Overcoming Limitations of Categorical Language Modeling

Shiran Dudy
Advisor: Steven Bedrick

A thesis presented for the degree of
Doctor of Philosophy

Center for Spoken Language Understanding
Oregon Health & Science University
November 2020

"Education means teaching a child to be curious, to wonder, to reflect, to enquire. The child who asks becomes a partner in the learning process, an active recipient. To ask is to grow".

Jonathan Sacks

Acknowldgements

There are many who I would like to thank, and who accompanied me throughout my journey. First, I am very grateful to have had Steven Bedrick as my advisor. I learned from him a lot: starting from the basics on how to ask a research question, to considering how and in what ways, in the grand scheme of things, our work adds to the general knowledge of our community. He also taught me how to attend to details more carefully, and to rigorously examine my steps and outcome in a methodological fashion. He always removed any roadblocks, and provided me with whatever assistance or advice about my work. Steven was *always* there when I asked (and I asked a lot). I am most appreciative of how he let me discover myself, his trust and support in me to follow my passion. He was everything I could ask for in a mentor.

I also would like to thank Melanie Fried-Oken who accepted me to her group and who exposed me to the world of assistive technology. Her dedication to relentlessly developing means to find the voices of the people who have lost their basic ability to communicate was inspiring to me. She taught me what it takes to run an interdisciplinary group. Most importantly, she has supported me throughout, and I am very fortunate for that. I also wanted to thank Peter Heeman who made it possible for me to graduate on time. Throughout the last year he has done everything he could to ensure I am provided with the resources and the knowledge to graduate and make a smooth transition onward. I do not take it for granted and appreciate him a lot for helping me make my first steps following my graduation. I wanted to thank Pat as well, as throughout the program she smoothly took care of every administrative issue. She was the first person who I saw when I arrived here and was always very friendly and welcoming. I am also grateful for Brian Roark who I learned from and consulted with whenever I got stuck. He always had the time and patience to listen, and offer good advice (which he always had). Finally, I would like to thank David Smith, who from time to time helped us in brainstorming over directions and ideas that I had in mind, and encouraged me to continue asking.

I wanted to thank my mother who was (and is) there for me throughout it all, who initially thought that pursuing a PhD so far from home was a crazy idea. Nonetheless, she still supported me from afar throughout. Ronen, my love, who I could not imagine going through the last stretch without. I am very fortunate to have him in my life. My sister who taught me that I can shape my reality in my own hands. To my dad who supported me and is very proud now. To my adopting mothers here, Dvora M., Karen, and Dvora T.. And especially to Dvora M. who was there for me whenever I needed a fresh perspective on life. To Naomi, Dudi, Yael, Ori and Cleiton who also my became my close family.

I wanted to thank my committee members Brian Roark, Peter Heeman, Meysam Asgari and Xubo Song for providing me feedback on this work and helping me strengthen my argument.

Overcoming Limitations of Categorical Language Modeling

Shiran Dudy

Abstract

Neural language models typically employ a categorical approach to prediction and training, leading to several well-known computational and numerical limitations. These limitations are particularly evident in applied settings where language models are employed as means for communication. From speller systems employed as assistive technology to texting applications on smartphones, all language models revolve around category-based prediction. Research shows that neural-category approaches to language modeling are questionable for predicting low-frequency words that are essential for user personalization. It is also challenging to adapt these architectures to a changing vocabulary due to the initially learned vocabulary constraints, which limit predictions of relevant categories (i.e., words) a user can type. Recently, such categorical models were shown to be relatively complex with long inference times, which may be detrimental for user engagement. In this thesis, I reevaluate neural-category approaches and propose an alternative: continuous output prediction.

Continuous output prediction is an underexplored alternative approach to language modeling that performs prediction directly against a continuous word embedding space. This approach splits the inference phase into two steps: a vector prediction followed by a vector decoding (mapping the vector to a category). Predicting a vector in an embedding space opens the door to a theoretically unlimited number of categories that can be represented and decoded using this technique. Technically, I show how given a trained model, continuous models' adaptation to a new vocabulary requires minimal architectural modifications compared to that of categorical alternatives. I also explore another important trait of continuous output prediction models: such models reach low-frequency vocabulary words that are often ignored by categorical models. I discuss the computational aspects of continuous output prediction, showing its promising results, especially in multiple-user settings and settings in which short inferences are required. Finally, to evaluate the diversity of categories predicted, including low-frequency words, I propose a simple metric based on the unique types predicted.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"No One Size Fits All"

A common notion in the assistive technology community is that 'there is no one size fits all' solution: in other words, that the choice of a solution greatly depends on the abilities of the user (Rubin, 2002). Users or patients who struggle to communicate due to motor-control or cognitive impairments may opt for augmentative and alternative communication (AAC) (Beukelman et al., 2020) devices that enable them to communicate with others in their surroundings.

Advances in AAC have led to a wide range of devices that allow users to engage with their environments, from eye-trackers (Stawicki et al., 2017; Barbara et al., 2016), to button presses (Shane et al., 2012; Gevarter et al., 2014), to brain-computer interfaces (BCIs) (Farwell et al., 1988; Donchin et al., 2000). Each of these modalities addresses different user needs and abilities, which may in turn affect the corresponding system's design. For instance, an eye-tracker is similar to typing on a keyboard, as both involve the coordination of voluntary controls (Elsahar et al., 2019), which in turn enables a direct selection modality. On the other hand, for individuals who may be lacking voluntary controls, button presses, switches, and many BCIs offer indirect selection (Elsahar et al., 2019). The goal of the various modalities is to elicit a signal to help these individuals spell a message.

This thesis focuses on the requirements for a communication device where oracles of the next token are proposed to the user in the form of text-entry prediction. The back-end of text-entry prediction often involves language models. Recently, language models have been integrated into BCIs in order to increase both communication rates as well as spelling quality (Speier et al., 2016). In this thesis, I focus on functional requirements a language model integrated into a BCI system should meet when employed for text-entry prediction, where conditions are exacerbated. This setting motivates setting higher demands to accommodate a user's communication needs, than the typical typing scenario.

At the center of this thesis are three requirements that form the axes according to which language models are evaluated: personalization, vocabulary adaptation, and low complexity requirements.

Personalization : Personalization in language modeling is usually expressed in modeling a user-specific language, which can be done by either augmenting the

model's existing vocabulary or adjusting the model's probabilities to make user-specific words more likely to be predicted. One motivation for personalization stems from the natural inclination of a speaker to seek reciprocity with their conversation partner through language style matching (Müller-Frommeyer et al., 2019). During a conversation, this reciprocity may be realized through both language patterns and word choice. A personalized language model offers more nuanced language that captures the unique vocabulary that characterizes a conversation, as shown by Wen et al. (2013), Vertanen et al. (2011), and Fowler et al. (2015). In this thesis, it is assumed that the words that are unique to particular conversations, or a user, tend to occur less frequently and therefore personalization is measured as the ability of a language model to predict the less frequent tokens in the text. Another aspect of personalization to consider is the deployment of a system that provides word prediction services for multiple users and as such is required to maintain personalized vocabularies. The cost of personalization across multiple users is investigated as well.

Vocabulary Adaptation: There are various motivations for working towards adaptive algorithms and adaptive language models in particular, the latter of which are often referred to as continual learning models (Parisi et al., 2019). Since humans adopt new words, it is important for language models to be able to learn about these new words. As humans, we regularly learn new vocabulary, whether through online communities (as shown by Danescu-Niculescu-Mizil et al. (2013)), our physically close circles (as shown by Crystal et al. (2008)), or self-inventions. This new vocabulary is integrated into *our* language. Language models need to be able to accommodate such expressive needs. Adaptive language models can address these communication needs by incorporating newly learned and relevant categories into a prediction list. They can also accommodate the changing needs of a user if their condition improves or deteriorates in such a way that causes a change in their communication needs.

Employing long-term algorithms can make for more sustainable models (Strubell et al., 2019; Schwartz et al., 2019) as well. The main reason long-term models may be more sustainable is that a fine-tuned model is superior to a newly trained model since the fine-tuned model is able to leverage previous knowledge to complete the task at hand (Howard et al., 2018). By preventing the need for initial, and expensive, training, such long-term models can provide improved performance with relatively lower effort (cost).[1] Therefore, scaffolding knowledge over time is less expensive than training new models each time. When defining the task of domain adaptation, one may consider pattern adaptation to a new domain. In this thesis, I suggest considering vocabulary adaptation to the new domain as well. Vocabulary adaptation refers to adaptation to the specific jargon that characterizes a domain.

Low Complexity: Szafir et al.'s (2012) study provides evidence that user engagement is affected by the immediacy of response. BCI systems have several moving parts that work together in a predictive text scenario. These parts include the signal processing module and the language modeling component. In this thesis, I evaluate the spatial and temporal complexities of the language model module in particular for the purpose of promoting sustainable user engagement. Communication rates in AAC in general (Elsahar et al., 2019), and in BCI (Mainsah et al., 2015) in par-

[1]Fine-tuning, by definition, requires a small amount of data; see Chapter 4.

ticular tend to be relatively slow, further emphasizing the need for low-complexity systems.

1.1 Problem Statement

Given the aforementioned language model requirements of personalization, adaptation, and low complexity, the current approach to language modeling in which a neural-categorical model is employed poses significant challenges. First, Holtzman et al. (2020) showed that common language models produce dull and repetitive text that is not akin to natural human language. Likewise, Dinan et al. (2019) showed that models predict infrequent words at a lower rate than these words are actually used by humans. Second, architecturally integrating new classes into a categorical model is a choice-dependent trade-off: either the categorical model must 'forget' the old classes that do not occur in the new dataset[2], or the model's complexity must increase to represent both the old and the new vocabulary classes. Third, these models' sustainability (Strubell et al., 2019; Schwartz et al., 2019) has been called into question due to their complexity costs, as already[3] noted by Jozefowicz et al. (2016). These costs stem from training, inference, and size.

In this thesis, I propose a strategy for designing language models that do not suffer from the aforementioned limitations. In particular, I propose a continuous output prediction language model, where predictions are made directly against an embedding space in the form of a regression. This embedding space represents vocabulary items (in the form of a vector) and is capable of containing varying (and theoretically infinite) numbers of entries, relaxing the need for a model to contain the learned entries. Instead of directly generating a probability distribution across the vocabulary, the model predicts a vector. Following the model's prediction of a vector, the vector is decoded to a particular class. Decoupling the prediction from its decoding step allows for a more controlled decoding process to take place (see Chapter 3). I show that when measuring models' class diversity and the prediction of low-frequency classes, a continuous model is superior to a categorical model. I also evaluate the computational burden incurred by continuous models and the potential gains of the models in their deployed settings. Finally, I propose considering language models as continual learning agents able to adapt to changing vocabulary patterns. I show that compared to categorical models, these models relatively have low-cost demands when they are required to adapt and can predict newly learned entries.

1.2 Thesis Contributions

This thesis makes several contributions. First, the thesis discusses the limitations of neural-based models, and in particular, category-based models. Second, it proposes an architecture that overcomes some of these limitations. I investigate these

[2]See, for instance, transfer learning (Howard et al., 2018; Gururangan et al., 2020) where the final layer is stripped off to accommodate the new task classes

[3]This research is pre-BERT era.

models in the setting of a word prediction task, and I propose a neural retrieval-based approach for language modeling and a new evaluation metric. Concretely, the following three contributions are made:

1.2.1 Retrieval-based language model

I introduce a GAN-based approach for continuous output prediction language modeling. This approach is inspired by Kumar et al. (2019), who designed a continuous prediction language model. Given the shortcomings of the categorical models, the proposed model is not architecturally bound to a set of classes; instead of generating a probability distribution over a *finite* set of classes, it directly predicts a continuous representation. Moreover, it is able to produce more expressive predictions thanks to the greater number of different word-types it predicts, including infrequent word-types, and it incurs reduced complexity costs compared to a categorical model.

1.2.2 Prediction diversity evaluation metric

Standard evaluation metrics used for language models often include accuracy and perplexity but may not account for other aspects that determine the models' usability. In this thesis,I propose an evaluation metric for prediction diversity. I show how in a word prediction setting, computing a model's relative rate of diverse types followed by a frequency breakdown of performance is related to the model's downstream performance over more and less frequent training examples. I also propose a 'usefulness' analysis that captures the semantic proximity of the non-exact matches a model predicts; that is, instances in which the model does not predict an expected target. This analysis reveals how many near-misses a model produces, where a prediction is similar yet not an exact match with regard to a target.

1.2.3 Adaptation in retrieval-based approaches

While domain adaptation is a common downstream task in NLP (Gururangan et al., 2020; Lee et al., 2020; Howard et al., 2018), I propose evaluating how well the continuous and the categorical approaches adapt to a new domain, as well as how successfully these approaches 'recall' previously learned domains. This contribution has the objective of advancing continual learning language models. This investigation exposes architectural flaws in the categorical models rendering them as less suitable to this task.

1.3 Organization of the Thesis

The next chapter of this thesis lays the groundwork for the original research that is presented in the following chapters. In Chapter 2, I describe the role of language and the goal of assistive technology in the context of AAC. I pay particular attention to BCI settings and various language systems that may be integrated into such devices. I also provide a review of language models and embedding spaces, and I elaborate on the challenges of using neural-category-based language models. In Chapter 3,

I argue for an alternative to the neural-category-based model: the continuous prediction model. This chapter presents the continuous approach, and in particular the proposed GAN architecture, and a thorough comparison of various categorical and continuous models is made. Chapter 4 investigates possible continual learning language models that incrementally adapt to new domains in a word prediction task. The categorical and the continuous approaches are compared and evaluated on various aspects related to continual learning. I argue that the continuous models offer a more promising architecture for this type of task. In Chapter 5, I investigate ways to enhance the proposed language model: first, by combining it with a categorical model; second, by using the proposed decoding mechanism; third, by learning about the role of embedding space in the process; and fourth, by evaluating several objective functions. Chapter 6 focuses on evaluating diversity in categorical state-of-the-art models. A variant of the metric that is used to measure diversity in Chapter 3 is proposed, shedding light on the diversity and long-tail prediction of state-of-the-art models. Another diversity metric is proposed for model analysis. This metric uncovers to what degree non-exact matches are semantically close to their targets, and a possible link is shown between low performance on a downstream task and low performance on the proposed evaluation metric. This thesis concludes with a summary of its findings.

Chapter 2

Preliminaries and Background

Society
noun
1. The aggregate of people living together in a more or less ordered community.

Oxford Dictionary

2.1 On the Roles of Language

An important aspect that differentiates humans from most animals is the ability for social intelligence (SI). Social intelligence is defined as the ability to 'get along well with others while winning their cooperation', according to Albrecht (2006).[1] Considering this, human cooperation is not restricted to the tool-making that propelled humans to the top of the food chain but in fact is expressed to a large degree in humans' ability to be sensitive to others' interests and perceive others' perspectives on the world. This skill gives humans the ability to not only understand but also predict others' behavior, in what is defined as the theory of mind (ToM) (Premack et al., 1978). As humans, throughout our lives we socially interact with other humans and frequently share our beliefs, intentions, desires, emotions, and knowledge, among other aspects. Yet what happens if this faculty is taken away from us? There may be various reasons that an individual struggles to establish social relationships, leading to social isolation and loneliness (Cacioppo et al., 2008).[2] This is particularly the case for individuals with speech motor disorders that impair their ability to partake easily in social interactions. However, perhaps an even more crucial outcome of communication incompetency is survival; indeed, the ability to communicate is indispensable in today's society. The term 'society' is based on humans' interactive nature (Oxford, n.d.). Those individuals who struggle to find their voices can today find ways to maximize their communication potential with the help of augmentative and alternative communication (AAC). Research in AAC is flourishing right

[1] It was originally defined in Thorndike (1920) as 'the ability to understand and manage men and women and boys and girls, to act wisely in human relations'.

[2] Loneliness as a scale recently has been systematized to describe the content levels of current relationships (Goodman et al., 2017).

now due to recent advances in hardware and software. According to the American Speech-Language-Hearing Association (ASHA), the goal of AAC is to

> *'study and compensate when necessary for temporary or permanent impairments, activity limitations, and participation restrictions of individuals with severe disorders of speech-language production and/or comprehension, including spoken and written modes of communication'.*(ASHA, 2004)

Speech-producing aids converting text-to-speech (TTS), hearing devices based on speech recognition algorithms, and even brain-computer interfaces (BCIs) are all aimed at facilitating communication and may help to create a more inclusive society.

2.2 Augmentative and Alternative Communication (AAC)

As mentioned earlier, the main purpose of AAC devices is to improve to the extent possible the communication of an individual who experiences communication disabilities. There are various ways to do so, as presented in the following sections. AAC can be divided into aided and unaided communication (Beukelman et al., 2020). Unaided AAC does not require an individual to have an external device; rather, the individual employs the body for communication. Many human beings augment their communication with body language such as gestures, signs, vocalizations (e.g., bursts, sighs), and eye blinks, as well as sign language. Some of these acts are shared conventions across cultures, but others may be predefined between a patient and their caregiver. For instance, a patient and their caregiver may establish the convention of blinking once for the word 'yes' and twice for 'no'. Aided AAC includes low- and high-tech solutions. An example of a low-tech device is a communication board, as seen in Figure 2.1a. A patient learns to compose a message by choosing from a symbol set board (containing figures and/or words). This board can help a patient overcome their speech production difficulties. A typing stick, as seen in Figure 2.1b, is another low-tech solution for individuals who may not be able to voluntarily move their hands.

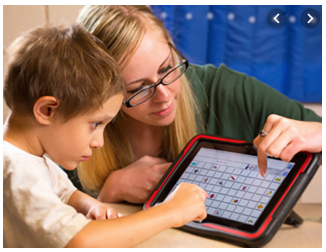
(a) Communication Board (Assistive Ware)

(b) Typing Stick (Allegro Medical)

Figure 2.1: Low-tech AAC devices.

One example of a high-tech AAC device is a digital communication board. Digital communication boards present rich symbol interfaces, as shown in Figure 2.2a;

unlike a low-tech board, a digital board may have a dynamic display. Another AAC example, a BCI, is shown in Figure 2.2b. A BCI is a way of 'utilizing the brain signals in a man-computer dialogue' (Vidal, 1973). The type of BCI presented in Figure 2.2b is for the purpose of verbal communication rather than movement control or simulations. In the following subsections, I elaborate on different BCI applications and interfaces.



(a) Digital Board (Lyra symbol to speech)

(b) Brain-Computer Interface (CAMBI labs)

Figure 2.2: High-tech AAC devices.

2.2.1 BCI systems

Various disorders can disrupt the neuromuscular channels through which the brain communicates with its external environment. Brainstem stroke, brain or spinal cord injury, amyotrophic lateral sclerosis (ALS), cerebral palsy, muscular dystrophy, multiple sclerosis, and other diseases impair the neural pathways that control the body's muscles. In the United States alone, nearly two million people are affected, and there are many more around the world (Ficke, 1992; Health et al., 1992; Murray et al., 1996). Having a BCI can provide these individuals with basic communication abilities (Rezeika et al., 2018). A BCI system allows an individual to control a computer through their brain signals. The system measures a specific brain activity and converts it in real time to commands for communication and control (Wolpaw et al., 2002). Originally BCIs were developed to control external apparatuses, or prosthetic devices (Vidal, 1973). This project was started in the 1970s by researchers at the University of California, Los Angeles, together with DARPA.[3] Today's full-fledged BCI systems allow for mobility through both movement control (Ma et al., 2017) and limb control (across ages, for various neuronal dysfunctions) (Chi et al., 2018; Manero et al., 2019). In the past decade BCIs have also been used for AAC communication, including message typing (Orhan et al., 2012), as mentioned earlier.[4],[5]

Invasive vs. non-invasive BCIs

There are two main types of BCIs: invasive and non-invasive. Until recently, the leading approach was non-invasive electroencephalography (EEG)-based BCIs.

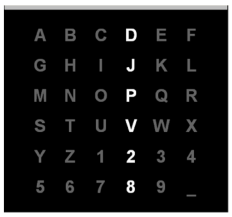
[3] Vidal (1973) mainly addressed limb injuries faced by military members in war zones

[4] I suspect communication BCIs eventually will be used not only to address communication disabilities, but also to facilitate communication within the wider population.

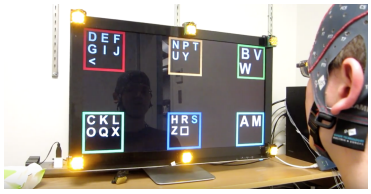
[5] BCIs are used in the gaming industry and other fields that are not covered here.

These BCIs elicited brain signals through an EEG cap made with electrodes attached to the scalp. EEG was discovered in the beginning of the 20th century when an animal's signal was first recorded (Pravdich-Neminsky, 1912). Over time the technology progressed to include a higher-quality signal acquisition device made possible by invasive electrocorticography (ECoG) (Palmini, 2006). Whether to employ invasive or non-invasive BCI is a question of trade-offs. Waldert (2016) offers a review of the trade-offs related to the different techniques, pointing to several criteria that merit consideration. First, the sensitivity level of a technique may be a factor. EEG requires many more neurons to produce a signal in order for the signal to propagate to the electrodes on the scalp, while extracellular action potentials and local field potentials (both of which are common forms of ECoG) can detect a signal from smaller, more localized regions (Waldert, 2016). When EEG detects a signal from a small region, the signal-to-noise ratio (SNR) is much greater than the alternatives. Second, non-invasive signals are limited to a small range of frequencies to be elicited from the brain as they act as low pass filters (< 90 Hz), while invasive techniques are capable of a greater range (up to several kHz ranges) (Waldert, 2016). Third, the indirect interface of an EEG cap produces a distorted signal resulting from the transition of the signal through the cerebrospinal fluid, skull, and scalp (Waldert, 2016), though ways to mitigate this distortion have been proposed (Michel et al., 2012). The type of signal retrieved from a user can be more complex when employing ECoGs. For instance, it can be read from the motor cortex to recover speech, as done in Angrick et al. (2019); however, this would require the user to think about saying a word. On the other hand, EEG often operates by the odd-ball paradigm (Donchin et al., 2000; Higger et al., 2016; Fried-Oken et al., 2015), such that the interface requires identifying the intended target and choosing it by producing binary responses to a set of proposed candidates. The information transfer rate is higher in ECoG in part due to that reason. On the other hand, Blabe et al. (2015) reports that individuals are less inclined to undergo the invasive procedures required when using ECoGs. There is a non-negligible rate of complications caused by using invasive techniques: Rolston et al. (2016) reported 3.4 and 9.6 major and minor complications among the 177 cases examined in their study. In addition, privacy concerns related to using ECoGs have been discussed by Ahmadi et al. (2014) and Chaudhary et al. (2018). While these concerns also apply to EEG, they are exacerbated by invasive techniques. Overall, according to Waldert et al. (2009), EEG yields lower performance than extracellular action potentials, as well as local field potentials. However, despite the high quality and the promise of ECoG, complications, risks, and user acceptance still stand in the way of its widespread usage.

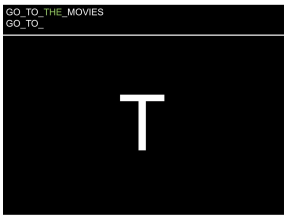
I focus on EEG-oriented interfaces for communicational BCIs in the following sections, as this is the project I participated in; however, the implications of this thesis are pertinent to invasive techniques that incorporate language models as well. In order to spell a message, a user has to interact with an interface. I discuss here the three different interfaces for BCIs that are presented in Figure 2.3. First, as shown in Figure 2.3a, a row-column scanner scans the each row in order to narrow the search for the row where the target letter is found, and then each column is scanned in order to find the target letter (conditioning on the row) for choosing the intended letter within that row. The signal elicited during scanning is the p-300 which is associated with decision-making, as first discovered by (Chapman et al., 1964). Second, as shown in Figure 2.3b, letters are grouped into boxes, and each



(a) Row-Column Scanner
Donchin et al. (2000)



(b) SSVEP
Higger et al. (2016)



(c) RSVP-Keyboard
Fried-Oken et al. (2015)

Figure 2.3: Communication interfaces of BCI.

box's frame flashes light in a different frequency in a steady-state evoked potentials (SSVEP) (Cecotti, 2010). When a match is found between a box and the elicited signal frequency (or harmonics), a splitting takes place (regrouping of the letters in the chosen box) in an iterative process until the target letter is found. Third, as shown in Figure 2.3c, a p-300 signal is elicited through a rapid serial visual presentation of letters: RSVP-Keyboard (Orhan et al., 2012). This interface is more flexible in nature as it allows for scaling to a large vocabulary symbol set. This may not be as simple for the previously presented interfaces, as they are conceptually more discriminative in nature, requiring other approaches to encompass the complete vocabulary during signal acquisition.

2.2.2 Icons

Icons in nature are aimed at augmenting communication. I will present a brief history about the origins of icons and then how icons are employed today in the context of AAC. Four millennia before icons were used to communicate and prior to the development of Mesopotamian script, around 8000–3000 BC, tokens were used. Tokens were geometric shapes such as cones, spheres, and discs, each of which conveyed a single concept associated with goods (Schmandt-Besserat, 2010). For instance, a disc and a cone meant a grain basket and an oil jar, respectively; x discs and y cones meant x baskets of grains and y oil jars, respectively. Tokens mainly conveyed semantics, which was part of a convention system, and they lacked any syntax. Around 3500 BC, the token system led to the development of writing. Tokens represented debt and were stored in an envelope that was a ball-shaped clay container. To prevent himself from opening it, the envelope owner made an impression mark on the external side of the envelope for bookkeeping purposes (Schmandt-Besserat, 2010). An additional milestone happened in about 3100 BC, when there was a transition from one-to-one mapping to plurality representation. Plurality representation allowed for more information to be conveyed with less 'bits'. Icon usage of pictograms is the most ancient method for visually representing verbal communication. Like any other writing system, this ancient method was based on a shared understanding of meanings and their character set. Later on, logography was introduced. Logography assigned a single pronunciation to a sign indicating the individual who owned goods for accounting purposes (Schmandt-Besserat et al., 2008). The man and mouth signs were associated with a single specific sound. Current writing systems evolved from icons that initially represented the conversants'

physical environment and gradually became a form of abstract representation. For example, the meaning of the fish symbol became detached from its shape. Abstract concepts detached from the physical world started to emerge and shape the languages humans speak today. For instance, the signs on the Mesopotamian block in Figure 2.4, in which each row represents a more advanced era, were gradually abstracted from the original environment. This graphic evolution occurred mainly during cuneiform and was a precursor for alphabet writing systems (Walker, 1987).

Figure 2.4: Cuneiform signs, with each sign's pictographic forms at 3000 BC, 2400 BC, and 650 BC presented in vertical order. (Walker, 1987)

Today icons can be employed in situations in which the symbol set of a language may not be accessible to an individual due to an injury, for instance, that prevents them from inferring meaning from a written language or easily producing intended words. Ponsford et al. (1995) indicated in their experiment that as many as 70% of individuals with traumatic brain injury (TBI) report word retrieval problems, while

35% report comprehension difficulties. Aphasia, as another example, is a condition defined by an inability to comprehend and formulate language caused by damage to specific brain regions (Damasio, 1992). A study by Koul et al. (1998) found a 'facilitative effect on the learning of graphic symbols by individuals with severe chronic aphasia'; in other words, icons were a supportive alternative for communication.[6] The usage of icons may also have to do with a younger age range, as icons are more intuitive for communicators to understand during the pre-reading skills period. The usage of icons in a picture exchange communication system (PECS) was found to increase communication in the form of requests and comments made by elementary school-aged children. A similar study by (Kravits et al., 2002) showed that icons used in functional communication training (FCT) elicited more requests from children with autism (Hines et al., 2008).

Icon-based AAC systems

The Blissymbols (Bliss, 1949) language was created by Charles Bliss, a Jewish person who escaped Nazi Germany to a concentration camp in China and developed an icon-based writing system. This system was widely adopted for AAC purposes in 1970-1980. Blissymbols was a compositional language containing equivalents to nouns, verbs, and adjectives in the form of material things, actions, and human evaluations. MinSpeak (Baker, 1982) is another symbol system whose goal was to enable individuals with impaired communication abilities to communicate through icons and icon combinations that were based on multiple-meaning iconic encoding. For example, an icon with an image of a `frog` could refer to the concepts of 'frog', 'green', 'jump', and 'water'. Pairing `frog` with other icons could lead to various concepts as well: `frog` + `cup` is 'pond', `frog` + `rainbow` is 'green', and `frog` + `arrow` is 'jump', as explained in Glennen et al. (1997) (see Figure 2.5 for another example). Patel et al. (2004) subsequently argued that additional flexibility may

Months Vocabulary	First Symbol	Second Symbol	Rationale
January	Calendar	Baby	New Year's Baby
February	Calendar	Heart	Valentine's Day
March	Calendar	Lamb	March Goes Out Like a Lamb
April	Calendar	Umbrella	April Showers
May	Calendar	Lei	May Flowers
June	Calendar	Outside	Summertime
July	Calendar	Flag	Fourth of July
August	Calendar	Pool	Hot/Go Swimming
September	Calendar	Teacher	Back to School
October	Calendar	Witch	Halloween
November	Calendar	Dinner	Thanksgiving
December	Calendar	Santa Claus	Christmas

Figure 2.5: MinSpeak rule-based picture symbol sequencing used in the interaction Education and Play Plus MAP system (Bruno, 1988).

be required and proposed relaxing the linear constraints of word order to produce

[6]The subjects were adults who were considered literate.

messages. For example, any combination of (`girl`, `go`, `home`) could be used to produce the following sequence: 'The girl is going home'. Patel et al. argued that this would make for more natural and efficient communication. Around the same time of Minspeak, DynaVox (Friedman et al., 1983) was producing another icon-based AAC system. To overcome the limitations of a limited icon display, DynaVox relied on a varying display from which individuals chose icons. This system also made an attempt at predictive typing. For instance, in an early DynaWrite version, the creators included a picture retrieval that was based on spelling: for instance, a child would type the first letter, and the images beginning with that letter would be displayed until a single image was left. Keskinen et al. (2012) advanced picture-based communication by developing an instant messaging service called SymbolChat. SymbolChat was fully customizable and did not require prior training to use. Wiegand et al.'s (2012) SymbolPath was another icon-based piece of software, allowing the user to generate a sentence through a touchscreen or by gazing over a board that grouped words by functionality, as shown in Figure 2.6 A

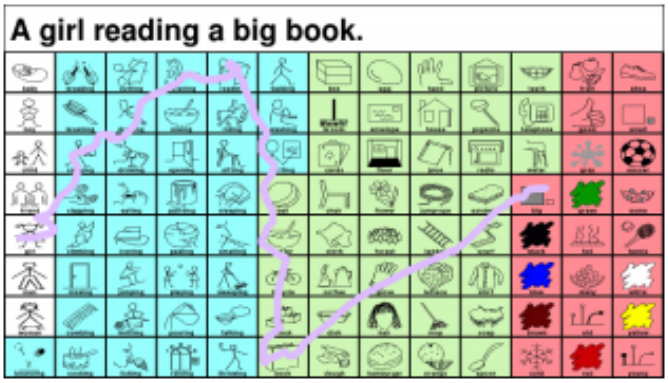


Figure 2.6: Sentence generation by traversing target words on the board (Wiegand et al., 2012b).

more pertinent example of an icon language is SymbolStix (Clark, 1997), which is used in several commercial AAC applications today including a weekly newspaper, learning tools, and educational games. Incorporating SymbolStix into spellers is a research question I previously addressed (Dudy et al., 2018), and triggered the questions that are raised in this thesis.

The aforementioned systems are focused on AAC and often form independent icon languages. In recent years emoticons and emojis have become an integral part of humans' symbol set, as they are embedded in almost every technology-based writing system. McCulloch (2020) described emojis as a supplement to and embodiment of written text. Text that is detached from emotions, informal text, and texting in particular can be challenging to decipher simply because the conversation partners cannot hear or see one another. Emojis and emoticons (:-D) attempt to address this void by symbolically representing emotions or other mental states, thereby making it seem as if the conversation partners are speaking to one another in person. McCulloch further split the emoji symbol set into two types of families: a group of co-speech devices and a group of emblems. One way to differentiate the two is by considering their use in sequences. Co-speech refers to making illustrative gestures in a fluid way to convey a single concept such as 'happy birthday' or 'danger', as if describing the concept to another person using body language. On the other hand,

emblems may repeat but are less likely to combine well with other emojis. This is because their conventional meanings may not be taken at face value but rather be more context-dependent and representative of discrete ideas. Therefore, emojis are also defined as AAC, as they help humans convey another dimension of information through text with the goal of augmenting communication.

This section described various types of assistive technology. I focused on AAC with BCIs in particular, as well as BCIs' different interfaces and alternative symbol systems for communication in AAC settings. In the following section, I focus on another core component of this work: language models, to which these icons (or various symbols sets in general) are often incorporated. Language model in the context of AAC help propose the next symbol to the user.

2.3 Language Models

Languages, and sentences in particular, are not random sequences of tokens; rather, they are formed under a certain logic that is realized in a set of patterns. The foundation for this assumption was laid in 1913 by Markov (an English-language translation based on the Russian source is in Markov (2006)). Markov chains were used to make predictions about whether the upcoming letter in Pushkin's 'Eugene Onegin' was a vowel or a consonant. Shannon (1948a), who is assumed to have been influenced by Markov chains, applied the n-gram approach to language modeling (described in section 2.3.2) to English word sequences. The foundations set by Markov and Shannon rely on the statistical deduction of probabilities based on textual data. In several influential papers (Chomsky, 1956; Chomsky et al., 1957; Miller et al., 1963), Chomsky argued that probabilistic approaches, and Markovian methods in particular, are not suited to describing the cognitive processes that take place in the brain to generate language. Chomsky claimed that while Markov and Shannon's probabilistic approaches provide an engineering solution that may at best describe how to generate language, the lack of accounting for why these methods may be similar to how language is produced makes these methods fruitless. Chomsky thus instead advocated for non-statistical approaches. In Chomsky (1956), Chomsky et al. (1957), and Miller et al. (1963), Chomsky argued for a generative grammar theory, which is a system of rules that explains how sentences are generated.[7] Among the many controversial objections made in the paper, one objection I found in Chomsky et al. (1957) to be especially compelling is that since these models are frequency-based, they may fail to produce valid sentences (grammatically correct sentences that describe plausible situations, as opposed to 'colorless ideas' phrases) such as those that humans generate, simply because these sentences may be constructed of less likely tokens or form less likely sequences. This comment in particular is investigated in Chapter 6. A couple of decades later, two laboratories worked towards speech recognition systems. One of them was at CMU (Baker, 1975), and the other was at an IBM research center. The laboratories collaborated in applying tri-grams following Shannon (1948a) to language modeling, marking the rise of statistical approaches (Baker, 1990; Jelinek, 1990). The n-gram technique dominated the field of natural language processing until nearly a decade ago, when neural networks started to become popular through Bengio et al. (2003) who introduced a more

[7] Deterministic/rule-based language models are not covered in this thesis.

robust modeling approach. In the following section, I mainly discuss n-grams in section 2.3.2 and neural networks in section 2.3.4 as means for language modeling; I also consider their relevant language model applications.

2.3.1 Language models' application

Today speller systems in AAC, and many text-entry prediction methods employed by our smartphones (Zaykovskiy, 2006), rely on back-end algorithms called language models (LM). Drawing on an interface perspective, Light et al. (1992) proposed a dynamic display for AAC systems that would enable scaling of the vocabulary. Given the predictive nature of language models, incorporating such models into AAC systems has been shown to reduce increase the rate of typing, as well as the quality of typing. To this end, Speier et al.'s (2016) review on incorporating language models into BCI speller systems lists several axes on which language models' incorporation improves the spelling process: output characters per minute (OCM), information transfer rate (ITR), theoretical bit rate (TBR), accuracy (ACC), and symbols per minute (SPM). Some of these measurements reflect speed and others quality. For instance, Ryan et al. (2010) reported 40% improvement on OCM when employing a word completion algorithm using a dictionary as a language model as opposed to a non-predictive speller. Orhan et al. (2011) reported an increased ACC when using various n-gram language models learned through recursive Bayesian estimation compared to relying solely on EEG signals. Speier et al.'s (2011) study showed a 50% increase in ITR on a p300 speller with an n-gram approach. Finally, Kindermans et al. (2013) showed a 61.4% SPM improvement when using n-gram-based p300 spellers. A particular limitation when considering the techniques of many of these studies, though, is that much of the research conducted was based on use patterns of healthy people (Speier et al., 2016) and very few laboratories evaluate their systems with target BCI users (Orhan et al., 2011; Oken et al., 2014; Mainsah et al., 2015). Language models have also been shown to be promising when incorporated with icon-based AAC systems. For instance, Wiegand et al. (2012a) proposed using a bag-of-words-like prediction instead of forcing a user to be subjected to word order when composing a message, and Dudy et al. (2018) simulated an icon language (based on textual patterns) in order to train a language model (which as mentioned triggered the questions in the following Chapters, though this work is not the focus, and will not be elaborated in what follows).

Language models play a crucial role in many speech and language applications as well. Katz (1987) presented a language model 'back-off' smoothing technique to overcome language sparsity. When Katz's (1987) approach was evaluated on IBM's ASR,the model demonstrated perplexity on par with that of alternative language models, assisting in real-time isolated word recognition. Nanjo et al. (2003) presented ways to improve speakers' recognition rates and accommodate pronunciation variations by directly adapting to speakers' data (including LM adjustments). Furui (2005) proposed a way to overcome speech recognition performance on spontaneous speech, pointing to a mismatch that requires collecting and training models on spontaneous speech rather then read speech when these ASRs were deployed on the latter's settings. Most work by the time of Nanjo et al. (2003) was based on count-based approaches. Graves et al. (2014) presented an end-to-end approach for ASR, with the connectionist temporal classification marking the beginning of the

neural network era in speech recognition. Various neural models were proposed in modeling speech that integrate LMs implicitly (Chorowski et al., 2015; Graves et al., 2013) and explicitly (Kim et al., 2017; Hinton et al., 2012). Language models were also introduced in vision-based applications of optical character recognition (OCR) (among other applications). Chang et al. (1995) was the first to propose a way to integrate a language model into OCR systems as a constraint for the types of words or strings the algorithm initially recognizes and through that to possibly recover unseen characters given sufficient context. Such models became a common practice (Kompalli et al., 2009; Fischer, 2012).

2.3.2 Statistical language models

Following the review of how language models are integrated in AAC systems and other applications, inevitably we ask how do language models work? As mentioned earlier, the statistical approach to modeling a language can be illustrated by formulating a probability distribution $p(s)$ over strings s that attempts to reflect how frequently string s occurs as a sequence. $p(s)$ is regarded as a sequence of tokens $p(w_1^n)$ that is modeled based on the joint probability of a string of tokens, as shown in Eq. 2.1 (and the chain rule):

$$\begin{aligned} p(w_1^n) &= p(w_1, w_2, w_3, ..., w_n) \\ &= p(w_1)p(w_2|w_1)p(w_3|w_1^2)p(w_n|w_1^{n-1}) \\ &= \prod_k^n p(w_k|w_1^{k-1}) \end{aligned} \tag{2.1}$$

Estimating these probabilities quickly becomes impossible due to the long dependencies involved in the computation of (time-wise) advanced tokens in the sequence. The dominant approach to addressing this problem had been to employ a stochastic count-based approach called n-grams (Chen et al., 1999).[8] N-grams are based on the Markovian assumption (Damerau, 1971) that simplifies the aforementioned dependencies to compute a token's probability based only on the first few tokens immediately preceding it, resulting in Eq. 2.2. For instance, a two-gram or bi-gram case is one in which the the current token is conditioned on the previous token.

$$p(w_1^n) = \prod_k^n p(w_k|w_{k-1}) \tag{2.2}$$

Given a textual corpus, computing a bi-gram probability requires calculating the ratio of the number of times a word and its context appeared in the text to the number of times the context appeared in the text irrespective of what followed. This is a maximum likelihood estimation (MLE) (Eq. 2.3):

$$p(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n)}{\sum_w C(w_{n-1})} \tag{2.3}$$

[8]There were also hidden-Markov language models that were presented by Kuhn et al. (1994), though they were not a common alternative.

A tri-gram probability is computed by conditioning a probability on the two tokens tokens immediately preceding the current one; therefore, an n-gram model describes the general case. Unigrams are computed based on the frequency of every token type in the corpus. N-grams allow both for feasible computation and a degree of dependency on previous observations.

One major advantage of employing an n-gram approach is that it is a white-box language model. In other words, one can reverse-engineer the decisions the model produces relatively easily. To illustrate the white-box strength, I built an n-gram toy model using the openfst (Allauzen et al., 2007) toolkit, which has a weighted finite state representation. Figure 2.7 shows a finite state representation in which the unit of prediction is a character that forms strings that produce words, with the goal of representing a word-based vocabulary of three different words: 'happy', 'home', and 'hole'. This is a compact representation that has states, arcs, and different symbols that are associated with transition probabilities. Since all the words in this language begin with 'h', 'h' is the prefix of every word and therefore it is associated with a probability of 1 to form a valid prefix in this language. On the other hand, if an 'h' prefix is typed (in an attempt to spell a word in this language), then there is about a one-third chance 'a' will be chosen to follow 'h' (producing the word 'happy') and twice as many chances that 'o' will be chosen (producing either 'home' or 'hole'). Understanding the transitions helps elucidate the internal process of the LM and thereby make it transparent; in other words, a white box. This helps with controlling the probabilities in a more straightforward way.

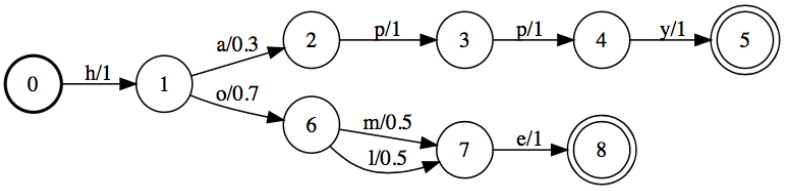


Figure 2.7: A letter-based language model lattice.

Another advantage is the count-based models can provide an estimation for an unseen term. As shown in section 2.3.2, this process helps in situations in which a certain sequence cannot be found in the lattice and the goal is to compute the probability of that particular sequence. For instance, computing the probability of the sequence 'how' would default to computing $\alpha p(\text{`}h\text{'})p(\text{`}o\text{'}|\text{`}h\text{'})$, as there is no information regarding the $p(\text{`}w\text{'}|\text{`}o\text{'})$. In a way, though, the problem of sparsity is inherent to the nature of n-grams. The problem is handled differently in neural approaches. The limitations of this approach in the context of neural networks are discussed further in section 2.3.5. In the following section, another approach to language modeling is described: neural network language models.

Smoothing

One question with such models is how to evaluate unseen probabilities. For instance, if one wishes to compute $p(w_n|w_{n-1})$ given that the textual document does not contain the word w_n. For example, let's compute the probability of the sequence 'how can I find the Arboretum', where w_n is associated with the word

'Arboretum', however, the word 'Arboretum' was not seen in the training set. Since this word-type ('Arboretum') exists in the language, it therefore should be assigned *some* probability, even if the word was not seen during training (a zero probability for the given sequence is the outcome of an un-smoothed language model). How should this probability be assigned? Various approaches have been devised for this purpose. While there are other techniques (Brown et al., 1992) for overcoming data sparsity, smoothing is a common one in n-grams. In this section I elaborate on several common n-gram-oriented smoothing techniques.

One way is to perform Laplace smoothing, which is based on the add-one estimate that is attributed to Jeffreys (1998) (this work is based on the theory by Laplace (1902)). During Laplace smoothing, bi-gram probabilities (as an example) are computed in the following way to avoid zero-probability assignments to unseen tokens:

$$p(w_n|w_{n-1}) = \frac{C(w_{n-1}, w_n) + 1}{\sum_w C(w_{n-1}) + |V|} \tag{2.4}$$

However, Gale et al. (1994) argued that the add one approach performs poorly.

Following the Good-Turing estimators by Good (1953), Church et al. (1991) and Gale et al. (1995) presented a Good-Turing estimation as a smoothing approach for n-grams. The Good-Turing smoothing is based on the idea that probability mass could be reallocated from n-grams that occur $r + 1$ times to n-grams that occur r times, with a focus on reallocating from single events to n-grams that were never seen. To that end, for each count an adjusted r is computed:

$$r* = (r + 1)\frac{n_{r+1}}{n_r} \tag{2.5}$$

where n_r is the number of n-grams seen exactly r times.

$$p_{GT} = (x : c(x) = r) = \frac{r*}{N} \tag{2.6}$$

Jelinek et al. (1990) presented n-gram smoothing with a fixed coefficient interpolation that combined MLE (Eq. 2.3) with interpolated probabilities, assigning smoothed probabilities recursively and relying on partial data.

$$p_{interp}(w_n|w_{n-i+1}^{n-1}) = \lambda_{n-i+1}^{n-1} p_{ML}(w_n|w_{n-i+1}^{n-1}) + (1 - \lambda_{n-i+1}^{n-1}) p_{interp}(w_n|w_{n-i+2}^{n-1}) \tag{2.7}$$

where λ_{n-i+1}^{n-1} can be estimated and is context-dependent (higher λs for longer contexts) and is calculated per context length. This has been applied to domain adaptation (Bellegarda, 2004) where a modified version is

$$p_{interp}(w_n|w_{n-i+1}^{n-1}) = \lambda p_{D_1}(w_n|w_{n-i+1}^{n-1}) + (1 - \lambda) p_{D_2}(w_n|w_{n-i+1}^{n-1}) \tag{2.8}$$

In domain adaptation usually the assumption is made that a small targeted domain (for a particular task) is available that could benefit from general patterns (and reduced sparsity) when mixed with a large general textual corpus. I apply this approach in section 5.2 for a purpose other than domain adaptation.

Another prominent smoothing approach was proposed by Katz (1987). This approach allows back-off to shorter contexts. Here I examine a tri-gram version (Eq. 2.9):

$$p(w_n|w_{n-2}w_{n-1}) = \begin{cases} p(w_n|w_{n-2}w_{n-1}) & if C(w_{n-2}w_{n-1}w_n) > 0 \\ \alpha p(w_n|w_{n-1}) & if C(w_{n-2}w_{n-1}w_n) = 0 \wedge C(w_{n-1}w_n) > 0 \\ \alpha p(w_n) & else \end{cases} \tag{2.9}$$

How can the best approach for a given task be determined? The next section discusses common evaluation metrics for structured prediction LMs assuming an expected target, as is the case in a word prediction task.

2.3.3 Evaluation metrics

The various smoothing approaches described and the improvements achieved were mainly evaluated following Shannon (1948b), who brought the concept of entropy from thermodynamics to information theory. This has become a common way of evaluating a model's probability distributions. Perplexity (PPX), which is a variation on entropy, indicates the level of confidence of a language model with regard to a certain textual document, as observed through the probability assignment to a certain target. As seen in Eq. 2.10, computing PPX involves averaging the probability mass ($q(x)$) assigned to an 'expected' word transition found in the textual document. A higher probability mass for the expected transition (one which is potentially more skewed towards an expected transition) indicates increased confidence in the language model for that particular transition and suggests that the model may represent the language patterns of that corpus more reliably than a model with low confidence. Technically, if $q(x)$ is high, the term in the power position will be lower, and therefore the lower the PPX, the more optimally the predicted patterns have been learned. The purpose of language models is therefore to predict the patterns of the language. Perplexity can be compared only within the same data set, and since it is not an absolute score, PPX is meaningful when compared to other models. It describes a relative difference; it is not an absolute scoring technique.

$$PPX(p) = 2^{-\sum_{x \in X} q(x) \log p(x)} \tag{2.10}$$

Another metric pertinent to the thesis topic called Distinct-N was proposed by Li et al. (2016a). This metric measures how diverse the tokens are given a generated textual response. To indicate how varied the response is, the metric represents the number of unique n-grams scaled by the total number of generated *tokens*.

Additional aspects evaluated in language models are complexity computations for efficiency and accuracy metrics when a target reference is expected. These are considered intrinsic evaluations. Currently many tasks in natural language processing (NLP) rely on language models for downstream predictions such as question-answering, natural language inference, and sentiment analysis that follow task-specific metrics by which to evaluate performance.[9]

2.3.4 Neural network language models

A more prominent type of language model than the n-gram models are the nerual language models. Perhaps a precursor to the neural network language models and neural networks in general was the Perceptron model introduced by Rosenblatt

[9]In speech recognition, the common metrics employed are phoneme error rate and word error rate; in tasks of machine translation, which I do not cover in this thesis, BLEU (Papineni et al., 2002) scores, METEOR (Lavie et al., 2009), and others have been applied.

(1957). The Perceptron model consisted a single layer of input functions and a single output. This model was criticized by Minsky et al. (1969) for the assumptions that it made that did not enable parity computing (such as computing XOR), resulting in what is referred to as the AI Winter (Crevier, 1993). Nonetheless, important breakthroughs gradually accumulated, contributing to the development of the current neural nets. Rumelhart et al. (1986) introduced the back-propagation concept, which suggests a network cannot only measure the error of a predicted representation in relation to a target representation, it can also inform the network how to improve its predictions so that they are closer to desired target. This can be done by updating the network's weights and has been demonstrated on a basic prediction model of language, a precursor to neural network language model (NNLM).

Following the introduction of the back-propagation concept, Elman (1990) introduced the first basic architecture for neural network language modeling: recurrent neural network language modeling (RNNLM). This architecture is comprised of a recursive-behavior component and a final layer to produce an estimate for a given target. The recursive component enables the processing of temporal sequences that do not require an additional dimension of an input to be represented, also referred to as a memory unit.[10],[11] In RNNLMs, a node can access the previously-processed input node and affect its subsequent 'behavior'. Elman (2004) subsequently entertained the idea that in his previous work (Elman, 1990), the RNN's hidden layer simulated activations that may take place in the brain, bridging the difference between the symbolic (e.g., n-grams) and the connectionist approaches, where activations of the same word can be realized differently depending on their contexts. For instance, in a connectionist model, there are different representations for the verb 'play' and the noun 'play', and these different representations are claimed to be a strength of this approach. The RNN unit has evolved to enhance the model's memory, as well as overcome the exploding/vanishing gradients noted by Pascanu et al. (2013). The basic recursive unit has evolved over time. Long short-term memory (LSTM), introduced by Hochreiter et al. (1997), offered a sophisticated gating strategy that was integrated into an RNN node; a gated recurrent unit (GRU) was subsequently introduced by Cho et al. (2014) as a less complex alternative that also was integrated into an RNN node. Recently Vaswani et al. (2017) proposed a way to allow for the parallel processing of sequences, presenting a new architecture called the Transformer. This architecture somewhat contradicts the underlying idea of the implicit decoding of time by introducing explicit temporal markers to be provided as inputs while speeding up the runtime under some settings.

Following the advancements in the field, Bengio et al. (2003) formalized NNLM and pointed to possible flaws in traditional n-grams to illustrate how neural language may provide an alternative to overcome them. First, NNLM can overcome the limited context of n-grams, making it possible to predicting a token conditioned on a greater number of previous tokens. Second, n-grams do not account for similarity between words. For instance, in the sentences 'the cat sat on the mat' and 'the dog laid on the carpet', the similarities of ('cat','dog'), ('sat', 'laid'), and ('mat', 'carpet') can be learned through a neural net contributing to generalized predictions since

[10] Elman (1990) resolved the debate over XOR (or parity computation) by demonstrating the network capabilities in learning that pattern.

[11] Smolensky (1988) proposed an alternative approach to symbolic systems, which are often attributed to universal grammars, for modeling language. This approach is regarded as subsymbolic.

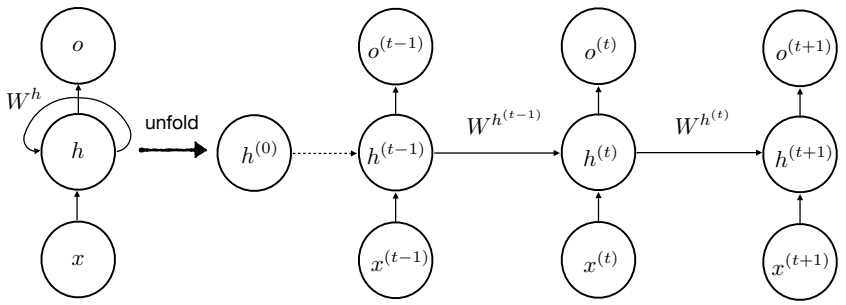


Figure 2.8: Sending forward the previous hidden state weights in RNNs.

semantically similar words form similar activation functions. N-grams may not draw on these similarities when constructing a lattice. Bengio et al. (2003) showed that NNLM obtained lower perplexity than smoothed tri-gram models on medium and large vocabularies of 1.2 million and 34 million tokens, respectively. Bengio et al. reported that the feed-forward network outperformed the RNN version. On the other hand, NNLM were found to fall short when predicting infrequent terms in the large vocabulary settings of GiGaWord (Napoles et al., 2012) and Penn Treebank (Marcus et al., 1994), as reported by both Chen et al. (2016) and Neubig et al. (2016).

Another advancement in NNLM was made by Sutskever et al. (2014), who developed a sequence-to-sequence (seq-2-seq) architecture that permits the varying-length sentence prediction that is desired in machine translation. This is because semantically parallel sentences in two languages may require a non-equal number of tokens to be realized. The seq-2-seq architecture is comprised of two LSTM units: one that encodes the input and another that generates an input-conditioned output (also referred as the encoder-decoder). Prior to this work, a single LSTM unit formed the network, making the output tokens correspond to the number of input tokens. Sutskever et al.'s (2014) work outperformed previous statistical machine-translation (MT) systems. Eq. 2.11 describes how the process of translating to a target language in encoder-decoder form generates a c vector. This vector represents a fixed vector of the input tokens (as a hidden state) that conditions the generation of y_i together with the previous tokens:

$$p(y_i|y_1, y_2, .., y_{i-1}, X) = \prod_{t=1}^{i} p(y_t|y_1, ..., y_{t-1}, c) \tag{2.11}$$

Another outcome of NNLM was the direct usage of internal layers for embedding space representation that was introduced by Mikolov et al. (2013a) (see Sections 2.4.1 and 2.4.2). Bahdanau et al. (2014) presented the attention mechanism in language models, wherein a learned set of weights is trained to 'attend' to a particular area in the token history that has been fed into the system. In Eq. 2.12, y_i is predicted based on the varying context vector c_i together with the previously predicted vector y_{i-1} and the previous hidden state of the decoder s_{i-1} to produce the current hidden state of the decoder:

$$p(y_i|y_1, y_2, .., y_{i-1}, X) = g(y_{i-1}, s_i, c_i) \tag{2.12}$$

where

$$s_i = f(s_{i-1}, y_{i-1}, c_i) \tag{2.13}$$

However, the attention vector c_i is learned as a function of the position of the output, decoding relevant features from the output by varying the weight of each of the input hidden states.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{2.14}$$

This model outperformed the previous simple RNN-based encoder-decoder on an MT task using BLEU scores (Papineni et al., 2002). Vaswani et al. (2017) who, as mentioned earlier, proposed restructuring RNNLM to allow for parallel processing, made a transformer work well in parallel due to several 'attention heads' and 'self attention layer' components. These helped the model to achieve higher BLEU scores than previous neural models on MT tasks.

Alec et al. (2018) presented a Transformer-based *unidirectional* language model called generative pre-training (GPT) that scored higher on nine out of eleven NLP tasks than earlier models. Devlin et al. (2018) then introduced the Transformer-based *Bidirectional* Encoder Representations (BERT), which outperformed GPT and previous models on eleven downstream tasks of natural language inference, including question-answering on the GLUE benchmark (Wang et al., 2018) and SQuAD (Rajpurkar et al., 2016a) (Stanford's Question-Answering Dataset). BERT's influence on the field has been widespread across NLP communities, leading to the release of various fine-tuned BERTs in the biomedical (Lee et al., 2020), medical (Rasmy et al., 2020), clinical (Alsentzer et al., 2019), and scientific (Beltagy et al., 2019) domains, among others.

Recently, BERT (or the field of Bertology) has drawn criticism; there has also been pushback against the current trend toward big models such as BERT. Bender et al. (2020) published an influential work that questioned the evaluation procedures and their contribution to natural language understanding. In the next section I explore less common neural architectures of NNLM that are based on generative, rather than discriminative, learning. The reason generative approaches are discussed is that my proposal is based on a generative model of language modeling. This subsection concludes with a summary comparison of the NNLM and count-based language model approaches.

Generative approaches

While generative approaches have not been at the forefront of neural language modeling, I cover two approaches that are related to the type of modeling proposed in this thesis. These generative approaches are an alternative to the current discriminative NNLM presented earlier. Due to the popularity of discriminatory NNLM, generative NNLM work has been devalued. It is important to note that the work on generative architectures' language models has focused on the category-based prediction of classes/words, and to the best of my knowledge, the continuous output prediction of generative approaches (or of embedding representations, which the next section covers) has not been researched yet.

The basic concept of a generative approach is modeling the joint probability distribution $p(x, y)$, while that of a discriminative approach is modeling the conditional probability distribution $p(y|x)$.

Generative adversarial networks (GAN) were first introduced by Goodfellow et al. (2014) as a min-max dynamic of two players, a generator and a discriminator with the following loss:

$$\min_G \max_D L(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))] \tag{2.15}$$

where the generator is aiming to imitate a real distribution $p(x)$ through learning a latent distribution $p(z)$ and the discriminator learns to not 'be fooled' by the generator's imitations. While both are expected to improve their performances over time, the GAN's theory suggests that optimal convergence is achieved by successfully training a generator such that the discriminator cannot distinguish the true from the real samples. Figure 2.9 illustrates the aforementioned process and

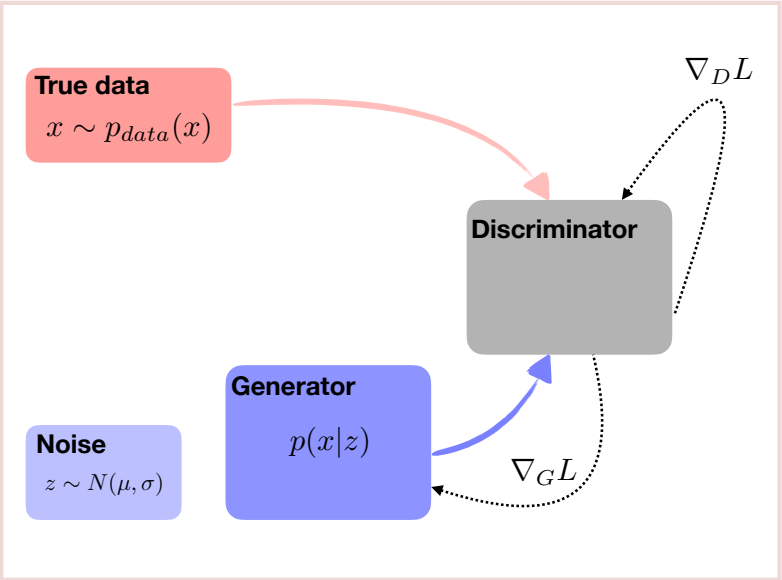


Figure 2.9: Illustration of GAN structure.

reflects the fact that the discriminator's decisions affect both itself and the generator. Zhang et al. (2016) proposed overcoming the discrete GAN outcome and rendering a non-attainable gradient (either correct or incorrect) in the system by replacing the predicted class with its corresponding embedding representation to create the continuous representation of a sentence. Yu et al. (2017) subsequently proposed SeqGAN to generate sequences of tokens, overcoming the gradient problem as well but by adopting a reinforcement learning approach. G is a stochastic policy generator and can be rewarded for choosing from a distribution of actions, and G is updated for its mean and variance (as it is assumed to be Gaussian), making for suitable gradient learning. Xu et al. (2018) proposed a diversity-promoting GAN for generating text. The proposed loss function incorporates rewards for novel and fluent words and penalties for repeated tokens. The researchers also proposed a cost function for the overall sentence novelty. Lamb et al. (2016) proposed providing the discriminator with the generator's intermediate hidden units rather than its discrete output (or a hot representation). This strategy makes the system differentiable and has been shown to achieve promising results in tasks' character-level language modeling as well as handwriting generation.

Variational autoencoders (VAE) were presented by Kingma et al. (2014) and Rezende et al. (2014). The goal of a variational autoencoder is to model the latent variables of x through two steps: first modeling $p(z|x)$ while z is a latent variable spanning x, and then reconstructing x from z through $p(x|z)$. Since $p(z|x) = \frac{p(x|z)p(z)}{p(x)}$, and $p(x)$'s computation is intractable, then $p(z|x)$ must be instead estimated. The way it is estimated is through an auxiliary function $q(z|x)$ that will learn to model $p(z|x)$. One goal, therefore, is to ensure that $q(z|x)$'s distribution is the closest possible to $p(z|x)$. To this end, in the process of learning, one objective is the following:

$$min(D_{KL}(q(z|x)||p(z|x))) \tag{2.16}$$

The second objective of a VAE is to model the data provided the latent variables, corresponding to learning the probability of $p(x|z)$, where the model learns how to reconstruct x. This basic architectural concept is shown in Figure 2.10: where our

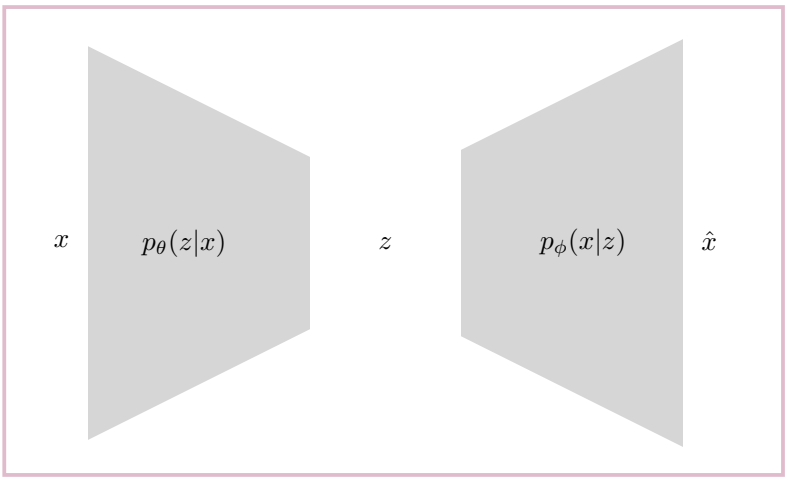


Figure 2.10: Illustration of VAE structure.

loss is

$$l_i(\theta, \phi) = -\mathbb{E}_{z \sim q_\theta(z|x_i)}[\log p_\phi(x_i|z)] + D_{KL}(q_\theta(z|x_i)||p(z)) \tag{2.17}$$

The first term is the reconstruction loss, which is computed as the expected negative log-likelihood loss for the i data point. The second term is a regularizer to match the approximated latent space to the true latent space and is often assumed to be of a Gaussian distribution (often by directly estimating μs and σs). As with other generative approaches in this field, research remains to be done, but the pace of research in language modeling in this context has slowed down. A VAE LM was proposed in Bowman et al. (2015b) where each of the components of the net was replaced with RNNs and a discrete prediction (employing a categorical LM). A similar version was developed for dialogue response in Serban et al. (2016), where several prompts were fed such that the latent units of z were conditioned to produce the prediction of the following prompt (similar to the 'context' element in an attention mechanism). Zhao et al. (2017b) proposed generating a diverse set of responses over the compared approaches for a dialogue response by modeling the *intent* and then conditioning the response directly on it. These LMs differ from the original in that a prediction rather than a reconstruction is made.

In Chapter 3 I propose a generative approach for language modeling based on a GAN, which is composed of non-categorical output (as opposed to the related work reviewed here) to predict the next word.

2.3.5 Neural models compared to count-based approaches

The neural models described in this section also bear limitations with regard to to count-based approaches. First, NNLMs are black boxes, while count-based approaches are not. Over the years, attempts have been made at probing these models via an attention mechanism (Choi et al., 2016; Xu et al., 2015) or other techniques (Alvarez-Melis et al., 2017; Lei et al., 2016), yet much remains unknown (Jain et al., 2019; Rogers et al., 2020). Second, computing a prediction in NNLM entails computing the probability of every word (or entry) in the model, which is an architectural outcome of the model.[12] The count-based models, on the other hand, do not require that a probability is assigned to every vocabulary word by default and have ways to compute on-demand probabilities when queried via smoothing (Katz, 1987). Finally, as mentioned, infrequent word prediction is found to be more optimal with the count-based approaches (Chen et al., 2016; Neubig et al., 2016).[13] A possible advantage is that a neural model may allow for relatively greater control over its size, which may not grow with more training data,[14] while a count-based approach may increase the lattice size if more unknown state transitions are introduced (this may be addressed through pruning if needed (Lesher et al., 1999), though this adds to the complexity). Another advantage is related to the basic accuracy and perplexity performance of NNLM, which has been proven to be higher (Mikolov et al., 2011b; Mikolov et al., 2013a; Mnih et al., 2007). In section 2.5, I describe the general limitations of NNLM that are part of the core argument of this thesis.

2.4 Word-Embedding Spaces

In this work, aside from incorporating neural language models, I also will be employing word embedding spaces. Word-embedding spaces (or representations) are incorporated explicitly into a language model to enhance learning, as this is a common way to augment different downstream tasks in general (Passos et al., 2014; Nguyen et al., 2014). In this section, I review static embeddings as well as contextual representations. First, however, it is important to ask what these representations are. Some representations are purely functional, converting a category of a word to a vector to represent it in a computational fashion instead of as text. Other representations are created to be more indicative of the semantics they may stand for.

Most of the representations I discuss here are aimed at identifying features that would represent terms such that similar terms would be 'closer'. Defining the proximity of embeddings is often done by measuring the cosine similarity of both representations (vectors) by calculating their inner products. Originally introduced for

[12] More on the depth of this limitation can be found in section 2.5.3.
[13] More on the depth of this limitation can be found in section 2.5.2.
[14] However, the fixed vocabulary size of the NNLM may be a limitation.

document retrieval purposes in information retrieval (IR), an early example is shown by Salton et al. (1975), where the similarity of a query to a document is searched. In the context of embedding spaces, we often ask how similar words or phrases are with outcomes ranging from least similar to most similar [-1,1], as shown in Eq. 2.18:[15]

$$cos(\theta) = \frac{a \cdot b}{\| a \| \| b \|} \tag{2.18}$$

The core premise of embedding spaces is based on the distributional hypothesis proposed by Harris (1954) suggesting that words that have similar contexts will have similar meanings. More precisely, the following methods are based on the principle of projecting words onto a space that forms semantic relations such that similar words are in closer proximity to one another. Similarity is defined by the distributional character of the contexts these words share.

2.4.1 Static embeddings

Static embeddings refer to embeddings that are fixed regarding an individual word. According the linguistic theory of Firth (1957), 'a word is characterized by the company it keeps'; in other words, by examining the contexts in which a word occurs, following the distributional semantics of Harris (1954), words that occur (or co-occur) in similar (or within the same) distributional contexts are likely to have similar roles, or semantic proximity. To illustrate these statements, the following example is provided:

(1) 'I enjoy drinking mocktails with you'

(2) 'I enjoy riding horses with you'

Given **(1)** (or **(2)**), 'I', 'enjoy', and 'you' tend to co-occur within the same context and can be considered semantically *related*, while given both **(1)** and **(2)**, 'mocktails' and 'horses' occur in similar contexts and can be considered semantically *similar*. Following the distributional semantic theory, dense embedding approaches capture both semantic relatedness and semantic similarity. Both aspects are described and investigated in Lavelli et al. (2004). The static approaches discussed in this section were found to be functionally be on-par (Levy et al., 2014b; Österlund et al., 2015), and their main difference lies in how these embeddings are formed and the computational costs involved when employing such embeddings.

Sparse embeddings

Originally the IR-based approach term-frequency inverse-document-frequency (tf-idf) was applied to context-word pairs. In this approach, words are assigned weights proportional to their frequency such that informative pairs receive higher weights, as shown in Huang et al. (2009). Another distributional approach involves assigning weights based on point-wise mutual information (PMI) (Church et al., 1990) measuring how likely it is that words occur together, as shown in Eq. 2.19:

[15] The angle can take $[0, \pi]$.

$$PMI(c, w) = \frac{\log(|c|, |w|)}{\log(|c|)\log(|w|)} \tag{2.19}$$

where $|\cdot|$ reflects the count (or frequency of occurrence) and c, w is the context and the word following it, respectively.

While the techniques described above are considered interpretable and easy to generate, they are hampered by sparse representation. This is due to a lack of data representing all possible pair examples as in tf-idf or PMI matrices. While different heuristics have been proposed over the years to find ways to project these approaches to denser representations, such as applying singular value decomposition (SVD) to tf-idf (Deerwester et al., 1990) or smoothing approaches to tf-idf, as shown in Hiemstra (2000), these representations remained particularly limited when increasing the context to more than a single word, making them sparser and longer. In recent years neural networks have become popular, leading to a new type of embeddings: dense embedding representations.

Dense embeddings

Dense embeddings are not conceptually different than the distributional approaches presented earlier; however, their appeal is in their lower dimensionality, which enables reduced computational complexity when employed in other downstream tasks. Moreover, these methods enable more complex representations by allowing longer contexts (more words, context of pre- and post word) to generate these embeddings due to their architecture.[16]

One way to form static dense embeddings' representations is by training a neural language model (section 2.3.4) specifically with an internal embedding layer that represents the learned features corresponding to the learned words. Technically, the model learns a matrix as a map where each row represents a distributed representation (vector) of a word in the vocabulary. The embedding layers are often a by-product in this process. We follow the review of Wang et al. (2020) describing various approaches for contextual embeddings.

Aside from training a language model architecture as mentioned, I describe the main lines of work of embedding that have been popular in the recent decade: CBOW, skipgrams, Glove, and FastText. These are methods for directly generating lexical embeddings (as opposed to general NNLM architectures where embeddings are a by product as mentioned). These embeddings are static in the sense that they do not change over time once learned. Mikolov et al. (2013a) introduced the word2vec algorithm, wherein the model learns word representations either by predicting a representation of a current word using continuous bag-of-words (CBOW) (see Eq. 2.20) or by predicting a context word based on a current word using skip-thought-vector (or skipgrams) (see Eq. 2.21):

$$p(w_t|c_t) = \frac{exp(C'(w_t)^T C(c_t))}{\sum_{i=1}^{|V|} exp(C'(w_i)^T C(c_t))} \tag{2.20}$$

[16]Practically, while it is possible to apply complex contexts to sparse embeddings, doing so would make the embeddings even longer and more computationally complex.

$$p(w_j|w_t) = \frac{exp(C'(w_j)^T C(w_t))}{\sum_{i=1}^{|V|} exp(C'(w_i)^T C(w_t))} \tag{2.21}$$

Both techniques are neural-based yet without a hidden unit, resulting in relatively less expensive training than NNLM. Figure 2.11 illustrates the processes of both CBOW and skipgrams, where in both cases, the final outcome after training is the internal matrix learned.According toMikolov et al. (2013a) skipgrams and CBOW were shown to outperform RNNLM and NNLM on syntactic and semantic word relations. Both techniques attained reduced complexity compared to NNLM.

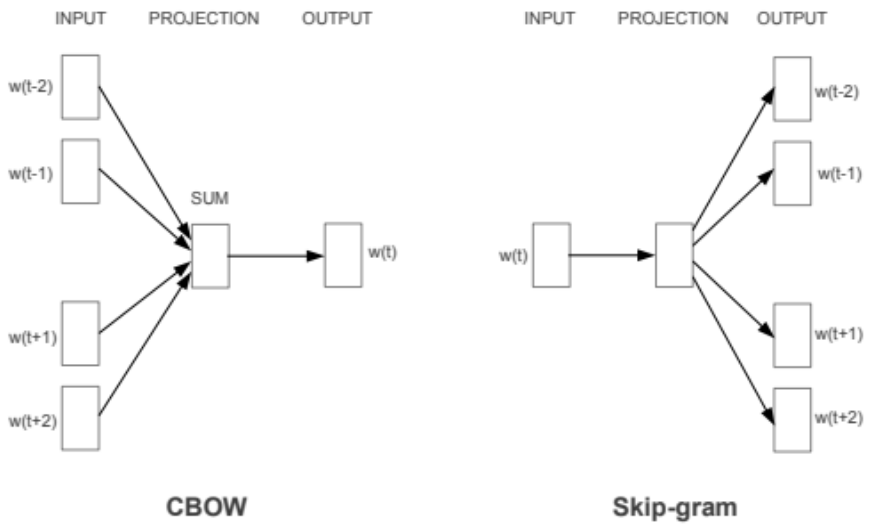


Figure 2.11: word2vec process by Mikolov et al. (2013a)

Pennington et al. (2014) presented another distributional representation for words by introducing GloVe. GloVe aims at capturing global information of the statistics of word-co-occurrence. GloVe's loss function is shown in Eq. 2.22:

$$L = \sum_{j,i=1}^{|V|} f(X_{ij})(C(w_i)^T C(w_j) + b_i + b_j - \log X_{ij})^2 \tag{2.22}$$

where X_{ij} denotes the number of times word w_i co-occurs with word w_j and $|V|$ is the vocabulary size. $f(x)$ is a weighing function to mitigate the imbalance of the rare and frequent words. Pennington et al. (2014) showed that GloVe vectors outperformed CBOW and skipgrams on word analogy tasks, specifically semantic and syntactic analogies, as well as on word similarity tasks, where a ranked list of word pairs is generated by computing a cosine score between them. GloVe showed superior performance over named entity recognition (NER). When compared directly to either CBOW or skipgrams, GloVe was found to have a steeper learning curve.

Bojanowski et al. (2017) proposed FastText to address the fact morphology was overlooked by the typical embedding algorithms and extended skipgram vectors to learn representations of n-grams, such that words are represented as the sum of the subword vectors composing them. Compared to CBOW and skipgrams, FastText was found to be more correlated with human judgements of similarity scores across

several different languages; it also performed more optimally on syntactic and semantic word analogy tasks. In addition, the authors showed reduced perplexity on language modeling tasks for various languages. Moreover, in a similarity task designed to identify closer (as judged by human subjects) rare or out-of-vocabulary words, to their similar in-vocabulary counterparts, FastText produced more optimal outcomes than skipgrams. Evaluating on different languages opened the door to more effective modeling for languages other than English, especially those with more involved morphologies (Conneau et al., 2017).

While the static dense embeddings attained high performance on the various tasks discussed above, polysemy remained a problem. Polysemy refers to the challenge of learning various representations for the same word. These representations correspond to the word's different senses. Since a word's sense may be contextually inferred, static, or fixed, representations that assume a single representation for all of a word's tokens cannot reflect senses in their final outcomes. While work (Chen et al., 2014; Neelakantan et al., 2014) has been done with static embeddings to tackle this issue, the breakthrough came when contextualized embeddings were introduced.

2.4.2 Contextualized embeddings

Contextualized embeddings are mainly an outcome of the unsupervised training of language models (Liu et al., 2020). They are considered contextualized because they produce different representations for the same word given different contexts. Similar to static embeddings produced by NNLM or RNNLM, contextualized embeddings are formed by an internal layer of a language model; unlike static embeddings, however, they are produced per token and not per word-type, which is key to representing various word senses' embeddings. In principle, contextual representations are learned by associating a token t_i with a representation that is a function of the entire input sequence (as shown in Eq. 2.23) where the other tokens are converted to a non-contextual representation prior to applying a form of aggregation with f.

$$Rep(t_i) = h_i = f(e_{t_0}, e_{t_1}, ..., e_{t_{i-1}}, e_{t_{i+1}}, .., e_{t_n}) \tag{2.23}$$

Peters et al. (2018a) developed embeddings from language models (ELMo), which is a bidirectional language model based on maximizing the log likelihood through the loss function shown in Eq. 2.24:

$$L = \sum_{t=1}^{N} (\log p(w_t|w_1, w_2, ..., w_{t-1}) + \log p(w_t|w_{t+1}, w_{t+2}, ..., w_N)) \tag{2.24}$$

ELMo employs a character-based convolutional neural net (CNN) layer that enables the processing of out-of-vocabulary (OOV) words and helps reduce the model's complexity. ELMo representation captured more optimal features than GloVe on SQuAD (Rajpurkar et al., 2016a), SNLI (Bowman et al., 2015c) (Stanford Natural Language Inference), and SRL (Pradhan et al., 2013) (the semantic role labeling dataset). Moreover, ELMo was shown to perform better on polysemy tasks than supervised approaches, word sense disambiguation, and the capturing of part-of-speech features.

Devlin et al. (2018) introduced BERT, incorporating both the left and the right side contexts. Similar to ELMo, this architecture can process OOV words effectively, though BERT employs a subword tokenization (instead of character-based tokenization) called word-pieces (Wu et al., 2016b). BERT is trained on two tasks: masked language modeling (MLM), which is aimed at predicting a word given both the left and right contexts, and next sentence prediction (NSP). Bert embeddings outperformed GloVE, ELMo, and GPT on the SWAG (Zellers et al., 2018) (Situations With Adversarial Generations) dataset and were on-par with the human scores on that task. BERT embedding remains an active area of research. For example, Alsentzer et al. (2019) presented publicly available, clinically oriented embeddings; Coenen et al. (2019) analyzed the geometry of BERT's embedding space and found that word senses are encoded at a very fine-grained level; and Hewitt et al. (2019) proposed a structural probe to find encoded syntax.

One assumption that was made when constructing BERT is that predicted tokens are independent of each other. For instance, modeling the probability of $p(t_2 = \text{quick}, t_4 = \text{fox}|t_1 = \text{the}, t_2 = \text{[MASK]}, t_3 = \text{brown}, t_4 = \text{[MASK]}, t_5 = \text{jumps})$ is factorized with BERT to another issue that $p(t_2 = \text{quick}|...), t_5 = \text{jumps}$ and $p(t_4 = \text{fox}|...)$, where t_2 and t_4 are conditionally independent. Another issue with BERT is the inability to use the learned '[MASK]' dependency during fine-tuning. This void was filled by Yang et al. (2019). During training, a token is predicted given permutations of factorizations, such that the task of prediction is conditionally dependent on previously predicted data. In other words, predicting w_4 can be done given partial predictions of the five-length input sequence (e.g., given w_1 or given w_2, w_5). For example:

$$L_{\textbf{BERT}} = \log p(\text{New}|\text{is a city}) + \log p(\text{York}|\text{is a city})$$

$$L_{\textbf{XLNet}} = \log p(\text{New}|\text{is a city}) + \log p(\text{York}|\text{New, is a city})$$

In the example above, BERT proposes a conditionally independent approach to predict the second token, while XLNet authors propose a way to capture the dependency between the pair given the first prediction. Huang et al. (2019) showed that XLNet outperformed BERT (and other baselines) when producing embedding representation of clinical text.

An important criticism of the methods I have described for embedding representations is the degree of semantic relations, and especially the degree of semantics that can be inferred from these spaces. In that regard, one may not *truly* know a word by the company it keeps; rather, one may know other words that are similar in some ways (similar contexts or frequencies (Gong et al., 2018)). Researchers are currently developing ways to ground meaning (Tamari et al., 2020; Bisk et al., 2020)[17] into relations that go beyond text.

2.4.3 Hot representation

While a type of sparse representation, hot representation is different then all the previously discussed methods in this section. Hot representation is the simplest version of a representing a word (or a class). Typically, the vector has a dimensionality

[17]The work mentioned is focused on grounding text in general, not grounding embedding representations in particular.

of the size of the vocabulary it represents. Therefore, it can be relatively costly to train a neural net (depending on its vocabulary size). In this technique, representing a word corresponds to activating a cell (Boolean of 'True') at an index that is associated with the word; see Figure 2.12. This encoding process is straightforward and does not require any learning to generate the encoded vectors.

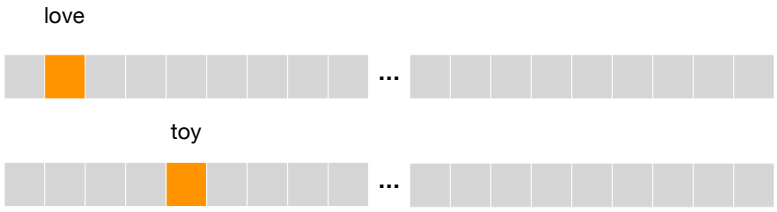


Figure 2.12: Illustrating 'hot' representations for the vectors of the words 'love' and 'toy'.

Hot representation is mainly used in learning a neural network into which these vectors are fed as an input. The actual feature learning of the different tokens is conducted internally in the network with as many as several additional embedding layers. Elman's (1990) research shows how these vectors were fed into early neuronal models (see (Elman, 1990), Table 5). Multi-hot encoding has also been incorporated where several categories are to be predicted/represented within the same vector in order to characterize several dimensions of the input (a comparative study on category based encoding approaches was presented by Potdar et al. (2017)).

2.5 Limitations of Neural-Categorical-Based Prediction

At this point in the chapter, I turn to consider the *limitations* of neuro-categorical-based models and elaborate on each in depth. The neuro-categorical based models include the various neural network models described in section 2.3.4 (generative and discriminative models), which predict a probability distribution (and an index) that corresponds to a category in their vocabulary. In the next chapter, these *limitations* are reexamined in relation to the proposal presented in this thesis.

2.5.1 Complexity limitations

Even before Transformers were introduced, according to Jozefowicz et al. (2016), 'the best (language) models are the largest we were able to fit into a GPU memory.' This statement suggests that good model performance is conditioned on access to heavy computational resources. This performance comes at a price: most current state-of-the-art models employ deep architectures that are computationally complex and require a significant number of parameters to be learned. One major reason that heavy computational resources is needed is that traditional approaches to language modeling (both neural and otherwise) model the task as *categorical* prediction. Neural language models following the traditional log-bilinear approach typically perform prediction by means of a *softmax* operation at the output layer that attempts to

estimate a discrete probability distribution over the symbol vocabulary (see, for example, Mikolov et al. (2010), Sutskever et al. (2011)).

$$p(y = c|X = x_k) = \frac{exp(f(w, x_k))}{\sum_{j \in |V|} exp(f(w, x_j))}, \forall c \in V \tag{2.25}$$

The softmax operation (shown in Eq. 2.25) is expensive both in terms of memory and computation and becomes more so the larger the size of the vocabulary. Increased runtimes may have to do with the estimated probability that is computed for each symbol, a softmax function that incurs a linear cost with regard to vocabulary size, and the increased memory requirement as the model is required to contain the trained entries that correspond to the size of the vocabulary. There are various approaches to addressing the computational concerns. An early method proposed reducing the size of the prediction space by eliminating infrequent word-types from the output space (Schwenk, 2007; Schwenk et al., 2014; Shah et al., 2015). A related approach collapses words into classes (Mikolov et al., 2011a) and predicts from there. These techniques have obvious limitations, though they may be appropriate in many application areas. Another more common family of approaches preserves the entire vocabulary while avoiding the large softmax parameter space and normalization step. For example, hierarchical softmax (Morin et al., 2005) organizes the vocabulary into a tree, thereby reducing the number of operations required to estimate a probability, albeit at the cost of increased training complexity and the possible loss of model generalizability due to the need to carefully construct the tree. Grave et al. (2017) subsequently proposed the adaptive softmax approach aimed at accommodating the hierarchical method to a GPU, reducing the training complexity possibility to a sublinear cost with regard to $|V|$. With a large vocabulary, however, even a sublinear scan over $|V|$ incurs a substantial cost. Another approach is noise-contrastive estimation (NCE) (Mnih et al., 2012; Gutmann et al., 2010; Zoph et al., 2016), which speeds up training by approximating the gradient calculation for the prediction layer.[18]

Another approach to addressing issues relating to vocabulary size in a language model is using subword units (first applied to neural language models by Mikolov et al. (2012) and used to improve NMT performance on rare words by Sennrich et al. (2016) and Wu et al. (2016a)). This approach reduces the vocabulary space of the model (thereby reducing the memory and computational requirements) and allows the model to function on sentences containing OOVs. This flexibility is indeed appealing, but it comes at a cost. One issue is that models employing subword units may fall short when used with highly inflected languages (Koehn et al., 2017). Such models have also been shown to under-perform on a variety of NLP benchmarks compared to word-level models (Li et al., 2019; Kumar et al., 2019). Furthermore, not every language and writing system can be conveniently decomposed to subword representations. For example, such decomposition is impossible when the text is written using Chinese characters or in an iconographic language (Dudy et al., 2018). A more practical issue is that in a text-entry context, it is preferable to offer

[18]Note that while NCE speeds up model training, it does not necessarily provide a speedup for all tasks that might require a language model. For example, using an NCE-trained model for explicit symbol prediction (as in a machine translation or predictive typing task) can still necessitate a linear scan over the entire vocabulary during inference.

users whole-word suggestions; repetitively approving incremental word segments is tedious for users, especially when there is a high selection cost (as in the case of an AAC application), or alternatively making the model generate possible words is longer/more expensive (although it will be addressed in Chapter 6.

2.5.2 Decoding limitations

Various NLP tasks are based on learning linguistic data that follows Zipf's law (Zipf, 1935) suggesting that the distribution of the data (and training examples in particular) is unbalanced, exhibiting high and low frequencies of different examples. Statistically based models, and in particular the softmax-based models, tend to maximize the overall accuracy or minimize the overall entropy (Dal Pozzolo et al., 2015), leading to the classification of observations with a relatively low-diversity set of predictions from the head of the distribution (most frequent probable examples). This produces dull and repetitive text that is not aligned with human-generated text Holtzman et al. (2020). This phenomenon was observed by Li et al. (2016a), who reported that '[t]hese models tend to generate safe, commonplace responses (e.g., "I don't know")', and Serban et al. (2015), who recounted that 'the majority of the predictions are generic, such as "I don't know" or "I'm sorry".'.[19] In addition, both Chen et al. (2016) and Neubig et al. (2016) found that n-gram language models produced lower perplexity on the lower-frequency bands (infrequent words) compared to several neural net baselines. Beyond statistically learned models' general sensitivity to frequency, softmax-based models' loss function plays a role in producing low-diversity predictions. The standard objective function these models learn is based on increasing the log-likelihood conducted through the cross entropy loss function (Eq. 3.3), which often promotes a single correct target through an entropy minimization objective. This leads to a relatively more skewed distribution (Ott et al., 2018), limiting the hypothesis search space by assigning high probability mass to very few items for a given prediction. On its own, this is a desired behavior. However, during learning the model infers which set of classes has the smallest loss, and the model thus ends up producing low-diversity predictions, as having the most frequent terms predicted is guaranteed to yield low loss simply because they can be regarded as 'safe bets'. Overcoming this problem often necessitates proposing alternative objective functions, new decoding strategies such as beam search, and various sampling approaches from the estimated prediction probability distribution of a model. Sampling is based on randomly choosing a token, often by restricting to a subset of the category set $v \in V$:

$$q(x_t|x_{<t}, p_\theta) = \begin{cases} p_\theta(x_t|x_{<t}) & x_t \in v \\ 0 & otherwise, \end{cases} \tag{2.26}$$

Sampling approaches often are restricted to either top-k (Fan et al., 2018; Ott et al., 2018), where v is directly mapped to k first entries, or an accumulated probability cut-off (Holtzman et al., 2020), where the entries are defined by the set of most likely

[19]While each of these examples describes a specific task of training a response to a prompt as part of a dialogue, these models use the same generic architecture. This architecture is based on a discrete category set with a softmax layer.

tokens that are within the probability mass cut-off. Sampling approaches can be conducted by increasing the temperature parameter τ (which was first introduced in cognitive science by Ackley et al. (1985)) in order to 'flatten' the distribution, as seen in Eq. 2.27:

$$p(y = c|X = x_k) = \frac{exp(f(w, x_k)/\tau)}{\sum_{j \in |V|} exp(f(w, x_j)/\tau)}, \forall c \in V \tag{2.27}$$

An aspect to consider is finding the appropriate k, or determining the probability mass to set as the threshold. This may vary from one task to another. Another reason for concern is that lower-frequency classes are often left out during this process as they are either cut out or assigned a lower probability, making them less likely to be chosen.

$$q(x_t|x_{<t}) = \underset{sequence}{\text{argmax}} \frac{1}{t} \sum_t \underset{token}{\text{argmax}} \sum_{x_t \in B} \log P(x_t|x_{<t}) \tag{2.28}$$

A more involved technique for decoding is based on beam search. Beam search is based on generating likely sequences for a given time step t. Eq. 2.28 describes how the most likely sequences are chosen based on the joint probability of the sequence predictions, limited to a beam width B. This allows for the most likely tokens to form a valid completion of a given sequence history. The normalization factor $\frac{1}{t}$ attempts to accommodate varying-length sentences. Vijayakumar et al. (2016) proposed a diverse beam-search algorithm where a diversification score is employed to penalize choices similar to those previously existing. Li et al. (2016b) proposed diversifying prediction by having the search algorithm discourage sequences from sharing common ancestors, resulting in a diversified list of candidates. Shao et al. (2016) proposed a stochastic beam search that involves sampling candidates to continue the given sequence while accounting for different ways which may not be the most likely class to increase diversity. However, in that paper and Li et al. (2016b), it was noted that beam search collapses to a similar sequence for longer sequences, indicating one failure mode.[20] In general, one concern regarding beam search is that despite the aforementioned attempts, beam search tends to lead to bland, incoherent, or repetitive text. Another concern revolves around sequence scoring, where the most likely score is based on the highest products of the sequence probabilities. This may encourage the generation of shorter over longer sequences and has been shown to collapse when longer sequences are required. These concerns together with the additional computations may have led to the lack of beam search's widespread adoption for decoding sequences.

Objective functions that are non-beam-search related have also been proposed to improve diversity in decoding. The reflective loss was proposed by Dieng et al. (2018) as an attempt to address class imbalance by demoting non-target likelihoods and promoting the likely target. In a follow-up work, this evolved to the unlikelihood loss. The unlikelihood loss Welleck et al. (2020), shown in Eq. 2.29, was recently proposed. It is able to produce increased token variability (at the expense of accuracy), where the maximum likelihood is applied with a penalty that demotes other candidates drawn from the immediate word contexts.

[20] Li et al. (2016b) also shared that some images in the image-captioning task required more diversity and others required less, indicating another concern regarding the proposed algorithm.

$$UL_t = \sum_{w_t \in W_t} -\alpha \sum_{c \in C^t} \log(1 - p_\theta(c|w_t)) - \log p_\theta(w_t|w_{<t}) \tag{2.29}$$

Another way to avoid low diversity through an objective function can be learned from the field of reinforcement learning (RL) where increasing, rather than minimizing the entropy (Williams et al., 1991; Todorov, 2007; Mnih et al., 2016) is commonly used. This approach is called maximum entropy (ME), and it is often applied to a discrete action space that can be defined by a softmax outcome, as in Yue et al. (2020). ME often contributes to expanding the space of the hypothesis search, producing less skewed distributions and allowing for exploration over exploitation in the policy learned by an RL agent. The ME component of those models is often added to the objective function as a regularizer.

Other decoding concerns were exposed by Yang et al. (2018b) and Kanai et al. (2018), who pointed to a softmax bottleneck problem wherein the final matrix that produces the softmax probabilities is of a low rank. This reduces the number of basis vectors the model is spanned by and consequently limits the expressiveness of the model.[21] On the other hand, these decoding issues may contribute to the high accuracy of these models. While narrowing down the search space, the model learns to assign higher likelihoods to only the most likely and frequent terms as such terms, on average, tend to introduce smaller penalties for the model. In other words, softmax-based models with the common MLE objective are trained to have high precision (expressed by correct predictions) but low recall (narrow pool of correct predictions).

Sub-word/partial sentence decoding (Sennrich et al., 2016; Wu et al., 2016b) has been widely used due to complexity reduction and the promise of rare word prediction. While the sub-word approach can be effective in alphabetic languages[22], reducing a model's prediction space to a very limited set of classes is challenging for languages that contain thousands of base-level units ("characters") such as Mandarin and Japanese Kanji script. One particular category of language where sub-word representations are not an option is that of the symbol-based communication systems used in AAC applications, such as the Symbolstix set of icons (Clark, 1997). This modality provides a large ($n \approx$35k) human-curated vocabulary of symbolic icons representing words, phrases, and concepts, and it is used by a variety of commercially available communication devices and platforms. Furthermore, in user-centered settings, the prediction of sub-units may not be a desired interface because the prediction of sub-units requires a more demanding engagement.

Overall, through various decoding approaches, across tasks of response generation, prompt completion, image captioning, and similar generative tasks, the softmax-based approach is lacking diversity of predictions. Instead it yields generic, repetitive outcomes that ultimately hurt the quality of the generated text. Sub-units may provide higher-quality predictions, though sub-units may not be optimal in every setting.

[21] Expressivness does not correspond to lexical expressiveness but rather to the mathematical conditions of a matrix.

[22] Ling et al. (2015) reported that their approach was particularly effective in morphologically rich languages such as Turkish.

2.5.3 Architectural limitations

While the model is realized by a software program, it is necessary to distinguish between the model's skeleton and operations. The skeleton, which is the focus of this section, is based on layers that occupy a physical location (as reflected by the architecture), whereas the operations are conducted over the skeleton's layers, as described in section 2.5.2. Another concern with the described categorical approach is that it is architecturally limited to a finite set of classes to be learned. This stems from the fact that the final layer *must* contain all classes and assign each its relative probability (see Eq. 2.25). The summation is applied across all vocabulary items, producing a normalization constant that allows for relative comparison; however, even in the absence of a normalization constant, a model's choice of the predicted term may require sorting across all entries at the cost of $O(|V|\log(|V|))$ (for sampling or beam search) or searching for the most likely token $O(|V|)$, which requires passing through all classes. This description is not an architectural concern, yet it is part of how the model represents its classes. The model's representation also has to do with the inefficiency of generating a probability distribution, resulting in redundant demands of allocating resources to infer the model's belief for every class. The main reason a categorical model computes a probability for each of its classes is its architecture. Count-based approaches that are also based on the Markovian theory (Damerau, 1971), for instance, do not provide/compute an overall distribution for $p(y = c|X = x_t), \forall c \in V$. Rather, they compute for all classes observed given a particular context. In cases of data sparsity, smoothing approaches, and back-off in particular (known as Katz smoothing (Katz, 1987)) can compute probabilities on demand for unseen events. The notion of class representation goes beyond this, however. What if a new class or several new classes need to be introduced? For example, how is it possible to move towards continual learning (Parisi et al., 2019) with class-based models, such that a model can learn new classes over time while maintaining its past knowledge? Currently, the prediction of untrained classes (unseen-during-training classes) cannot be performed in a straightforward manner under categorical models. In order to perform this prediction, one has to essentially change the model architecturally and then re-train the model (more details in Chapter 4). This process includes stripping off the original layer to form new one that will allow new classes to be reflected as vocabulary entries of the model. Figure 2.13 illustrates

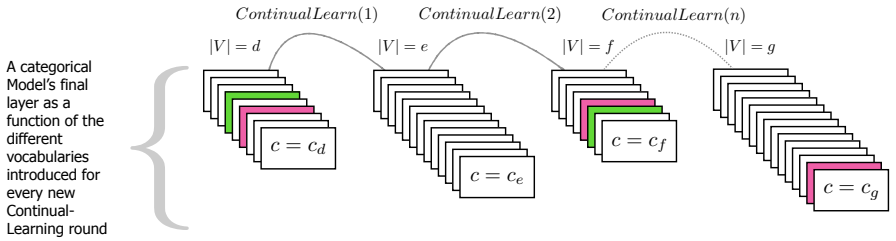


Figure 2.13: Continual learning in categorical models requires stripping off the final layer to introduce new vocabulary items for every new training (there is no access to past data). $|V|$ and c are vocabulary sizes and vocabulary classes, respectively.

the process of continual learning conducted over a neural-categorical model where access to old data is impossible. After the first training, the model's vocabulary is made of d classes; in the next training, as there is no access to the previously learned

data, the model learns a new set of entries at the size of e, yet it cannot represent the previously learned 'green' and 'pink' classes. In the third iteration, the datasets for continual learning are relatively smaller than in the previous rounds, making the model less expressive in terms of overall class. This is expected, given that the model is limited to the current data. If the model had had access to previous data, the number of classes would have grown over time, incurring an increasing computational burden, and that would be the limiting factor. One reservation may be that even in the presence of classes in the architecture, a model may not be able to predict them well for computational, rather than architectural, reasons. These reasons may include catastrophic forgetting (McCloskey et al., 1989), or as described earlier, the dull and repetitive nature of the categorical models. However, these problems cannot be considered in the context of continual/lifelong Learning thoroughly since these classes may not be represented over time. There are different ways to mitigate the problem. First, categorical models could be trained from scratch for every new task, though that may call into question the sustainability of such models, incurring high costs vs. adapting a pre-trained models as shown by Howard et al. (2018). A single multi-task general purpose model containing many possible outcomes could also be trained, although this poses complexity concerns as the categorical model's complexity is also a function of its vocabulary. Finally, other architectures for training lifelong agents (as has been argued by Herbelot et al. (2017)) could be found. These architectures could progressively acquire knowledge over time and would be able to represent past data as well.

2.5.4 Evaluation limitations

This section considers a concern that is not a direct limitation of the categorical model per se, but rather an indicator of the need to establish higher model standards and add new ones (through additional metrics) to mitigate the problems discussed in Sections 2.5.1 and 2.5.2.

First, the complexity and efficiency concerns discussed earlier are not limited to the particular model dissected in 2.5.1. Computing demands increased $300,000x$ between $2012-2018$, as reported in Amodei et al. (2018), and this increase is assumed to be partly the outcome of increased usage of the neuronal models. To illustrate the increase, Strubell et al. (2019) showed that the training of a big transformer, together with a neural architecture search emit four times more CO_2 than a car, including fuel, over its lifetime. Schwartz et al. (2019) showed that in recent years, AI papers have, as expected, tended to target accuracy more often than efficiency. However, the financial implications of this trend gradually may hinder academics, students, researchers, and especially for those from developing economies from actively participating in the field. In addition, in a user-centric setting where real-time response (Martin, 1965) is crucial for user engagement, short runtimes determine the usefulness of a model.

The second concern, as mentioned in the previous paragraph, is that accuracy has become the main metric for evaluating a model's performance.[23] Traditionally, perplexity has also been measured to show the relative gains from one model to another. Both metrics may reveal important dimensions, yet they may be insufficient for understanding what the model has learned. Accuracy, which is a measurement of

[23]This argument applies to Bleu assessment as well.

how well a model predicts an expected target, does not consider that the performance of the more frequent terms is different than that of rare terms. Given that linguistic data follows Zipf's law (Zipf, 1935), where there is a small group of high-frequency terms and a large group of low-frequency terms, as well as the sensitivity of these models to frequency as shown in section 2.5.2, accuracy may not reveal how well a model predicts lower-frequency areas. This leads to the failure modes of tedious text described earlier. Perplexity/entropy computations (shown in Eq. 2.30) may also be dominated by the probabilities of the most frequent examples.

$$ppx = -\frac{1}{|X|}\sum_{x \in X} q(x)\log(p(x)) \tag{2.30}$$

A metric that can evaluate the explicit performance of the model as a function of its frequency could reveal stronger and weaker areas that translate to more and less reliable outputs, respectively. Identifying these areas may be the first step towards improving these model's performance. Here are several areas where low-frequency training examples ranging from sub-words to sentence generation may be important for different tasks: the translation of rare words (Sennrich et al., 2016; Wu et al., 2016b), and the prediction of rare words in technical writing or the biomedical domain across various textual tasks (Bahdanau et al., 2017). See et al. (2019) found that human judgements of conversational aspects can be improved when chatbots combine relatively rare words in a response. A textual domain was found to be characterized mainly by its lexical features (Plank et al., 2010) (domain sensitivity); this may suggest that if we assume that too many of the most frequent word-types at the head of the distribution are stop-words[24], the shoulder and the tail words of the distribution are where the domain jargon is found and therefore are crucial for domain adaptation.

As this discussion suggests, there are a number of different tasks for which infrequent words are desirable. Depending on the task, infrequent words may be evaluated at different granularities: for instance, whether they are introduced in a sentence or what particular units are employed. This argument goes beyond words as a unit of test, but evaluating a model on its less frequent examples in general could be helpful for improving text-based tasks like those mentioned above. Therefore, given that accuracy metrics are important as well, considering a holistic evaluation that includes both prediction accuracy and prediction expressiveness/diversity may not only provide overall alignment with tested targets but also reflect how well various classes, or training examples, are predicted in a given task.

[24]and are more functional in nature, hence the high frequency

Chapter 3

Towards Continuous-Output prediction of Language Models

> "Change does not roll in on the wheels of inevitability, but comes through continuous struggle"
>
> Martin L. King

3.1 Introduction

In this chapter I propose a GAN-based continuous output prediction model. The continuous models predict a continuous representation indicating a location in a (fixed) embedding space, which is later on mapped to a particular word-type. In other words, the predicted vector retrieves a word vector from the given embedding space. This approach relaxes the constraint forcing a model to predict among the finite number of vocabulary items that were introduced during training[1] by using the model for feature extraction purposes and then decoding the predicted vector through a retrieval process. I show how the proposed approach to language modeling not only predicts more diverse terms, but is also computationally efficient compared to current categorical approaches.

3.1.1 Motivation for using a continuous approach

As discussed in Sections 2.5.1 and 2.5.3, the *categorical* nature of prediction is shown to introduce costs derived from architectural constraints during the decoding process. During the decoding process, every category's likelihood is computed and compared using a sorting algorithm; otherwise, a basic comparison, which can be costly, is made across all vocabulary items. As discussed in section 2.5.2, categorical prediction also struggles with repetition and the prediction of high-frequency terms (producing generic text). Together with its objective function, categorical prediction

[1]The retrieval part is bound by the number of word embeddings found in a given embedding space or the external data used for decoding.

promotes decreased entropy, making for tedious text. However, there may be other mechanisms by which a prediction model operates.

I draw *inspiration* from work done on the brain. Studies have shown that the human brain may have a mechanism for the retrieval of concepts. Using neuro-imaging, Huth et al. (2012) and Huth et al. (2016) revealed that categories (both objects and actions) are represented as locations in a continuous semantic space; in other words, concepts are organized in a distributed fashion across more than a single region. Huth et al. (2012) claimed that it is unlikely that a category is represented in a single location. Instead, a single concept is found to activate several different regions of the brain during inference. These regions (and their activation intensity) are predictable and correspond to dimensions by which the cortex characterizes the concept; moreover, they are shared across subjects. Huth et al. (2012) hypothesized that this distributed representation evolved due to efficiency considerations of storage, noting that the brain 'represents diversity of categories in a compact space'. A similar neuro-imaging study by Martin (2016) corroborated these findings, showing that concepts in the human brain are encoded in a distributional fashion across various regions representing specific object properties. These properties are shaped in part by our action, perception, and emotion systems. Linking back to the search for an alternative prediction mechanism for neural models, these studies argue that continuous representation is likely to be the mechanism undergirding concept representation in the human brain; language models can also represent their concepts continuously. In artificial neural models, continuous/distributed representations of concepts have been attributed to the connectionist theory in the form of neural network layers. In NLP in particular, layers and other techniques have been devised to generate embedding space representations (for more on this, see section 2.3.4, and 2.4). Having these artificial representations (e.g., for words) suggests that there are ways to represent concepts continuously, but perhaps the prediction of continuous representations in language models has not been discussed.

Dell et al. (1997) conducted behavioral studies in cognitive psychology about speech production. They showed that unsuccessful attempts by individuals with aphasia to retrieve a certain concept in a word retrieval task generally *resemble* the target relatively well in sound or in meaning; the attempts and the target are not unrelated. It can be inferred that the erroneous words resulting from unsuccessful attempts have closer proximity to the target word during the retrieval process, accounting for the confusion. Linking back to one aspect of an alternative language model, one could hypothesize that instead of a categorical output prediction model being predicted, a distributed representation of a concept is predicted. That particular prediction, even if incorrect, may say more about the semantics of the prediction than choosing a likely category from a list would, since embedding spaces are organized in part by semantic relations.

A work in cognitive linguistics by Croft (2012) considered verbs (actions), and as a result events, that are geometrically represented across several regions in the brain to be the most plausible explanation for how events and actions are represented in the brain. This theory is backed up by Meltzer-Asscher et al. (2013), who examined brain activation using neuro-imaging. Considering different types of verbs, the researchers showed that while an individual is reading words and pseudo-words, each verb is associated with more than a single region. Gärdenfors' (2014) research, in line with Croft's (2012) work, focused on describing how concepts in the hu-

man brain are organized semantically in a geometric structure, thereby providing a cognitive science perspective. In their book, Gärdenfors hypothesized that representing categories over several dimensions is likely to be motivated by the easier learning of new concepts. In their earlier work, Gärdenfors et al. (2001) suggested that often when an individual is introduced to a new concept, they learn during the inquiry process what this concept is similar to, which helps with mapping the concept 'closer' to concepts that have similar attributes (dimensions). This theory also offers an explanation for how individuals retrieve open-ended queries. For example, if a child says, 'I want to play with the brown thing', one might consider possible brown objects with which they think the child is familiar and which of those objects most look like toys (provided the person has no prior knowledge to aid the search). In this way, the person searches their concept space. According to Gärdenfors et al., this process involves activating the objects that share the particular attribute of being brown, intersected with the concept region that corresponds to toys. The retrieval of concepts thus may be more targeted than searching across the complete set of concepts (as with a categorical machine-learning model), and it may be driven directly by the activations/dimensionalities characterizing a concept.

Linking to the prediction of continuous representation, one might hypothesize that a model has learned to predict concepts from an existing embedding space, and a new (unseen) word is added to that space that the model has not been trained on. This new word can be retrieved if it is in a region in the space the model learned to predict. For instance, if the model learned the location of the word `increase` but another vector appears during the decoding process associated with the word `enhance`, the latter word might be retrieved 'for free' simply due to its semantic resemblance (and therefore representation) to a known vector of `increase` the model had learned to predict. This is an out-of-vocabulary (OOV) prediction. This example may be one that an artificial continuous model can benefit from as well, which cannot exist in neuro-categorical-based models (see section 4.5.5 for more).

Elman (1990) argued that there is no limit to the number of concepts that can be represented with a finite set of units, referring to the hidden units on a neural network model where concepts are represented in a distributed fashion ('concepts are expressed as activation patterns over a fixed number of nodes'). I would like to extend this idea to the model's final layer, or in other words, train a continuous output prediction network to gain the benefit discussed in Elman's (1990) work[2]

The aforementioned work provides evidence for a system (in this case, the human brain) that is likely working in a continuous fashion. Drawing on this work, I described some analogies to artificial continuous prediction and the ways in which an artificial continuous language model could be more fruitful than a categorical alternative. Some deficiencies of the categorical approach, such as repetition and lack of diversity, are tackled in the following paragraph. Variations on continuous output prediction approaches have recently emerged from the field of natural language processing, where language models are *augmented* with additional resources

[2]The claim regarding an infinite number of concept representations is technical and can stand on its own; however, my work is driven by a motivation similar to that of Elman's (1990), who argued for the transition from symbolic models to distributed representation models. I argue for the transition from symbolic output representation in the category neural models to activation pattern representation (that in principle can allow for an infinite number of word-types to be represented and predicted).

that enhance their predictions. First, Khandelwal et al. (2019) proposed a hybrid model that combines a softmax-based approach with a retrieval-based approach. The final embedding in the transformer is matched against an embedding dictionary that decodes the representation to a word entry (in section 5.2, we experiment with a similar idea). Decoupling the retrieval process from the model prediction phase may help address the changing data and allow for vocabulary personalization. Guu et al. (2020) presented a language model in which a neural knowledge retrieval returns potentially similar documents to the query instead of employing a self-contained transformer. These documents are then encoded together with the query to generate an answer. Guu et al. noted that the retrieval process can provide open-domain knowledge that is *rarely* observed in the text. The most similar work to that presented in this chapter comes from Kumar et al. (2019), who proposed a continuous output prediction for a machine translation task. They showed that the performance of the continuous encoder-decoder model was on par with that of an equivalent categorical one; they also reported increased usage/prediction of rare words for the target translations and a reduced number of model parameters and runtimes. Lewis et al. (2020) also demonstrated a retrieval-based LM addressing a number of NLP tasks. Weston et al. (2018) produced dialogue responses through retrieval and found them to be less generic, longer in sequence, and inclusive of more rare words. The research shown here re-frames (and reforms) the process of prediction in language models to a retrieval approach that is similar to methods in information retrieval. This is similar to situations in which new data arrives, as in the case of news outlets where a search engine is required to retrieve both updated and personalized (less generic) outcomes, as shown by Gabriel De Souza et al. (2019). Sheu et al. (2020) extended Gabriel De Souza et al.'s (2019) work and refined the article representation to a graph embedding representation.

Continuous output prediction can be useful when the vocabulary is represented in a continuous fashion, as in Dudy et al. (2018), where we developed a predictive typing model for an icon-based vocabulary that leverages the properties of continuous-space word embeddings to represent icons' compositional nature. Vocabulary and domain adaptation are essential in the work of Dudy et al., where we aimed to personalize the language model for icon-based language models in AAC. Continuous representation has opened the door for new icons to be regularly introduced and composed in complex ways (personalization is investigated in this chapter, and adaptation is investigated in Chapter 4).

Given the examples revolving around language models and recommendation systems from related fields, it seems that splitting the inference process in language models into prediction and retrieval may contribute to reducing the models' complexity, enabling searches across external large corpora and the retrieval of more novel examples. Moreover, the separate decoding step may enable more optimal usage related to control over the type of data decoded by allowing the model's developer to choose a corpus that does not recover undesired data, for instance. This independence can be helpful not only when the trained data is unavailable but also when changing/updating the data from which to decode. A continuous representation can theoretically represent an endless number of items and allow for additional items to be added. It may also be advantageous in locating/learning positions of new/unseen concepts if similar concepts have been seen, as the new and already-familiar concepts would share similar activations (similar vectors have

similar representations). These are the reasons I propose a continuous approach in this research.

Following the literature review, I compare the performance of the continuous approach to those of several categorical models on a word prediction task, considering both prediction accuracy and prediction diversity/expressiveness. I also explore the diversity of a model over various word frequencies and produce an estimate of the computational costs, accounting for multiple users given the settings that motivated this work.

In the following section, I contextualize the work within other related subfields. Then I present the methods (including the datasets, models, embeddings, decoding step, metrics, and baselines) and reveal the results, introducing additional models to examine in my investigation and that allow me to focus on various comparisons separately.

3.2 Related Work

3.2.1 Predictive language models

While there exists a great deal of work regarding categorical prediction language models (Devlin et al., 2018; Peters et al., 2018a; Liu et al., 2019; Alec et al., 2018; Alec et al., 2019) (which covered thoroughly in Chapter 2), continuous prediction models are less well-studied. The closest and most relevant recent work is that of Kumar et al. (2019), who also explored the direct prediction of continuous word embeddings, albeit in the context of neural machine translation. They proposed a new loss function (Von Mises-Fisher (VMF) loss) for that purpose and demonstrated their model's feasibility over a large-vocabulary task. The model trained was of an encoder-decoder architecture. The VMF model of continuous prediction was found to be on par with an equivalent neural-categorical MLE model across three language translation tasks. They showed that this approach was faster and required fewer computational resources than the categorical alternatives. Kumar et al.'s (2019) approach also demonstrated good performance compared to categorical baselines when translating low-frequency words. Subsequent work by Li et al. (2019) focused on improving training times for ELMo models by using continuous output layers and stressed the resulting reduction in computational costs while maintaining on-par performance on standard evaluation tasks. In particular, they reported a fourfold speed increase and the elimination of 80% of the trainable parameters for continuous output models compared to their categorical counterparts. My work differs from Kumar et al.'s (2019) research on continuous space language modeling in that I focus specifically on a word prediction task rather than a machine translation (MT) task. Additionally, my primary focus is on accurately modeling diverse vocabulary entries (shown in this chapter) and the adaptation performance of continuous models (shown in Chapter 4), as well as investigating different embedding spaces, decoding techniques, and their impacts (shown in Chapter 5). Even more importantly, I compare an architecture inspired by their paper[3], that of a simple LSTM (Hochreiter et al., 1997) suitable for language modeling tasks, to the novel GAN architecture proposed in this chapter.

[3] Kumar et al. employed a decoder-encoder suitable for MT tasks

3.2.2 Adversarial language model training

Generative adversarial networks' (GANs) basic dynamic is explained in 2.3.4. GANs are infrequently used in language modeling, and existing work has emphasized language generation (Subramanian et al., 2017; Press et al., 2017). In part, this is due to a fundamental mismatch between the discrete, sequential, and variable nature of traditional language modeling and the more static, real-valued, continuous space in which GANs work. Yu et al. (2017) proposed a method for integrating an RNN into the generator component of a GAN and solved certain issues relating to gradient updates. This led a successful MT architecture (Yang et al., 2018a).

My model (see section 3.3.2) follows the conditional GAN architecture described by Mirza et al. (2014), where the discriminator is provided with both the input and output (or two separate input channels) $w_{<t}$ and either $\hat{w}_t$ or w_t (depending on whether the embedding is of a fake or of real word) instead of concatenating the entire sequence and feeding the discriminator with that. In this work, I applied Mirza et al.'s (2014) architecture to continuous space word embeddings that are fed to the discriminator. To the best of my knowledge, this application has yet to be proposed in the context of language modeling.

3.2.3 Rare words

Rare words are a persistent source of difficulty in neural approaches to NLP. The problem of generating rare words has to do both with the Zipfian nature (Zipf, 1935) of text, in which rare words are not found frequently, and the fact that neural models, similar to many other statistical approaches, are susceptible to frequency, as shown in Chapter 6. The more examples a given token has, the more likely it is the model will assign it a high probability. Fewer examples of a word prevents a model from learning a good internal representation of it, and as a result, the model will not predict it as frequently. Another reason revolves around the MLE objective. This reason is discussed in section 2.5.2, specifically in relation to entropy minimization versus maximization. In the context of NMT, Sennrich et al. (2016) demonstrated that rare words are more optimally predicted by using subword units, and this has since become a common practice. On the other hand, a different work by Czarnowska et al. (2019) showed that subword units can cause problems in morphologically rich languages (known to have many infrequent word-types (Gerz et al., 2018)) and that systems struggle with achieving both semantic and morphological correctness. Following the ambiguous research conclusions, the practice of subword-units is called into question in this chapter to learn about the prediction of infrequent words. This may be particularly problematic in domains that exhibit high type-to-token ratios (Gerz et al., 2018), where more types are introduced in a given text, contributing to a longer-tail (of the Zipfian distribution). Among these domains are the domains of medical/biomedical/scientific literature, which contain an abundance of terminology and jargon, and the set of morphologically rich languages, including Arabic, Hebrew, Turkish, and Czech, where inflections are many as well. In this work I explore the degree of prediction diversity, and in particular, the long-tail word-type prediction using both news and biomedical domains.

Embedding spaces are another area of focus for rare words. Embedding spaces that to some degree organize terms by semantic relations are not as reliable when positioning infrequent words (contrary to frequent terms) due to the low number

of location updates these words are given during the process of generating an embedding space, as was shown in Gong et al. (2018). Pinter et al. (2017) devised an approach for adding high-quality representation of rare terms to an embedding space by generating new items based on learning a word's embedding from its characters. This contributes to the relocation of these words to a more semantically related area. Schick et al. (2019) extended this research by integrating a word's contexts into Pinter et al. (2017) process of representation. This addition further improves the positioning of a word, as two sources of information are involved: the spelling and the context. Bojanowski et al. (2017) also addressed rare words by extending a skipgram model (Mikolov et al., 2013a) to include subword units, thereby improving the representation of rare terms (see section 2.4.1 for background information on this). This is called into question in section 5.4. Due to a lack of standard evaluation metrics for prediction diversity, in this work I propose a way to measure a model's performance as a function of class frequency, focusing on the prediction diversity of word-types (classes). Further work on the topic can be found in Chapter 6.

3.3 Methods

I conducted a series of word prediction experiments in which I compared the performance of a traditional categorical prediction model with my novel GAN-based continuous approach. In what follows I elaborate on the methods and then review the results.

3.3.1 Datasets

I performed experiments on two corpora: newswire text from the New York Times (NYT) section of the Annotated English Gigaword corpus (Ferraro et al., 2018, LDC2018T20), and full-text biomedical journal articles from the open-access subset of PubMed Central (Beck, 2010) (PMC). The PMC corpus was part-of-speech tagged using ScispaCy (Neumann et al., 2019, Version 0.2.2, model: `en_core_sci_md`), and its sentences were split with spaCy. These two corpora differ not only in their content but also in their vocabulary distribution. PMC contains a much larger variety of word-types than NYT, but more importantly, it includes many more tokens from from low-frequency types ("long-tail types"), as shown in Figure 3.1. The PMC dataset was chosen in addition to NYT to serve as a stress-test in the experiment. It allowed me to learn how the models operate when there is a *greater* number of low-frequency training examples (i.e., word-types). For the purposes of my experimental evaluation, I attempted to match the two corpora's vocabulary sizes, so as to ensure similar output dimensionality across the categorical models. The NYT corpus is notably larger than the PMC corpus in terms of token count. As such, I completed this task by tokenizing NYT and then tokenizing various sizes of PMC until the two datasets were approximately matched in vocabulary size. There were 950k types and 734M tokens in the NYT corpus, and 860k types and 458M tokens in the PMC corpus. Each vocabulary type observed fewer than four times in the training set was replaced with an *unk* symbol, and numeric tokens were replaced with class tokens.

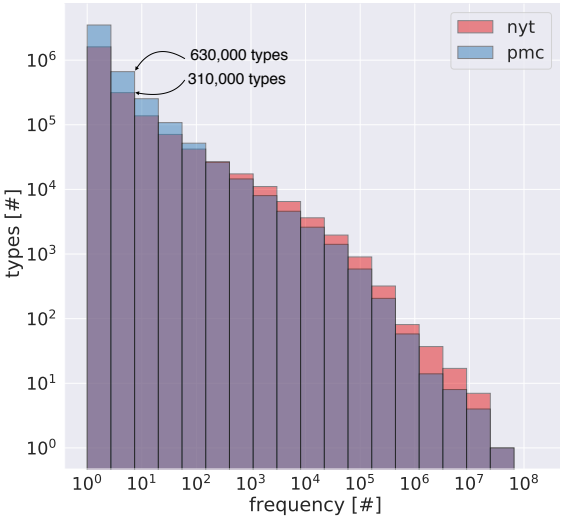


Figure 3.1: Token-type distribution.

3.3.2 Models

I conducted a series of word prediction experiments in which I compared the performance of a traditional categorical prediction model with my novel continuous approach. These experiments and models are interleaved throughout this chapter. The models described throughout this work were designed to be as similar as possible architecturally[4], thus ensuring that the final layer of each model is the main source of change. I start by discussing two very basic types of models: a categorical model and a continuous model. The categorical model (referred to as `ctg` and which is based on the method of Sutskever et al. (2011)) contains an RNN layer, as shown in Eq. 3.1, and is fed with an embedding representation of a word or a sequence of words.

$$h_t = LSTM(w_{t-1-n}, .., w_{t-1}), w_i \in \mathbb{R}^{d\times 1}, h_t \in \mathbb{R}^{m\times 1} \tag{3.1}$$

This LSTM layer is followed by a softmax layer computed using Eq. 3.2,

$$\hat{w}_t = \frac{exp(h_t^T \theta_k)}{\sum_{j\in|V|} exp(h_t^T \theta_j)}, \forall k \in V, \theta \in \mathbb{R}^{m\times v} \tag{3.2}$$

where the model's goal is to estimate a probability distribution over the set of the model's vocabulary items. The `ctg` model is then trained based on maximum likelihood estimation (MLE) through a cross entropy objective, as seen in Eq. 3.3:

$$L_t = \sum_{w_t \in W_t} T(w_t) \log(P(\hat{w}_t)) \tag{3.3}$$

The second model is a continuous output prediction model, which is a regression model that was designed to be as similar in design as possible to `ctg` model. This second model (referred to as `c` and which was inspired by Kumar et al. (2019)) also

[4]The models investigated are by no means state-of-the-art models, but they are given in simplified form to facilitate comparison between them, focusing on the differences resulting from the different decoding processes.

has a single LSTM (Eq. 3.1) that receives as input of an embedding sequence. The model then has a linear layer followed by a non-linear one,

$$g_t = \frac{e^{h_t^T \theta} - e^{-h_t^T \theta}}{e^{h_t^T \theta} + e^{-h_t^T \theta}}, \theta \in \mathbb{R}^{m \times d} \tag{3.4}$$

and a normalizing layer

$$\hat{w_t} = \frac{g_t}{\|g_t\|_{L_2}}, \hat{w_t} \in \mathbb{R}^{1 \times d} \tag{3.5}$$

and the cosine loss function shown in Eq. 3.6. This particular loss was chosen because measuring the similarity of word embeddings is usually done through cosine similarity. Therefore, if the objective is to make a vector *similar* to a particular target, this can directly inform the model of how to set the gradient to make it more similar. However, during development, I also experimented with MSE and max-margin losses and found that cosine loss consistently performed best, as described in section 5.5.

$$L_t = \sum_{w_t \in W_t} |2 \times (1 - < \hat{w_t}, w_t >)|^{0.5} \tag{3.6}$$

where the goal is the model's estimate, $\hat{w_t}$, in the form of a vector representation. The learned representations are based on a (fixed) pre-existing embedding space in which each vector is associated with a corresponding category (of a word entry). All the models in this chapter are fed with embeddings as input in an attempt to set the decoding stage to be the only part that differs across models.

3.3.3 Embeddings

In all my experiments, I trained the embedding models from scratch on the same data used for training the language models. To make the experiments as similar as possible, for all the different models described in this work, I used an embedding or embedding sequences. I explored two dimensionalities, 50 and 200, to maintain low model complexity. Unless stated otherwise explicitly, word2vec was the embedding technique used. In section 5.4 I evaluate the impact of the word embedding space on three different pre-trained embedding spaces: word2vec (Mikolov et al., 2013b)[5], GloVe (Pennington et al., 2014)[6], and FastText (Bojanowski et al., 2017)[7] (`w2v`, `ft`, and `glv` models), each with a dimensionality of 50. The reason contextualized embeddings were not employed is that learning the locations of all tokens (vs. all word-types) may be inefficient both for the learning model (prediction of continuous vectors) and for the process of decoding a vector to a word, thereby creating an inflated space from which to decode a term and in doing so increasing complexity. In this thesis I propose a method to extend Kumar et al. (2019). Following this work, I employed static representation vectors despite the polysemy limitation that remains to be addressed in future work.

[5]I used the implementation by Řehůřek et al. (2010).
[6]Please see https://github.com/stanfordnlp/GloVe.
[7]Please see https://github.com/facebookresearch/fastText.

3.3.4 Decoding

The decoding process maps the embedding prediction to a point in the embedding space to associate the prediction with a word-type. I describe simple decoding in the following. I also propose a new decoding mechanism, which is described in section 5.3.

Simple Decoding: I apply a nearest-neighbor search to match the predicted vector with an embedding vector representation that will recover the predicted word. I apply maximum inner product search (MIPS) (Shrivastava et al., 2014). Given a large data vector W collection (of an embedding space) of size N, where $W \subset \mathbb{R}^D$ and a query point $q \in \mathbb{R}^D$, I can search for the vocabulary entry that maximizes the inner product of the predicted vector and any possible word-embedding candidate (Eq. 3.7)

$$p = \underset{w \in W}{\operatorname{argmax}} \langle q, w \rangle \tag{3.7}$$

To perform an efficient MIPS, I made use of the `Annoy` library (Bernhardsson, 2018) with angular distance.[8] The most challenging part of the process is the decoding part. Therefore, to ensure that the cost of the continuous output prediction is low, it is crucial that this part is efficient. Annoy has an optimized nearest-neighbor search in high-dimensional spaces. First, if the dimensionality of such a space is considered too high $>> x$, then hashing methods are applied to project the vectors to a lower dimension and in doing so reduce the computational costs of the inner product. Second, Annoy makes the search more efficient by making k binary trees (creating a forest of trees). This reduces the computational costs of the search. While this algorithm does not construct a single binary tree, the resulting forest of trees has an approximate cost $\propto O(\log |V|)$.

3.3.5 Process

The dataset split was 60/20/20 train/dev/test. The dev set was used for early stopping (Yao et al., 2007) of the training based on the dev loss. At this point I am able to illustrate the entire process of continuous output prediction. Figure 3.2 captures the process: (1) tokens are converted to embeddings and are fed into the language model; (2) a predicted vector (referred as $\hat{w}_t$) is generated; (3) this predicted vector is mapped to the nearest embedding vector; and (4) it is mapped into a category (via a simple dictionary data structure). This category is the word the model is considered to have predicted.

3.3.6 Metrics

In my word prediction experiments, the key unit of evaluation was the prediction attempt: given a word history, was the model able to correctly predict the following token? This is referred to as a "hit" – this is 1-best. Because a prediction is in a continuous embedding space, the various neighbors of the 1-best word may be relevant or appropriate; furthermore, as many real-world word prediction tasks involve

[8] Please see https://github.com/spotify/annoy.

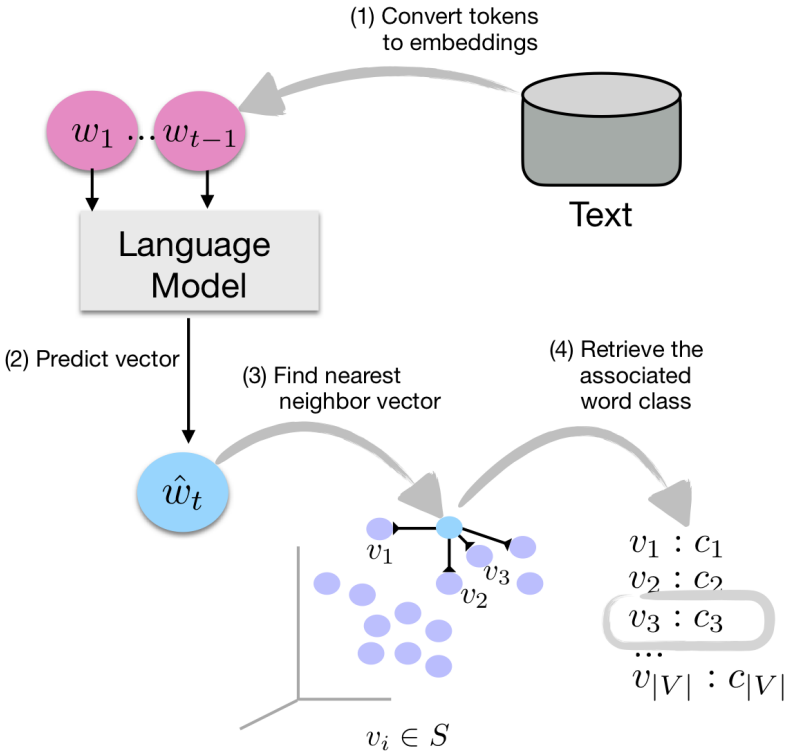


Figure 3.2: Continuous output model learning schema.

presenting a user with a short set of candidates, in practice, a near-miss may still be a useful result. As such, I also considered the neighborhood of the ten closest words to the predicted vector. In addition to exploring raw prediction performance, I was particularly interested in the *diversity* of predicted types (see section 2.5.2 for a discussion of factors motivating the evaluation of prediction diversity); a model that only ever predicts extremely common words may achieve a high accuracy score but is of no practical use. The metric used in the literature to measure diversity that is most similar to the metric I describe is Distinct1, proposed by Li et al. (2016a) (see section 2.3.3). This metric is different than mine, however, as it measures the overall number of unique items predicted rather than a given prompt, thereby revealing the overall degree of type coverage by the model. I used following metrics to evaluate my models:

- $top_1(top_{10})$ - percentage of trials that constituted a "hit" (i.e., target is within top $1/10$ neighbor(s)) vs. a "miss."
- $T_1(T_{10})$ - number of unique *types* that the model correctly predicted (i.e., that appeared in at least one $top_{1/10}$ hit).

Example: To illustrate these metrics, assume that 20% of the target words in a given test set are 'the' and that a dummy model predicts the word 'the' for every query in the test set. The model's performance on top_1 would be 20%, but for T_1, since the model's hits on the test set are based only a single word-type ('the'), the number of diverse types the model would correctly predict is 1. Note that the T_1 is computed across all the correct predictions in the test set to form a unique list.

I computed these metrics over the predictions on the entire test set; I also also broke them out into vocabulary frequency bins. This allowed me to investigate the

models' performance on rare words.
A note on perplexity: While perplexity is a traditional metric for evaluating language model performance, I did not use it in this set of experiments. This choice was driven first by the fact that my predictions, which are in a continuous space, do not lend themselves naturally to perplexity calculation, and second, that my focus is on a practical application-oriented task (word prediction) where metrics such as top_1 and T_1 are more closely linked to my outcome of interest. Note that the metrics proposed here reveal what perplexity may overlook, which is how well the model performs in practice, particularly regarding the variety of types the approach succeeds in predicting.

3.3.7 Baselines

In these experiments, I compared continuous models to categorical ones starting with the architectures of the `c` and `ctg` models, as well as those of two additional baseline models: `freq`, which consistently predicts the ten most common words in the test corpus, and `ugrm`, which randomly samples ten words from the test corpus's unigram distribution.

The purpose of the `freq` baseline is to account for the possibility of *mode collapse*, a common failure mode in which a model predicts a very limited number of frequently observed values. This concern mainly applies to generative approaches (of the continuous output type), which are notoriously prone to mode collapse, as shown by Che et al. (2016) and Srivastava et al. (2017). If a model predicts a small number of modes (which in this particular context is exhibited by predicting a small number of word-types, typically the most frequent), failure mode can be concluded due to the mode-collapse situation. I was particularly concerned about this failure mode given the heavily skewed distribution of words in a linguistic corpus stemming from its Zipfian nature (Zipf, 1935) and the fact that the distributional properties of word embedding spaces tend to be affected strongly by word frequency (Gong et al., 2018), which may make the model gravitate towards a particular region in the embedding space instead of attempting to learn various locations in the space. `ugrm` baseline is another lower-bound model used to deduce the degree of diversity when a relatively more sophisticated model is found based directly on the training frequency.

I therefore devised the `freq` and `ugrm` baselines in order to simulate models suffering mode collapse.

3.4 Results

Having finished discussing the methods, I turn to review the results.

3.4.1 High-level analysis

Table 3.1 presents the combined results of the two models (`ctg` and `c`), with the appropriate baselines (`freq` and `ugrm`) and when evaluated against a newswire corpus. The `freq` model shows that simply picking the most frequent word achieves a top_1 hit rate of 0.89%, and the ten most frequent words are accurate 23.39% of the time. This shows that by predicting a very small number of distinct types, a model can

model	top_1 (top_{10})	T_1	(T_{10})
`freq`	00.89 (23.39)	1	(10)
`ugrm`	00.71 (08.46)	2,190	(5,592)
$\texttt{ctg}_{50}$	19.19 (46.02)	3,982	(7,559)
$\texttt{c}_{50}$	17.31 (28.70)	8,917	(22,509)
$\texttt{ctg}_{200}$	21.21 (47.87)	4,163	(7,683)
$\texttt{c}_{200}$	18.94 (30.90)	4,335	(13,087)

Table 3.1: Experimental results on large NYT corpus.

significantly improve its hit rates. The `ugrm` model correctly predicts 2,190 types, giving a reasonable estimate of what a stratified model might achieve in terms of type diversity, but it achieves a much lower hit rate than the `freq` baseline.

$\texttt{ctg}_{50}$ and $\texttt{c}_{50}$ in Table 3.1 demonstrate an interesting trade-off. While the categorical baseline's (`ctg`) top_1 is optimal among all the categorical models and is higher than that of the continuous model, the latter outperforms `ctg` in terms of T_1, successfully predicting more than twice the number of distinct types that `ctg` is able to predict (the effect is even more pronounced in terms of T_{10}). When the embedding dimensionality is increased to 200, as shown in Table 3.1, the continuous approach ($\texttt{c}_{200}$) again achieves a lower absolute hit rate but maintains its advantage in type diversity (shown in T_{10}).

model	top_1 (top_{10})	T_1	(T_{10})
`freq`	00.89 (25.53)	1	(10)
`ugrm`	00.76 (09.03)	1,790	(4,619)
$\texttt{ctg}_{50}$	22.12 (48.02)	4,764	(9,020)
$\texttt{c}_{50}$	19.89 (32.04)	11,947	(34,641)
$\texttt{ctg}_{200}$	22.92 (48.47)	3,678	(7,006)
$\texttt{c}_{200}$	17.06 (30.88)	6,083	(18,533)

Table 3.2: Experimental results on large PMC corpus.

Table 3.2 presents a pattern in baseline performance similar to that of Table 3.1 with regard to the PMC dataset. This table shows stronger trends for the non-baseline approaches than it does for the baseline approaches. According to Table 3.2, in the first triples of dimensionality 50 when the continuous approaches are two points below in top_1, the type diversity T_1 of $\texttt{c}_{50}$ is about threefold with regard to `ctg`. Increasing dimensionality[9] to 200 decreases the hit rate as well as the number of types for the continuous approach, yet the types are still more than 1.5 for $\texttt{c}_{200}$. In the PMC experiment, increasing dimensionality to 200 contributes to higher hit rates for $\texttt{ctg}_{200}$ but reduces the number of types. $\texttt{ctg}_{200}$ in the NYT experiment

[9]Recall that all models are fed with an embedding sequence to maintain a controlled environment.

maintains a relatively similar number of types. Overall across domains, while the hit rates for the `ctg` approach are higher, the continuous models often predict many more types correctly, indicating that the models have learned more patterns that are unique to the particular domain at hand.

3.4.2 Proposing an adversarial continuous output model (GAN)

Following the results seen in both Tables 3.1 and 3.2, I propose a new continuous output prediction model that is based on generative adversarial learning (GAN) (Goodfellow et al., 2014). For the GAN model, I employed the continuous model (`c`) as the generator (`G`). The continuous model was trained jointly with a discriminator (`D`) to provide stimuli for training the generator. The dynamic is as follows: `G` generates a fake embedding representation for a given history. `D` is then fed through two channels, one with the given history and another with fake/real samples that are provided by either the fake vector created by `G` or with the real word embedding taken from the data. In `D` each of the real or fake vectors is compared to the outcome of `D`'s RNN (internally imitating the process in `G`); this comparison results in a probability score for the real or fake sample. `D` tries to learn what the real data patterns are like, while `G` tries to learn how to fool `D`. Both goals are reflected in the models' objectives. Architecturally, `D` and `G` are similar, but inspired by Mirza et al. (2014), I decided on having two channels to send the data. The step in which the discriminator compares its generated embedding with the provided sample to make a prediction about the sample's authenticity is described in Eq. 3.8.

$$D_t = \sigma\left((\hat{w_t}^D - \{\hat{w_t}^{fake}, w_t^{real}\})^T\theta + b\right) \tag{3.8}$$

My final loss calculation (Eq. 3.9) closely follows the architecture used by Yang et al. (2018a) and Yu et al. (2017) in that it conditions over a history of token vectors as input. However, it is different in its output. Rather than directly choosing a symbolic representation of item category within the model, the final loss calculation aims to generate a vector representation that later is mapped to an existing embedding representation associated with a category:

$$\begin{aligned}\min_G \max_D L(D,G) =& \mathbb{E}_{w\sim p_{data}(w)}[\log D(w_t|w_{history})]+ \\ & \mathbb{E}_{\hat{w}\sim p_{\hat{w}}(\hat{w})}[\log(1 - D(G(\hat{w_t}|w_{history})))]\end{aligned} \tag{3.9}$$

To the best of my knowledge, this is the first attempt to employ this type of model for vector embedding prediction. Whereas GANs are trained using a joint loss function, the generator and discriminator are trained under MSE loss, as shown in Eq. 3.9. The generator uses the same cosine loss as is used in the `c` model as its embedding loss.

Figure 3.3 shows the high-level architecture of the generator (`G`) on the left and the discriminator (`D`) on the right. On the left, the `G` is fed with a history input sequence, together with a Gaussian noise initializing the first hidden layers of the RNN, to ultimately generate a 'fake' continuous representation ($\hat{w_t}^{fake}$). This 'fake' representation, or a 'real' representation (w_t^{real}) (the appropriate word embedding of which continues the $w_{1:t-1}$ sequence), is fed to `D` together with the history. A score results from comparing the embedding that `D` generated internally ($\hat{w_t}^D$) to that

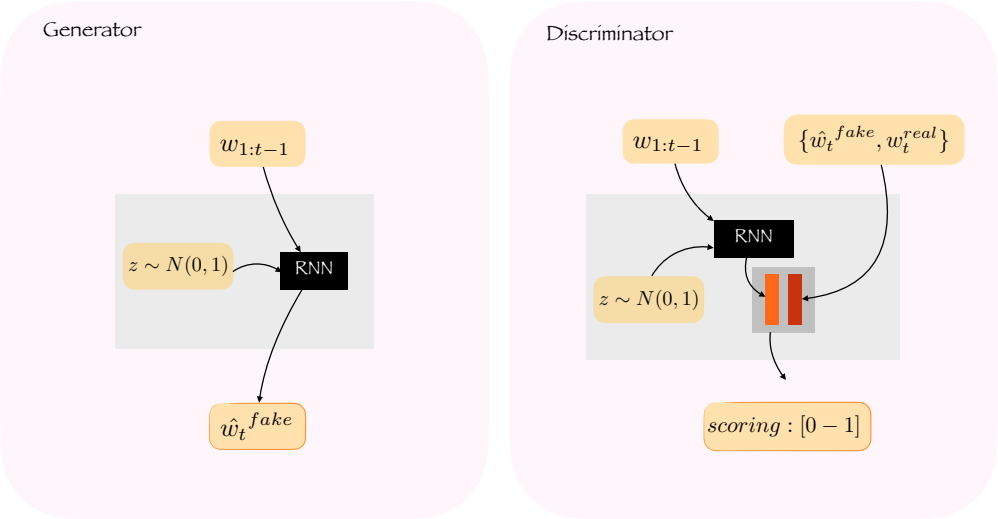


Figure 3.3: Word prediction conditional GAN schema.

which was fed. `D` tries to learn how to distinguish between real and fake embeddings, but in this game `G` will fool `D` until convergence. At the end of the training, only the generator `G` is required for producing a prediction as this is the language model.

The reason I explored this type of architecture is that a continuous output is differentiable, making the losses propagate more optimally than a discrete output (Yu et al., 2017). The problem happens whenever argmax is employed in a GAN dynamic, which is expected given the softmax architecture. `G` tries to increase the likelihood `D` assigns to its predictions by converting `G`'s predictions to hot representations (via argmax) and feeding them into `D` (minimizing $1 - D(G(z))$). However, back-propagating `D`'s loss through the hot-encoded tokens is problematic. The question then becomes: why apply argmax over the tokens in the output sequence? The reason is that since the real sentences are hot-encoded, if the fake are presented as a probability distribution, then it is easy to distinguish between real and fake examples. Another alternative that avoids this problem is Kusner et al. (2016).

3.4.3 GAN's model performance

Tables 3.3 and 3.4 show that in most cases, `G` has a decreased performance in terms of hit rates for top_1 and top_{10}, but it provides the most diverse number of types across all experiments, as shown in T_1 and T_{10}. Dim 50 tends to be greater in the number of types, while dim 200 is greater in terms of hit rates.

3.4.4 Long-tail analysis

Recall that I am particularly interested in the models' performance on predicting rare words and aim to have the continuous models learn to predict items in both frequent and rare neighborhoods in the embedding space. I therefore performed a stratified analysis by target word frequency to assess whether the models' behavior differed for rare vs. common words.

Figure 3.4 presents type coverage (T_1) for the NYT experiment in three frequency bins: *high*, *mid*, and *low*, which correspond to $x \in [10^3, \text{inf})$, $x \in [10^2, 10^3)$, and $x \in [10^1, 10^2)$, respectively. x is each target type's frequency, following Kumar et al.'s

model	top_1 (top_{10})	T_1	(T_{10})
`freq`	00.89 (23.39)	1	(10)
`ugrm`	00.71 (08.46)	2,190	(5,592)
$\texttt{ctg}_{50}$	19.19 (46.02)	3,982	(7,559)
$\texttt{c}_{50}$	17.31 (28.70)	8,917	(22,509)
$\texttt{G}_{50}$	16.46 (27.45)	11,534	(27,921)
$\texttt{ctg}_{200}$	21.21 (47.87)	4,163	(7,683)
$\texttt{c}_{200}$	18.94 (30.90)	4,335	(13,087)
$\texttt{G}_{200}$	18.15 (29.15)	6,194	(17,905)

Table 3.3: GANs learning on large NYT corpus.

model	top_1 (top_{10})	T_1	(T_{10})
`freq`	00.89 (25.53)	1	(10)
`ugrm`	00.76 (09.03)	1,790	(4,619)
$\texttt{ctg}_{50}$	22.12 (48.02)	4,764	(9,020)
$\texttt{c}_{50}$	19.89 (32.04)	11,947	(34,641)
$\texttt{G}_{50}$	19.04 (30.41)	18,040	(47,550)
$\texttt{ctg}_{200}$	22.92 (48.47)	3,678	(7,006)
$\texttt{c}_{200}$	17.06 (30.88)	6,083	(18,533)
$\texttt{G}_{200}$	20.67 (32.96)	9,411	(28,164)

Table 3.4: GANs learning on large PMC corpus.

(2019) split that was shown to capture meaningful learning differences. I observe that the type coverage patterns of the models vary greatly across the frequency bins. In the high-frequency bin, the type coverage for `ctg` was about 20% for both the 50 and 200 dimensions (of a total of $n = 19,204$ types). The continuous approaches were more diverse than the categorical with 7,700, 6,400, and 4,100 types $\texttt{G}_{50}$, $\texttt{c}_{50}$, $\texttt{ctg}_{50}$. For 200 dim, I observed no meaningful differences. For the mid- and low-frequency bins, the continuous approaches clearly outperformed their respective `ctg` models. In the mid-frequency bin, the $\texttt{ctg}_{50}$ model was predicting as low as 55 types, whereas $\texttt{c}_{50}$ and $\texttt{G}_{50}$ predicted 2,100 and 3,100 types, respectively. Similarly, 40, 760, and 1,300 were the numbers of correctly predicted types for $\texttt{ctg}_{200}$, $\texttt{c}_{200}$, and $\texttt{G}_{200}$, respectively. In the low-frequency bin, there were no types predicted for any of the `ctg` models, whereas 729, 386, 256, and 78 types were found in $\texttt{G}_{50}$, $\texttt{c}_{50}$, $\texttt{G}_{200}$, and $\texttt{c}_{200}$, respectively.

In addition to investigating type coverage, I performed a stratified analysis of *token* coverage (top_1), as shown in Figure 3.5. Stratifying both accuracy and diversity by frequency helps when interpreting accuracy in the context of type diversity as they are complementary. This way the accuracy can be attributed to a diverse prediction or to the limited number of types that were predicted, which is less desirable. As before, there was a major difference across frequency bins, with all models

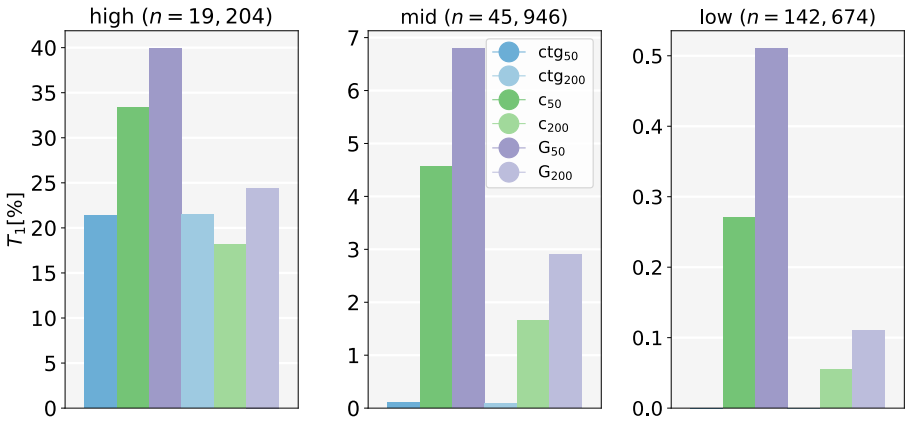


Figure 3.4: NYT *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

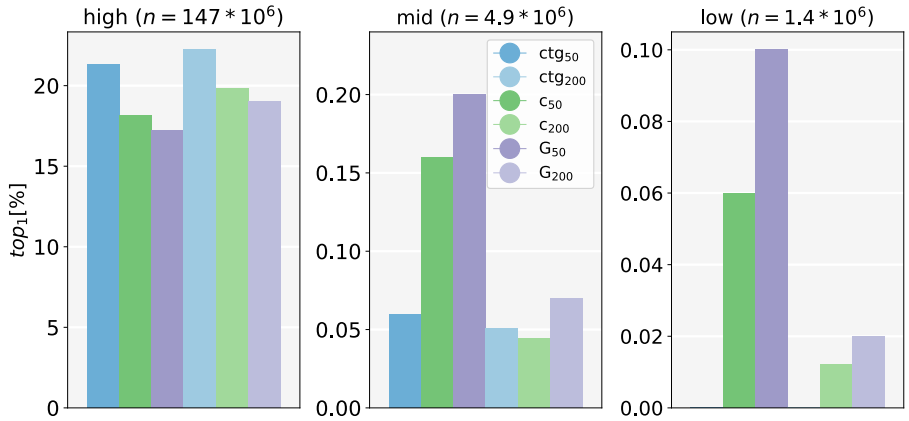


Figure 3.5: NYT *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

performing substantially better on high-frequency terms but with the traditional categorical approach struggling to make correct predictions on mid-frequency terms and completely failing on low-frequency terms. Note that the mid-frequency bin dim of 200 was not especially different from the other models. My continuous models often were able to correctly predict more mid- and low-frequency terms than their categorical counterparts.

I applied the same analytical method to the results obtained using the PMC dataset, which exhibited very different patterns of type distribution than the NYT dataset, as shown in Figure 3.1. Figure 3.6 demonstrates the type coverage of PMC across frequency bins. Here, too, the differences were more pronounced when the dimensionality was 50. Yet for the majority of bins, the continuous approaches predicted more types than the `ctg` models. In the mid-frequency bin, the numbers of types were 7,200, 3,600, and 265 for $\texttt{G}_{50}$, $\texttt{c}_{50}$, and $\texttt{ctg}_{50}$, respectively, and 3,700, 1,700, and 68 for $\texttt{G}_{200}$, $\texttt{c}_{200}$, and $\texttt{ctg}_{200}$, respectively. Again, neither of the `ctg` models was able to predict any types at all in the low-frequency bin; however, 3,000, 1,700, 810, and 565 types were found in $\texttt{G}_{50}$, $\texttt{c}_{50}$, $\texttt{G}_{200}$, and $\texttt{c}_{200}$, respectively.

Figure 3.7 illustrates patterns similar to those seen in Figure 3.5, demonstrating decreased performance in terms of hit rate in dim 200 across all models in the mid- and low- bins, yet higher hit rates for continuous models within each dimension.

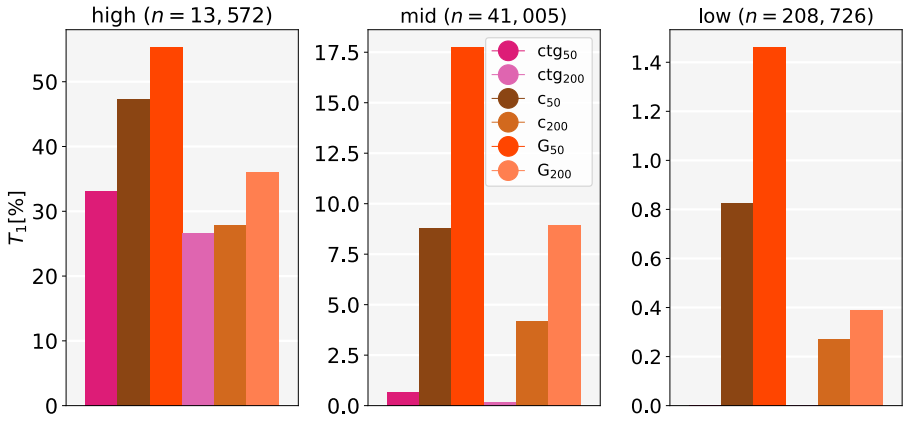


Figure 3.6: PMC *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

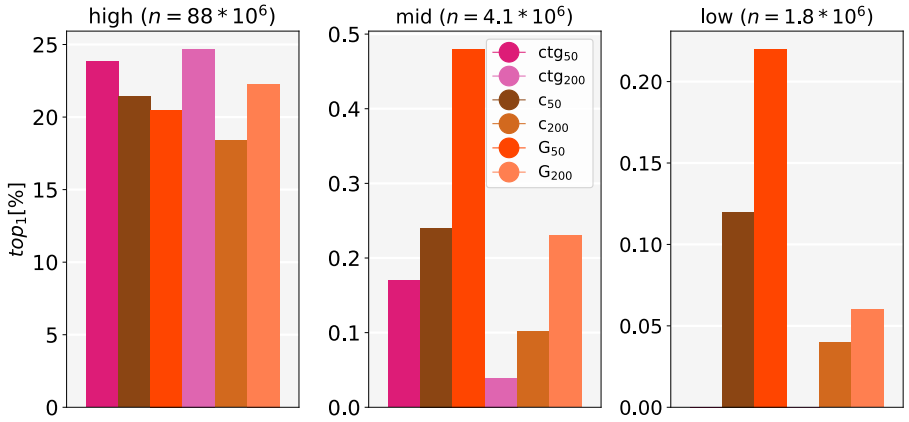


Figure 3.7: PMC *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

This analysis shows that in longer-tail areas across domains, and longer-tail domains themselves, the continuous models were much stronger in terms of type diversity.

Qualitative examples

In this section I provide a few qualitative examples of the mid- and low- bins' predictions for the different models[10] In the examples that follow, the predicted token (i.e., the token that the model was required to predict) is in *italics*. These examples illustrate correct predictions made by the continuous models and incorrect predictions made by the categorical models, respectively. Where appropriate, I include a portion of the sentence following the target token for context; recall that the models did not have access to this information at the time that they were queried.

`G (NYT,mid):` my piano teacher at the paris *conservatory* a woman ...

`ctg(NYT,mid):` my piano teacher at the paris *hotel* a woman ...

`G (PMC,low)` ... owing to the anthropophilic and *endophilic* behaviour of ...

[10]More examples appear in Appendix A.1.1.

`ctg(PMC,low)`: ... owing to the anthropophilic and *unk* behaviour of ...

`G (PMC,mid)`: human infections are common following handling or processing of infected turkeys or *ducks*

`ctg(PMC,mid)`: human infections are common following handling or processing of infected turkeys or *by*

`G (PMC,mid)`: ... this is further confirmed by detection of apoptotic and *nonapoptotic* death ...

`ctg(PMC,mid)`: ... this is further confirmed by detection of apoptotic and *cell* death ...

The examples provided illustrate typical differences between `ctg` and `G` models. Observe that compared to `ctg`, `G` retrieves rarer and more context-sensitive tokens such as (*piano*,*conservatory*), (*anthropophilic*,*endophilic*), (*turkeys*,*ducks*), and (*apoptotic*,*nonapoptotic*). These tokens reflect greater similarities to previous keywords in the sentence. These examples, together with those in Appendix A.1.1, qualitatively demonstrate the strength of the approach proposed in this chapter.

Statistical evaluation

I verified that the `ctg` models' frequency-binned performances were significantly statistically different from those of the continuous approaches. To compare the distribution of predicted types across the frequency bins, I applied a χ^2 two-sample test given the overall uneven sizes of the hit populations[11] My concern was that, given the large differences in scale between the bins, the between-model differences that I observed in the mid- and low-frequency bins might be due to chance and the models might be more equivalent than they appeared. Hence my goal was to understand whether at least one bin population (whether high-, mid-, or low-frequency) of tokens predicted by a continuous model was different from the categorical populations. The null hypothesis for this test was that the continuous models proposed in this work follow the same distribution as the categorical distribution. There were three degrees of freedom corresponding to the three bins (as the samples were not equal in size); I used Bonferroni correction for multiple comparisons. Eq. 3.10 describes the test:

$$Q^2 = \sum_{i=1}^{k} \frac{m_1 O_i - m_2 Ei}{O_i + Ei} m_1 = \sqrt{\frac{\sum_i E_i}{\sum_i O_i}}, m_2 = \sqrt{\frac{\sum_i O_i}{\sum_i E_i}} \tag{3.10}$$

In this equation, O_i and E_i are the observed and expected counts in the ith bin, respectively (i.e., the hit counts for the two models under comparison), while m_1 and m_2 are the scaling parameters used to handle the difference in sample size.
NYT: For the hit distribution shown in Figure 3.5, the null hypothesis was rejected for every dimension-matched of a continuous-ctg pair with $p < 0.005$.
PMC: For the hit distribution shown in Figure 3.7, the null hypothesis was rejected for every dimension-matched of a continuous-ctg pair with $p < 0.005$.

[11]For a description, see https://www.itl.nist.gov/div898/software/dataplot/refman1/auxillar/chi2samp.htm.

It can thus be concluded that the continuous model presented a different population in at least one frequency bin, thereby corroborating the evidence provided in the main paper that both approaches essentially produce different distributions of predictions.

In the next two sections, I explore additional categorical baselines, as advances in the field have been made in an attempt to address the concerns and limitations of the categorical model described earlier.

3.4.5 An improved categorical model: The unlikelihood loss function

As mentioned earlier,[12] the lack of diversity in prediction in language models has been observed by Holtzman et al. (2020) and Dinan et al. (2019). It was also demonstrated with GPT-2 by Welleck et al. (2020). The need to diversify prediction has been explored through both improved decoding strategies (Li et al., 2016a; Serban et al., 2015) and alternative objective functions. Several different approaches to beam decoding, which prevents predicting tokens from being proposed again in an attempt to avoid repetition, have been proposed. Many of them revolve around penalizing or excluding word-types that are similar to previously predicted types in a beam search (Li et al., 2016b; Kulikov et al., 2018; Holtzman et al., 2018). Other decoding approaches include pure sampling and increased temperature (as shown in Holtzman et al. (2020)). Still other approaches rely on new objective functions. For instance, He et al. (2019) presented a negative loss function with which to update a model after MLE-based training if an undesired prediction is present. This negative loss demotes the currently predicted type through additional model updates. Edunov et al. (2018) proposed incorporating sequence-level loss with a task-specific cost function (e.g., BLEU) to encourage less-frequent sequences. This technique has been shown to produce improved results on machine translation as well as text summarizing tasks. Pereyra et al. (2016) proposed training with a smoothed token distribution to reduce a model's confidence that is created by a typical negative log likelihood loss (updated with 'hot' representation). I chose a recent and successful approach called unlikelihood loss, which was proposed by Welleck et al. (2020).

$$UL_t = \sum_{w_t \in W_t} -\alpha \sum_{c \in C^t} \log(1 - p_\theta(c|w_t)) - \log p_\theta(w_t|w_{<}t) \tag{3.11}$$

This enhanced loss function has two components. The first one appears on the left-hand side of Eq. 3.11: $(\log(1 - p_\theta(c|w_t)))$, which demotes up to five word candidates (by choosing them from the previous context words). The second, shown on the right-hand side of Eq. 3.11, is an MLE loss through cross entropy aimed at promoting the target word that is also used in the categorical model (see Eq. 3.3). This loss function provides a more informed updating process as the model learns both what the appropriate prediction is and what predictions are unlikely to be appropriate, resulting in a richer feedback signal. Moreover, since the immediate (previous) context is demoted, this loss function can intuitively explain why repetition may be discouraged.

[12] This argument is made and substantiated in section 2.5.2.

3.4.6 An improved categorical model: Employing subword unit tokenization

Another enhanced categorical model is based on the recent tokenization approaches of subword representations (Sennrich et al., 2016; Wu et al., 2016b; Lample et al., 2019), which have become common means of representing tokens (Devlin et al., 2018; Liu et al., 2019; Alec et al., 2018; Alec et al., 2019). These approaches were devised for two main reasons. First, they promote complexity reduction, as employing partial sentences or word segments reduces the overall units used to represent a vocabulary, thereby leading to smaller computational costs. Second, the building blocks formed by the word segments can handle rare words (Sennrich et al., 2016; Wu et al., 2016b; Lample et al., 2019). A ancillary advantage is open vocabulary prediction; in other words, these segments can be sequenced in ways that are not seen during training, thereby creating words outside of the trained vocabulary.

The different tokenization approaches segment words or sentences based on the fundamental principle of Huffman (1952) in which more common sequences are encoded with fewer bits, thereby requiring fewer units to be spelled. WordPiece's (Wu et al., 2016b) and XLM's approaches to tokenization (Lample et al., 2019) create segments of words, while BytePair's (Lample et al., 2019) approach is based on sentence segmentation. The following is how each approach segments words:

WordPiece `['to', '##ken', '##ization']`

XLM subword `['to', 'ken', 'ization</w>']`

BytePair `['to', 'ken', 'ization ']`

In WordPiece an internal word segmentation is indicated with two pound signs `'##'`; in XLM the final suffix is indicated with the end of word sign `'</w>'`; and in BytePair, tokens are formed from segmented sentences in which spaces are included. For this experiment I found WordPiece tokenization to be especially easy for word evaluation as its unit of segmentation is words;[13] however, the experiment would have been possible using sentence segments from BytePair as well. In the remaining part of section 3.4.6, I describe the decoding process for words.

Decoding words from their WordPieces

Since WordPiece (Wu et al., 2016b) tokenization is based on word boundaries, it is easier to evaluate word units' generation. To ensure flexible decoding, first-rank predictions are produced by initially feeding the model with the context sequence (as is done with every model) and then continuing to feed it with the most likely guesses for as many rounds as needed until the model produces a complete word (or the model stops producing valid suffixes). Once the model is finished, the WordPieces are concatenated to form the predicted word that is then compared to the target word. Note that various segments are allowed to form a complete predicted word as the *word* is the unit of evaluation. Producing the first ten guesses is especially complex. The model goes through a depth-first search (DFS). This way every possible valid word that can be spelled to match a target is extracted.

[13] XLM's tokenizer is an optimal choice, too, for the same reason.

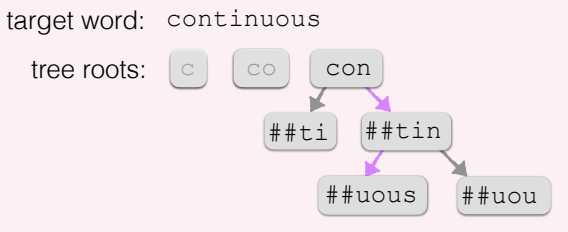


Figure 3.8: Depth-first search for top ten guesses with WordPiece tokenization.

In Figure 3.8, during the initial round of prediction, if valid roots (i.e., valid prefixes) are found within the top ten guesses, they are stored. The following rounds exhaust possible additional word segments to continue a valid prefix, and they are stacked until either a predicted target pair is matched or all valid paths have been exhausted. This ensures subword flexibility for the top ten words. The algorithms for the first and the top ten guesses are Algorithm 1 and Algorithm 2.

3.4.7 Overall performance across experiments

High-level analysis

Tables 3.5 and 3.6 add to Tables 3.3 and 3.4, reflecting the additional categorical baselines: a categorical model trained with WordPiece that was trained with a custom tokenizer (`wpe`) where the tokenizers were trained on the training sets, and the unlikelihood loss function model (`UL`). In Tables 3.5 and 3.6, the basic categorical model called `ctg` is renamed `mle`, representing the simple MLE loss function.

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{mle}_{50}$	19.19 (46.02)	3,982	(7,559)
$\texttt{wpe}_{50,nyt}$	<u>28.28</u> (52.61)	4,037	(7,630)
$\texttt{UL}_{50}$	17.50 (39.99)	4,523	(8,782)
$\texttt{c}_{50}$	17.31 (28.70)	8,917	(22,509)
$\texttt{G}_{50}$	16.46 (27.45)	<u>11,534</u>	(27,921)
$\texttt{mle}_{200}$	21.21 (47.87)	4,163	(7,683)
$\texttt{wpe}_{200,nyt}$	<u>29.07</u> (53.67)	4,452	(8,262)
$\texttt{UL}_{200}$	18.33 (41.24)	5,596	(10,352)
$\texttt{c}_{200}$	18.94 (30.90)	4,335	(13,087)
$\texttt{G}_{200}$	18.15 (29.15)	<u>6,194</u>	(17,905)

Table 3.5: Complete experimental results on NYT corpus.

As shown in Tables 3.5 and 3.6, the `wpe` accuracy rate is the highest by a large margin; on the other hand, the number of unique types that were correctly predicted is not meaningfully different (and is in fact worse at times) compared to `mle`. The increased hit rate may be attributed to the flexible decoding strategy (see section 3.4.6) allowing for more than a single possible spelling. However, `wpe`'s T_1 indicates that this approach may not be particularly effective in the prediction of diverse word-types. `UL` produced a lower hit rate while predicting a greater number of

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{mle}_{50}$	22.12 (48.02)	4,764	(9,020)
$\texttt{wpe}_{50,pb}$	<u>27.29</u> (50.23)	3,020	(6,849)
$\texttt{UL}_{50}$	17.52 (39.69)	4,780	(9,657)
$\texttt{c}_{50}$	19.89 (32.04)	11,947	(34,641)
$\texttt{G}_{50}$	19.04 (30.41)	<u>18,040</u>	(47,550)
$\texttt{mle}_{200}$	22.92 (48.47)	3,678	(7,006)
$\texttt{wpe}_{200,pb}$	<u>27.85</u> (50.94)	3,182	(7,389)
$\texttt{UL}_{200}$	18.80 (41.13)	5,888	(10,785)
$\texttt{c}_{200}$	17.06 (30.88)	6,083	(18,533)
$\texttt{G}_{200}$	20.67 (32.96)	<u>9,411</u>	(28,164)

Table 3.6: Complete experimental results on large PMC corpus.

types than any of its categorical counterparts, and this outcome can be attributed to its objective function. Despite the improved performance of `UL`, in most experiments, its performance was worse than that of `c`, and in all experiments, its performance was worse than `G` in prediction diversity. `G` was found to be the most diverse method. Next, I break type diversity down by frequency to investigate it in greater detail across training rates. In Chapter 6 I conduct a similar experiment over a different dataset, evaluating state-of-the-art models. There, I show that the accuracy numbers are similar to the ones presented here, so while the chapters' experiments are not comparable, the outcome in this chapter, based on simple models, is a good proxy for state-of-the-art models.

Long-tail analysis

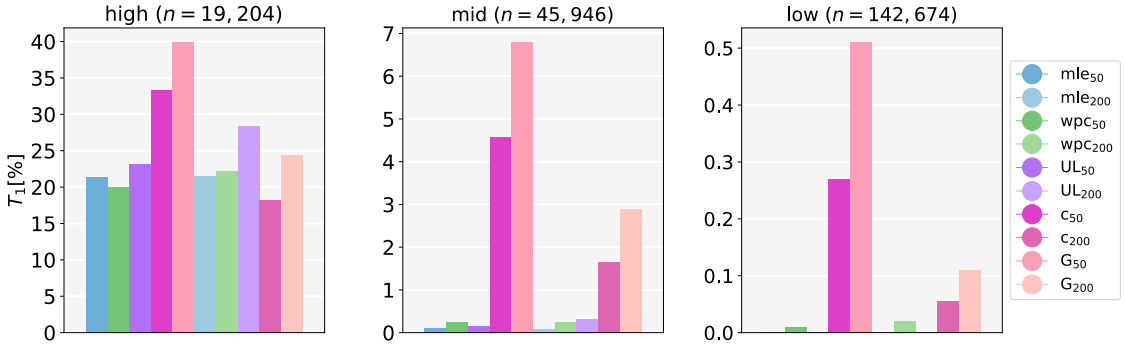


Figure 3.9: NYT *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

The same experiments that are represented in Tables 3.5, and 3.6 are shown in Figures 3.9-3.12. `wpc` in Figure 3.10 and 3.12 (shown in shades of green and brown, respectively) presented the highest hit rates in most experiments, though considering its relatively low diversity, as shown in Figure 3.9 and 3.11, these hit rates are likely to be attributed to the very few types it was producing correctly. Overall, this

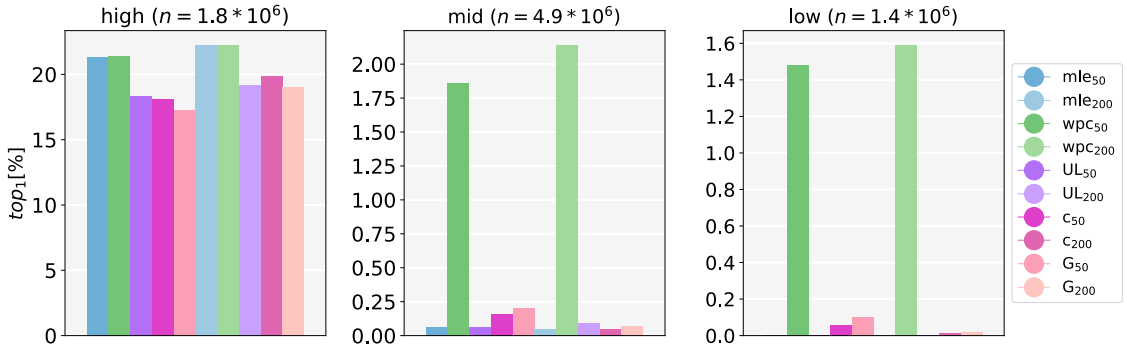


Figure 3.10: NYT *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

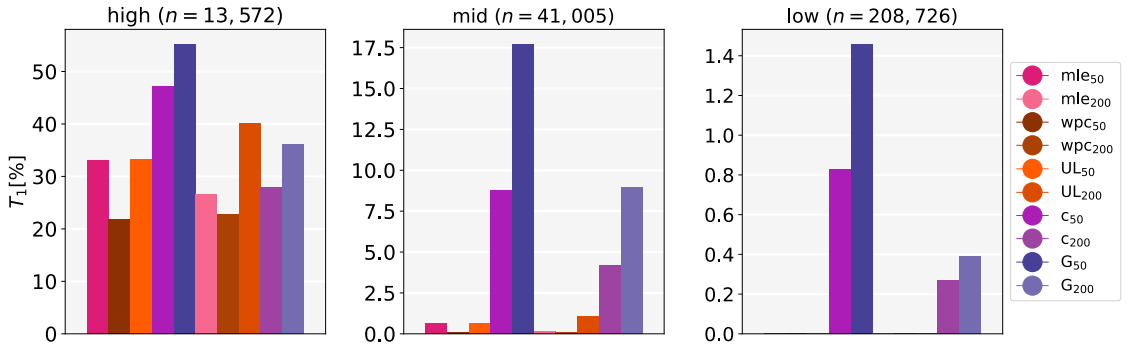


Figure 3.11: PMC *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

analysis shows that in longer-tail areas across domains, and longer-tail domains themselves, the continuous models were much stronger in terms of type diversity (shown in pink and light pink in Figure 3.9 and purple and dark blue in 3.11). GAN in particular was found to be the most diverse model. Having investigated three categorical models and two continuous models across domains and dimensionalities, I conclude that the each of the different approaches may be incentivized differently. The categorical approach promotes high accuracy, while the continuous models are richer in types. In the next section, I probe the computational differences between the continuous and categorical models.

3.4.8 Computational costs

The need to reduce the computational costs associated with machine learning models (Schwartz et al., 2019) is especially pressing today, as the center of focus is on heavy models such as BERT (Lan et al., 2019; Zafrir et al., 2019). In this section, I look at the theoretical and practical costs of the models presented thus far in terms of training, inference, and memory requirements.

The theoretical costs discussed mainly reflect the final layer of each of the different models and *not* an entire model itself, while the real complexity costs present the model's overall real cost. Note that models using the categorical approach represent both the costs of the `mle` and the `UL` models, as they have similar degrees of complexity (up to a constant). `c` and `G` are represented under the continuous approach

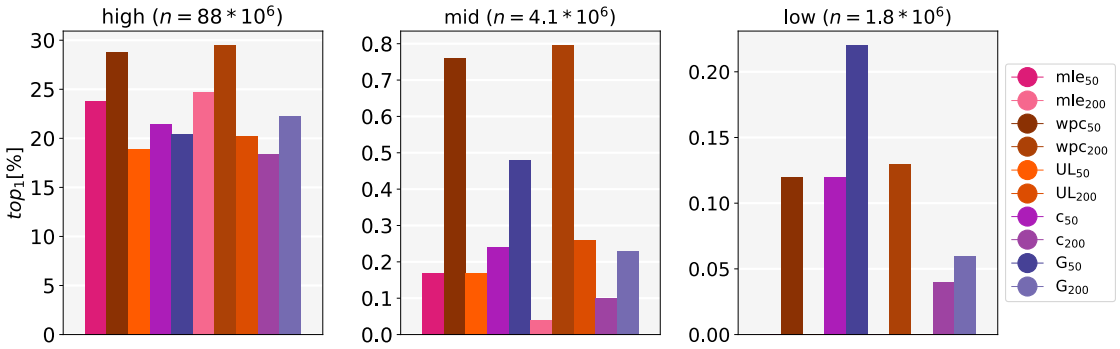


Figure 3.12: PMC *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

for the same reason. The tokenized categorical model is represented separately.

Assume that $|V| > |v| >> d$; in other words, the word-level vocabulary size is larger than the subword-level vocabulary size, the latter of which is much bigger than the embedding dimensionality being predicted (as in this chapter's experiments). In these experiments $900,000 > 50,000 >> 50$ (reflecting one order and three orders of magnitude of difference, respectively). As shown in Table 3.7, under this assumption, the continuous models have lower algorithmic complexity compared to the categorical models during each forward pass at training time. Comparing a continuous ELMo to an equivalent categorical one (of subwords or a sampled softmax of words),Li et al. (2019) pointed to a four-fold speedup in training while eliminating 80% of the required trainable parameters. As discussed earlier, this is due to the fact that there is no need to scan the entire vocabulary space ($|V|$, or $|v|$) as part of the softmax step of the loss calculation (no decoding takes place during the training of the continuous models). In the case of categorical models using methods such as adaptive softmax (Grave et al., 2017), the models may reduce their training complexity to sublinear (with regard to $|V|$ or $|v|$); with a large vocabulary, though, even a sublinear scan of $|V|$ can come at a substantial cost.

Even if noise-contrastive estimation approaches are employed (Mnih et al., 2012; Gutmann et al., 2010; Zoph et al., 2016) for a categorical model, thereby resulting in a reduced training cost at inference time, this model still has to make a pass over the entire vocabulary space in order to predict a word.[14] The continuous architecture seen here uses an approximate nearest-neighbor algorithm (Bernhardsson, 2018) (see section 3.3.4) to search for the optimal vocabulary item during decoding, resulting in computational complexity $\propto O(log|V|)$, .

Memory requirements of the continuous approach are also leaner than those of the simple categorical model but may be greater than those of the tokenized version. My model's output layer does not need to contain parameters for every single vocabulary entry; it only must store a single vocabulary list of embeddings, which can be done very efficiently. This advantage becomes even more important when storing prediction models that are customized for individual users, which is

[14]While we acknowledge hierarchical softmax's (HSM) (Morin et al., 2005) theoretical gains, most models today do not employ this approach. This was true even before subword usage became prevalent. Therefore, in our comparison we consider alternatives to currently employed approaches. Moreover, Grave et al. (2017) proposed an approach that is more suitable to GPUs than the HSM, suggesting that this may be why HSM has been less commonly adopted.

one of the goals of this work, as mentioned in 1. In such a scenario, a categorical approach would involve, at the very least, maintaining a separate model and thereby a separate set of output layer parameters for each user, making the model expensive for large vocabularies. My continuous approach, in contrast, would involve storing a much smaller matrix of per-user output parameters $O(d)$, and the upfront cost of the vocabulary index would be able to be amortized across the entire user population $O(|V|)$ by *sharing* the embedding space.

model	*categorical*	$categorical_{tok}$	*continuous*
train runtime	$O(\|V\|)$	$O(\|v\|)$	$O(d)$
test runtime	$O(\|V\|)$	$O(\|v\|)$	$\propto O(log\|V\|)$
mem. (X usrs)	$O(\|V\|)*$X	$O(\|v\|)*$X	$O(d)*$X$+O(\|V\|)$

Table 3.7: Theoretical complexity analysis for each of the models' final layer (decoding and memory requirements).

Table 3.8 presents the models' *overall* floating point operations (FLOps) costs. I computed a linear layer FLOps through $(2 * I - 1) * J * mb(100)$ such that J, I are adjacent matrices and mb is a mini-batch size. An LSTM layer was computed as $l(2) * ts(25) * 8(gates) * 2(hlayers) * h(200) * mb(100) * (h + 1)$ and had similar dimensions across all models (l, ts, $gates$, and h are LSTM layers, times-teps, LSTM gates, and hidden-layer size, respectively). Recall that all continuous models have the same dimensionality as that of categorical models for the LSTM layer and a different dimensionality than that of the final linear layer and the decoding part taking place. The continuous decoding part is based on the size of the complete vocabulary divided by 1000 clusters (for an average number of comparisons) and multiplied by d (computing angular distance), and it is estimated to be $450k$, which is negligible relative to $128M$[15] Building the Annoy forest of trees[16] requires a one-time cost of $4.5G$ and it is loaded for each call. The memory allocation for each of the continuous models is based on the file size of the model and the embedding space size. In the models presented here, the main difference is found in their final layers; other than that, they are relatively similar. As such, it can be deduced that the difference in the number of approximate calculations in train and inference (FLOps) has to do with the additional operations of the *categorical*'s final layer. If there are very few users in this hypothetical scenario, then memory-wise it may make sense to use the tokenized model and avoid the non-tokenized categorical model. However, when increasing the number of users, the continuous models have the smallest scaling factor (by two orders of magnitude less than the tokenized and non-tokenized categorical models), making for a more appealing approach to scaled deployment.

In future work I will shed light on the wall-clock comparisons of the three models; at this point, however, I note that the continuous models' cost trade-offs shift the burden of costs in ways that may allow for easier scaling across users.

[15] Annoy's library (Bernhardsson, 2018) employs random projections (Bingham et al., 2001) to reduce vector dimensions if needed, so the estimated computation is an upper bound using d.

[16] See Section3.3.4.

model	*categorical*	$categorical_{tok}$	*continuous*
Train appx. calc. (FLOps)	$488M$	$148M$	$128M$
Inference appx. calc. (FLOps)	$488M$	$148M$	$128M$
mem. (X usrs) (bytes)	$731M$*X	$41M$*X	$2.1M$*X+$548M$

Table 3.8: Overall model real complexity costs in floating point operations (FLOps) ($|V| = 900,000$, $|v| = 50,000$, $d = 50$).

3.5 Future Directions

One natural direction the experiments described in this chapter could take involves executing this protocol to examine the different models in relation to morphologically rich languages (MRLs) that are known to have higher type-to-token ratios (Gerz et al., 2018) resulting from language inflections. I hypothesize that the GAN-based approach proposed in this chapter is a promising approach to such languages as the model was shown to be superior on type diversity when using a high type-to-token rate dataset: the biomedical dataset of PMC (where the long-tail of infrequent words was attributed to jargon rather than various inflections). This is also motivated by Czarnowska et al. and (2019), who suggest that current language models are sub-optimal for MRLs. Czarnowska et al. proposed splitting the task into semantic and syntactic tasks. The researchers addressed the syntactic element, and this work complements theirs by adding the semantic element. Furthermore, the feature-based decoding will be discussed in section 5.3 could be employed to provide morphological oracles that are learned separately.

Another possible direction is to devise a set of real-world tasks similar to those of Guu et al. (2020). The researchers showed to what degree a self-contained model can handle such tasks compared with a retrieval LM that has access to external data. They also showed that some settings could benefit from different training and decoding datasets to yield high performance. The outcome is likely to vary from one task to another, but perhaps quantifying the degree of novel returns[17] retrieved through a continuous model would highlight another possible strength (OOV will be explored in Chapter 4).

Recently, a new objective function was proposed (Choi et al., 2020). This new objective function specifically targets prediction diversity and is likely to be more competitive than the `UL` (Welleck et al., 2020) objective implemented in this chapter. This is another possible future direction. This chapter proposed another direction that goes beyond diversity. While the prediction of diversity is central to this chapter, continuous prediction was found to scale well across users, as shown in Tables 3.7 and 3.8. In Chapter 4, I will show that this approach also allows for growth and continual learning due to its architecture, making for more sustainable models.

3.6 Conclusion

In this chapter I proposed a retrieval-based architecture of an adversarial learning nature and evaluated it on three main axes: prediction accuracy, prediction diversity,

[17]In this experiment, this question translates to OOV retrievals.

and complexity costs. The experiments were conducted across a news-based domain and a biomedical domain. Several models were compared: a basic categorical model with MLE, a categorical sub-word model, a categorical diversity-enhancing model, a continuous model, and my proposed GAN-based continuous approach. A trade-off between prediction accuracy and prediction diversity was observed; this trade-off was found in Luo et al.'s (2020) work on image captioning as well. I showed that the high-accuracy performance attributed to the categorical models can be attributed to the narrow list of classes predicted, which encourages the model to be less exploratory[18] and more exploitative of given learned examples. On the other hand, the continuous prediction approaches scored higher on diversity, including with regard to rarely seen training examples. Computational costs were considered as well, and shorter runtimes, FLOps, and memory were found across users of the retrieval based-approaches.

In the context of AAC, the language model proposed in this chapter allows a user to retrieve a richer number of icons/words contributing to a more expressive message at relatively lower costs. Predicting a more diverse set of words may indirectly contribute to engagement and is a way to develop more trust in the system. As there are additional axes on which to compare these models, in the following chapter I focus on the architectural potential of the retrieval models presented here.

[18]Here I borrow reinforcement learning terminology.

Chapter 4

Incremental Domain Adaptation in Language Models

"The measure of intelligence is the ability to change"

Albert Einstein

4.1 Introduction

This chapter is aimed at exploring the promise continuous prediction models hold. In this chapter I explore how to move towards continual learning models for language modeling and evaluate desirable characteristics for this task. Following Parisi et al. (2019), continual learning agents are models that can continue to learn over time and adapt to *new* distributions while retaining the *previously* learned distributions. In this chapter I develop a continual-learning language-model agent that learns to adapt to different domains for text-entry purposes, where the user's language may change over time or change as a function of their conversation partner or situation, as shown by Müller-Frommeyer et al. (2019), and by Tannen et al. (2005) and Biber et al. (2009). Specifically, the categorical approach is compared to continuous prediction models. While the categorical approach is shown to outperform the continuous approach on the main metrics, note that the meta-metrics in this chapter, which define the optimal criteria for continual learning agents, arguably play a more important role when considering both model approaches for the purpose of continual-learning language-models.

4.2 Related Work

4.2.1 Domain adaptation in NLP

Pratt (1993) introduced the concept of transfer learning, a network learned on source data, subsequently rescales its weights on target data. Following the definitions of Pan et al. (2009), domain adaptation, which is the main focus of this chapter, is one type of transfer learning. In domain adaptation, labeled data exist for the source domain, which is the initial domain a model often is trained on, while one or

more additional domains for the same task are to be learned as well (regardless of whether there are labels for the following domains). Even though domain adaptation was a field of research prior to the following work, it was formalized by Daume III et al. (2006), where training and testing are based on two different but related distributions of data. The initial training is based on an out-of-domain distribution and then optimized for the in-domain data:

$$\hat{\theta} = \arg\max_{\theta} p(\theta|\arg\max_{\eta} p(\eta)p(D^o|\eta))p(D^i|\eta) \tag{4.1}$$

Eq. 4.1 defines domain adaptation based on Daume III et al. (2006), where a model's parameters (θ) are learned based on the hyper-parameters (η) and the out-of-domain distribution D^o but then are refined to maximize the likelihood of the in-domain distribution. There is also earlier work that was done in the field. Woodland et al. (1999) presented language model interpolation in an HTK version that showed decreased perplexity when combining a category-based language model and an n-gram model (see Eq. 2.8). The category-based model is computed in Eq. 4.2 (based on Kneser et al. (1993)):

$$p_{category}(w|w_{i-1}) = p(w|C(w))p(C(w)|C(w_{i-1})) \tag{4.2}$$

Another way to mix the distribution, known as count-merging, was shown by Bacchiani et al. (2003) and Ljolje et al. (2000) and appears in Eq. 4.3:

$$p(w_i|w_1^{i-1}) = p(w_i|h) = \frac{\alpha C_1(hw_i) + \beta C_2(hw_i)}{\alpha C_1(h) + \beta C_2(h)} \tag{4.3}$$

where C_x are counts from a particular domain, and α, β are scalars. Bacchiani et al. (2003) reported that both unsupervised approaches were found to improve performance, given limited in-domain data. Federico (1999) proposed another interpretation of adaptation called minimum discrimination information. This interpretation involves combining both the in-domain and out-of-domain datasets, as shown in Eq. 4.4:

$$p(w_i|h) = \frac{p_o(w_i|h)\alpha(w_i)}{\sum p_o(w_i'|h)\alpha(w_i')} \tag{4.4}$$

where

$$\alpha(w_i) = \frac{p_i(w_i)}{p_o(w_i)} \tag{4.5}$$

The $\alpha(w)$ component generates a distribution based on both unigram probabilities of the distributions. Gretter et al. (2001) conducted an online adaptation based on a confidence measure such that the probability generated is based on both an n-gram lattice and a current hypothesis. The online adaptation's tokens are considered only if they cross a likelihood threshold. Blitzer et al. (2006) proposed a way to recover wrong labels given a news source containing both (X, y) of words and labels (parts-of-speech), and a biomedical target domain containing words X and tags based on the news domain-trained tagger. The goal was to recover corrupted tags resulting from the domain mismatch. Adapting to the new domain and recovering the biomedical tags were completed through structural correspondence where the

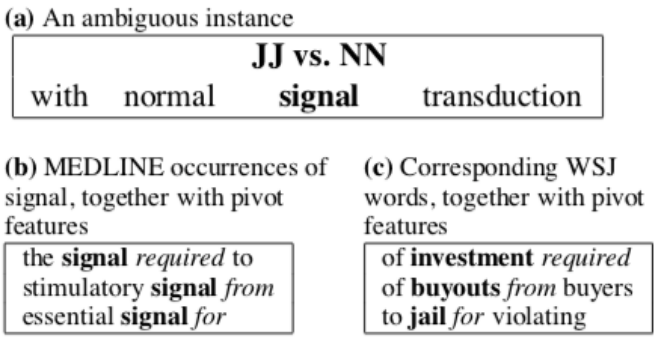


Figure 4.1: Recovering labels through structural correspondence (from the example given in Blitzer et al. (2006)).

tag's word was searched and extracted with a set of 'pivot' words (that function similarly across these domains). This helped recover the label via the patterns found in the new labeled corpus. Figure 4.1 illustrates how to recover a label from a corrupted tagger for a biomedical text by anchoring the target word in pivot words and searching for the likely label given the labeled news corpora.

Johnson et al. (2005) devised an approach to project several domains into a low-dimensional shared domain. This is done by minimizing the loss in Eq. 4.6 in a form of co-training:

$$[\theta, \hat{f}_{1:l}] = \arg\min_{\theta, f_l} \sum_{l=1}^{m} (\sum_{i=1}^{n_l} \frac{L(f_l(\theta, X_i^l), y_i^l)}{n_l} + r(f_l)) \tag{4.6}$$

where every classification data of X, y learns a projection to a lower-dimensional space, together with a regularization function to prevent over-fitting or to promote generalization. Johnson et al. (2005) also proposed a semi-supervised approach where one can recover labels when only a few are available for a particular classification I will refer to as A. The recovery can take place if A is co-trained with other classification problems, leading to the generation of a rich space that can generalize A from only a few examples of it. Daumé III (2009) proposed a way to feed features into a classifier that indicates whether an input feature is common to both domains or specific to a source or a target domain. The feature representation is based on hot decoding (see section 2.4.3) with designated positions for each domain. Kim et al. (2016) took another step towards applying the work of both Daumé III (2009) and Johnson et al. (2005) to a neural network and generalized to output labels beyond binary classification. Kim et al. (2016) showed that training K LSTMs with a generic LSTM on all domain training data was superior to the alternatives (including Daumé III's (2009) model). Following Daumé III, Sun et al. (2015) proposed an unsupervised adaptation where both the source and the target domains' distributions are aligned via the alignment of their co-variance matrices as shown in Eq. 4.7:

$$\min_A \parallel C_S - C_T \parallel = \min_A \parallel A^\top C_S A - C_T \parallel \tag{4.7}$$

Pan et al. (2010) proposed a transfer component analysis (TSA) where 'transfer' components are used to map from one domain to another, while maintaining the variance of the data and reducing the distance between the two distributions. Pan et

al. (2010) demonstrated improved performance over a text classification task. Glorot et al. (2011) presented a deep learning architecture for unsupervised learning by projecting the input into a lower-dimensional space for improved generalization across domains. Glorot et al. employed a neural approach involving a stacked autoencoder architecture and showed superior performance in a sentiment classification task of 12 Amazon product domain shifts.

Fine-tuning is based on *rescaling* weights on a target domain, where the available target data is *scarce* (Dauphin et al., 2012).[1] This is a more general type of transfer learning compared to domain adaptation, as it is not restricted to the original task it was trained on with the source domain. Recently fine-tuning has become a common means used by various transformers (Devlin et al., 2018; Peters et al., 2018b; Liu et al., 2019) over various downstream tasks. Howard et al. (2018) introduced the universal language model fine-tuning (ULMfit), which is an LSTM-based model. ULMfit emphasizes the practice of fine-tuning over a language model for attaining task-specific models, instead of training these models from scratch. The authors also introduced a layer-dependent learning rate for weight updates during training, as shown in Eq.4.8:

$$\theta_t^l = \theta_{t-1}^l - \eta l \cdot \nabla_{\theta^l} L(\theta) \tag{4.8}$$

where the assumption is that the different layers maintain different features (lower layers were found to capture syntactic features, while higher layers were found to capture semantic features). Therefore, learning preserves the general linguistic features while the model remains flexible to learn others, which is the justification for applying different learning rates. Following transformer research, Dai et al. (2015) showed an effective approach for neural language model adaptation. This approach begins with training a general model, which is then adapted to a smaller dataset. Dai et al.'s (2015) approach had superior performances across various domains. Gururangan et al. (2020) extensively evaluated neural adaptation through fine-tuning both on a textual domain and on task-based text. Gururangan et al. pointed out the different aspects of adaptation, including adapting to the pattern of a text and adapting to a domain vocabulary. They demonstrated that while a news corpus presented overlapped word-types with a reviews corpus, textual patterns did not overlap word-types. Moreover, given limited computing resources, they proposed augmenting a small-sized domain with quality data by finding the nearest-neighbor sentence embeddings from larger in-domain data, achieving very close performance compared to training a full-size domain. Lee et al. (2020) presented bioBert, based a BERT model architecture which was pre-trained on the biomedical domain and then fine-tuned using three biomedical domain downstream tasks. The main reason for training bioBert was the vocabulary difference and requirement of training from scratch, such that fine-tuning can be within a sub-domain of bioBert, rather than a BERT fine-tuned on a completely different domain. Several more fine-tuned BERTs were released (including biomedical (Lee et al., 2020), medical (Rasmy et al., 2020), clinical (Alsentzer et al., 2019), and scientific (Beltagy et al., 2019) domains, among others, as mentioned in section 2.3.4). In this chapter I will evaluate language model adaptation, and in particular continual learning of language models.

[1]The target data scarcity setting was also assumed in the studies presented earlier in this chapter, though most of those studies revolved around adaptation. This part of the chapter reviews mostly broader transfer learning works. Adaptation does not assume the scarcity of target domain by definition.

4.2.2 Continual learning

'The ability to continually learn over time by accommodating new knowledge while retaining previously learned experiences is referred to as continual or lifelong learning,' according to Parisi et al. (2019). The fine-grained distinction is that while lifelong learning (LL) considers a broad range of models and was introduced by Thrun et al. (1995), continual learning (CL) is often referred to a neural-based model (Chen et al., 2018).

One of the biggest challenges of LL, and CL in particular, has to do with the catastrophic forgetting (Parisi et al., 2019) phenomenon. Catastrophic forgetting/inference (CF) was already observed by McCloskey et al. (1989), who described the problem of degradation of performance that results from learning a sequence of tasks that specifically hurts *early-learned* tasks. This problem describes the tension between plasticity, which is the ability to adapt and increase the synaptic strength following the Hebbian theory (Hebb, 1949), and the stability of the knowledge acquired through homeostatic mechanisms observed in biological systems, as described in Mermillod et al. (2013). In this regard, a subtlety defining CL is that it is a form of transfer learning where the knowledge gained from one task is expected to be used for the same task, rather than being re-purposed for a different one as in a typical instance of transfer learning. Therefore, CL evaluation considers previously trained knowledge/tasks as well.

Ven et al. (2019) characterized three scenarios for continual learning: Task-IL (incremental learning) in which models are *informed* which task they are required to solve (could be multi-headed the more tasks are presented); Domain-IL in which a model solves the task at hand (and is not instructed what task to solve); and Class-IL in which a model solves the task and is required in addition to infer the task ID. The problem shown in this chapter follows Domain-IL, as the model is required to predict an outcome and is not provided with the task ID but is also not required to predict it.

Ven et al. (2019) provided a review of different approaches to overcome CF. I will review the main ways to overcome CF in neural models. Kirkpatrick et al. (2017) proposed the elastic weight consolidation (EWC) model for supervised and reinforcement learning scenarios. The approach consists of a quadratic penalty between the parameters for the old and the new tasks that slows down the learning for task-relevant weights coding to retain previously learned knowledge. Since the posterior distribution to compute the model's parameters based on the current and previous domains is intractable,

$$p(\theta|D) = \frac{p(D_B|\theta)p(\theta|D_A)}{D_B}. \tag{4.9}$$

The parameters are approximated through the loss presented in Eq. 4.10, with goal of penalizing parameter changes that are assumed to be meaningful to the old domains through regularization.

$$L(\theta) = L_B(\theta) + \sum_i \frac{\lambda}{2} F(\theta_i - \theta^*_{A,i})^2 \tag{4.10}$$

Eq. 4.10 contains the loss computed for the current domain together with a regularization component applied to retain Domain A's knowledge. Another regularization-based approach called learning without forgetting (LwF) was proposed by Li et al.

(2017) where parameters of the old tasks, shared parameters, and new task parameters θ_o, θ_s, and θ_n are employed for learning. The loss computed from the new task is regularized through the old task and the shared model's parameters. Shin et al. (2017) proposed a replay mechanism where examples from previous tasks are replayed indirectly. Training a model with the new data and then replaying old data was proposed by Robins (1993), under the assumption that the replay would address the forgetfulness of a model. The deep generative replay approach of Shin et al. (2017) is based on training an external 'solver' that imitates the old task model and produces $\hat{y}$s (soft targets) instead of recording the original targets (hard targets). Then in order to replay examples to the 'scholar' (the main model), a pair of $(x, \hat{y})$ is introduced to accommodate the old tasks, together with the typical training on the new task, where hard targets are introduced. Using a 'solver' may save memory demands of data (specifically the labels) associated with previous tasks and has been shown to improve performance when trained together with LwF, compared to just LwF. The approach employed in this chapter is based on replay described by Shin et al. (2017), where old examples are presented again during the next training. More technical details are described in section 4.4.5.

4.3 Continual Learning of Language Models

4.3.1 Problem definition

In this chapter I examine the language models architectures introduced in Chapter 3 for performing continual learning. Ideally after deploying a language model, it should continue to adapt to user patterns, while also recall previously learned patterns. Therefore, the questions asked in this part are about continual learning qualities a language model maintains. Evaluating the models is done through both meta-metrics that will be described in section 4.4.6 as well as basic performance tests that were introduced in section 3.3.6. In what follows I formalize the problem and elaborate on the methods.

4.3.2 Problem formalization

I denote domain A distribution, which is the first domain being trained, as a subset of N examples from $D^A \sim \mathcal{D}^A$, and domain B distribution, which is the second domain being trained, as a subset of M examples from $D^B \sim \mathcal{D}^B$. $M < N$ as this condition reflects a scenario where gradual learning of a user is attained. The goal is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ with a low expected loss with regard to $\mathcal{D}^B$, while retaining a relatively low loss for $\mathcal{D}^A$ (under a reduced catastrophic forgetting assumption). $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^d$ represent up to several and a single embedding vectors respectively such that each of the vectors corresponds to a single word-type.

4.4 Methods

I conduct a series of word prediction experiments in which the performance was comparing a traditional categorical prediction model, a simple continuous model, and

the proposed GAN-based continuous approach in an incremental domain adaptation setting.

4.4.1 Datasets

Similar to section 3.3.1, I performed experiments on two corpora: newswire text from the New York Times section of the Annotated English Gigaword corpus (Ferraro et al., 2018, LDC2018T20) (NYT) and full-text biomedical journal articles from the open-access subset of PubMed Central (Beck, 2010) (PMC). NYT and PMC were chosen to be $\mathcal{D}^A$, $\mathcal{D}^B$, respectively. These domains were chosen as they have reduced overlap of vocabulary, simulating a stress-test under adaptation settings. There were 514k types and 731M tokens in the NYT corpus and 861k types and 453M tokens in the PMC corpus. The word-type overlap is 116k, and the token overlap is 270M. Each vocabulary type observed fewer than four times in each training set was replaced with an *unk* symbol, and each numeric token was replaced with a class token. The dataset split is 60/20/20 train/dev/test. All models underwent a single pass on the train set (due to complexity considerations), during which the most optimal model was chosen. The early stopping (Yao et al., 2007) approach was taken when evaluating the models' losses on dev set.

4.4.2 Models

The categorical and continuous models employed in this set of experiments followed the architecture described in section 3.3.2, the GAN-based model followed the architecture described and in section 3.4.3. However, these are not the same models; they were especially trained for this set of experiments using the same architecture but different conditions.

4.4.3 Embeddings

I trained embedding spaces from scratch on both domains' datasets. All models were fed with embeddings as input to make them as architecturally similar as possible. This set of experiments was conducted on the dimensionality of 50 for two reasons. First, doing so reduced complexity across all models, and second, most of the diverse models in Chapter 3 were found in that dimensionality. I leave experiments with higher dimensionalities for future work. Following the results that will be shown in section 5.4[2], I custom-trained Word2Vec (Mikolov et al., 2013b)[3] on both the NYT and PMC datasets, making for a single shared space. The motivation for the shared space is described in the following section. A total of 1.27 M entries covering both domains were represented on the space.

4.4.4 Decoding

Similar to section 3.3.4, I performed a **simple decoding** where I applied a nearest-neighbor search to match a predicted vector with an embedding vector representation

[2]Even though Chapter 4 is presented prior to Chapter 5, experiments of the latter took place earlier in the process

[3]I used the implementation by Řehůřek et al. (2010).

that would recover the predicted word. I applied maximum inner product search (MIPS) (Shrivastava et al., 2014) `Annoy` library (Bernhardsson, 2018) with angular distance.[4] During the decoding process of the test-set trained on $\mathcal{D}^A$, unique $\mathcal{D}^B$ entries (i.e., those that were not shared across domains) were masked in the shared embedding space to simulate a space that initially represented a single domain $\mathcal{D}^A$. Over time (i.e., in the second training), the new domain $\mathcal{D}^B$ was unmasked. This new domain contained additional words from which the model could decode. The decoding process did not influence the training. The training was affected by the loss with regard to a domain's prediction rather than how well each prediction mapped to a particular embedding term. This description is illustrated in Figure 4.2. The main purpose of the masking was to *simulate* real settings where new words are learned over time.[5]

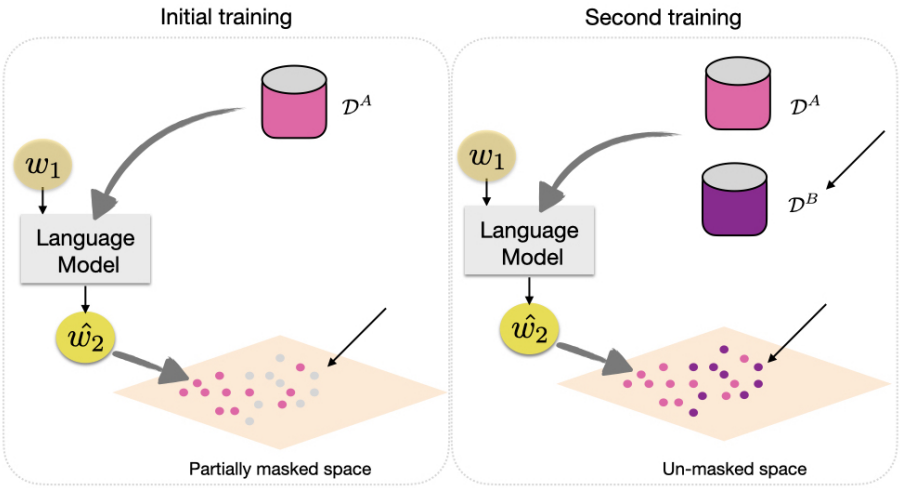


Figure 4.2: Decoding protocols: initially a $\mathcal{D}^B$-masked embedding space and then an unmasked embedding space.

4.4.5 Experiments

In this chapter I focus on the single task of incremental domain adaptation, which involves the continual learning of domains for a word prediction task. The type of evaluation conducted in this chapter is motivated by the task of continual learning, which goes beyond domain adaptation and measures a model's performance on both the recently and previously learned domains.

In my experiment to overcome catastrophic forgetting, I *replayed* examples similar to Shin et al. (2017) in the sense that a subset of training examples from the previously trained domain was replayed $n < N$ during the training of $\mathcal{D}^B$. These examples were replayed directly rather than through training a solver as described in Shin et al. (2017). This choice was made since in this experiment, the goal was not to evaluate various approaches to overcome catastrophic forgetting. Though I may consider various approaches in future work, in this experiment I applied the same approach across models. Another, more important, reason for not training a

[4] https://github.com/spotify/annoy

[5] Please see the discussion section in this chapter for further information.

'solver' and simplifying the replay process is that computationally, especially for the categorical approach, training a 'solver' to store all probability distributions even for a subset of n examples is very costly since the task is applied under a large vocabulary setting. Replaying previous data directly therefore assumes limited access to the data, which may simulate an optimal way to 'recall' previously seen data.

Table 4.1 shows the number of samples for each type of experiment. Note that in the replay condition, both PMC and a subset of NYT were trained together.

set	NYT	PMC	NYT_{subset}
train	22.5M	11.8M	5.5M
valid	7.5M	3.9M	1.8M
test	7.5M	3.9M	–

Table 4.1: Number of sentences per set

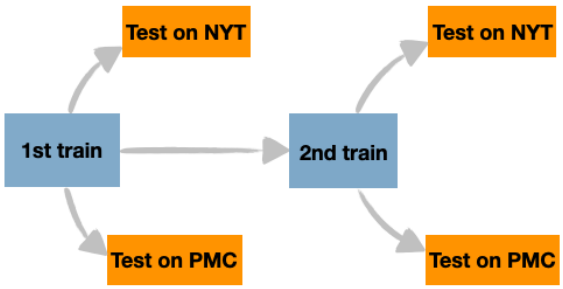


Figure 4.3: Training procedure of evaluation and testing for the first and second trainings.

Figure 4.3 illustrates the training procedure for the adaptation experiments.I conducted three types of experiments. The main condition involved training first on $\mathcal{D}^A$ (N examples) and then with replay such that $\mathcal{D}^B$ was trained together with a subset of $\mathcal{D}^A$ replaying the previous domain ($M + n$ examples). To illustrate the impact of replay on the models, a second condition was devised as a baseline, wherein no replay took place in the second training. In other words, the first training was similar to replay where the model was trained on $\mathcal{D}^A$ (N examples), and the second training was not similar, as the model was trained on $\mathcal{D}^B$ only (M examples). The third condition involved training on both $\mathcal{D}^A$ and $\mathcal{D}^B$ at once ($N + M$ examples) as a potential upper-bound baseline, though this may not have presented a realistic setting under a continual learning condition.

These conditions were applied to all three models: `ctg`, `c`, and `G`. For experiments that required a second training, the `ctg` model was divided into two separate experiments. In one case, the original model $\texttt{ctg}_1$ maintained its original final layer, where the vocabulary corresponded to the first domain V^A. In the other case, the final layer was replaced with a new layer that represented the final vocabulary trained (which varied across conditions). In the experiment where `ctg` was trained on both domains, the training vocabularies of both the V^A and V^B domains were represented.

Table 4.2 shows the models' results for the first training. Note there are two types of outcomes: $\texttt{model}_1$ and $\texttt{model}_{both}$. The former was used in a second round whose

results appear in Table 4.3 under 'Model.' The latter was when both domains were trained at once, which was a single-round training condition referred to as *both*. The

Model	1^{st} training	$(\mathcal{D}^A)$	*both*	$(\mathcal{D}^A \cup \mathcal{D}^B)$
`ctg`	$\texttt{ctg}_1$	(N)	$\texttt{ctg}_{both}$	$(N+M)$
`c`	$\texttt{c}_1$	(N)	$\texttt{c}_{both}$	$(N+M)$
`G`	$\texttt{G}_1$	(N)	$\texttt{G}_{both}$	$(N+M)$

Table 4.2: First-round training experiment.

models under '1^{st} training' from Table 4.2 were the starting models for the second round, where a `ctg` model was split into two: either its last layer was kept and referred to as $\texttt{ctg}_1^k$, or its last layer was replaced, in which case it was referred to as $\texttt{ctg}_1^r$.

Model	*fine-tune*	$(\mathcal{D}^B)$	*replay*	$(\mathcal{D}^B \cup (d \subset \mathcal{D}^A))$
$\texttt{ctg}_1^k$	$\texttt{ctg}_{ft}^k$	(M)	$\texttt{ctg}_{rp}^k$	$(M+n)$
$\texttt{ctg}_1^r$	$\texttt{ctg}_{ft}^r$	(M)	$\texttt{ctg}_{rp}^r$	$(M+n)$
$\texttt{c}_1$	$\texttt{c}_{ft}$	(M)	$\texttt{c}_{rp}$	$(M+n)$
$\texttt{G}_1$	$\texttt{G}_{ft}$	(M)	$\texttt{G}_{rp}$	$(M+n)$

Table 4.3: Second-round training experiment.

The results shown in Table 4.3 are those of models that were trained with a *replay* condition and were called $\texttt{model}_{rp}$ or are those of models that were only trained on the second domain, as in the case of a simple *fine-tune* condition, and were called $\texttt{model}_{ft}$.

Motivation for applying *keep* and *replace* to the `ctg` model

The motivation for setting up two conditions, *keep* or *replace*, was to exhaust all possible configurations of a categorical model. Since a categorical model's vocabulary is based on the categories the model is exposed to through the initial training, it is limited when training the model again. One can either maintain the original layer, though then it cannot predict new terms from the second training. Or alternatively, if the original layer is replaced with a new layer it cannot represent the types of categories formed based on the previous vocabulary (that are not found in the second training).

This is also the reason for the *replace* condition, where the final layer of the old model is replaced with a new one, such that the model represents the vocabulary types of the current training. In *both* conditions, both domains' vocabularies are represented in the final layer of the categorical model, as both datasets are shown during the single training the model undergoes. *keep*,*replace* are important to understanding the results discussed in the following section. Note that this is not the case with the continuous models, as they do not store types in their architecture; therefore, the second training does not require architectural modification of the model.

Technicalities for `ctg` training

The conditions of *keep* and *replace* required particular accommodations for the training procedures. During the second training, a model was not trained to recognize a type for which it had no representation, as doing so may not have sent back (during the backprop) an effective learning signal. This means that the training examples were not equal across *keep* categorical models and the rest of the models.

An alternative may have been to assign an additional *unk* class (there was already an *unk* class for words with fewer than four occurrences) and have the model learn to predict it once an unrepresented class occurs. However, doing so may bias the model to the *unk* category, thereby harming the learning. Both ways of overcoming unrepresented classes have flaws and represent trade-offs associated with different aspects of the process. Ideally, I would have experimented with both; however, I chose to avoid training on unrepresented terms, as biasing the model towards a particular class seemed to generate a weaker model.

4.4.6 Metrics

The goal of this study was to evaluate the effectiveness of a model when considering aspects of long-term adaptation (i.e., continual learning). I define an optimal model as a model whose outcome allows for:

- Adapting to *new* distributional patterns of the recently trained dataset.
- Retaining memory of *previously* seen patterns (co-existence).
- Allowing for words/concepts to be introduced and predicted *over time*.
- Requiring *minimal* modifications/costs for a continual process.

These meta-metrics are both evaluated and discussed through the quantitative metrics below. I performed a set of experiments in which the various models and settings described in Sections 4.4.2 and 4.4.5 were trained and put through a word prediction task. The models' goal was to predict the next word in the sequence, given a history; I refer to this as a 'hit'. Because I was predicting in a continuous embedding space, it may have been that various neighbors of the 1-best word were relevant or appropriate; as such, I also considered the neighborhood comprised of the ten closest words to the predicted vector. Evaluation was measured through the same metrics I presented in Chapter 3 and were as follows:

- $top_1(top_{10})$ - percentage of trials that constituted a 'hit' (i.e., target is within top 1/10 neighbor(s)) vs. a 'miss'
- $T_1(T_{10})$ - number of unique *types* that the model correctly predicted (i.e., that appeared in at least one $top_{1/10}$ hit).

4.5 Results

4.5.1 Performance following the second training

One of the main questions concerning continual learning is how well the models performed on the previously learned domain, NYC. Table 4.4 provides results that help answer that question. The hit rates of the continuous approaches were lower than those of the categorical alternatives across the board. However, for T_1, the categorical approaches were not always found to be the most diverse. In fact, `G` models were more diverse under the *ft* and *rp* conditions, while under-performing under the *both* condition. For the NYT domain, the *both* condition was not found to provide an upper-bound baseline for the continuous models, while it was more optimally performing for `ctg`. The *ft* condition was not found to be any different than the *rp* condition for the continuous approaches. Among the continuous models, `G` outperformed `c` across conditions. Another question is how well the models predicted

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{ctg}^k_{rp}$	19.93 (46.13)	5,767	(10,196)
$\texttt{ctg}^r_{rp}$	19.65 (45.85)	4,806	(8,460)
$\texttt{c}_{rp}$	12.42 (27.95)	1,557	(5,000)
$\texttt{G}_{rp}$	12.47 (28.38)	6,919	(18,751)
$\texttt{ctg}^k_{ft}$	20.78 (47.96)	8,316	(13,755)
$\texttt{ctg}^r_{ft}$	13.85 (34.42)	2,023	(3,819)
$\texttt{c}_{ft}$	12.32 (28.23)	1,563	(5,267)
$\texttt{G}_{ft}$	12.51 (28.28)	6,930	(18,769)
$\texttt{ctg}_{both}$	19.72 (46.16)	7,809	(12,873)
$\texttt{c}_{both}$	11.85 (26.94)	2,087	(6,456)
$\texttt{G}_{both}$	12.05 (27.05)	3,828	(11,315)

Table 4.4: Second training evaluated on NYT test.

on the currently learned domain. Table 4.5 gives the results for the current domain, PMC. Here, too, the continuous approaches were sub-par to the categorical models on accuracy, as well as diversity. In this test-set, the *both* condition was shown to provide an upper bound for all methods in most respects. The *rp* condition was shown to improve the `c` and $\texttt{ctg}^k$ models' performances. On the other hand, *ft* was a more optimal condition for the replaced layer of the categorical model $\texttt{ctg}^r$. `c` performed the worst across models. Finally, while `G` had a low level of accuracy, it was similar to or more diverse in top_{10} compared to `ctg`.

Discussion

The categorical approaches outperformed the continuous models. However, while $\texttt{ctg}_{rp}$ presented similar performances, each was stronger on a different dataset for expected reasons. $\texttt{ctg}^k_{rp}$ was stronger on NYT, which may have to do with the fact that the original layer of the NYT vocabulary was kept. $\texttt{ctg}^r_{rp}$ was more accurate and diverse on PMC, which is perhaps due to the replacement of the layer expressing the new vocabulary of PMC. A possible reason that the continuous approaches underperformed may be due to the model conducting the prediction on a *large* embedding space where both domains' vocabularies were accessible (see section 4.4.4). This is different than the experiments shown the Chapter 3, where decoding was employed

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{ctg}^k_{rp}$	21.83 (46.58)	4,524	(8,028)
$\texttt{ctg}^r_{rp}$	22.04 (47.99)	5,107	(9,108)
$\texttt{c}_{rp}$	11.97 (26.24)	535	(1,959)
$\texttt{G}_{rp}$	7.65 (22.95)	2,879	(9,741)
$\texttt{ctg}^k_{ft}$	15.44 (35.62)	2,741	(5,860)
$\texttt{ctg}^r_{ft}$	23.48 (49.45)	4,404	(7,322)
$\texttt{c}_{ft}$	7.22 (21.47)	424	(1,640)
$\texttt{G}_{ft}$	7.66 (22.99)	2,875	(9,770)
$\texttt{ctg}_{both}$	22.03 (47.76)	7,368	(12,211)
$\texttt{c}_{both}$	13.63 (28.66)	2,655	(8,702)
$\texttt{G}_{both}$	12.36 (26.65)	5,680	(18,681)

Table 4.5: Second training evaluated on PMC test.

Table 4.6: Relative gain of second training w.r.t first training evaluated on NYT testset (subtraction of old from new).

model	Δtop_1	Δtop_{10}	ΔT_1	ΔT_{10}
$\texttt{ctg}^k_{rp}$	−1.10	−1.61	−401	−565
$\texttt{ctg}^r_{rp}$	−1.38	−1.88	−1362	−2301
$\texttt{c}_{rp}$	−0.08	−0.89	+32	−113
$\texttt{G}_{rp}$	+0.24	−0.27	+887	+1,826

(a) Replay condition.

model	Δtop_1	Δtop_{10}	ΔT_1	ΔT_{10}
$\texttt{ctg}^k_{ft}$	−0.25	+0.22	+2,148	+2,994
$\texttt{ctg}^r_{ft}$	−7.18	−13.32	−4,145	−6,942
$\texttt{c}_{ft}$	−0.18	−0.61	+38	+154
$\texttt{G}_{ft}$	+0.28	−0.37	+898	+1,844

(b) Fine-tuning condition.

on a smaller vocabulary and a single domain. This embedding space may have required more memory capacity and 'orientation' in the space, which may be beyond the continuous models' capabilities. Even though the continuous models were underperforming on accuracy, `G` was very diverse (in particular, it outperformed the `ctg` models on T_{10}), demonstrating the strength of the continuous approach.

4.5.2 Relative gains after the second training

Tables 4.6a, and 4.6b describe the relative gain attained over the NYT dataset following the second training. I measure to what degree NYT patterns remained as I compare the first training to the second. The continuous models did not seem to benefit from the *rp* condition where part of the old data was replayed; their outcome for replay condition was similar to fine-tuning, though `G` became more diverse under both conditions. The categorical model $\texttt{ctg}^r$ benefited from replay, with a smaller decrease in performance than seen during fine-tuning. $\texttt{ctg}^k$ behaved the opposite: it became more diverse on the NYT types and was less hurt in accuracy.

Tables 4.7a, and 4.7b displaying the performance following when evaluated on PMC show more positive gains overall, which is expected as this table evaluates the gains following the training on the same domain (not the same datasets though). The `G` models again seem not to have exhibited any change under the *replay* condition compared with *fine-tuning*, but they did become more diverse under the *both*

Table 4.7: Relative gain of second training w.r.t first training evaluated on PMC testset (subtraction of old from new).

model	Δtop_1	Δtop_{10}	ΔT_1	ΔT_{10}
$\texttt{ctg}^k_{rp}$	+6.08	+11.06	+2,253	+3,170
$\texttt{ctg}^r_{rp}$	+6.29	+12.47	+2836	+4,250
$\texttt{c}_{rp}$	+0.04	−0.83	+65	+166
$\texttt{G}_{rp}$	−0.10	−0.04	+243	+864

(a) Replay condition.

model	Δtop_1	Δtop_{10}	ΔT_1	ΔT_{10}
$\texttt{ctg}^k_{ft}$	−0.31	+0.10	+470	+1,002
$\texttt{ctg}^r_{ft}$	+7.73	+13.93	+2,133	+2,464
$\texttt{c}_{ft}$	−4.53	−5.60	−46	−153
$\texttt{G}_{ft}$	−0.09	0.00	+239	+893

(b) Fine-tuning condition.

condition. `c` benefited slightly from the *replay* condition, and $\texttt{ctg}^k$ benefited greatly from the *replay* condition. $\texttt{ctg}^r$ was more accurate with *fine-tuning*, but it was more diverse on *replay* condition.

Discussion

`G` was not different under either condition across domains and was able to learn more terms within a domain under both *fine-tuning* and *replay*. Perhaps the *replay* technique employed for these experiments was not effective for `G`; however, new terms were added under both conditions at the same rate. `c` also was not different when evaluated on NYT. It could be that presenting more patterns through *replay* was helpful for improving over the shared terms (PMC's and NYT's overlapping terms), such that having the model encounter a word more frequently (in the initial and second training) strengthened its learning. The replay improved the performance of $\texttt{ctg}^r$ on NYT since the NYT replay-vocabulary helped with expressing the terms that were missing in fine-tuning. The replay hurt $\texttt{ctg}^k$ performance on NYT, but I am uncertain why. For PMC, the replay helped `c` to overcome CF. Replay was helpful for both $\texttt{ctg}^k$ on PMC, potentially because the replay dataset strengthened vocabulary items that were already observed mixed with new vocabulary, as opposed to training over a new domain completely on a layer trained on the previous domain. Replay could have been a bit distracting for $\texttt{ctg}^r$, as it spread the focus that could have been channeled to learn only the new vocabulary of PMC (as it was trained on NYT patterns as well).

4.5.3 Evaluating the recall of old terms and acquisition of new terms

The overall vocabulary set generated by the domains $\mathcal{D}^A$ and $\mathcal{D}^B$ is the unity of $V = V^A \cup V^B$. A vocabulary is defined as a unique set of types; therefore, unionizing two vocabularies results in the unique representation of types from both vocabularies' types. V can also be represented as $V = V_{D^A \setminus D^B} \cup V_{D^A \cap D^B} \cup V_{D^B \setminus D^A}$; that is, a union of the unique types in each domain, together with the overlapping types both domains share. In the following, I evaluate how well each model performed on the unique types of each domain, recalled the old types of NYT, and learned the new types of PMC following the second training.

Tables 4.8a and 4.8b describe how well the models correctly predicted old terms unique to the initial trained domain of NYT and how well the models predicted

Table 4.8: Recalling old terms and learning new terms.

model	Recall Old		Acquire New	
	top_1	T_1[%]	top_1	T_1[%]
$\texttt{ctg}^k_{rp}$	75.08	0.25	–	–
$\texttt{ctg}^r_{rp}$	3.41	0.10	22.91	0.16
$\texttt{c}_{rp}$	68.32	0.06	0.00	0.00
$\texttt{G}_{rp}$	64.78	0.39	0.05	0.03

(a) Replay condition.

model	Recall Old		Acquire New	
	top_1	T_1[%]	top_1	T_1[%]
$\texttt{ctg}^k_{ft}$	77.52	0.41	–	–
$\texttt{ctg}^r_{ft}$	–	–	14.02	0.2
$\texttt{c}_{ft}$	66.95	0.06	0.00	0.00
$\texttt{G}_{ft}$	64.8	0.39	0.07	0.03

(b) Fine-tuning condition.

unique terms of the PMC domain respectively. The results are shown for both the *replay* condition and the *fine-tuning* condition.

The categorical approach reached high levels of accuracy and demonstrated diverse predictions across all experiments. However, most of the categorical models were successful at either old terms' recollection or new terms' acquisition. The `c` model could only recall old terms, and `G` was capable of both recalling and acquiring new terms (though to a very small degree), and similar to `G`, `c` performed more optimally on recalling old terms than acquiring new ones. The reason $\texttt{ctg}^r_{rp}$ was able to predict old terms has to do with the replay mechanism, where *some* of the old terms are presented during the second training while the layer is replaced (i.e., $\texttt{model}^r$); this means that the new layer not only represents the new domain's vocabulary, but also represents some of the old domain (based on the subset of sentences' vocabulary sampled for replay). Otherwise, a categorical model cannot represent both classes of terms. This is the main difference between the continuous models and the categorical models. While the continuous approaches do not provide compelling results in these experiments, they open the door for representing both classes within the same model at no further cost, which is a key takeaway of this chapter.

A lack of access to previous data, therefore, can be harmful for the expressivity of a categorical model (before addressing the issue of catastrophic forgetting). It becomes a question of trade-offs: either increasing the complexity of the model by replaying previous data in future training, or not representing new terms and only adapting to new patterns as much as possible. In this regard, the continuous models can do both, as their complexity is not directly affected by the size of the vocabulary they learn.

A question to ask, though, is why the continuous models *seem* to perform well on the initial domain and worse on the new domain. I note that comparing performance across domains is risky. These distributions are not comparable since they are driven by different patterns and word frequencies. Nonetheless, the continuous approaches were able to model the biomedical distribution in isolation, as shown in section 3.4.3; therefore, expecting the continuous models to perform well on the unique terms of PMC is based on previous evidence[6] (lower performance with respect to NYT, but it was lower in the biomendical domain). One experiment to try is to swap the domains' order of training and train first on PMC and then on NYT to learn whether the performance varies depending on the order of training. It is also possible that the very thin model size of the continuous approaches may have not been

[6]Both NYT and PMC performed worse in this set of experiment relative to Chapter 3 experiments

conducive to performance. As a future direction one may compare performance over similar-complexity models instead of similar-architecture models, assuming that more parameters can be useful for learning and retaining more patterns.

4.5.4 Long-Tail prediction

In this section I investigate how well the models predicted as a function of frequency. Figures 4.4 and 4.5 capture the stratified models' performance on the T_1 and top_1 metrics on the NYT test-set after the second training. As expected, the categorical models were stronger than the continuous models both in types and tokens. While accuracy was low for all the GAN-based models, $\texttt{G}_{rp}$ and $\texttt{G}_{ft}$ were shown to be among the more diverse models (shown in light and dark purple). $\texttt{ctg}^k_{ft}$ was stronger (type and token prediction) than $\texttt{ctg}^k_{rp}$ on the high- and mid-bins, though the latter was more accurate on the low-bin (shown in light and dark pink). On the other hand, $\texttt{ctg}^r_{rp}$ performed more optimally over $\texttt{ctg}^r_{ft}$ (shown in dark and light brown). These findings shed more light on Tables 4.8a and 4.8b, and as mentioned, the decrease in $\texttt{ctg}^k_{rp}$ is likely due to generalization across a bigger training set under the *rp* condition. Figure 4.6 and 4.7 capture the stratified models' performance on the T_1

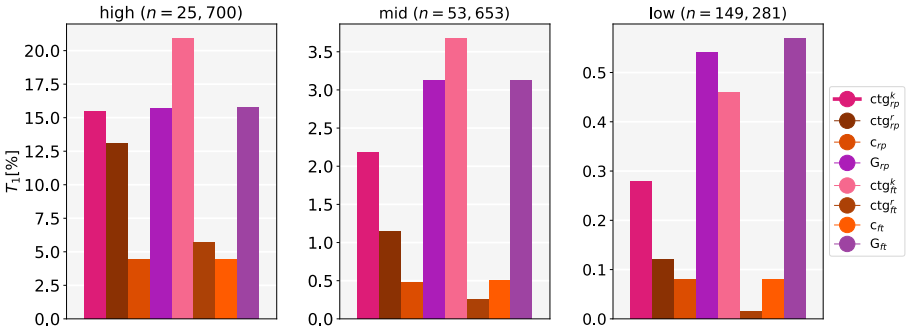


Figure 4.4: Type distribution on NYT test-set.

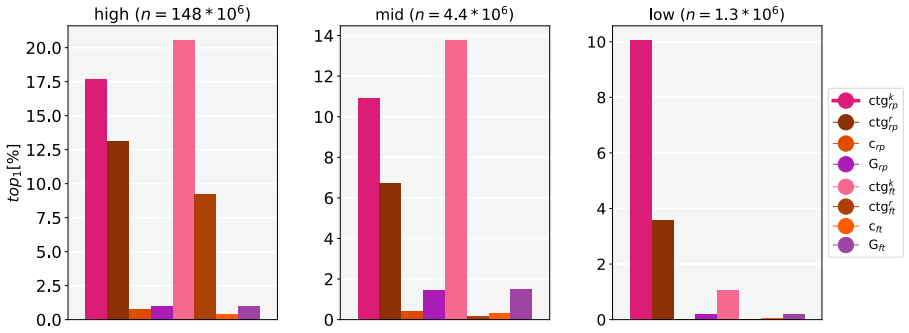


Figure 4.5: Token distribution on NYT test-set.

and top_1 metrics on the PMC test-set after the second training. The GAN-based models were relatively diverse but much less accurate. The $\texttt{c}$ models were neither diverse nor accurate. The strongest categorical models were $\texttt{ctg}^r_{rp}$ and $\texttt{ctg}^r_{ft}$, as they were likely to learn and represent the PMC vocabulary more reliably, though $\texttt{ctg}^k_{rp}$ was very accurate on the high- and mid-bins. While the categorical approaches were more diverse and more accurate, the GAN-based approach, despite its small

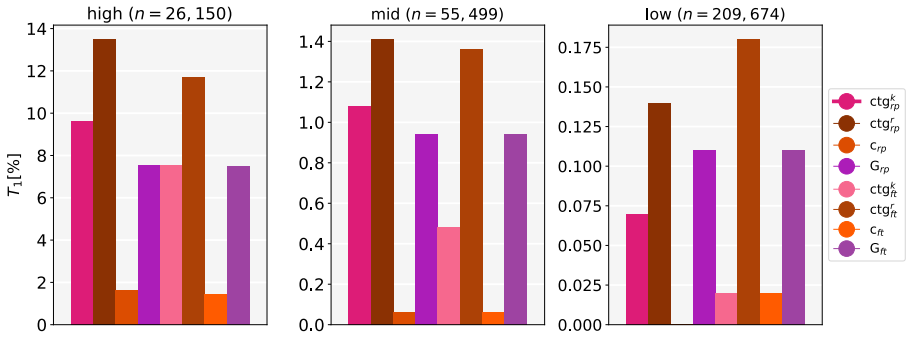


Figure 4.6: Type distribution on PMC test-set.

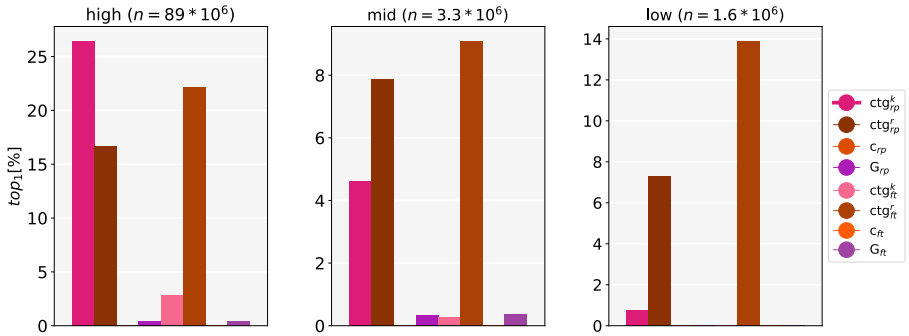


Figure 4.7: Token distribution on PMC test-set.

size, was relatively diverse as well, though its accuracy levels were low. The GAN-based models did not demonstrate the relative accuracy levels shown in Chapter 3 in the current set of experiments. As mentioned, the reason for this may have to do with the different settings of CL learning as well as the overall vocabulary size, which may require an increased model size to understand the degree to which these models can be accurate.

4.5.5 Were OOV words being predicted?

In this section I discuss the out-of-vocabulary (OOV) of a term using a strict definition that considers OOV to be any word-type that did not occur during *either* of the two training sessions. The OOV rate discussed in this section was evaluated on the model after the second training.

Tables 4.9a and 4.9b show the OOV rates for the different models. The strict

Table 4.9: Out-of-vocabulary (OOV) predictions (strict).

model	NYT		PMC	
	$top_1[\#]$	T_1	$top_1[\#]$	T_1
$\texttt{ctg}^k_{rp}$	–	–	–	–
$\texttt{ctg}^r_{rp}$	–	–	–	–
$\texttt{c}_{rp}$	31	7	101	11
$\texttt{G}_{rp}$	17	5	1,154	98

(a) Replay condition.

model	NYT		PMC	
	$top_1[\#]$	T_1	$top_1[\#]$	T_1
$\texttt{ctg}^k_{ft}$	–	–	–	–
$\texttt{ctg}^r_{ft}$	–	–	–	–
$\texttt{c}_{ft}$	0	0	91	14
$\texttt{G}_{ft}$	15	5	1,152	97

(b) Fine-tuning condition.

definition of OOV renders the categorical models impotent as they represent only items that were seen during training (in these experiments, only a subset of the training vocabulary was represented, subject to the *keep*, *replace* condition; this choice that was made based on computational considerations).

Tables 4.9a and 4.9b, therefore, capture any token that was not observed during training but may have occurred during testing or was found in dev. The reason the continuous approaches retrieved these terms is that the embedding spaces generated were based on the datasets of both domains and as a result contained words outside the train-sets' vocabularies. While the number of absolute tokens or types is not significant, I performed this experiment to demonstrate the potential of this approach, allowing to benefit from the fact the continuous models learn to represent a vector of a term, regardless of whether that particular term was seen (since these models learn locations on the space). In other words, by learning the locations of terms on the space, these models learn to some degree the semantics of a word, since we assume that these spaces reflect semantic relations at a non-negligible degree. These locations/representations (as much as they exist) in the space, and as much as the model was exposed to them, allow for the retrieval of unseen words, whose semantics/representation was learned for free, while training on (i.e. seeing) close vectors to the unseen word (provided that the unseen-during-training words provided found in the space). As long as the location was learned, decoding a prediction to a particular word is likely to be coherent irrespective of whether these vectors were learned during training.

The type of OOV prediction that characterizes the continuous models is different than the OOV prediction attributed to subword tokenization systems (Wu et al., 2016b; Sennrich et al., 2016). In the latter, predicting unseen subword combinations is technically enabled (since the building blocks are there), yet the semantics of these building blocks are often missing, since the way these subwords are generated has to do with frequency and not meaning (see Byte-pair (Gage, 1994) as an example of the algorithm to generate subwords by frequency of occurrence). Demonstrating the different OOV types may be easier with examples. A continuous model can find a word irrespective of its spelling, as long as the word occurs near a word vector the model was trained on. Therefore, if `increase` is found in the training set, `enhance` can be retrieved due to its similar vector representation. On the other hand, a subword categorical model is less likely to make such a prediction by guessing random letters that compose an unseen word. What it can do, however, is predict an OOV word by learning morpho-syntactic patterns. For instance, if the model was trained on the `I have learn ed` sequence, given an unseen history of `I have walk`, it would be able to produce `ed` and through that generate an unseen word `walked`. Predicting similarity may be potentially more useful than following syntactic patterns indicating a semantic, rather than syntactic, generalization.

Predicting semantically related out-of-vocabulary terms illustrates one type of generalization over the learned data, as it retrieves concepts that are correctly guessed and were not observed before. This section demonstrates the aforementioned trait of the continuous prediction approach and presents possibilities that cannot be attained in an obvious way through categorical models.

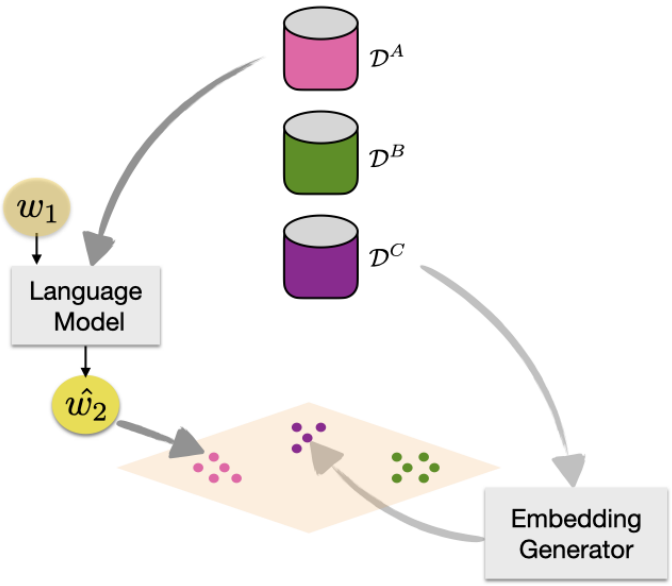


Figure 4.8: Towards a continual learning language model agent: adapting a pre-trained embedding space to new domains while enabling the continuous adaptation of a language model.

4.6 Future Directions

There are limitations to certain decisions made regarding the setting of the experiments that may be fruitful to address in future experiments. The first has to do with the continuous models compared in this experiment. While the continuous models' architecture produced performances on par with their categorical counterparts in Chapter 3, in the setting of this chapter it was not the case. A possible reason for the poor performance of the continuous models may be the small number of parameters of these models limiting the learning capacity. Therefore, as mentioned earlier, comparing same-complexity models instead of similar-architecture models may be another way to investigate the difference in this setting as the models are required to continue learning under large vocabulary setting over time. Another consideration is the degree to which the embedding space was organized in an optimal way for the model to learn the embeddings' locations or find alternative embedding spaces. For instance, Faruqui et al. (2014) projected two languages into the same space by employing canonical correlation analysis, which learns a joint representation of two domains by maximizing the correlation of the features when projected to a common subspace.[7] Exploring continuous models with advanced techniques to overcome CF would be another step towards continual language models such as learning without forgetting (Li et al., 2017) or elastic weight consolidation (Kirkpatrick et al., 2017). Furthermore, in order to make a continual learning agent fully automated, more research on adding new terms to a pre-trained embedding space is needed, specifically from other domains. Pinter et al. (2017) proposed an approach to generate representations of rare and unseen words based on their characters, given a pre-trained space, and Schick et al. (2019) extended it to capture a word's context and thereby refine the word's representation. Figure 4.8 illustrates this process. A continual

[7]However, this technique may not be less suitable for a continual adaptation setting.

learning language model agent learns new regions in the embedding space through model training, and new domains are added to the embedding space, preparing it for the next model's training.

4.7 Conclusion

In this chapter I presented a set of experiments comparing the categorical and the continuous approaches. The comparison showed that across many quantifiable evaluations of models' diversity and accuracy, the categorical models were superior to the continuous models. However, there are key areas, or meta-metrics, where the continuous models were found promising. These areas merit further consideration. First, word-types, even those that are mutually exclusive across domains, learned from various domains can be predicted within the same model (as shown in section 4.5.3, as there is no limit on the number of words a model can predict. Second, OOV words were also predicted without being presented through any training. OOV words are predicted with the retrieval-based approach 'for free' (no additional training), whereas the categorical models could only represent a *subset* of the training vocabulary they was exposed to (as shown in section 4.5.5). Third, the continuous approach requires no modifications to its architecture throughout its lifetime. The categorical approach is more involved, as it may require stripping off the final layer in order to accommodate new words, and its complexity is dependent on the number of words/classes it predicts (as shown in section 3.4.8). These three reasons are why I believe these models are promising for a sustainable continual-learning language-model agent. Linking this work to AAC, this chapter proposed to explore the setting where an agent learns and evolves with its user, where its previous expression habits are retained and their new ones are learned. The continuous models were shown to allow for an organic growth under this condition and were enabling not only new and old terms to be retrieved (even if to a low degree), they can present words that were not introduced by the user, even though the user might be interested in using them (OOV). Having said that more research is needed to fully develop such a system.

Chapter 5

Enhancing Continuous Prediction Models

5.1 Introduction

This exploratory chapter is aimed at further understanding and characterizing the continuous prediction models presented in Chapter 3: either by finding ways to augment these models or by investigating the conditions under which these models work optimally. The goal of this chapter is achieved in several ways. Chapter 3 presented the continuous approach models, and in particular, their unique ability to capture a richer variety of types during prediction. On the other hand, when these models were compared to a categorical alternative, the latter exhibited higher hit rates. In this chapter I examine several ways to better understand the mechanism of the continuous models. First, I explore a combination of both categorical and continuous models, seeking to achieve the advantages of both approaches. Next, I propose a new decoding mechanism in order to improve the continuous approach hit rates and further examine the models' mechanism of prediction. Third, I investigate the embedding spaces and their role in the process. Fourth, since *optimal* training protocols often require trial and error, I report the various loss functions considered throughout development. Finally, I conclude this section by describing my findings from investigating these models.

5.2 Can We Enjoy Both Worlds? On Combining Two Models

5.2.1 Introduction

Following the main results in Chapter 3, which showed that the current categorical models produce higher accuracy while the presented continuous approaches were richer in diversity, it is necessary to consider how well the approaches can perform together. In this section I discuss the degree of improvement over top_x and T_x metrics when combining the approaches. While there are various ways to combine two models, I chose to conduct model interpolation (also explained in section 2.3.2). Model interpolation is mainly done in one of two ways. The first is in a linear

fashion, as shown in Eq. 5.1:

$$P(w_t|w_{t-1-n}^{t-1}) = \sum_m \lambda_m P_m(w_t|w_{t-1-n}^{t-1}) \tag{5.1}$$

having a λ constraint of

$$\sum_m \lambda_m = 1, \lambda_m > 0 \tag{5.2}$$

where the model interpolation is essentially the outcome of a *union* of m model distributions, with global λs, where each λ describes the 'usefulness' of a model. The model in Eq. 5.1 was first introduced by Jelinek et al. (1980) as a way to provide 'broader' probability distributions for sparse data settings resulting from the unification process ($\text{data}_A \cup \text{data}_B$). It has prevailed in model adaptation for statistical language models (LMs) (Liu et al., 2008; Foster et al., 2010; Zhao et al., 2004). Interpolation was subsequently applied for combining an n-gram model with a neural model (Duh et al., 2013). Recent work (Grave et al., 2016) assumes that after a word appears once, it is likely to occur again; therefore, to augment a model's prediction, the authors proposed interpolating it with a neural cache. This cache stores recent hidden states with their target words, and given a current hidden state, it finds the word associated with the closest stored hidden state.

The second method of interpolation is the log-linear interpolation, as seen in Eq. 5.3:

$$P(w_t|w_{t-1-n}^{t-1}) = \frac{1}{Z_{w_{t-1-n}^{t-1}}} \exp \sum_m \lambda_m \log P_m(w_t|w_{t-1-n}^{t-1}) \tag{5.3}$$

In this interpolation, the exponent over the log of probabilities makes *similarities* among models to be more likely, which is a form of an *intersection* across model distributions Darroch et al. (1972) and Och et al. (2002) ($\text{data}_A \cap \text{data}_B$).

Perhaps the closest research to that presented in this section was introduced by Grave et al. (2016) and Khandelwal et al. (2019). In both of these studies, a softmax-based language model is combined with a contextual-embedding search model to produce an interpolated probability for word prediction. In Khandelwal et al. (2019), the contextual-embedding search model was a datastore containing key-value pairs of embedding-term entries, such that given a query-embedding during testing, the closest embedding from the stored dictionary would provide the query's nearest-neighbor distance distribution of terms in the space. The main shortcoming of this approach is the high complexity mostly resulting from the storage requirement of a huge embedding dictionary[1], which consequently affects runtimes.

5.2.2 Naive approach

In this work I chose the linear interpolation approach as the *union* of distributions allowed the models to be compensated by the strength of its counterpart (as opposed to their intersection). I refer to it as the naive approach since in section 5.2.4 I

[1]In my experiment I match nearest-neighbor embeddings based on a single representation of a word-type, yet in the contextual space, instances of the same word-type form separate embeddings that can significantly increase the search space.

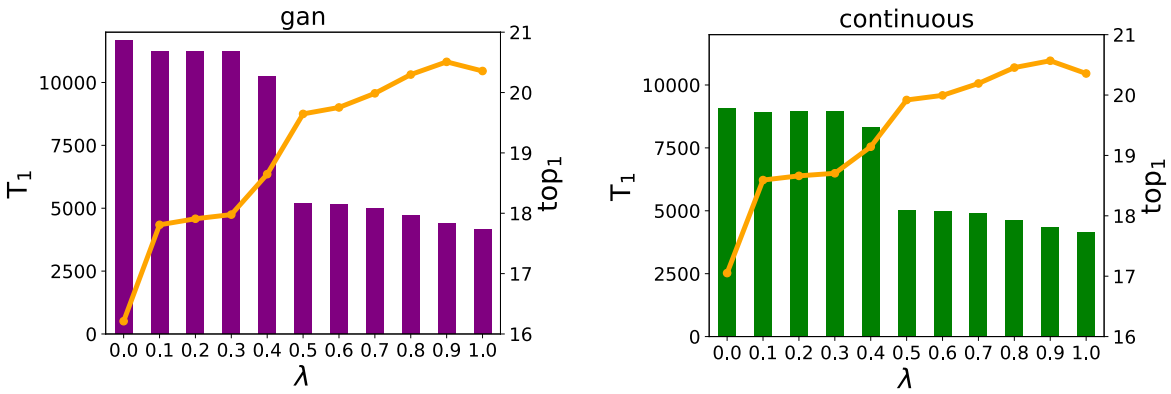
present a more sophisticated approach to combining both models. I followed Eq. 5.4 (based on Eq. 5.1):

$$P(w_t|w_{t-1-n}^{t-1}) = \lambda P_{ctg}(w_t|w_{t-1-n}^{t-1}) + (1-\lambda)P_{con}(w_t|w_{t-1-n}^{t-1}) \tag{5.4}$$

where I combined the categorical model with a continuous model with a $\lambda \in [0, 1]$ to learn about the possible operating points for the model interpolation.

5.2.3 Results of the naive approach

Figure 5.1 describes the two models of G_{50}, and c_{50}, combined separately with a ctg_{50} trained on the NYT dataset (that was presented in the previous experiments). For both experiments, $\lambda = 0.0$, and $\lambda = 1.0$ are data points where the continuous model and the categorical model are represented respectively. The purple and green bars (GAN- and continuous-based approaches, respectively) are associated with T_1, while the yellow trend line is associated with the top_1 metric. In Figure 5.1, the accuracy (in top_1) increases alongside λs as the model becomes more categorical, while the diversity decreases (shown in T_1) for both the GAN- and the continuous-based models. The main differences are that in $\lambda < 0.5$, the GAN-based model is more diverse than the continuous-based interpolation, while the continuous one is more accurate.



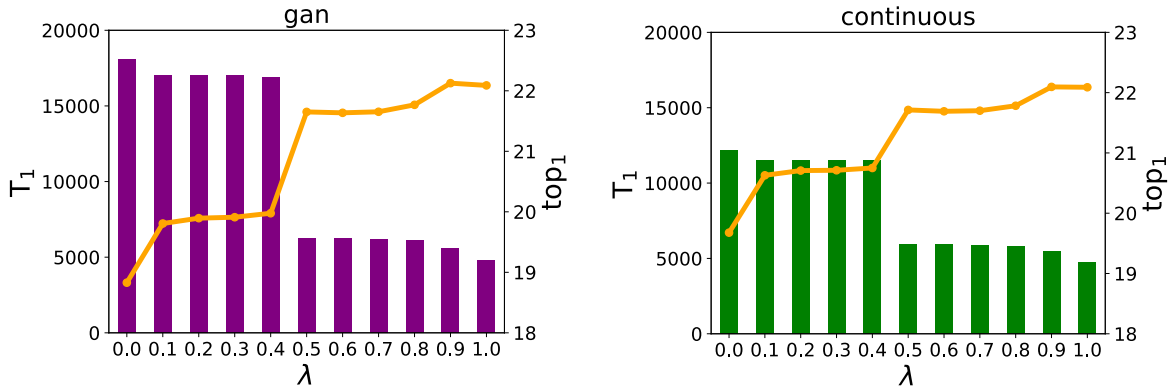
(a) `G` and `ctg` (b) `c` and `ctg`

Figure 5.1: NYT combined models
$\lambda = 0.0$, $\lambda = 1.0$ correspond to the continuous and categorical models, respectively.

This stems from the initial performances of the different continuous models, where G_{50} was more diverse, while c_{50} was more accurate. Similar patterns can be seen in the PMC dataset shown in Figure 5.2, which is also based on the previously presented models.

Apart from the absolute range difference between Figure 5.1 and Figure 5.2, an interesting difference lies in *gap* between the GAN- and the continuous-based model interpolation within a dataset. The initial difference between G_{50} c_{50} was $3,500$ additional types in favor of by the former model in the NYT setting, while about 6000 additional types were predicted additionally by G_{50} in the PMC setting (as seen in Tables 3.5 and 3.6).

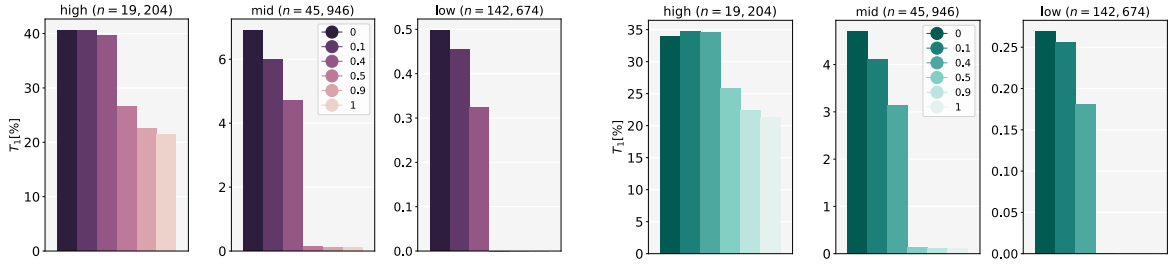
Given the higher-level performance, I chose several possible λ operating points to better elucidate how well the interpolated models predict around the less frequent



(a) `G` and `ctg` (b) `c` and `ctg`

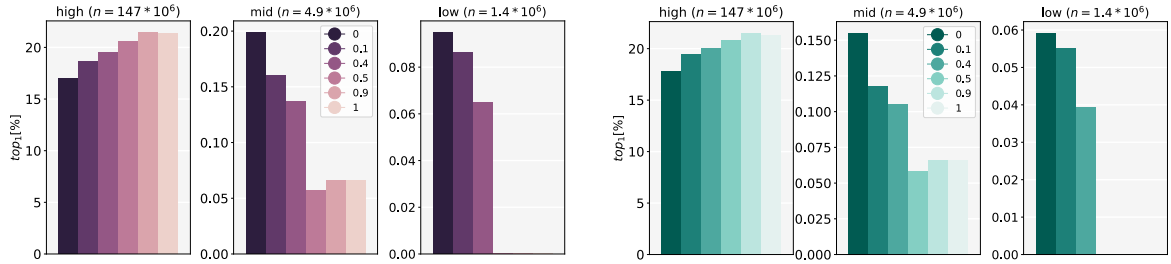
Figure 5.2: PMC combined models
$\lambda = 0.0$, $\lambda = 1.0$ correspond to the continuous and categorical models, respectively.

areas. I focused on $\lambda = 0.1, 0.4, 0.5, 0.9$, as they indicate steeper transitions in the figures presented.



(a) `G` and `ctg` (b) `c` and `ctg`

Figure 5.3: NYT T_1 (types) of different λs for model interpolation.



(a) `G` and `ctg` (b) `c` and `ctg`

Figure 5.4: NYT top_1 (tokens) of different λs for model interpolation.

Figures 5.3 and 5.4 illustrate performance as a function of training frequency over types and tokens for the chosen λs.

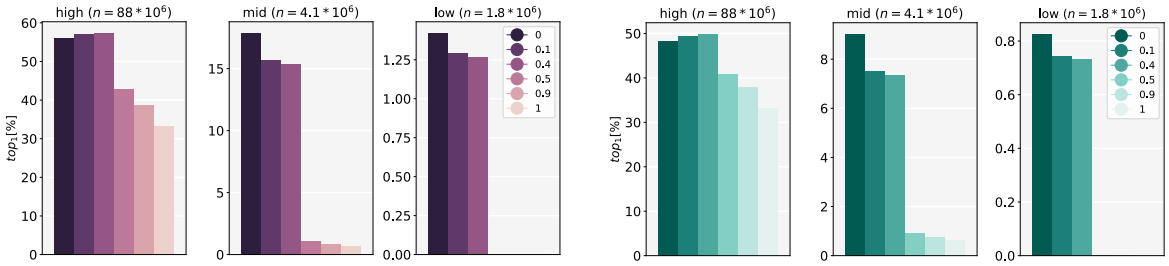
$\lambda = 0.1$ A relative increase in top_1 while maintaining diversity rates similar to those of the original continuous model (ideal for diversity-promoting prediction) can be observed.

$\lambda = 0.4$ shows an additional increase in top_1 and a minimal decrease in T_1.

$\lambda = 0.5$ is a breaking point at which very few predictions are made in the mid- and low-bins (in types and tokens), while accuracy increases at a similar rate.

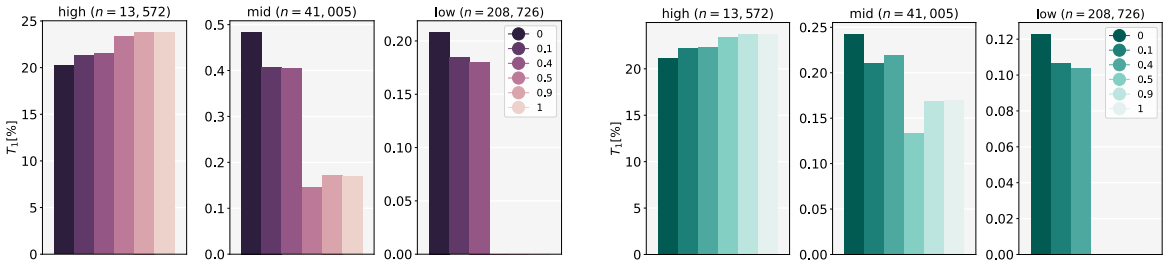
$\lambda = 0.9$ is slightly more diverse than ctg_{50} in NYT; it also resulted in slightly higher accuracy in the high- and mid-bins as well, capturing fewer types but more of them.

A similar pattern was observed for the models over the PMC dataset shown in Figures 5.5 and 5.6.



(a) `G` and `ctg` (b) `c` and `ctg`

Figure 5.5: PMC T_1 (types) of different λs for model interpolation.



(a) `G` and `ctg` (b) `c` and `ctg`

Figure 5.6: PMC top_1 (tokens) of different λs for model interpolation.

5.2.4 Context-dependent approach

Following the experiments with the naive approach shown in section 5.2.3 where λ is constant, I propose a way for learning a context-sensitive λ through a designated categorical model I will train. Formally, the model's goal is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ where $\mathcal{X} = w_{t-1-n}^{t-1}$, which is a sequential history of tokens; and $\mathcal{Y}$ is $\lambda \in \mathbb{R}$ with a range of $(0-1)$, where the lower and the upper ranges reflect continuous and categorical leaning λs, respectively, and h approximates $p(\lambda|w_{t-1-n}^{t-1})$. Learning a varying λ can be done in various ways, and this section discusses two simple ways to do so following the aforementioned formalization. Both ways are based on training a simple LSTM-based model (Hochreiter et al., 1997). Technically, this model is fed with partial sentences as w_{t-1-n}^{t-1}, and it is trained to predict either 0 or 1 labels representing the *continuous* or the *categorical* model, depending on which distribution correctly predicts the given context (trained with cross entropy loss; see Eq. 3.3). The models that provide the labels are the same models from the previous section; in other words, they are *fixed*, and a prediction is considered

false or true based on the original target reference. When preparing the training set for the contextual λ, given both models' predictions (`ctg`, `G`), the predictions were either both correct or both wrong, or one of the models was correct. When both were correct or incorrect, this led to a random assignment of a λ label $(0, 1)$. Notice that given the low hit rate of top_1 (of around 20%) of `ctg` and `G`, about 80% of the labels were randomly assigned. Formally, Eq. 5.4 is relaxed to represent λ as a function of the time step, as shown in Eq. 5.5,

$$P(w_t|w_{t-1-n}^{t-1}) = \hat{\lambda}_t P_{ctg}(w_t|w_{t-1-n}^{t-1}) + (1 - \hat{\lambda}_t)P_{con}(w_t|w_{t-1-n}^{t-1}) \tag{5.5}$$

where λ is estimated directly by using the probability estimate provided by the λ-trained model

$$\hat{\lambda}_t = \lambda_{\texttt{pr}} = P_\lambda(\lambda|w_{t-1-n}^{t-1}) \tag{5.6}$$

or by rounding to the closest integer [0,1]

$$\hat{\lambda}_t = \lambda_{\texttt{rnd}} = \lfloor P_\lambda(\lambda|w_{t-1-n}^{t-1}) \rceil \tag{5.7}$$

The assumption is that the proposed models will learn which contexts require a general or a more detailed class to be predicted. It is important to test to which degree the learned λs are effective, given the evidence in Chapter 3, and section 5.2.3, indicating that categorical and continuous models have non-overlapping regions of correctly predicting a query which a learning model can discover. These non-overlapping regions were shown to be frequency-dependent classes in Chapter 3. For instance, jumping back to section 3.4.4, for the context of 'my piano teacher at the paris', where *conservatory* is the target prediction, a model may learn that a continuous output prediction may be more appropriate than the categorical one, as the modifier 'paris' may call for a less frequent word-type. On the other hand, the context of 'my piano teacher at', with a target of *the*, may learn a λ that indicating the categorical model's distribution should be considered.

One possible limitation that might arise is that the non-overlapping regions assumption is a crude assumption. There may be overlapping regions for the continuous and categorical models that do not produce effective learning (and as a result an optimal λ by which to combine the models' distribution). In what follows, I examine to what degree such a model can be found. Another possible limitation is if top_1 is vastly different between the models, which may create a bias towards one class and hinder the development of a model that combines the two effectively. This is not the case since both models have similar rates (20%) (in the context of this experiment, which is different than the context of Chapter 3) and 80% incorrect labels, making a random assignment.

5.2.5 Results of the context-dependent approach

In this set of experiments I focused on investigating the GAN-based approach (`G`) as it exhibited higher diversity rates (compared to `c`) and potentially greater polarity with regard to the categorical model. As previously noted, the more different the `ctg` model (i.e., the greater the non-overlapping performance between the models), the better the chance that λ learned may determine when a context should be

followed by a certain model distribution. I now compare the results of NYT and PMC shown in Tables 5.1 and 5.2. $\lambda_{\texttt{pr}}$ presents the results based on Eq. 5.6 and Eq. 5.7. Both Tables 5.1 and 5.2 show a more optimal balance in the $\lambda_{\texttt{pr}}$ model,

model	top_1	T_1
$\lambda_{\texttt{pr}}$	17.90	11,240
$\lambda_{\texttt{rnd}}$	16.29	11,716

Table 5.1: NYT contextual λs `G` combined with `ctg`.

model	top_1	T_1
$\lambda_{\texttt{pr}}$	19.90	17,045
$\lambda_{\texttt{rnd}}$	22.09	4,790

Table 5.2: PMC contextual λs `G` combined with `ctg`.

whereas the $\lambda_{\texttt{rnd}}$ model is either producing a very high diversity and a low accuracy, as in NYT, or a high accuracy and a low diversity, as in PMC. It may be easier to understand the contextual λ experiments compared to the constant λ models, as shown in Figures 5.7, and 5.8. Both the NYT and PMC domains exhibit

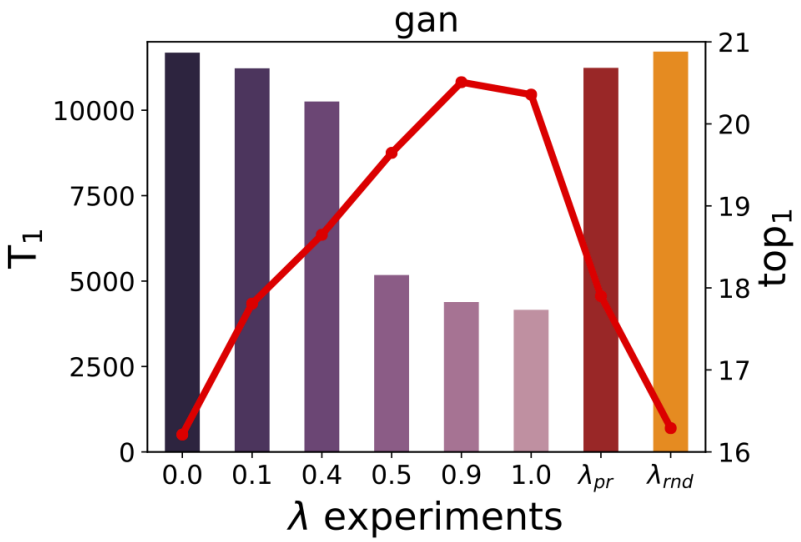


Figure 5.7: `G` and `ctg` combined with different λs in NYT
(the two right-side columns are contextual λs).

similar patterns. First, $\lambda_{\texttt{pr}}$ exhibited a performance on T_1 and top_1 that is similar to $\lambda \in [0.1 - 0.4]$, while $\lambda_{\texttt{rnd}}$ was more similar in performance to either a `G` model in NYT or a `ctg` model in PMC (showing a less well combined model approach leaning sharply towards one or the other model). This suggests that $\lambda_{\texttt{pr}}$ combined the probabilities of both models in a more effective way than $\lambda_{\texttt{rnd}}$, as both metrics (T_1, top_1) improved rather than a single one.[2]

[2]In NYT, $\lambda_{\texttt{rnd}}$ improved diversity at the expense of accuracy, while the opposite happened in PMC.

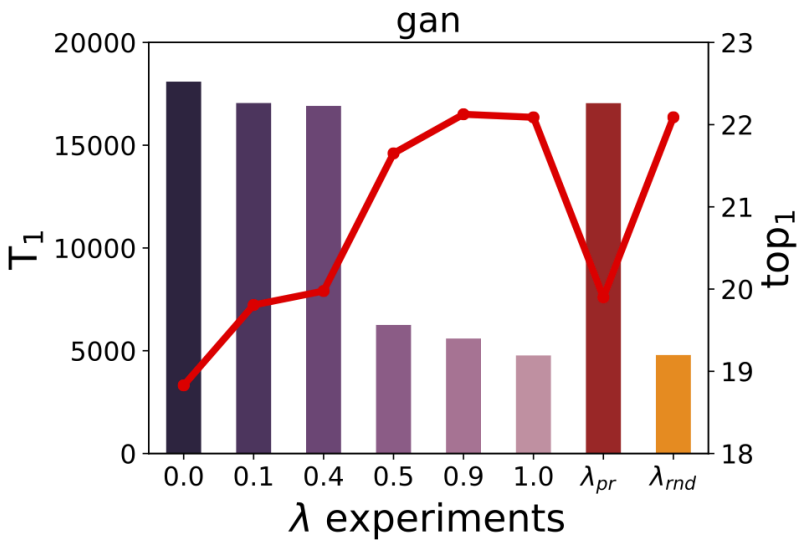


Figure 5.8: `G` and `ctg` combined with different λs in PMC (the two right-side columns are contextual λs).

Following this analysis, the performance over frequency for NYT is shown in Figures 5.9 and 5.10, where the colored dashed (- - -) and non-dashed (—) lines show similarities between the context-dependent λs learned to the performance of the constant λs described in the previous sections. $\lambda =$ `pr` (i.e., $\lambda_{\texttt{pr}}$, abbreviated as `pr`) has top_1 and T_1 outcome that is similar to those of $\lambda = 0.1$ (as the non-dashed line meets *both* `pr` and $\lambda = 0.1$), and $\lambda =$ `rnd`. In other words, $\lambda_{\texttt{rnd}}$, abbreviated as `rnd`, seems to be less effective in combining both approaches and similar to the pure GAN-based approach (as the dashed line meets *both* `rnd` and the continuous model).

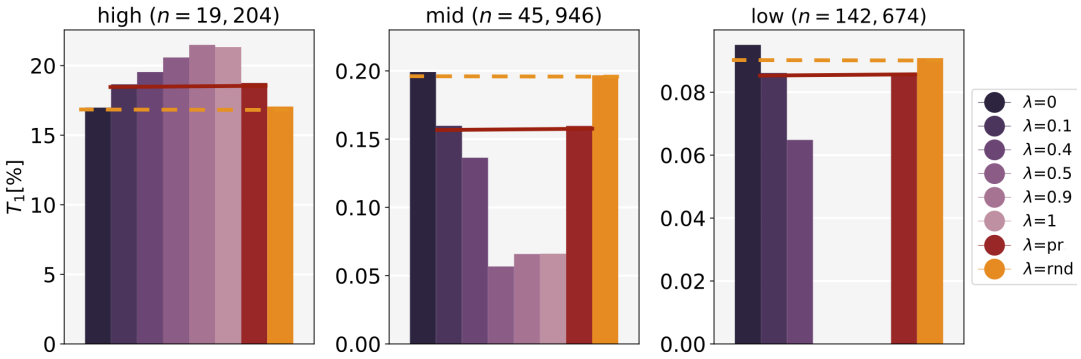


Figure 5.9: `G` and `ctg` combined with different λs in NYT (types).

Figures 5.11 and 5.12 illustrate the performance over frequency for the PMC dataset. While `pr` for PMC was similar to $\lambda = 0.1$ both on types and tokens (as the non-dashed line meets *both* `pr` and $\lambda = 0.1$), `rnd` was found to be a middle ground between the approaches as it did not meet any constant λ (the dashed line for `rnd` is halfway from the various λs).

Following the frequency-based analysis, `pr`, which estimated a probability value for the factor combining both approaches, has learned a diverse λ that is similar in performance to an optimal λ of 0.1. `rnd`'s binary decision makes the quality of the combination of the distribution less clear: for NYT it mainly highlighted the

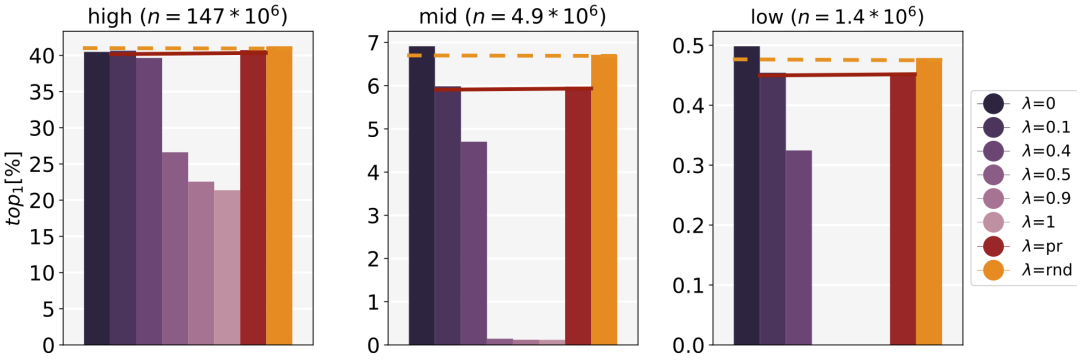


Figure 5.10: `G` and `ctg` combined with different λs in NYT (tokens).

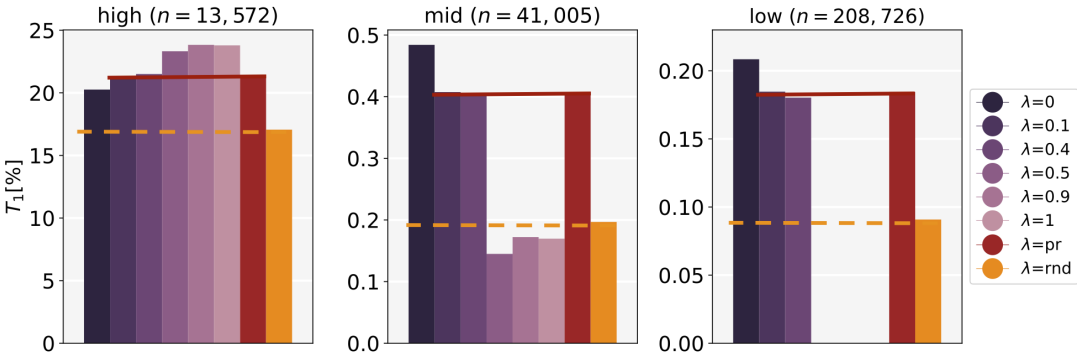


Figure 5.11: `G` and `ctg` combined with different experimental settings λs in PMC (types).

continuous model, while for PMC it combined models in a way that was not similar to other λ experiments. Analyzing `rnd` in PMC may help place this experiment between the 0.4 according to Figure 5.11 and 0.5 according to Figure 5.12, making it under the hood leaning towards the continuous approach (as was shown in the frequency-based analysis). When considering the setting for the purpose of explaining the results, it merits noting that at least 80% of the labels for the contextual experiment were chosen randomly. Assuming there is a fair coin, no biases should be created, as the expectancy of a fair coin is 50% for each side. However, this may have not been the case. The random labels assigned might have been leaning towards one label over the other, which may explain why both the NYT and PMC contextual experiments leaned towards one type of label (assigning more 0 type labels that are associated with the continuous label).

5.2.6 Future work

In this section, I proposed one way to combine these models effectively: model interpolation. However, there are various other approaches that can be taken to achieve this goal. For instance, one could follow an approach similar to Grave et al.'s (2017), where the dictionary V is partitioned by frequency to its head V_{head} and its tail V_{tail}, and access to each of the partitions is through hierarchical representation where $v \in V_{head}$ can be directly predicted from a shortlist, and accessing V_{tail} can be done by first predicting a *tail* class that leads to the second layer where $v \in V_{tail}$ are found, as proposed by Mikolov et al. (2011c). Alternatively a two-level tree can be

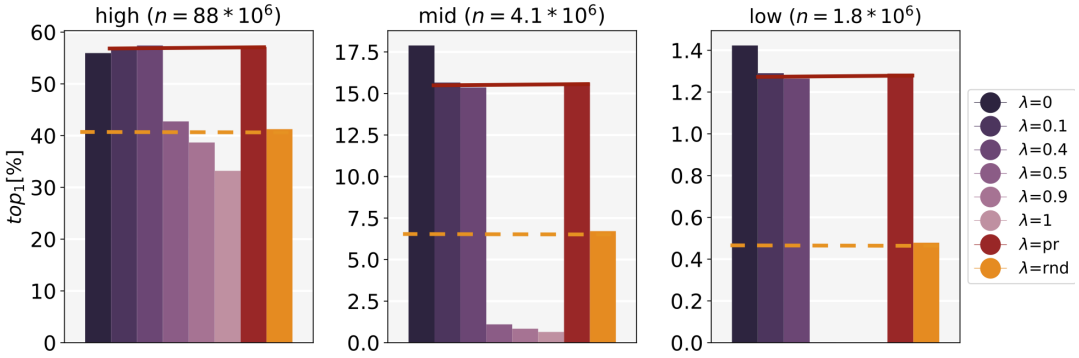


Figure 5.12: `G` and `ctg` combined with different λs in PMC (tokens).

constructed such that first, the model determines to which class (head or tail) the predicted word belongs and then guesses the word, as seen in Eq. 5.8 and presented in Le et al. (2011):

$$p(w_t|h_t) = p_1(C(w_t)|h_t)p_2(w_t|C(w_t), h_t) \tag{5.8}$$

The former makes smaller amortized costs due to Zipf's law (Zipf, 1935), where directly predicting a small group of frequent words that makes up the majority of the text is cheaper than indirectly as in the fully hierarchical method, and therefore reduces the number of computations.

5.2.7 Conclusion

In this section I presented ways to effectively benefit from both models' strengths, as the categorical approach demonstrated strong prediction accuracy and the continuous approaches had greater prediction diversity (see Chapter 3). I examined model interpolation approaches that linearly combine both models either through a constant or a context-dependent factor. One possible limitation to consider is that combining both models increases the overall costs involved in making a prediction due to the use of the categorical model. Importantly, the results show that combining these models can yield enhanced performance, and that in our experiments, a context-dependent λ where a probability score generated results comparable to one of the optimal constant λs that was found ($\lambda = 0.1$). Therefore, given the additional costs of the context-sensitive technique, choosing a constant λ is a less expensive way to combine the approaches while yielding a similar outcome. In section 5.2.6, additional research directions related to combining models were proposed.

5.3 Feature-Based Decoding

5.3.1 Introduction

In Chapter 3 I showed that the continuous approaches, applied to both the NYT and PMC datasets, produced models that were overall more diverse in their predictions than the categorical-based approaches. Knowing this, I searched for a way to augment the hit rates of the continuous models. In this section I propose a new decoding technique for improving the continuous models' hit rates.

Recall that during the decoding process described in Section 3.3.4, the closest embedding prediction candidates are matched against a target through a nearest-neighbor search. Given that the embedding space these models learn is organized by semantic relations (e.g., words sharing similar context are likely to be found closer to one another), I investigated whether adding *syntactic* information could enhance the underlying results. For the purpose of this evaluation, I employed feature-based decoding, which is described in section 5.3.2, using oracle part-of-speech (PoS) tags of a target in order to refine the list of prediction candidates in the nearest-neighbor search.

Syntactic constraints for text generation have been applied by previous researches to paraphrase extraction: Pang et al. (2003) proposed extracting similar PoS patterns using a finite state automaton (FSA) to find semantically similar phrases. Callison-Burch (2008) proposed extracting similar PoS patterns from one language to find paraphrases in a parallel corpus for machine translation. This may not be applicable to every language pair, as some languages may not reflect similar syntactic patterns. PoS constraints were also applied recently in metaphor extraction (Yu et al., 2019), where the constraint served to find each verb and its context, such that given a target verb, a new metaphor can be generated by replacing a similar (yet simpler) verb with a target in a sentence.

Why use PoS decoding?

The choice to use PoS as a feature to assist in decoding was motivated by an attempt to decompose word prediction into a process with both syntactic and semantic components. Traditional categorical language models are known to encode syntactic information, as discussed in (Goldberg, 2019; Linzen et al., 2016), among others, but when decoding, this information does not seem to sufficiently inform word prediction in the continuous models. Following the lead of Czarnowska et al. (2019), who focused on syntax prediction by fixing LM semantics, I fixed/simulated the process of predicting PoS tags and focused on the semantic aspect of word prediction. In practical use, the PoS-enabled decoding method would require an additional predictive model to generate PoS tags; however, as previously noted (Goldberg, 2019; Linzen et al., 2016), this is a much simpler prediction task and is one at which categorical models are known to do well. Specifically, there are two reasons the prediction of PoS tags is likely to work well: first, the classification of a *limited* set of classes such as the MNIST dataset LeCun et al. (1998), and second, the tendency of categorical models to exhibit high *prediction accuracies* (which was shown in Chapter 3 as well). Given this, and that my experimental purpose was to assess the utility of additional features in decoding, I found it appropriate to separate the feature-generation task from the decoding task and evaluate the latter in isolation.

Note that this approach is highly flexible, as almost any feature (or combination of features) may be used to assist the model in decoding. Given that, choosing *which* feature(s) to use becomes an important question. If I had been using a word embedding space directly informed by syntax, as described by Levy et al. (2014a), or frequency (Gong et al., 2018), a different feature may have worked better.

5.3.2 Applying a feature-based constraint

For this experiment, I *simulated*[3] a predictive model of PoS. First, based on the PoS tags observed in training set, I created a lookup table that maps each term seen during training to its possible PoS. During testing, I provided the correct PoS tag of the predicted word, which conditioned the search only on the nearest-neighbor words (with regard to a prediction) that had the same PoS tag (based on the lookup table). In this way, I generated a refined list of candidates. As noted, while the PoS of unseen tokens were provided as is based on PoS-tagged corpora of Ferraro et al. (2018, LDC2018T20) for NYT and Beck (2010) for PMC, learning to predict these tags is possible using models that can estimate Eq. 5.9

$$p(PoS_t) = p(PoS_t|PoS_{<t}, w_{<t}) \tag{5.9}$$

or any variation of it.

5.3.3 Results

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{ctg}_{50_p}$	<u>29.30</u> (51.38)	4,809	(8,607)
$\texttt{c}_{50_p}$	20.09 (31.83)	10,326	(25,616)
$\texttt{G}_{50_p}$	19.56 (30.30)	<u>13,540</u>	(32,140)
$\texttt{ctg}_{200_p}$	<u>30.00</u> (52.16)	4,752	(8,420)
$\texttt{c}_{200_p}$	21.73 (32.93)	5,348	(15,611)
$\texttt{G}_{200_p}$	20.91 (31.06)	<u>7,659</u>	(21,221)

Table 5.3: NYT PoS decoding

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{ctg}_{50}$	19.19 (46.02)	3,982	(7,559)
$\texttt{c}_{50}$	17.31 (28.70)	8,917	(22,509)
$\texttt{G}_{50}$	16.46 (27.45)	<u>11,534</u>	(27,921)
$\texttt{ctg}_{200}$	21.21 (47.87)	4,163	(7,683)
$\texttt{c}_{200}$	18.94 (30.90)	4,335	(13,087)
$\texttt{G}_{200}$	18.15 (29.15)	<u>6,194</u>	(17,905)

Table 5.4: NYT simple decoding

Tables 5.3 and 5.5 associate each model with a subscipt p indicating PoS decoding. The original tables with the simple decoding, as presented in Chapter 3 (3.3 and 3.4), are found below each feature-based decoding.

[3]I leave implementation of an oracle-predicting model (instead of simulation) to future work, but as explained in the previous section, this type of model has been shown to be effective in similar environments, and under this assumption, the focus of this section is whether decoding with these oracles can improve performance and if so, how.

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{cg}_{50_p}$	<u>30.00</u> (52.38)	6,006	(10,538)
$\texttt{c}_{50_p}$	22.35 (36.68)	15,909	(42,082)
$\texttt{G}_{50_p}$	21.32 (34.73)	<u>23,544</u>	(56,059)
$\texttt{cg}_{200_p}$	<u>30.26</u> (52.52)	4,654	(8,306)
$\texttt{c}_{200_p}$	21.73 (35.99)	8,374	(23,292)
$\texttt{G}_{200_p}$	23.89 (37.57)	<u>12,952</u>	(34,800)

Table 5.5: PMC PoS decoding

model	top_1 (top_{10})	T_1	(T_{10})
$\texttt{ctg}_{50}$	22.12 (48.02)	4,764	(9,020)
$\texttt{c}_{50}$	19.89 (32.04)	11,947	(34,641)
$\texttt{G}_{50}$	19.04 (30.41)	<u>18,040</u>	(47,550)
$\texttt{ctg}_{200}$	22.92 (48.47)	3,678	(7,006)
$\texttt{c}_{200}$	17.06 (30.88)	6,083	(18,533)
$\texttt{G}_{200}$	20.67 (32.96)	<u>9,411</u>	(28,164)

Table 5.6: PMC simple decoding

Both decoding results shown in Tables 5.3 and 5.5 show that this decoding technique behaved differently depending on the model. For the categorical models, the technique advanced the hit rate of tokens, while for the continuous approaches, it enhanced the diversity of types predicted. This reinforces the plausibility of inherently different mechanisms of prediction for the two different approaches (as shown in Tables 5.4 and 5.6). This decoding strategy strengthens the already strong aspects of these models, emphasizing how they operate. The categorical and continuous approaches exhibit different angles of the bias-variance trade-offs: the categorical model is more biased towards the head of the distribution, achieving high hit rates, while the continuous approach has greater variety in prediction, attaining high prediction diversity rates. PoS decoding increased the categorical model's hit rate by 50% (top_1, NYT, $\texttt{ctg}_{200_p}$), while the same decoding produced 37% more types than the simple decoding in the continuous GAN models. In this regard, the PoS decoding served as a diagnostic tool revealing the models' priorities. Compared to the NYT domain, the PMC domain showed more significant improvement in terms of absolute numbers for prediction diversity in the continuous models; therefore, these models could potentially thrive at long-tail distributions or domains.

Initially, the goal of the feature-based (or PoS) decoding technique was to improve the hit rate of the continuous approaches; however, the proposed decoding instead increased the diversity of the continuous model (while increasing the hit rate of the categorical model). Nonetheless, the hunch to add syntactic information to the decoding process for the continuous models was proven justified, as doing so created more successful predictions (against the metrics presented).

5.3.4 Future work

A possible direction for feature-based decoding involves addressing morpohlogically rich languages. As Czarnowska et al. (2019) demonstrated, one way forward with multiple word inflection languages may be to separate the problem into two simpler problems: one that addresses the rich semantics (such as the continuous output models as proposed in this thesis), and another that applies a constraint over the syntactic features. For example, in a Turkish dataset curated by Öztürk et al. (2014), the syntactic features include not only PoS but also morphological analysis. Applying both constraints through a predictive model may be a promising way of handling the large number of types, as each model would require prediction from a limited number of classes.

5.3.5 Conclusion

While the initial goal of this section was to augment the continuous prediction approach by means of decoding, this analysis exposed deeper observations about the models at hand. First, the decoding approach indeed improved the hit rates of the continuous models; however, this decoding actually improved the diversity of these models to a much greater rate. Moreover, when applied to the categorical model, the approach improved a different property, which was accuracy.[4] Boosting these approaches has given rise to what these approaches are optimized for: the continuous model increases its access to new terms, while the categorical model refines its re-ranked list of similar choices (similar as the tables show that only a small number of types were added). Given this, this decoding emphasizes the bias-variance tradeoff, where the categorical models tend to be more biased, while the continuous models have greater variance (at the expense of accuracy). Hence, the proposed decoding was informative as an analysis tool by pointing to what each type of model promotes. Feature-based decoding is proposed as a way to approach morphologically rich languages or domains with relatively large numbers of infrequent word-types.

5.4 The Role of Pre-trained Embedding Space in Continuous Models

5.4.1 Introduction

Another important aspect of continuous prediction language modeling is the embedding space on which the models are trained and prediction takes place. Understanding embedding spaces' role in the process could assist researchers in choosing spaces that produce optimal outcomes for continuous approach predictions with respect to the top_x and T_x metrics. The choice of static embeddings (described in section 2.4) was made for two reasons. The first reason is to reduce costs driven by alternatively employing contextual embeddings in which every token in the training set is represented differently, as shown by Grave et al. (2016) and Khandelwal et al. (2019). Therefore, for large vocabularies, contextual embeddings may be deemed

[4]There was slight improvement to diversity.

an impractical choice. The second reason is that the model is required to learn locations across a space and within a training session; hence, fixing locations would make for more stable learning. The possible trade-off of this choice of static spaces is the lack of multi-sense word representations. In Kumar et al. (2019), from which I drew inspiration, the authors have trained on static spaces as well. However, since the goal was one of machine translation, the embedding space on the input differed from the one on the output.

5.4.2 Approach

In Chapter 3 the experiments were only evaluated on a word2vec custom embedding space generated based on the appropriate datasets. In this section I elaborate on the analysis that led me to arrive at the decision to use word2vec. I evaluated my models' behavior across three different spaces: GloVe, FastText, and word2vec (as described in section 3.3.3). All embedding spaces were trained on the training set of each domain and either had a representation for each word in the training (a unique representation for every grapheme) or collapsed to *unk* for words occurring fewer than four times. The following section, therefore, aims to investigate the effect on continuous model performance when controlling for all parameters except the embedding space. In this section I experimented with the continuous model of `c`, since I assumed there would not be major differences between the `c` and `G` models with respect to the effect of the embedding space. This assumption was made because in previous experiments, both continuous models behaved similarly yet with slightly different intensities across the examined metrics.

5.4.3 Results

space	top_1 (top_{10})	T_1	(T_{10})
$\texttt{glv}_{50}$	10.56 (16.85)	1,484	(3,367)
$\texttt{ft}_{50}$	13.64 (21.99)	1,478	(4,880)
$\texttt{w2v}_{50}$	17.31 (28.70)	<u>8,917</u>	(22,509)
$\texttt{glv}_{200}$	13.79 (18.97)	638	(1,616)
$\texttt{ft}_{200}$	16.40 (24.19)	655	(2,322)
$\texttt{w2v}_{200}$	<u>18.94</u> (30.90)	4,335	(13,087)

Table 5.7: NYT embedding spaces' effect on $\texttt{c}_{50}$.

Tables 5.7 and 5.8 show the results of this analysis for both the NYT and PMC corpora when using a simple nearest-neighbor decoding method. Models using word2vec algorithm (`w2v`) were consistently superior to those using GloVe (`glv`) or FastText (`ft`).

While increasing embedding dimensionality is often perceived as a way to better inform the model about the features of the data (as seen in an increased $top_1(top_{10})$)[5], in this case, this increased dimensionality harmed the performance in other ways,

[5]This may come at the risk of overfitting.

space	top_1 (top_{10})	T_1	(T_{10})
$\texttt{glv}_{50}$	12.37 (17.61)	2,942	(8,553)
$\texttt{ft}_{50}$	9.71 (16.71)	667	(3,898)
$\texttt{w2v}_{50}$	19.89 (32.04)	11,947	(34,641)
$\texttt{glv}_{200}$	14.63 (18.74)	676	(2,106)
$\texttt{ft}_{200}$	9.07 (13.49)	49	(275)
$\texttt{w2v}_{200}$	17.06 (30.88)	6,083	(18,533)

Table 5.8: PMC embedding spaces' effect on $\texttt{c}_{50}$.

including by leading to a reduced number of types ($T_{1/10}$). This finding illustrates another bias-variance tradeoff conditioned on vector dimensionality. Tables 5.7 and 5.8 were stratified to examine performance over frequency bins for type and token distributions. As Figures 5.13, 5.14, 5.15, and 5.16 show, `w2v` outperformed the other embedding spaces in hit rate and predicted more infrequent terms; moreover, the alternative spaces did not score well at all on prediction diversity.

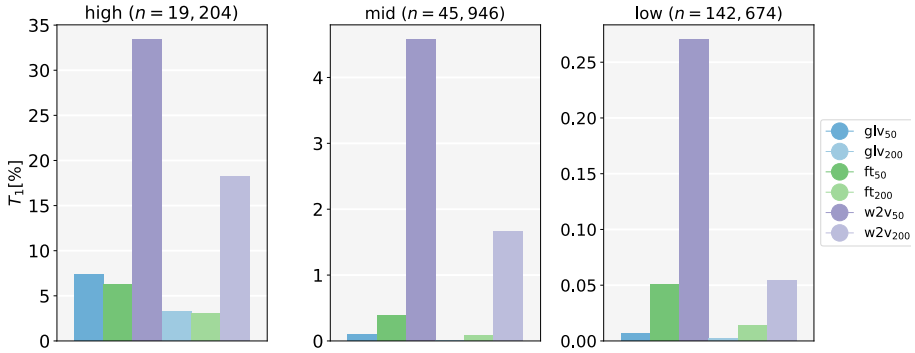


Figure 5.13: NYT type coverage by training frequency and embedding space.

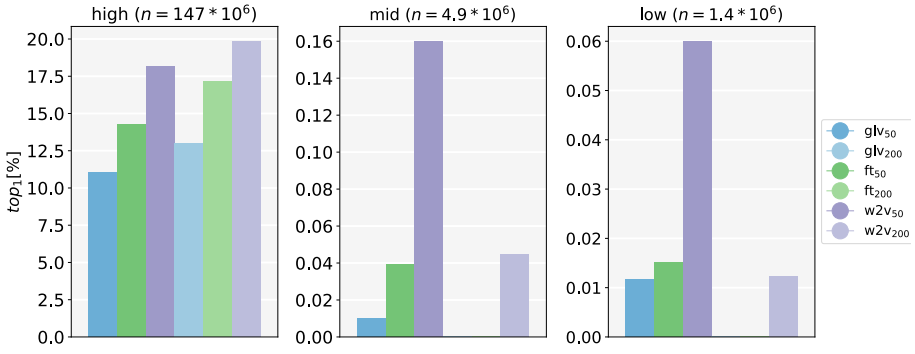


Figure 5.14: NYT token hit rate by training frequency and embedding space.

A possible reason for the low performance of `ft` may have to do with the fact that FastText algorithm does not directly generate the entire set of embedding words. Rather, it generates the most frequent ones, together with additional subwords made of pieces of less frequent words. Then, to represent the experiment's infrequent words in FastText space it has to compose subwords together, and the degree to

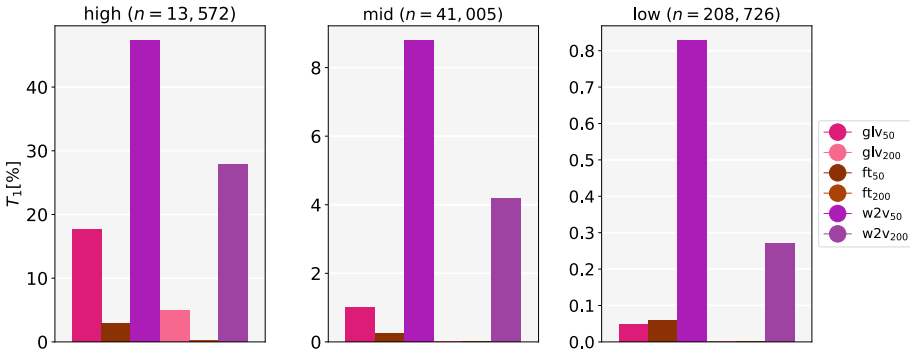


Figure 5.15: PMC type coverage by training frequency and embedding space.

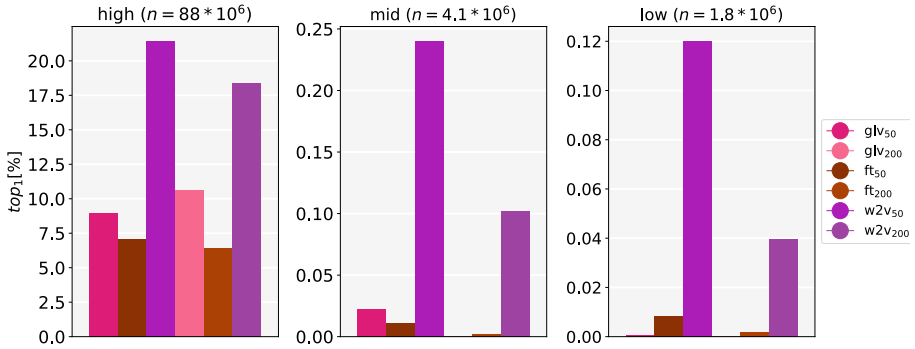


Figure 5.16: PMC token hit rate by training frequency and embedding space.

which this composition produces meaningful semantic relations is questionable. A possible limitation in FastText is, therefore, the ability to capture the context of those rare words and represent them meaningfully in the space. Before moving forward, two conclusion can be drawn from the experiments I presented. First, word2vec was found to be the most optimal embedding space over the evaluation metrics of interest. Second, increasing dimensionality may improve accuracy, but was found to harm the type diversity. In what follows (section 5.4.3) I dive deep into learning about what are possible differences between `glv` to `w2v` to account for the different outcomes shown in the table and figures presented in this section.

On differences between GloVe and word2vec

In this section I take a closer look at some differences noted while exploring the effect of the embedding space on the `c` model. The results are based on a toy dataset of NYT (Ferraro et al., 2018, LDC2018T20), containing a limited number of sentences: 45k, 15k, 15k for train/dev/test, following the same training protocols described in section 3.3.

model	top_1	(top_{10})	T_1	(T_{10})
w2v$_{50}$	13.84	(23.48)	523	(1,660)
glv$_{50}$	4.95	(9.11)	128	(347)

Table 5.9: different dimension results.

Table 5.9 illustrates the differences observed between the `glv` model and the `w2v`

technique (supporting the findings presented earlier in Tables 5.7 and 5.8). Since `glv` performed poorly, I looked for ways to mitigate this, or alternatively, reasons for this behavior. To better understand the difference, I plotted the hit/miss distribution of `glv` compared with the `w2v` model. Figure 5.17 presents the hit/miss rate over the different predictions the models made. The x-axis describes the cosine similarity of the predicted vector to its target (on a log scale).

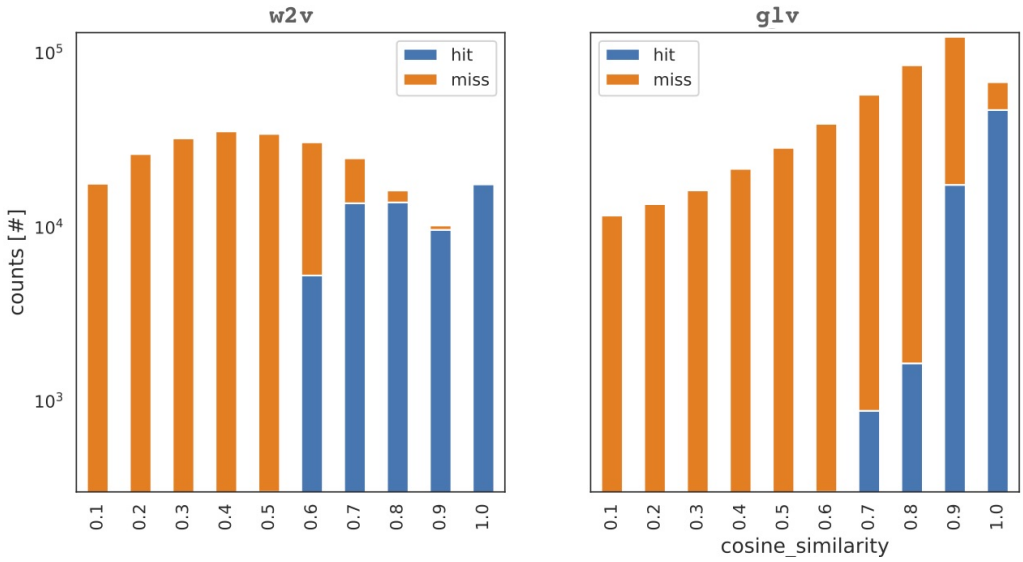


Figure 5.17: Prediction to target cosine similarity.

Figure 5.17 reveals areas in which the `glv` model was able to make predictions that were very close (0.9 similarity) to the target, yet were not considered to be among their respective targets' closest neighbors (notice, on the left-hand side of the figure, the orange color on the 0.9 bar, which is considered a 'miss'). On the other hand, for `w2v`, there were situations in which the similarity was relatively looser (0.6 similarity), but the model's prediction was found to be in the desired vicinity of the target (notice, on the left-hand side of the figure, the blue color on the 0.6 bar, which is considered a 'hit'). Why, despite predicting high-similarity vectors, can the `glv` model not reach its targets? Naturally, I hypothesized that when high similarity is measured and the target and prediction turn out to be insufficiently close, the cause has to do with the potentially dense nature of the region. In other words, because the target is surrounded by many *highly* similar neighbors, the prediction struggles to penetrate its closest circles. To examine this hypothesis, I measured the density of the input, focusing on the tokens of the dataset.

In Figure 5.18, the distance of the target's median neighbor (out of ten neighbors) is boxplotted for all the tokens in the test dataset. The median distance from the target, according to Figure 5.18, is longer in `w2v` than in `glv` (one-tailed Mann-Whitney U, $p \ll 0.0001$). Given that the only difference in the plots is the embedding mechanism by which the plots were generated, I suspect that the reason for the behavior observed in Figure 5.17 has to do with the manner in which the vectors occupy the space, and more precisely, the difference in density of the vectors. The fact that Figure 5.18 is based only on the vectors and the dataset, and not on the model, reinforces this hypothesis. Since the figures provide evidence that `glv` may exhibit high clusterness regions, inevitably one must ask whether a less dense vector space would provide a better foundation on which to train a model. To test this

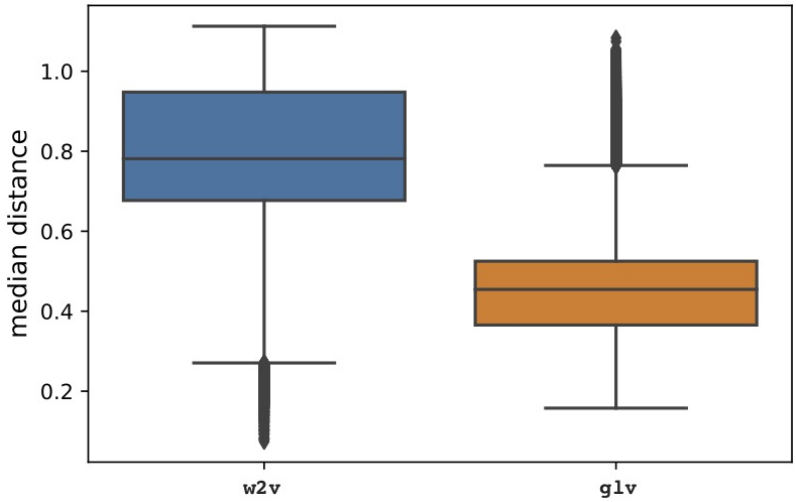


Figure 5.18: Median neighbor's distance from target.

hypothesis, I transformed the GloVe space to a less clustered one and trained a new model on it.

I provide the set of steps to find a viable kernel transform in the Appendix A.2.1. Table 5.10 presents the pre- and post-transform results of the GloVe models.

model	top_1	(top_{10})	T_1	(T_{10})
$\texttt{glv}_{50}$	4.95	(9.11)	128	(347)
$\texttt{glvT}_{50}$	6.40	(11.95)	177	(459)

Table 5.10: Pre- and post-transform results.

Clearly, the post-trasform GloVe space, shown in `glvT`, enhanced the model's performance. The phenomenon of clusterness, or observing spaces that contain high-density regions of textual spaces, was described by Radovanović et al. (2009) and Radovanović et al. (2010) and is referred to as the *hubness* phenomenon. Since *hubness* is not a phenomenon unique to GloVe (or to word embeddings; it happens in image and speech spaces as well) (Radovanović et al., 2010), and since GloVe is a common embedding space that has been widely used by the NLP community, learning about its nature is key to understanding its impact on different models. This probing provided tools to analyze and enhance the performance of similar embedding spaces that may exhibit low performance for this reason. The GloVe space was shown to exhibit hubness (clusterness of embeddings), harming the learning abilities of a continuous model.

5.4.4 Future work

One limitation of the embedding models presented is the degree of semantic relations expressed in these models. Ideally, proximity would reflect the semantic relations between similar words and would not be conflated with frequency, as was shown by Gong et al. (2018). Another possible concern is the reliability of the neighborhoods of infrequent words; that is, to what degree are other similar

words close when only a limited number of examples is observed?[6] It is possible that an embedding space of ConceptNet (Speer et al., 2016), which is a distributional semantic-based embedding space created by incorporating human knowledge (through retrofitting (Faruqui et al., 2015)) would be more reliable for determining the similarity of less-frequent words. Another limitation is found in section 5.4.3, where the experiment described was performed on a smaller dataset and there were fewer repetitions to corroborate the findings. A general limitation of this experiment and of the model training described in this thesis overall is the failure to employ contextualized vectors, which [7] were shown to capture multi-sense meanings and represent words not in a *fixed* manner. While a possible future direction is multi-sense representation, the space a model learns to predict is assumed to be *fixed* to some degree[8] This allows for the meaningful learning of various locations/regions across space, leading to meaningful decoding such that a model can recall those locations given a similar query later on.

5.4.5 Conclusion

In this section, embedding spaces were investigated. word2vec was found to be superior to the other spaces both in types and tokens. I hypothesized as to why FastText was underperforming, and I further experimented with GloVe to learn about its nature of prediction by comparing it closely with word2vec. GloVe was found to create more clustered spaces, thereby hindering a model in retrieving a target vector. More advanced embedding spaces, such as those of ConceptNet, may be able to be employed in future research. In these more advanced embedding spaces, less-frequent words maintain semantic relations, and if a model is not found to hit close enough to a target term, its neighbors can provide insights into the underlying semantics the model predicted.

5.5 How to Find an Optimal Loss Function for a Continuous Model

5.5.1 Introduction

The loss function also plays a role when developing a neural architecture. Loss functions set different objectives for a model, and together with back-propagation (Rumelhart et al., 1986), reduce errors between a model's prediction and an expected target. As explained in section 2.3.4, these errors update the net weights through the chain rule. This section examines several loss functions that are appropriate to regression, as regression is the learning type occurring in the continuous output prediction models in this thesis. Specifically, I aim to minimize the error between a predicted location and a specific target word vector. Regularization may take place as well to prevent over-fitting to the training set. While the losses in this section revolve around the loss applied to both `c` and `G` models, the additional loss of `G` stemming

[6] Bahdanau et al. (2017) pointed to the rare word representation problem.

[7] For an explanation, see section 2.4.2.

[8] Locations across space can change from one training session to another, though, so in that regard they may change over time.

from the GAN dynamic (Eq. 3.9) is MSE, as binary cross entropy did not produce meaningful output.

5.5.2 Loss functions

In this section I share the preliminary results of the different loss functions I explored for the continuous models. I evaluated these losses on the toy dataset NYT (Ferraro et al., 2018, LDC2018T20), containing 45k,15k,15k for train/dev/test, following the same training protocols described in section 3.3. Both models, `G` and `c`, were trained and evaluated on the metrics of interest: top_x and T_x. The costs function studied were the mean-squared error (MSE) and cosine similarity (CoS), as presented in Eq. 5.11, 5.10:

$$L_t = \sum_{w_t \in W_t} |\hat{w}_t - w_t|^2 \tag{5.10}$$

I experimented with MSE loss as Mikolov et al. (2013a) and Bolukbasi et al. (2016) showed that the embedding space to some degree encodes certain relations through simple addition and subtraction operations

$$L_t = \sum_{w_t \in W_t} |2 \times (1 - < \hat{w}_t, w_t >)|^{0.5} \tag{5.11}$$

CoS was chosen as an objective function as well since it is commonly used to measure word embedding similarity (as shown in Mikolov et al. (2013c)). Lazaridou et al. (2015) explained how the outcomes of CoS and MSE may exhibit low variance relative to their targets and predict average representations of vectors (the mean vector of the space), since the mechanism of these losses causes the model to be drawn to high-density areas in the vector space. Following their method, I employed an additional loss function, the max-margin (MM) loss shown in Eq. 5.12:

$$\sum_{j \neq i}^{k} \max\{0, \gamma + D(\hat{w}_i, w_i) - D(\hat{w}_i, w_j)\}, k > 1, \gamma > 0 \tag{5.12}$$

This loss is a variation of hinge loss (Rosasco et al., 2004) where updates take place if a prediction is not found to be the closest to a target among other vectors[9] and where instead of a binary classification, the training procedure is a one vs. all. Max-margin is seen in Eq. 5.12. Table 5.11 shows the results of the models for the continuous approach.

Table 5.11 shows that both MSE and CoS were close in the number of types, while the hit rate of CoS seems to be higher that that of MSE across models. MM exhibited low performance for the type of models I used. It is important to note that the loss performance has to do with the type of distance metric used in the decoding step, which was based on angular distance. This may explain why CoS was chosen in this setting as an objective function, while in Khandelwal et al. (2019), the authors chose MSE as an objective and did their decoding through MSE as well. I note that the decoding process has no direct effect on model training.

[9]The other loss functions' updates are based on whether there exists an error irrespective of other vectors.

model	$top_1(top_{10})$	T_1	(T_{10})
$\mathtt{c}_{MSE}$	11.25 (21.68)	546	$(1,718)$
$\mathtt{G}_{MSE}$	10.39 (19.77)	562	$(1,821)$
$\mathtt{c}_{CoS}$	13.84 (23.48)	523	$(1,660)$
$\mathtt{G}_{CoS}$	12.42 (21.43)	588	$(1,858)$
$\mathtt{c}_{MM}$	11.15 (20.89)	194	(824)
$\mathtt{G}_{MM}$	10.07 (19.47)	222	(892)

Table 5.11: Continuous models' loss functions exploration.

Von-Mises Fisher loss function

The final loss function I investigated was the Von-Mises Fisher (VMF); however, I was not able to employ it following its definition by Kumar et al. (2019). In the following I explain the mismatch between the models I proposed and the loss. The VMF loss function is as follows:

$$NLLvMF(\hat{e}, e) = -\log(C_m(\|\hat{e}\|)) - \hat{e}^T e \tag{5.13}$$

where $C_m(\kappa)$ is given as

$$C_m(\kappa) = \frac{\kappa^{m/2-1}}{2\pi^{m/2} I_{m/2-1}(\kappa)} \tag{5.14}$$

and the gradient approximation of the first component is

$$\Delta \log(C_m(\kappa)) = -\frac{I_{m/2}(\kappa)}{I_{m/2-1}(\kappa)} \tag{5.15}$$

Since according to Eq. 5.13 and 5.14 $\kappa = \|\hat{e}\|$, and since in the case of normalized vectors $\|\hat{e}\| = 1$, the first component's gradient presented in 5.15 remains the same across every vector predicted such that

$$\begin{aligned} \Delta \log(C_m(\kappa))\Big|_{\kappa=1} &= -\frac{I_{m/2}(1)}{I_{m/2-1}(1)}\Big|_{m=50} \\ &= -\frac{I_{50/2}(1)}{I_{50/2-1}(1)} \\ &= const \end{aligned} \tag{5.16}$$

making the described gradient insensitive to different samples as all share the same norm. As a result, the loss adds a component that acts as a constant *bias* term that is not learning meaningful gradients and can be reduced to Eq. 5.17:

$$\begin{aligned} NLLvMF(\hat{e}, e)\Big|_{\|\hat{e}\|=1} &= -\log(C_m(\|\hat{e}\|)) - \hat{e}^T e \\ &= bias. - \hat{e}^T e \end{aligned} \tag{5.17}$$

Therefore under the current conditions, the $NLLvMF$ loss function is reduced to the loss function seen in Eq. 5.11.

Regularization

The previous section described various objective functions; however, another detail to note for the purpose of reproducibility is that another loss component that was found useful was the L_2 norm. The L_2 norm was applied to the parameters of the model during training under a large vocabulary setting. While this may not have produced the minimum error over the training set, it allowed for the generation of a stable training error; without this error, the model had poor gradients. This was not the case with the small dataset training described here.

Up to this point, I have explored several objective functions and shown what was found to be a relevant and optimal loss for the proposed language models.

5.5.3 Future directions

Exploring other architectures such as variational autoencoders (VAEs)[10] may be a another way to find stronger models for continuous output prediction. As with any architectural search, this may require investigating various loss functions, though ones that are typical to VAEs' architecture. One concern regarding VAEs is the resolution of its prediction, as VAEs have been found to generate blurry images (Dosovitskiy et al., 2016; Zhao et al., 2017a). This is because VAEs' auxiliary function may not learn a good estimate for the real distribution of the data; in other words, samples may have lower quality when reconstructing their targets.

5.5.4 Conclusions

In this section, I examined several loss functions: MSE, CoS, MM, and VMF. I found that CoS optimally performed on top_x and T_x, which may have to do with the type of decoding involving angular distance that was applied. VMF was shown not to be fruitful for a normalized vector space. Additional architectures could be explored in the future for retrieval purposes, and VAE could be approach to consider in pursuit of this goal.

5.6 Summary

In this chapter, I presented several means to enhance the prediction outcome over the metrics of interest. The first is model interpolation, wherein both models were combined and could be compensated by the other model's strengths. Context-dependent combination approaches were mostly effective for this, though the context-independent approacher was easier and less costly. Second, I found that the feature-based decoding increased the diversity (and not the types) and revealed the underlying mechanisms these models prioritize. Third, it was shown that word2vec created embedding spaces that produced more fruitful predictions both in prediction accuracy and prediction diversity. I also investigated the GloVe model characteristics to hypothesize about the cause of its sub-par performance. Finally, in the loss function search, I compared the outcomes of the various models and reasoned why the Von-Mises Fisher loss was less appropriate for the type of spaces (and models) presented. Each section discussed possible directions for further enhancing or learning about the

[10]The method is described in section 2.3.4.

models. This chapter looked into technical optimizations when employing a continuous model. Such optimizations can subsequently affect the experience or the strengths of the continuous approach when choosing the next icon/word. In the next chapter, I will move away from these models and focus on exposing the problem of diversity in SotA models, as well as the possible consequences for for failing to consider diversity of predictions.

Chapter 6

Capturing Diversity with new Evaluation Metrics[1]

"Strength lies in differences, not in similarities"

Stephen R. Covey

6.1 Introduction

Language models are foundational components in many NLP systems, and as such it is crucial to be able to empirically evaluate their behavior. Traditionally, language models are evaluated using performance metrics that relate to the model's ability to accurately predict words given some context (e.g., perplexity). Following the paradigm described by Galliers et al. (1993), this can be thought of as an *intrinsic* evaluation criterion (and perplexity an intrinsic metric), as it relates to the *objective* of the language model itself.

In recent years, it has become common to also evaluate language models *extrinsically*, in terms of the model's *function*. This is done by measuring a model's performance when used as a component in a downstream task.[2] For example, Devlin et al. (2019) evaluated BERT in terms of its performance when used as the language model component in benchmark tasks such as question answering and "commonsense inference."[3] This shift towards extrinsic and task-oriented evaluation is welcome, and has the potential to make language model evaluation more ecologically valid.[4] As useful as task-oriented evaluation metrics are, however, we believe

[1] A disclaimer: parts of this work were not written by me, but written by Steven Bedrick, my advisor who I collaborated with on this research. These include the introduction section and the methods section. I conducted the experiments and reported the results

[2] In part, this trend has been driven by the increasing use by developers of language models of downstream tasks as ancillary training objective functions, which renders problematic the classical notion of intrinsic and extrinsic evaluation as a binary construct.

[3] SQuAD versions 1.1 (Rajpurkar et al., 2016b) and 2.0 (Rajpurkar et al., 2018), and SWAG (Zellers et al., 2018), respectively, in the case of the original BERT paper.

[4] "Ecological validity" is a dimension of experimental validity that is concerned with the question of whether an observed effect reflects "what happens in everyday life"(Brewer et al., 2014), i.e. beyond the artificial setting of the experiment itself. In an NLP context, a researcher working on question answering who was concerned with ecological validity would ensure that the questions on

that this approach brings with it certain practical limitations, and that there remains a strong need for robust and meaningful intrinsic evaluation metrics that can be used to characterize and compare the performance of language models.

In this work, we outline and propose a variation on the standard next-word-prediction language modeling task that is designed for use in evaluating and comparing language models and is robust to implementation differences (tokenization method, etc.) that complicate the comparison of modern models in terms of token-level predictions. Our proposed metrics are richer and more meaningful measures than traditional intrinsic metrics such as perplexity, which is insensitive to *which* tokens are matched, and as such may be confounded by distributional properties of their evaluation corpora. Our approach accounts not only for the *accuracy* of a model's word predictions, but also the *diversity of types* that it predicts, across different lexical frequency bins. We further propose a formulation for the next-word-prediction task that explicitly allows for language- and task-level details to be captured in the resulting metrics, thereby blurring the line between intrinsic and extrinsic language model evaluation. Our methods provide greater ecological validity than traditional intrinsic evaluation methods, while still remaining simple to interpret and easy to calculate.

6.1.1 Formalities: Language Models and Word Prediction

For our present purposes, we will consider a language model to be a model that, given a sequence W of n tokens $w_{1:n}$ from a fixed vocabulary of types V, estimates the joint probability of $P(W)$. There are many different ways that such a model can be constructed and trained, ranging from traditional count-based methods (often originally credited to Shannon (1948a)) to more modern neural approaches (Bengio et al., 2003; Peters et al., 2018b). While the methods of building such a model vary widely, they have in common the goal of learning to approximate the distribution of tokens and types in some corpus.

Importantly, different models may use different units of prediction (i.e., the entries in V may correspond to different linguistic constructs). A character-based model (such as that found in many mobile text entry keyboards) predicts at the level of a individual grapheme; the tokens in a word-level model correspond to individual words (possibly in an inflected form, as appropriate). For reasons of efficiency and robustness, many modern neural models actually predict at the level of a sub-word/sub-sentence unit (via e.g. byte-pair encoding (Sennrich et al., 2016), wordpieces (Wu et al., 2016a), etc.).

Given such a model, we can typically also estimate the *conditional* probability distribution $P(w_t|w_1 \dots w_{t-1})$ over possible words occurring after a given history h consisting of t tokens. We refer to this as the next-word-prediction problem[5] of predicting $\hat{w}_t = \text{argmax}_w P(w|h)$. Using the terminology of conditional text generation, this is akin to generating a single token via greedy decoding given a context. This is of more than theoretical interest from a language modeling perspective. Language models trained using the standard cross-entropy loss function are in effect being optimized to perform this very task, and furthermore, many NLP applications rely

which they trained and evaluated their system (in form and content) match the ones on which the system was designed to be used.

[5]Also known as the "Shannon Game" (Shannon, 1951).

in practice on effective and robust word prediction.

A standard and widely-used metric for evaluating language model performance is with *perplexity* (PPX), which is closely related to this prediction task. When computed for a given token prediction event by a language model, PPX captures how "predictable" that event was for the model:

$$PPX(p, q) = -\sum_{X} q(x) \log p(x) \tag{6.1}$$

where X corresponds to V (the model's vocabulary of possible tokens it must choose between), $p(x)$ represents the "true" or "target" distribution and $q(x)$ the model's estimated distribution. The closer the predicted distribution matches the target distribution, the lower the perplexity. When averaged over many prediction events, and computed on a held-out test dataset, perplexity attempts to capture the degree to which the model has optimally learned to represent its target distribution. A more accurate (i.e., "better") model should result in lower average perplexity (as the model will more often predict a high probability for the correct target), whereas a less accurate model should be reflected in a higher perplexity.

6.1.2 Evaluation Considerations

Perplexity is a classic example of an *intrinsic* evaluation metric, in that it is measuring the model's ability to carry out its immediate objective. As mentioned previously, modern language models are often evaluated according to their performance when used as components in a downstream task of some kind.[6] We find this increasing prevalence of *extrinsic* evaluation to be a very positive development, and do not in any way wish to argue *against* use of downstream tasks for evaluation. However, we see several limitations to an extrinsic-only evaluation paradigm, and argue for more robust intrinsic measures.[7] Extrinsic evaluation is necessarily dependent on the selection of specific benchmark tasks to include, and this process is fraught with difficulty, for several reasons.

First, there are many possible benchmark tasks from which one could choose, each attempting to measure something different. Different authors will naturally choose different combinations of tasks when evaluating their language models, as they may be focused on different aspects of their models' behavior. While scientifically appropriate, this does make for a heterogeneous evaluation landscape, and complicates comparisons between published results.

Second, new tasks are constantly being created, and existing tasks are regularly updated. This results in a complex and unstable evaluation landscape in which evaluation tasks change from year to year, and allows for much confusion around versions of datasets and benchmarks.

Third, downstream NLP tasks and datasets often have their own issues around validity. For example, the commonly-used SNLI natural language inference corpus (Bowman et al., 2015a) was later found to have substantial issues resulting from artifacts in how its annotations were collected (Gururangan et al., 2018). How

[6] Galliers et al. (1993) refer to this as the model's "function" (in contrast to its "objective.")

[7] In this, we follow Ito et al. (1999), who wrote about language models in the context of their use in automated speech recognition systems, warned against relying solely on evaluation metrics that were specific to that task (specifically, word error rate).

should one now assess a language model evaluated using this downstream task, knowing that the metrics may be of very limited validity? Finally, we note that widely-used and well-studied downstream evaluation tasks are often not available in "low-resource" languages, and so may not be an option in many scenarios. For these reasons, we believe that intrinsic measures should still play an important role in language model evaluation.

The question then becomes that of *what* to measure. Perplexity has the advantage of being well-understood and easy to calculate, and is closely linked to the standard cross-entropy training loss frequently used in language modeling. However, it has long been observed that perplexity itself often fails to correlate with downstream task performance (Iyer et al., 1997; Ito et al., 1999), suggesting that it may have limited external validity as a metric.

There is an additional, more subtle limitation to the use of perplexity in cross-model comparison. As previously mentioned, many modern language models use sub-word units of prediction. One of the consequences of this heterogeneity is that evaluation metrics that relate to individual base-level prediction events (as is the case with perplexity) are not comparable across models, even if they are trained and evaluated on the same corpus: different tokenizations and vocabularies will result in different numbers of prediction events, as well as a differently-sized space of possible choices at each event. From the perspective of the perplexity metric, two models with different approaches to tokenization are performing fundamentally different and numerically incomparable tasks.

Beyond this statistical problem, there is a problem with the underlying semantics of using perplexity as a measure when working with sub-word units. Any actual application of a language model that involves explicit word prediction[8] will ultimately demand not *fragments* of words, but rather *entire* words. In other words, even models whose native unit of prediction is at the sub-word level must make predictions that can *eventually* be able to be decoded into whole words at *some* point.

Given that, raw perplexity becomes a somewhat confusing evaluation metric, as the underlying phenomenon that it is measuring is quite distinct from the model's actual objective (i.e., predicting a whole word). Imagine, for instance, a model that predicts at the sub-word level, and now must predict a word given the history "*The tyrannosaurus was chased by the.*" The correct continuing word is "*velociraptor*", and under the sub-word tokenization used by this model, this will necessitate several separate prediction events (as "velociraptor" is both a long and an infrequently-occurring word). The model may or may not be able to correctly make each of those several predictions; presumably, the first sub-word prediction will be the most "challenging" for the model, and the mid-word units "easier." From the perspective of the perplexity metric, however, there will be no difference between the first unit or the third.[9] Whatever the perplexity metric is telling us about the model's behavior during this process will likely tell us little about the model's ability to actually predict "velociraptor" given this particular word history.

[8] For whatever definition of "word" is appropriate in the language under consideration.

[9] Or, for that matter, from the previous token, "*the.*"

6.1.3 Recentering on Words

We propose that intrinsic evaluation of language models be done in terms of the *whole-word* prediction task, regardless of the specific tokenization practices of any particular model. This would have the advantage of making cross-model comparison easier, and of the resulting metric bearing a closer resemblance to what we intuitively expect such a metric to capture (i.e., the model's performance at its primary objective). While computing perplexity at the level of whole words (see section 6.2) is a step in the right direction, we also propose several additional intrinsic metrics relating to the word prediction task.

Word Prediction Accuracy: We propose directly measuring and reporting the model's raw *accuracy* at word-level predictions (i.e., the proportion of words that were predicted correctly). This has the advantage over perplexity of grounding the number more closely to the concrete performance objective that we are concerned with. Furthermore, it is easily extended to account for various attributes of model behavior that may be of interest in terms of downstream tasks, while still remaining in the realm of intrinsic evaluation.

In the experiments we describe in section 6.2, we experiment with variations on this metric that capture different notions of "accuracy." For example, we explore "top n" accuracy (i.e., if the target word is in within top n most likely predictions, that prediction counts as a "hit"). This could be of use in a text entry scenario, in which the model is responsible for generating candidate words for further selection or refinement by an end-user (as in a mobile phone keyboard application, or an Augmentative and Alternative Communication (AAC) system for individuals with motor and/or speech impairments). Many other possible downstream tasks for language models involve techniques that would also benefit from having the target word given better placement in the ranked prediction space, and thus would benefit from a metric explicitly measuring this property.

"Soft-Match" Prediction Accuracy: We propose extending simple prediction accuracy to allow for "near miss" predictions, where the predicted word is "similar" to the target (for a specified definition of "similar"). In many applications of language modeling, there may be multiple possible valid predictions. This problem has long been understood in the context of machine translation evaluation; in their description of the motivation behind the METEOR metric, Lavie et al. (2009) addressed the "problem of reference translation variability by utilizing flexible word matching, allowing for morphological variants and synonyms to be taken into account as legitimate correspondences." In a word prediction task, we could allow an explicit synonym to count as a correct prediction; depending on the application or domain in question, one could use external language resources to model much more complex and task-specific notions of similarity (e.g., in a biomedical NLP context, one might give the model credit at evaluation time for predicting a medication that is from the same functional class as the target). In the experiments described in section 6.2.4, we use a method based on word neighborhoods in an embedding space. Depending on the nature of the task under consideration, other features could be used.

Consider a typing task in a morphologically rich language, in which a user might be willing to accept predictions that involve the correct lexeme but with an incorrect inflection. Allowing for this sort of flexibility in the evaluation of a word prediction model has the potential to greatly increase the ecological validity of the experiment, in that, that the experimenter is able to easily encode their own task-specific notions

of relevance while still staying in a fairly constrained and easy-to-analyze evaluation setting.

Lexical Frequency & Diversity: One important limitation of raw classification accuracy as a metric is its susceptibility to being biased by imbalanced class distributions. For example, if some classes occur much more frequently than others, a model may achieve a high accuracy score by learning to focus on these frequent classes to the exclusion of infrequent ones. In written language, the distribution of classes (i.e., of word *types*) are notoriously skewed (Zipf, 1935), and exhibit a "long tail" of words that occur relatively infrequently, with a small set of "head" words that make up a large proportion of individual *tokens* observed in the training and test data.

We observe that language models often exhibit very different performance characteristics when predicting more common types than less common types; in fact, our experiments in this paper demonstrate that, for some commonly-used language models, the actual number of infrequent types that are *ever* correctly predicted is surprisingly small (see section 6.3.1). This over-emphasis on frequent types, when carried forward into downstream generation tasks, may lead to the failure mode described by Holtzman et al. (2020) in which generated text is "dull and repetitive." This phenomenon is not limited to words alone; morphologically-rich languages (MRLs) exhibit a similar Zipfian distributional pattern in terms of the occurrence of different morphological phenomena, which in turn affects the performance of systems designed to process such features of language (Czarnowska et al., 2019; Tsarfaty et al., 2020). We believe that this behavior can be explained through the lens of the bias-variance tradeoff common to all statistical learning problems. As observed by Lazaridou et al. (2015), neural models have a tendency towards the "bias" end of that tradeoff, which in the context of language modeling results in a strong preference for head words and against tail words.

This is a serious enough problem in machine translation and text generation systems that there is a growing body of literature looking at ways to increase the lexical diversity in model output. Some authors (Li et al., 2016a; Welleck et al., 2020) have examined training strategies and loss functions that optimize for diverse output, while others (Vijayakumar et al., 2016; Ippolito et al., 2019) focus on alternatives to greedy decoding and identify several ways to generate more diverse sequences of words. Questions of evaluation arise, as the construct of "diversity" itself is surprisingly difficult to characterize, as pointed out by Tevet et al. (2020).

In the context of our word prediction task, we propose two evaluation measures that account for the Zipfian skew in type distributions, and illuminate differences in model performance across the type frequency spectrum. We will *stratify* our evaluation of prediction accuracy by frequency similar to previous chapters (see section 3.4.4 as an example)Second, will measure the *overall proportion of possible types* that the model was able to predict at least once during evaluation (similar to one presented in 3.3.6 though in this chapter this metric is based on proportions).

6.2 Methods

In this section we describe a series of experiments in which we use our proposed evaluation metrics to explore the behavior of several widely-used and large-scale language models (obtained using the HuggingFace (Wolf et al., 2019) Transformers library).

Specifically, we examine GPT-2 (Alec et al., 2019) (`gpt-2`), GPT (Alec et al., 2018) (`openai-gpt`), BERT (Devlin et al., 2019) (`bert-base-uncased`), RoBERTa (Liu et al., 2019) (`roberta-base`).

6.2.1 Training & Datasets

Since the pre-trained models were all trained in widely varying ways on different corpora, we ran each model through a single pass of fine-tuning on a common corpus to attempt to bring them more closely into alignment. For this fine-tuning (and for the ensuing experiments), we used WikiText 103 (Merity et al., 2016), which consists of a large ($n = 28,475$) training set of English-language Wikipedia articles and a small ($n = 60$) test set of 60 articles, with one sentence per line. The fine-tuning task was on a word prediction task in a unidirectional fashion, in which the context is based only past history (i.e., not on future tokens).[10] We note that for BERT and RoBERTa, this usage does differ somewhat from the prediction paradigm under which they were trained, which is implicitly bidirectional.

6.2.2 Whole-word decoding

As previously described, modern language models typically use sub-word/sub-sentences units as their native unit of prediction. In order to perform a meaningful evaluation of cross-model word prediction accuracy, it is necessary to obtain word-level predictions, which for the mentioned models may involve more than one model-level prediction event. The models we worked with in this set of experiments used two different tokenization strategies (wordpieces for BERT, and GPT, and BPE for RoBERTa and GPT-2), and as such we developed algorithms for decoding whole words by sequentially decoding individual sub-word units. While the algorithms differ slightly in their implementation between the model families, the overall method is similar.

Our single-word decoding algorithm shown in Algorithm 1, extracts the first word candidate by the model through concatenating tokens until *end-of-word* is indicated[11], and then compared with a target word.

To extract multiple candidate words, given a target word we run a Depth-First Search to find whether a valid path of tokens exist, having each model prediction spanning its top ten guesses. Our pseudo-syntax is presented in Algorithm 2. We note that once a possible sequence is found, contrary to Algorithm 1, we do not enforce end-of-word for finishing the sequence, if that particular token is missing from current unit. This is not a typical beam-search based on likelihoods, rather based on existence of valid units (in first K options) for a given target word, simulating user choices given a context.

In addition to decoding whole words, we would like to be able to obtain a probability estimate of the resulting prediction, for use in computing a word-level perplexity measure. We approximate this by taking the product of the prediction-level

[10]Bert and Roberta were given a '[MASK]' token at the end of a sequence for unidirectional prediction

[11]the code is available for all model types we present in this paper, and for the different tokenization approaches they are trained by

Algorithm 1 Top1 Target Word Search

```
1: procedure TargetFind1(w_<t, w_t, model)
2:     cxt ← w_<t // set cxt as the original context for the current word prediction
3:     word ← empty
4:     while str(word) in str(w_t) do      // is word a valid prefix of w_t (target)
5:         cxt ← Cat(cxt, word)            // add the valid prefix to the context
6:         P_wpc_t ← Predict(cxt, model)   // predict the next token
7:         top_1 ← ArgMax(P_wpc_t)         // take argmax
8:         word ← Cat(word, top_1)         // concat token to prefix
9:         if str(word) = str(w_t) then    // check for complete match
10:            return True
11:        end if
12:        if str(word) not in str(w_t) then   // prefix is not part of target word
13:            return False
14:        end if
15:    end while
16: end procedure
```

Algorithm 2 TopK Depth-First Search

```
1: procedure TargetFind2(w_<t, w_t, model)
2:     cxt ← w_<t // set cxt as the original context for the current word prediction
3:     P_wpc_t ← Predict(cxt, model)       // retrieve likelihood distribution
4:     top_10 ← Top10(P_wpc_t)             // get top 10 likelihoods
5:     for root in top_10 do               // find valid prefixes
6:         roots.append(root)
7:     end for
8:     for root in roots do                // a root is a valid prefix token
9:         paths ← List(root)
10:        while paths do
11:            path ← Pop(paths)           // the prefix of the predicted word
12:            basic_cxt ← w_<t            // original context
13:            cxt ← Cat(basic_cxt, path)  // concat both context and path
14:            P_wpc_t ← Predict(cxt, model)   // predict the next token
15:            top_10 ← Top10(P_wpc_t)
16:            for wpc_t in top_10 do      // is any top_10 guesses forms a valid prefix
17:                word ← str(wpc_pre, wpc_t)
18:                if word = str(w_t) then     // check for complete match
19:                    return True
20:                else if word in str(w_t) then   // add for partial match
21:                    new ← Cat(path, wpc_t)
22:                    paths.append(new)
23:                end if
24:            end for
25:        end while
26:    end for
27:    return False                        // all possible paths were exhausted
28: end procedure
```

probabilities (i.e., the model's estimate of the probability of each constituent unit in a given decoded word), which we can then use for a perplexity-like score:

$$ppx = -\sum^{words} q(w) \log(\prod^{units} p(u)) \tag{6.2}$$

If a word was not found it is assigned a fixed probability reflecting a low likelihood.

6.2.3 Experiments

We performed a word-level prediction experiment on the test dataset described in section 6.2.2, using each of the models in section 6.2. For each test example, we performed incremental unidirectional word prediction using Algorithm 1 to generate whole-word predictions. In other words, for each test example W comprised of $w_1 \dots w_n$ words, we queried the model $n-1$ times, to predict $\hat{w}_i = \underset{w}{\text{argmax}} P(w_i | w_{1:i-1})$ for $i \in [2, n]$. Additionally, we used the algorithm previously described to decode the top k ranked word predictions (for $k = 10$), $\hat{\mathbf{w}}_t^k$. In other words, for the test input "the dinosaur ate the ..." we would sequentially predict $p(w_2|$"the"$)$, $p(w_3|$"the dinosaur"$)$, and so on. At each prediction event, we compared the predicted $\hat{w}_t$ to the ground-truth w_t according to the various metrics described in the next section. We counted as "hits" word-level prediction events where the comparison matched (for the different definitions of "matched"), and "misses" otherwise.

6.2.4 Calculation of Metrics

Prediction Accuracy

We measure token-level prediction accuracy[12] using an exact-match criterion, top_1. In other words, if $w_t = \hat{w}_t$, a "hit" is counted; otherwise, a miss. We also computed a higher-recall metric top_k, in which a "hit" is counted if $w_t \in \hat{\mathbf{w}}_t^k$ — i.e., if the target word is in the top k predictions, it counts as a "hit." For our experiments, we computed top_{10} (i.e., $k = 10$).

Soft-Match Accuracy

As described in section 6.1.3, there are a number of criteria by which one might implement a soft-matching algorithm. From the perspective of evaluation, the key is to design a criterion in such a way as to capture the aspect of user behavior that one may wish to support.

We performed our soft-matching experiments with a text entry scenario in mind, in which a user is able to choose among the language model's top n predictions. Under this scenario, if the model fails to predict the target word but instead predicts a *related* word (a synonym, perhaps), the user may still be able to convey their message. To simulate this, we may define the soft-match operation as follows:

[12]For tokens— i.e., words— in the test set, as opposed to tokens from the perspective of the model being evaluated.

$$\text{SoftMatch}(a, b, s) = \begin{cases} a = b & True \\ is\_sim(a, b) & True \end{cases} \tag{6.3}$$

Where the arguments to SoftMatch are two candidate words a and b, as well as a similarity function s. Softmatch(a, b, s) is true if a and b are a match, or if s indicates similarity. For our experiments here, we used a method based on similarity in word embedding space, under the theory that words with similar embeddings may be (relatively) appropriate substitutions in a word prediction task.

We used the word2vec algorithm (Mikolov et al., 2013b) to train 50-dimensional word embeddings on the "train" subset of the WikiText-103 corpus. We then defined our softmatch similarity function $s_{knn}(a, b) = a \in knn(b)$, where $knn(b, k)$ retrieves the k nearest neighbors of target word b in the embedding space. Using our softmatch function, we then re-scored the prediction accuracy such that a positive softmatch counted as a "hit." We used the Annoy library (Bernhardsson, 2018) to perform efficient nearest-neighbor retrieval. We conducted experiments in which we varied the k parameter; in other words, by allowing a match deeper into the k-nearest neighbors of the target. Our motivation for this was that, if all things are being equal, a model that mis-predicts a target but at least guesses something that lies in the right semantic neighborhood is more useful than one that does not.

Lexical Diversity

In order to measure type diversity given all the hits in top_1/top_{10}, we counted how many unique types were correctly predicted for first and top ten guesses and present it in T_1 (T_{10}) respectively. To illustrate the utility of measuring the rate of unique types that were correctly predicted, consider a hypothetical dataset in which 20% of the tokens consist of the word *the*, and that the model at hand predicts only this word for every sample in the test set. In this scenario, top_1 accuracy will be 20%, as *the* is a correct prediction for 20% of the times, yet T_1 is based on only one type[13] as there was only a single type that was correctly predicted— suggesting sub-optimal learning of the input distribution, or a lack on the model's parameters of the ability to reflect that distribution during test.

6.3 Results

6.3.1 Diversity Evaluation

model	top_1 (top_{10})	T_1 (T_{10})	ppx
`GPT-2`	35.63 (67.76)	26.60 (47.27)	34.8
`GPT`	29.37 (60.89)	15.96 (30.80)	37.9
`RoBerta`	28.18 (59.55)	24.73 (42.63)	42.2
`Bert`	22.11 (50.98)	15.59 (29.61)	50.7

Table 6.1: Experimental results on Wiki-103 corpus.

[13] T_1 is the relative percentage of one over the overall number of types

Table 6.1 shows the results for the different models. `GPT-2` and `GPT`, which were pre-trained on a word prediction task, exhibited the lowest *ppx*. `GPT-2` had the highest hit rate and type diversity. However, when comparing `GPT` to `RoBerta`, while the models' accuracy seems to be similar, the *ppx* is lower for `GPT`. Moreover, `RoBerta` was found to be much more diverse than `GPT`, suggesting that the reasons for similar hit rates (28.18, 29.37) are different, as shown by the different T_x metrics. On the other hand, while `Bert` and `GPT` had similar diversity rates, `GPT` had higher accuracy than its counterpart, making for a different accuracy/diversity ratio than that of `Bert`. This different ratio may also suggest different prediction behaviors for `GPT` and `Bert`. Understanding type diversity requires addressing which types are predicted well and which types are harder to predict. Thus, I stratified both T_1 and top_1 as a function of frequency: *high*, *mid*, and *low*, for $x \in [10^3, \text{inf})$, $x \in [10^2, 10^3)$, and $x \in [10^1, 10^2)$, respectively, where x is each target type's frequency.

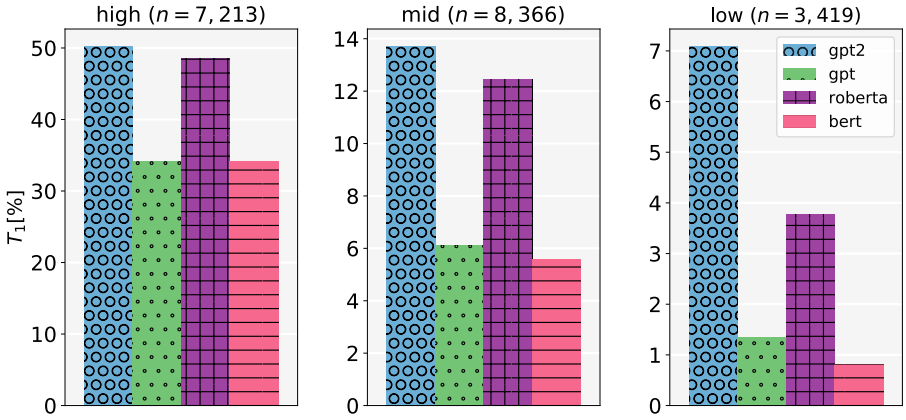


Figure 6.1: Wiki-103 *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

Figure 6.1 illustrates the models' type distributions. High diversity was found for both `GPT-2` and `RoBerta`, although `GPT-2` picked from the low bin twice as often as `RoBerta`. Notice the stark difference between `RoBerta` and `GPT`. `RoBerta` outperformed `GPT` across every bin, illustrating its diversity strength (given the models' similar hit rates shown earlier). Although they performed worse than `RoBerta`, both `GPT` and `Bert` had similar rates of diversity, with `GPT` performing almost twice as well on the lowest bin. Finally, even `GPT-2`, which attained the highest diversity, covered only 50%, 14%, and 7% of the trained types on which it was evaluated. This shows there is room for reflecting the input data more optimally.

Figure 6.2 presents the hit rate distribution. This figure explains the gaps between `GPT-2` and `RoBerta`, showing that while the models were not so different in diversity, `RoBerta` missed hits mostly from the most frequent bin, with a 10% gap, and had sub-optimal prediction in the mid- and low-bins. The similar hit rates of `RoBerta` and `GPT` are clearly distributed differently, as `RoBerta` reached parts of the long tail of the distribution more often than `GPT`. `Bert` and `GPT` exhibited the biggest gap in the most frequent bin with an 8% difference, while the mid and low bins were similar. Overall evaluations of diversity predictions can shed light on models' priorities. Measuring type diversity reveals that models that have similar hit rates can vary immensely in diversity, which may impact downstream tasks. Evaluating diversity may not only indicate to what degree the learned distribution is reflected,

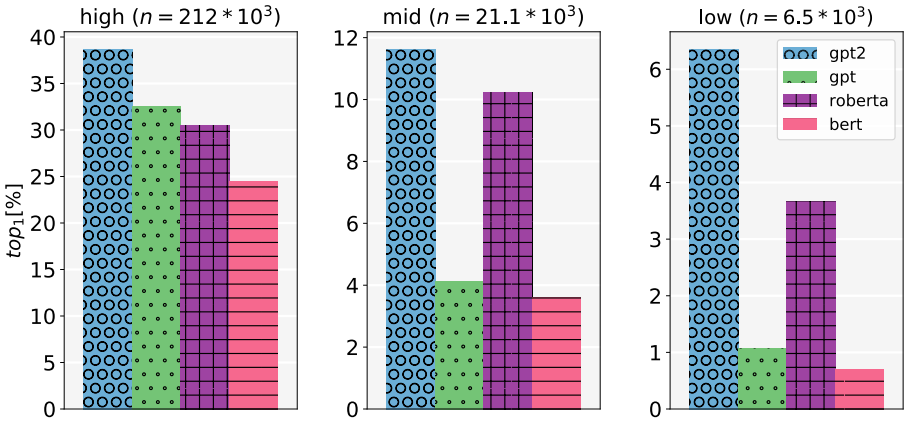


Figure 6.2: Wiki-103 *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

but may also directly point to the missing types and the weaknesses of the model. All the models in this experiment were shown to be weaker in the lower bins or biased by frequency, a problem that is necessary to start addressing. Addressing this problem would indirectly contribute to higher accuracies as well. In section 6.4, I use a case study to illustrate why learning diverse types, and low-frequency types in particular, can be useful. Next, I present a way to further understand models, even if they do not directly find the target word in a prediction.

How effective was the fine-tuning for types and hits?

In this part, I measure the degree to which a single-pass fine-tuning improved type diversity as well as type accuracy. The answer to this question is necessary to determine to what degree the single-pass protocol was useful for improving the pre-trained models' performance on the tested data. An insufficient degree (expressed in poor improvement over the metrics) requires the consideration of a different protocol (more than a single-pass round, for instance). The reason a single pass was chosen is that it has the steepest learning curve with the lowest costs possible to runtime. Table 6.2 shows the results for the *pre-trained* models (before fine-tuning took place). The four models presented in Table 6.1 are compared to the Δ gains after fine-tuning.

model	top_1	(top_{10})	T_1	(T_{10})	ppx
`GPT-2`	6.84	(5.42)	4.27	(3.4)	-4.9
`GPT`	5.47	(8.18)	3.67	(5.15)	-8.8
`RoBerta`	19.05	(16.53)	12.49	(9.54)	-16.0
`Bert`	17.97	(30.58)	14.22	(18.44)	-30.7

Table 6.2: Δ change from pre-trained models to the Wiki-103 fine-tuned models on the Wiki-103 test (with regard to Table 6.1).

Table 6.2 reveals that the raw gains for the bidirectional models `RoBerta` and `Bert` were much larger when fine-tuned compared to those for `GPT-2` and `GPT`. The *ppx* values also decreased more dramatically in the bert-based models. The models'

performance before and after fine-tuning can be observed from a different angle through the % change from the pre-trained models shown in Table 6.3. The percent change was more noticeable for `RoBerta` and `Bert` than the other models, with a 530 increase resulting from learning.

model	top_1 (top_{10})	T_1 (T_{10})	ppx
`GPT-2`	124 (109)	119 (108)	-86
`GPT`	123 (116)	130 (120)	-81
`RoBerta`	309 (138)	202 (129)	-72
`Bert`	530 (250)	1138 (265)	-62

Table 6.3: Δ in % change from pre-trained models to the Wiki-103 fine-tuned on the Wiki-103 test (with regard to Table 6.1).

It was interesting to consider how the observed gains were distributed, breaking down the types by word frequency as done before. To this end, Tables 6.3 and 6.4 show improvement as a function of word frequency. The biggest gains were in the highest frequency bin. While `GPT-2` and `GPT` exhibited relatively smaller changes in performance across the bins, `Bert` gave no evidence of any Wiki-103 low-frequency bin (and very little of any mid-frequency bin) words that were shown during the test prior to fine-tuning. In other words, `Bert`'s overall performance can essentially be attributed to fine-tuning. Fine-tuning also dramatically improved `RoBerta`'s performance, with about half of the types attained in post-Wiki training in the highest bin and more than two-thirds and three-quarters in the mid- and low-frequency bins, respectively.

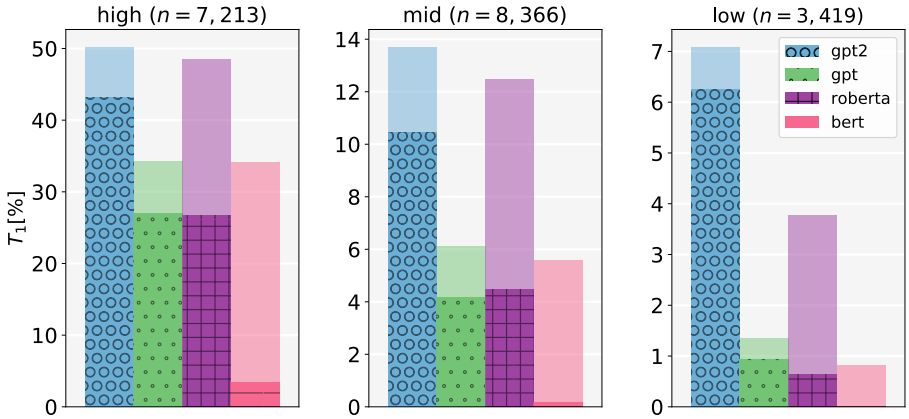


Figure 6.3: Wiki-103 *type* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

Figure 6.3 shows that before fine-tuning, `GPT` and `Bert` were not close in number of types, as seen in Table 6.1. Figure 6.4 shows that `RoBerta` and `GPT` were farther away in terms of hit rate before fine-tuning as well.

6.3.2 Soft-Match Evaluation

Figure 6.5 illustrates `GPT-2` and `GPT`'s T_1 and top_1 performance on the left (bars) and right (line) axes. Both models gradually (@3-@100) captured more types as

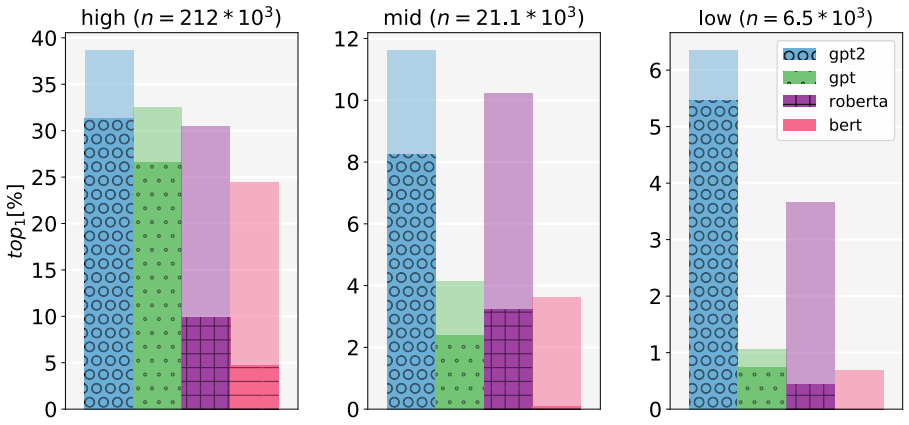


Figure 6.4: Wiki-103 *token* coverage by training frequency bin. n: number of items in each bin; y-axes are percentages over n (note different scales).

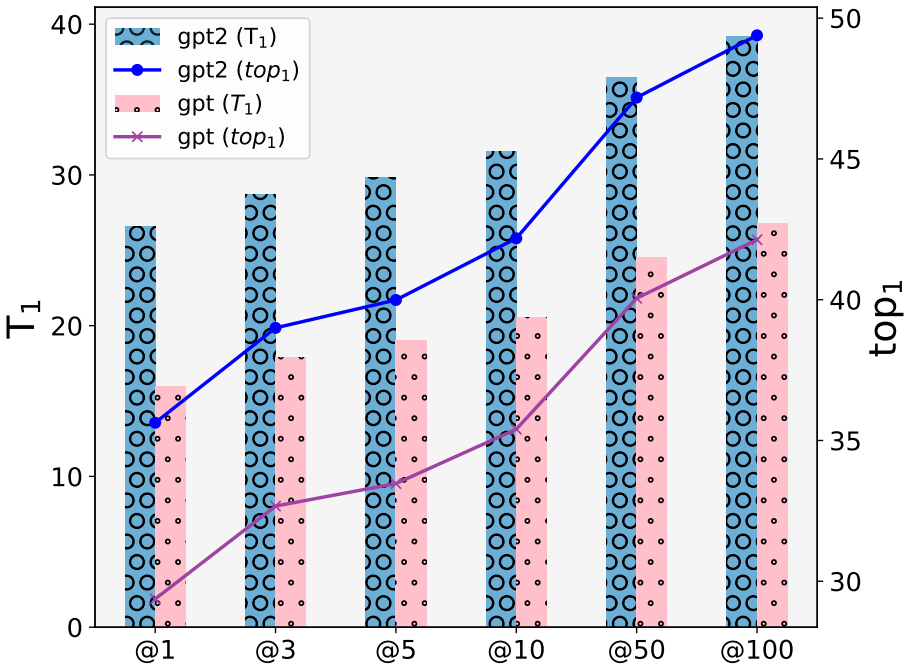


Figure 6.5: Soft-match for `GPT-2` and `GPT` (Wiki-103).

the beam of k in *knn* was increased (considering more target neighbors), leading to increased hits. This evaluation shows that `GPT-2`'s exact matches (@1) were higher but that its misses enriched the pool of unique types with 14% (@100) additional unique types (light blue), whereas `GPT-2` covered only 11% more types (light pink). Both models' increases in accuracy were similar. This reinforces that the models' prediction mechanisms are slightly different, as similar gains in accuracy were translated to either more learned high types or more diverse patterns, as shown in Figures 6.1, 6.2. This analysis shows that although there were mismatches, some of them were actual near-misses and were related to what the models were expected to predict. This can be of practical use for different users, including for those who wish to analyze how wrong the mismatches were as part of an error analysis process.

6.4 A Case Study of Paraphrasing

In this section, I look at the impact of model inference performance on the particular downstream task of paraphrasing. To this end, I employed a state-of-the-art algorithm, Bertscore (Zhang et al., 2019), to compute the similarity scores of sentence pairs in part by comparing embeddings derived from a language model. Under Bertscore, a higher similarity score indicates the greater semantic similarity of a pair of sentences, such that one is a close *paraphrase* of the other. Note that the critique that arises at the end of this section is not about the Bertscore tool as such, but is rather about a certain type of pattern that the models employed by this tool may insufficiently learn.

Why Paraphrasing? I chose the downstream task of paraphrasing to measure the semantic similarity of sentence pairs as it can be easily manipulated to consider a single word modification. Consider the following example sentence involving the word triple (poodle, dog, cat): [14]

(a) which `dog` has longer hair ?

(b) which `cat` has longer hair ?

(c) which `poodle` has longer hair ?

The pair (a, b) ought to score lower (i.e., be considered by Bertscore to be more dissimilar) than the pair (a, c), as a is a valid paraphrase of c while b is not. This, of course, assumes that the language model being used as the underlying source of embeddings for the Bertscore algorithm has accurately captured the semantic meaning of the three words under consideration. If not, there may be an inversion of results such that (a, b) (incorrectly) appears to be more similar than (a, c), suggesting that the model in question should perhaps not be used for paraphrase-related tasks.

To explore the impact of word frequency on model representations with this chapter's fine-tuned models, I generated 50 rare and 50 common triples from the Wiki-103 trainset. Each of the triples contains a rare/common word, its hypernym, and a sibling hypernym extracted from WordNet (Miller, 1995) (using `nltk` (Loper et al., 2002)). The hypernym and its sibling hypernym are represented by (x_r, x_h, x_a) and (x_c, x_h, x_a), respectively. For each word in a triple, I identified a sentence in which the rare word naturally occurs and generated probe sentences in which I replaced the rare word with x_h and x_a.[15]

I then used *bertscore* to compare the sentences in terms of their similarity. In principle, one would expect that $bertscore(s(x_h), s(x_r)) > bertscore(s(x_h), s(x_a))$. In other words, the similarity score for the pair made of a sentence with `dog` and a sentence with `poodle` is expected to be higher than that of the pair made of a sentence with `dog` and a sentence with `cat`. This is because `dog` and `poodle` are closer semantically than `dog` and `cat` and therefore are a closer paraphrase of each other. However, if the model's word representations are being confounded by lexical frequency, one might instead observe the opposite pattern (i.e., the sentences with the more common words mistakenly appearing to be more similar to one another despite the words' semantic difference). I considered cases in which the

[14] Note that the word `poodle` occurs much less frequently in English than either `dog` or `cat`.

[15] See Appendix A.3.1 for details on sentence selection and generation.

model correctly identified the paraphrase (e.g., if $bertscore(s(\texttt{dog}), s(\texttt{poodle})) > bertscore(s(\texttt{dog}), s(\texttt{cat}))$) as hits and misidentifications as misses. My "null hypothesis" was that there should not be any difference in hit rates between high- and low-frequency words (i.e., word frequency should not affect a model's ability to identify paraphrases). Furthermore, I compared the performance of two fine-tuned models trained on Wiki-103: `Bert` and `RoBerta`. Given the results of my earlier experiments, I hypothesized that if there is a difference in hit rate, `RoBerta` will prove more robust to the rare words condition, given its superior performance at predicting (and thus representing) rare words.

model	*hits*	*misses*	*total*
$\texttt{Bert}_{rare}$	14	36	50
$\texttt{RoBerta}_{rare}$	11	39	50
$\texttt{Bert}_{common}$	40	10	50
$\texttt{RoBerta}_{common}$	39	11	50

Table 6.4: Paraphrasing sentences with Wiki-103 words.

I note that this is something of a "toy" experiment given its small size, which limits the conclusions that can be drawn from it. However, Table 6.4 shows a difference in performance between the rare set of words and the common, such that the models do appear to have failed to capture the semantics of rare words, as reflected in the greater number of *misses* (χ^2; $p < 0.001$ for both models).

I found that `RoBerta` and `Bert` did not differ greatly in performance, suggesting that with this method, the strong effect of word frequency outweighed the between-model difference observed in my earlier experiments. This null result could easily be an artifact of the very small sample size of 100 probe sentences, though. I also noticed a substantial number of *misses* with the set of common words. Overall, although these results are based only on a small sample, it does seem that the poorer performance of both models on the rare words is unlikely to be a coincidence. Note that Raunak et al. (2020) suspected that the dot product of rare vectors in a next-token prediction task would fail due to the rare vectors' ineffective embedding representation learned for low-frequency words shown by Gong et al. (2018). In this section, I provide corroborating evidence for this claim. I hope to be able to experiment with a greater sample size to better elucidate the degree to which rare word inferences are reliable in producing outcomes aligned with human semantics on various downstream tasks.

How effective was the fine-tuning for the paraphrasing task?

In order to address this question, I ran the paraphrasing experiment described previously on the pre-trained models. Table 6.5 presents the raw results. There were small differences slightly favoring the fine-tuned model for rare words in `RoBerta` and a slight decrease for both models on common words. I assume that had the sample size been bigger, or alternatively had there been gaps between models, or if the gaps between the pre- and fine-tuned models had been larger, a stronger conclusion could have been drawn with regard to the condition that indicates better performance. The main conclusion, though, remains: rare words were found to be semantically similar to their hypernyms at a much lower rate than common words.

model	*hits*	*misses*	*total*
$\texttt{Bert}_{rare}$	14	36	50
$\texttt{RoBerta}_{rare}$	8	42	50
$\texttt{Bert}_{common}$	41	9	50
$\texttt{RoBerta}_{common}$	40	10	50

Table 6.5: Paraphrasing sentences with Wiki-103 words pre-trained model performance.

6.5 Future Work

The soft-match technique proposed in this chapter is aimed at finding *word* paraphrases, though a future direction might involve evaluating similarity in the meanings of generated *prompts*. In this case, the problem would be considered message recovery, permitting a minimal loss as long as the 'core' *intent* was conveyed (as is the case with lossy compression). In other words, is it possible to evaluate whether the generated text is semantically similar in the eyes of the reader?

Future evaluation metrics accordingly could benefit the approach developed by Shannon (1948a) introducing the noisy channel theory, in which the goal is to recover the source on the destination side with as little noise as possible. In this case, the focus would be on the *semantic* loss when recovering a message. In AAC, for example, a user ideally could opt for relaxing exact matches over characters, replacing them with close matches over the semantic space (i.e., the user's intent) of a phrase. This naturally poses challenges. The definition of close matches over semantics may vary across users and be highly context-dependent. I believe that focusing on correctly predicting semantics would be a critical improvement, especially in typing settings and AAC in particular.

When adapting to a user, one scenario to consider is that in which a user and their interlocutor have formed a code-book in which short words encode rich messages. On the one hand, an AAC system should optimize for the way a user may choose to use the system; on the other hand, employing a code-book may come at the expense of the system's semantic inference of this language (as it may be obfuscated from the system). Striking a balance across all of a user's needs is challenging, and the ability to do so ultimately determines the system's usefulness. Evaluation metrics, when designed for users, should reflect the dynamic/changing needs under which the users work; that is why there is no one-size-fits-all. Ideally, an individual who uses an AAC device would naturally choose to communicate in an urgent mode, a calm mode, a close-match mode, or another mode that is not discussed here. Finally, I propose another direction as a future work. An ideal system would need to adapt and learn directly from users' input, as the users would decide whether a message was recovered by their predicting system. Not only positive feedback but also negative feedback can improve the system. This would contribute to a continuous learning of the system.

Immediate future goals include conducting a larger-scale experiment of the paraphrasing task presented. Furthermore, I hope to continue evaluating language models' prediction diversity and its effects on additional downstream tasks (for example, tasks where human speech is anticipated) since prediction diversity evaluation may

vary from one task to another. There are other units of evaluation besides words; for instance, the unit of evaluation may be defined at various textual granularities, such as phrases, depending on the prediction diversity desired. I also leave for future work questions of to what degree different tokenization approaches or model sizes affect prediction diversity.

In my experiments, I did observe differences in performance between models with different tokenization strategies (e.g., `GPT-2` and `RoBerta` as compared with their architectural counterparts); however, these models also varied substantially from one another in other respects (e.g., size), and as such, it is difficult to attribute this performance gap to tokenization alone. It may also be the case that the bigger the model (in terms of number of parameters), the more diverse it is likely to be; under this hypothesis, one would expect today's ever-larger models (e.g., `GPT-3`) to outperform their predecessors in terms of diversity. However, I do not believe that it is sustainable (Strubell et al., 2019; Schwartz et al., 2019) to rely on increasing model complexity as an approach to addressing the frequency-related challenges that I observed in my experiments; rather, fundamentally different approaches to language model training are needed.

6.6 Limitations

While the principle of soft-matching was discussed in section 6.2.4 through a particular technique, there is more than a single way by which a soft-match evaluation can be realized. A possible limitation of employing Word2Vec has to do with the type of neighbors *knn* extracts from an embedding space. Neighbors may be found to be close due to frequency, as noted in Gong et al. (2018), or association of similar context, yet they may not necessarily be synonyms. An alternative approach would be comparing a prediction against a list of human-annotated synonyms from WordNet (Miller, 1995) or other languages' equivalents. If one developed a model for the medical domain, UMLS (Bodenreider, 2004) could be used to provide valid alternatives in a more restricted fashion.

One limitation of the described paraphrase experiment is its relatively small sample size (50 for each condition). The small sample size was adopted because of the need for a human-in-the-loop process for choosing appropriate sentences. In the future, the paraphrasing task could be conducted on a large set of triples to further validate the pattern observed in this chapter. Though I did not observe a meaningful difference between the two model types (such a difference could have indicated a fine-grain link between the intrinsic evaluation of type diversity and performance in paraphrasing), I presented evidence for an overall link between the degree to which rare words are predicted in a word prediction task and the rate of correct inference on rare words. Given that this subgroup consisted of reduced number of training examples, it may be necessary to look into this subgroup more closely (subgroup of infrequent types) if one of the main goals is to reflect the input distribution.

6.7 Conclusion

In this chapter, I presented two types of evaluation techniques by which to investigate the performance of a model across its input distribution, thereby revealing which areas are easier and more challenging for the model to learn. Through this analysis I showed that models are susceptible to training frequency during training and underperforms when less-frequent examples are tested, hurting the overall performance. In addition, I proposed a way to identify the degree to which a model is semantically close to a target when an exact match is not predicted, indicating a usefulness property. Finally, I showed how the downstream task of paraphrasing may be rendered less reliable, as the models employed struggle to produce true statements when rare words were involved.

I believe that models should reflect the trained distribution more optimally than they do currently. It is critical to recognize their bias toward frequency, which renders them unfair towards some words and potentially harmful for downstream tasks. I also believe it is important to participate in setting benchmarks for models' diversity. Finally, given that distributional representation goes beyond words, I hope to address more complicated tasks in future work.

Chapter 7

Conclusion

7.1 Summary

The primary goal of this thesis was to investigate a novel approach to language modeling in the context of text-entry prediction. Inspired by mechanism for language retrieval and storage in the human brain, I proposed a retrieval-based approach to language modeling and demonstrated its performance on functional requirements, driven by a brain-computer interface setting, where short inference times, personalization, and continual adaptation are desired.

First, the retrieval-based approach was introduced and compared to a basic continuous approach, and more importantly, to an equivalent categorical model. This comparison was on two domains: news text and biomedical text, which have different rates of infrequent word-types. This comparison showed the differences between the categorical and the continuous models. The former was shown to be more accurate, while the latter was shown to be more diverse.

The continuous approach was shown to be more diverse and accurate on the long-tail where infrequent word-types are found, which may indicate more personalized predictions. In the context of AAC this translates of predicting correctly infrequent icons or words the user intends to express, that may be associated with their specific expressive needs (such as a name of a movie they liked). Additional categorical baselines were compared including a diversity-promoting objective function and a subword tokenized model. The former improved diversity to some degree yet lagged compared to the GAN-based approach. The latter model was shown to increase accuracy rates, but not diversity. Costs were compared as well, and the continuous approach was shown to be promising compared to the categorical model in terms of providing short inference times and having a smaller memory footprint, especially when scaled across multiple users.

The continuous models were subsequently evaluated for continual learning, where the task is both to adapt new domains and to recall older ones. On this task, the categorical models presented higher accuracy and high diversity, while the GAN-based approach was shown to be diverse yet overall was sub-par. However, when these approaches were evaluated for their suitability to the task, the categorical models were shown to exhibit limitations when required to represent both old and new domain terms and to be architecturally more complicated for domain adaptation in general. The continuous models, on the other hand, not only were able to represent both old and new word-types of a domain within the same model at no additional

cost, but also did not require any architectural modification for adaptation. Furthermore, these models were able to retrieve out-of-vocabulary words that were not introduced during the training but were semantically related to the trained tokens. Therefore, despite the low performance shown, the continuous models were found more promising (for long-term) continual learning language models. These models allow for adaptation of new icons, as well as old, subjected to user's expressive needs.

Given the promising results shown, I proposed several ways to enhance the GAN-based approach. First, I combined the categorical and the continuous models, showing that their combined outcome was balanced both accuracy and diversity. Then, feature-based decoding was presented, where prediction candidates were filtered to match the target's part of speech. This approach was found to increase diversity in the continuous models, and more interestingly, it emphasized the bias-variance mechanism by which each model operates. Several embedding spaces were also investigated, and word2vec was found superior for the task. Various objective functions were explored to learn about optimal settings for continuous prediction as well as the factors that come into play, such as the decoding step, when deciding on an objective.

Finally, state-of-the-art models were shown to be susceptible to frequency when employing the proposed diversity metric that measures predictions across frequency bins. Evaluating non-exact matches was proposed as well as means of learning the potential usability of a language model in a typing scenario. A downstream paraphrasing task failed when the paraphrases included similar words, one of which was infrequent, illustrating the implications of the poor modeling of long-tail words.

The GAN-based approach proposed in this thesis was shown to have merit in its ability to predict diverse and potentially personalized vocabulary, making it suitable for the continual learning of its users, especially for AAC where personalization is important. This approach demonstrated reduced complexity which is a crucial aspect for user engagement, and is even more important in a BCI system that is known to have high latencies (see section 2.2.1). The Diversity as an evaluation metric was shown to measure how well the input distribution is reflected in language models. Both the approach and the evaluation metric are ways to start addressing limitations of neural categorical models.

7.2 Future Work

The GAN-based approach was shown to achieve high diversity rates, particularly on the biomedical domain, in Chapter 3. For this reason, I believe it is important to investigate the degree to which this model can retrieve diverse predictions on morphologically rich languages that, similar to the biomedical domain, exhibit high type-to-token ratios (Gerz et al., 2018). In addition, as shown in Chapter 5, feature-based decoding merits further experimentation. Instead of (or in addition to) part-of-speech decoding, morphological/inflectional oracles can be used to enhance decoding. Another related direction is to train, rather than simulate, oracles that guess an expected feature distribution.

Another research direction related to the continuous models involves exploring more architectures and modeling approaches such as reinforcement learning dynamic. As an example, Yu et al. (2017) showed that seqGAN produced a high-performing GAN-based language model that was combined with a reinforcement

objective. Drawing on that, continuous models can be powerful and may be found to be more promising given the axes in this thesis, which include personalization, adaptation, and low costs.

One way to continue Chapter 4's line of inquiry is devising a similar-complexity experiment comparing the performance of categorical and continuous models on a similar model size. Increasing the continuous models' size may enhance the accuracy rates of the models, marking another step towards using continual learning agents for language modeling. Following Gärdenfors et al. (2001), I suggest working towards grounded embedding spaces. I would like to explore ways to add new terms to a pre-trained embedding space in such a way that takes advantage of the knowledge in the space. For instance, assume the model has a representation for the words 'blue' and 'green', and the unseen word 'turquoise' is described as a color between 'blue' and 'green'. Ideally positioning this word should thus be a possible operation. This follows Gärdenfors et al. (2001)'s theory of how new terms are populated in the human mind. I would also like this space to represent dimensions beyond language, including dimensions of perception such as vision, sensation, and sound following Martin (2016).

Combining both the continuous and the categorical approaches as described in Chapter 5 could be realized through other architectures. For instance, a categorical model could be trained to directly predict a frequent word-type or an infrequent class label, where the prediction is based on the continuous models, similar to the work of Grave et al. (2017).

There are at least two ways to continue the research outlined in Chapter 6. First, researchers should consider diversity metrics beyond word-level unit tests, as many natural generation tasks are evaluated on phrases and sentences, and evaluating diversity on these tasks may require a different strategy. Tevet et al. (2020) showed that when evaluating sentence diversity, different words can convey similar meanings and devising evaluations for semantically different sentences can therefore be challenging. Another direction of research involves conducting a large-scale paraphrasing task to validate observations indicating poor performance when paraphrasing with infrequent word-types. I hope that the work presented in this thesis contributes to surfacing diversity considerations in language modeling, as well as overlooked barriers posed by categorical models. Moreover, I hope that this thesis promotes alternative ways to overcome these limitations.

Bibliography

Ackley, David H, Geoffrey E Hinton, and Terrence J Sejnowski (1985). "A Learning Algorithm for Boltzmann Machines". In: *Cognitive science* 9.1, pp. 147–169.

Ahmadi, Maryam and Leila Ahmadi (2014). "Privacy aspects of nanoneuroimplants from the point of view of a human dignity perspective in related international conventions". In: *Journal of Biomaterials and Tissue Engineering* 4.4, pp. 315–337.

Albrecht, Karl (2006). *Social Intelligence: The New Science of Success*. John Wiley & Sons.

Alec, Radford, Wu Jeffrey, Child Rewon, Luan David, Amodei Dario, and Sutskever Ilya (2019). "Language models are unsupervised multitask learners". In:

Alec, Radford, Narasimhan Karthik, Salimans Tim, and Sutskever Ilya (2018). *Improving language understanding by generative pre-training*.

Allauzen, Cyril, Michael Riley, Johan Schalkwyk, Wojciech Skut, and Mehryar Mohri (2007). "OpenFst: A general and efficient weighted finite-state transducer library". In: *International Conference on Implementation and Application of Automata*. Springer, pp. 11–23.

Alsentzer, Emily, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott (2019). "Publicly Available Clinical BERT Embeddings". In: *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pp. 72–78.

Alvarez-Melis, David and Tommi Jaakkola (2017). "A causal framework for explaining the predictions of black-box sequence-to-sequence models". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 412–421.

Amodei, Dario and Danny Hernandez (2018). "AI and Compute". In: *Blog post*.

Angrick, Miguel, Christian Herff, Emily Mugler, Matthew C Tate, Marc W Slutzky, Dean J Krusienski, and Tanja Schultz (2019). "Speech synthesis from ECoG using densely connected 3D convolutional neural networks". In: *Journal of neural engineering* 16.3, p. 036019.

ASHA (2004). "Roles and Responsibilities of Speech-Language Pathologists with respect to Augmentative and Alternative Communication: Technical report". In:

Bacchiani, Michiel and Brian Roark (2003). "Unsupervised language model adaptation". In: *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03)*. Vol. 1. IEEE, pp. I–I.

Bahdanau, Dzmitry, Tom Bosc, Stanisław Jastrzębski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio (2017). "Learning to Compute Word Embeddings on the fly". In: *arXiv preprint arXiv:1706.00286*.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Baker, BR (1982). *Minspeak, a semantic compaction system that makes self-expression easier for communicatively disabled individuals*. Byte.

Baker, James (1975). "The DRAGON system–An overview". In: *IEEE Transactions on Acoustics, speech, and signal Processing* 23.1, pp. 24–29.

Baker, James K (1990). "Stochastic modeling for automatic speech understanding". In: *Reprinted in A. Waibel and Kai-Fu Lee (eds) Readings in Speech recognition, Morgan Kaufmann, San Mateo CA*, pp. 297–307.

Barbara, Nathaniel and Tracey A Camilleri (2016). "Interfacing with a speller using EOG glasses". In: *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, pp. 001069–001074.

Beck, Jeff (2010). "Report From the Field: PubMed Central, an XML-Based Archive of Life Sciences Journal Articles". In: *International Symposium on XML for the Long Haul: Issues in the Long-term Preservation of XML, Montréal, Canada*.

Bellegarda, Jerome R (2004). "Statistical language model adaptation: review and perspectives". In: *Speech communication* 42.1, pp. 93–108.

Beltagy, Iz, Kyle Lo, and Arman Cohan (2019). "SciBERT: A Pretrained Language Model for Scientific Text". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3606–3611.

Bender, Emily M. and Alexander Koller (July 2020). "Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics. DOI: 10.18653/v1/2020.acl-main.463.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Jauvin (2003). "A Neural Probabilistic Language Model". In: *Journal of machine learning research* 3.Feb, pp. 1137–1155.

Bernhardsson, Erik (2018). *Annoy*.

Beukelman, David R et al. (2020). *Augmentative and alternative communication, fifth edition*.

Biber, Douglas and Susan Conrad (2009). *Register, genre, and style*. Cambridge University Press.

Bingham, Ella and Heikki Mannila (2001). "Random projection in dimensionality reduction: applications to image and text data". In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 245–250.

Bisk, Yonatan, Ari Holtzman, Jesse Thomason, Jacob Andreas, Yoshua Bengio, Joyce Chai, Mirella Lapata, Angeliki Lazaridou, Jonathan May, Aleksandr Nisnevich, Nicolas Pinto, and Joseph Turian (2020). "Experience Grounds Language". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics. DOI: 10.18653/v1/2020.emnlp-main.703.

Blabe, Christine H, Vikash Gilja, Cindy A Chestek, Krishna V Shenoy, Kim D Anderson, and Jaimie M Henderson (2015). "Assessment of brain–machine in-

terfaces from the perspective of people with paralysis". In: *Journal of neural engineering* 12.4, p. 043002.

Bliss, Charles Kasiel (1949). *Semantography: A Non-alphabetical Symbol Writing, Readable in All Languages; a Practical Tool for General International Communication, Especially in Science, Industry, Commerce, Traffic, Etc., and for Semantical Education, Based on the Principles of Ideographic Writing and Chemical Symbolism*. Institute for Semantography.

Blitzer, John, Ryan McDonald, and Fernando Pereira (2006). "Domain adaptation with structural correspondence learning". In: *Proceedings of the 2006 conference on empirical methods in natural language processing*, pp. 120–128.

Bodenreider, Olivier (2004). "The Unified Medical Language System (UMLS): integrating biomedical terminology". In: *Nucleic Acids Research* 32.Database issue, p. D267.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146.

Bolukbasi, Tolga, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai (2016). "Man is to computer programmer as woman is to homemaker? debiasing word embeddings". In: *Advances in neural information processing systems*, pp. 4349–4357.

Bowman, Samuel R., Gabor Angeli, Christopher Potts, and Christopher D. Manning (Sept. 2015a). "A Large Annotated Corpus for Learning Natural Language Inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 632–642. DOI: 10.18653/v1/D15-1075.

Bowman, Samuel R, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio (2015b). "Generating sentences from a continuous space". In: *arXiv preprint arXiv:1511.06349*.

Bowman, Samuel, Gabor Angeli, Christopher Potts, and Christopher D Manning (2015c). "A large annotated corpus for learning natural language inference". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642.

Brewer, Marilynn B. and William D. Crano (2014). "Research Design and Issues of Validity". In: *Handbook of Research Methods in Social and Personality Psychology*. Ed. by Harry T. Reis and Charles M.Editors Judd. 2nd ed. Cambridge University Press, pp. 11–26. DOI: 10.1017/CBO9780511996481.005.

Brown, Peter F, Vincent J Della Pietra, Peter V Desouza, Jennifer C Lai, and Robert L Mercer (1992). "Class-based n-gram models of natural language". In: *Computational linguistics* 18.4, pp. 467–480.

Bruno, J Joan (1988). "Picture and word association performance in preliterate children and adults". PhD thesis. Verlag nicht ermittelbar.

Cacioppo, John T and William Patrick (2008). *Loneliness: Human Nature and the Need for Social Connection*. WW Norton & Company.

Callison-Burch, Chris (2008). "Syntactic Constraints on Paraphrases Extracted from Parallel Corpora". In: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pp. 196–205.

Cecotti, Hubert (2010). "A self-paced and calibration-less SSVEP-based Brain–Computer Interface Speller". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 18.2, pp. 127–133.

Chang, Jason S and Shun-Der Chen (1995). "The Postprocessing of Optical Character Recognition based on Statistical Noisy Channel and Language Model". In: *Proceedings of the 10th Pacific Asia Conference on Language, Information and Computation*, pp. 127–132.

Chapman, Robert M and Henry R Bragdon (1964). "Evoked Responses to Numerical and Non-Numerical Visual Stimuli while Problem Solving". In: *Nature* 203.4950, pp. 1155–1157.

Chaudhary, Poonam and Rashmi Agrawal (2018). "Emerging threats to security and privacy in brain computer interface". In: *International Journal of Advanced Studies of Scientific Research* 3.12.

Che, Tong, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li (2016). "Mode regularized generative adversarial networks". In: *arXiv preprint arXiv:1612.02136*.

Chen, Stanley F and Joshua Goodman (1999). "An empirical study of smoothing techniques for language modeling". In: *Computer Speech & Language* 13.4, pp. 359–394.

Chen, Wenlin, David Grangier, and Michael Auli (2016). "Strategies for Training Large Vocabulary Neural Language Models". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1975–1985.

Chen, Xinxiong, Zhiyuan Liu, and Maosong Sun (2014). "A unified model for word sense representation and disambiguation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1025–1035.

Chen, Zhiyuan and Bing Liu (2018). "Lifelong machine learning". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 12.3, pp. 1–207.

Chi, Albert, Sawyer Smith, Isaac Womack, and Robert Armiger (2018). "The Evolution of Man and Machine—a Review of Current Surgical Techniques and Cutting Technologies After Upper Extremity Amputation". In: *Current Trauma Reports* 4.4, pp. 339–347.

Cho, Kyunghyun, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio (2014). "On the Properties of Neural Machine Translation: Encoder–Decoder Approaches". In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111.

Choi, Byung-Ju, Jimin Hong, David Keetae Park, and Sang Wan Lee (2020). "Fˆ 2-Softmax: Diversifying Neural Text Generation via Frequency Factorized Softmax". In: *arXiv preprint arXiv:2009.09417*.

Choi, Edward, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart (2016). "Retain: An interpretable predictive model for healthcare using reverse time attention mechanism". In: *Advances in Neural Information Processing Systems*, pp. 3504–3512.

Chomsky, Noam (1956). "Three models for the description of language". In: *IRE Transactions on information theory* 2.3, pp. 113–124.

Chomsky, Noam et al. (1957). "Syntactic structures". In: *The Hague: Mouton*.

Chorowski, Jan K, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio (2015). "Attention-based models for speech recognition". In: *Advances in neural information processing systems*, pp. 577–585.

Church, Kenneth and Patrick Hanks (1990). "Word association norms, mutual information, and lexicography". In: *Computational linguistics* 16.1, pp. 22–29.

Church, KW, WA Gale, and JB Kruskal (1991). "The Good Turing Theorem. Appendix A". In: *Church, KW; Gale, WA: A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams, Jg* 5, pp. 19–54.

Clark, Jacquie (1997). *Symbolstix*. News 2 You.

Coenen, Andy, Emily Reif, Ann Yuan, Been Kim, Adam Pearce, Fernanda Viégas, and Martin Wattenberg (2019). "Visualizing and Measuring the Geometry of BERT". In: *arXiv preprint arXiv:1906.02715*.

Conneau, Alexis, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou (2017). "Word translation without parallel data". In: *arXiv preprint arXiv:1710.04087*.

Crevier, Daniel (1993). *AI: the tumultuous history of the search for artificial intelligence*. Basic Books, Inc.

Croft, William (2012). *Verbs: Aspect and causal structure*. OUP Oxford.

Crystal, David, Philip Pullman, Jilly Cooper, Jeremy Vine, and Meera SyalEnglish (2008). *Kitchen Table Lingo*. Virgin. ISBN: 9780753518199.

Czarnowska, Paula, Sebastian Ruder, Edouard Grave, Ryan Cotterell, and Ann Copestake (Nov. 2019). "Don't Forget the Long Tail! A Comprehensive Analysis of Morphological Generalization in Bilingual Lexicon Induction". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 974–983. DOI: 10.18653/v1/D19-1090.

Dai, Andrew M and Quoc V Le (2015). "Semi-supervised sequence learning". In: *Advances in neural information processing systems*, pp. 3079–3087.

Dal Pozzolo, Andrea, Olivier Caelen, and Gianluca Bontempi (2015). "When is undersampling effective in unbalanced classification tasks?" In: *Joint european conference on machine learning and knowledge discovery in databases*. Springer, pp. 200–215.

Damasio, Antonio R (1992). "Aphasia". In: *New England Journal of Medicine* 326.8, pp. 531–539.

Damerau, Friederick J. (1971). *Markov Models and Linguistic Theory*. Berlin, Boston: De Gruyter Mouton. ISBN: 978-3-11-090858-9. DOI: https://doi.org/10.1515/9783110908589.

Danescu-Niculescu-Mizil, Cristian, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts (2013). "No country for old members: User lifecycle and linguistic change in online communities". In: *Proceedings of the 22nd international conference on World Wide Web*, pp. 307–318.

Darroch, John N and Douglas Ratcliff (1972). "Generalized Iterative Scaling for Log-Linear Models". In: *The annals of mathematical statistics*, pp. 1470–1480.

Daumé III, Hal (2009). "Frustratingly easy domain adaptation". In: *arXiv preprint arXiv:0907.1815*.

Daume III, Hal and Daniel Marcu (2006). "Domain adaptation for statistical classifiers". In: *Journal of artificial Intelligence research* 26, pp. 101–126.

Dauphin, Grégoire Mesnil Yann, Xavier Glorot, Salah Rifai, Yoshua Bengio, Ian Goodfellow, Erick Lavoie, Xavier Muller, Guillaume Desjardins, David Warde-

Farley, Pascal Vincent, et al. (2012). "Unsupervised and transfer learning challenge: a deep learning approach". In: *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pp. 97–110.

Deerwester, Scott, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman (1990). "Indexing by latent semantic analysis". In: *Journal of the American society for information science* 41.6, pp. 391–407.

Dell, Gary S, Myrna F Schwartz, Nadine Martin, Eleanor M Saffran, and Deborah A Gagnon (1997). "Lexical access in aphasic and nonaphasic speakers." In: *Psychological review* 104.4, p. 801.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805*.

— (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

Dieng, Adji B, Kyunghyun Cho, David M Blei, and Yann LeCun (2018). "Learning with Reflective Likelihoods". In:

Dinan, Emily, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. (2019). "The second conversational intelligence challenge (convai2)". In: *arXiv preprint arXiv:1902.00098*.

Donchin, Emanuel, Kevin M Spencer, and Ranjith Wijesinghe (2000). "The Mental Prosthesis: Assessing the Speed of a P300-based Brain-Computer Interface". In: *IEEE transactions on rehabilitation engineering* 8.2, pp. 174–179.

Dosovitskiy, Alexey and Thomas Brox (2016). "Generating images with perceptual similarity metrics based on deep networks". In: *Advances in neural information processing systems*, pp. 658–666.

Dudy, Shiran and Steven Bedrick (2018). "Compositional Language Modeling for Icon-Based Augmentative and Alternative Communication". In: *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, pp. 25–32.

Duh, Kevin, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada (2013). "Adaptation Data Selection Using neural Language Models: Experiments in machine translation". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 678–683.

Edunov, Sergey, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato (2018). "Classical Structured Prediction Losses for Sequence to Sequence Learning". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 355–364.

Elman, Jeffrey L (1990). "Finding structure in time". In: *Cognitive Science* 14.2, pp. 179–211. DOI: https://doi.org/10.1016/0364-0213(90)90002-E.

— (2004). "An alternative view of the mental lexicon". In: *TRENDS in Cognitive Sciences* 8.7.

Elsahar, Yasmin, Sijung Hu, Kaddour Bouazza-Marouf, David Kerr, and Annysa Mansor (2019). "Augmentative and alternative communication (AAC) advances:

A review of configurations for individuals with a speech disability". In: *Sensors* 19.8, p. 1911.

Fan, Angela, Mike Lewis, and Yann Dauphin (2018). "Hierarchical Neural Story Generation". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 889–898.

Faruqui, Manaal, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith (2015). "Retrofitting Word Vectors to Semantic Lexicons". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1606–1615.

Faruqui, Manaal and Chris Dyer (2014). "Improving vector space word representations using multilingual correlation". In: *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 462–471.

Farwell, Lawrence Ashley and Emanuel Donchin (1988). "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials". In: *Electroencephalography and clinical Neurophysiology* 70.6, pp. 510–523.

Federico, Marcello (1999). "Efficient language model adaptation through MDI estimation". In: *Sixth European Conference on Speech Communication and Technology*.

Ferraro, Francis, Max Thomas, Matthew, Travis Wolfe R. Gormley, Craig Harman, and Benjamin Van Durme (2018). "Concretely Annotated English Gigaword". In: *Linguistic Data Consortium, Philadelphia* 4, p. 1.

Ficke, Robert C (1992). "Digest of Data on Persons with Disabilities." In:

Firth, John R (1957). "A synopsis of linguistic theory, 1930-1955". In: *Studies in linguistic analysis*.

Fischer, Andreas (2012). "Handwriting recognition in historical documents". PhD thesis. Verlag nicht ermittelbar.

Foster, George, Cyril Goutte, and Roland Kuhn (2010). "Discriminative Instance Weighting for Domain Adaptation in Statistical Machine Translation". In: *Proceedings of the 2010 conference on empirical methods in natural language processing*, pp. 451–459.

Fowler, Andrew, Kurt Partridge, Ciprian Chelba, Xiaojun Bi, Tom Ouyang, and Shumin Zhai (2015). "Effects of language modeling and its personalization on touchscreen typing performance". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 649–658.

Fried-Oken, Melanie, Aimee Mooney, Betts Peters, and Barry Oken (2015). "A Clinical Screening Protocol for the RSVP Keyboard Brain–Computer Interface". In: *Disability and Rehabilitation: Assistive Technology* 10.1, pp. 11–18.

Friedman, Mark, Bennett Tilden, and Killiany (1983). *DynaVox Tobii*. Dynavox.

Furui, Sadaoki (2005). "Recent progress in corpus-based spontaneous speech recognition". In: *IEICE transactions on information and systems* 88.3, pp. 366–375.

Gabriel De Souza, P Moreira, Dietmar Jannach, and Adilson Marques Da Cunha (2019). "Contextual Hybrid Session-based News Recommendation with Recurrent Neural Networks". In: *IEEE Access* 7, pp. 169185–169203.

Gage, Philip (1994). "A new algorithm for data compression". In: *C Users Journal* 12.2, pp. 23–38.

Gale, William A and Geoffrey Sampson (1995). "Good-turing frequency estimation without tears". In: *Journal of quantitative linguistics* 2.3, pp. 217–237.

Gale, William and Kenneth Church (1994). "What's wrong with adding one". In: *Corpus-Based Research into Language: In honour of Jan Aarts*, pp. 189–200.

Galliers, J.R. and Karen Spärck Jones (Feb. 1993). *Evaluating Natural Language Processing Systems*. Tech. rep. UCAM-CL-TR-291. University of Cambridge, Computer Laboratory.

Gärdenfors, Peter (2014). *The Geometry of Meaning: Semantics Based on Conceptual Spaces*. MIT Press.

Gärdenfors, Peter and Mary-Anne Williams (2001). "Reasoning about categories in conceptual spaces". In: *Proceedings of the 17th international joint conference on Artificial intelligence-Volume 1*, pp. 385–392.

Gerz, Daniela, Ivan Vulić, Edoardo Ponti, Jason Naradowsky, Roi Reichart, and Anna Korhonen (2018). "Language modeling for morphologically rich languages: Character-aware modeling for word-level prediction". In: *Transactions of the Association for Computational Linguistics* 6, pp. 451–465.

Gevarter, Cindy, Mark F O'Reilly, Laura Rojeski, Nicolette Sammarco, Jeff Sigafoos, Giulio E Lancioni, and Russell Lang (2014). "Comparing acquisition of AAC-based mands in three young children with autism spectrum disorder using iPad® applications with different display and design elements". In: *Journal of Autism and Developmental Disorders* 44.10, pp. 2464–2474.

Glennen, Sharon and Denise C DeCoste (1997). *The handbook of augmentative and alternative communication*. Cengage Learning.

Glorot, Xavier, Antoine Bordes, and Yoshua Bengio (2011). "Domain adaptation for large-scale sentiment classification: A deep learning approach". In: *ICML*.

Goldberg, Yoav (2019). "Assessing BERT's Syntactic Abilities". In: *arXiv preprint arXiv:1901.05287*.

Gong, Chengyue, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu (2018). "FRAGE: Frequency-Agnostic Word Representation". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Curran Associates, Inc., pp. 1334–1345.

Good, Irving J (1953). "The population frequencies of species and the estimation of population parameters". In: *Biometrika* 40.3-4, pp. 237–264.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets". In: *Advances in neural information processing systems*, pp. 2672–2680.

Goodman, A, J Wrigley, K Silversides, and N Venus-Balgobin (2017). *Measuring Your Impact on Loneliness in Later Life*.

Grave, Edouard, Armand Joulin, Moustapha Cissé, Hervé Jégou, et al. (2017). "Efficient Softmax Approximation for GPUs". In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pp. 1302–1310.

Grave, Edouard, Armand Joulin, and Nicolas Usunier (2016). "Improving Neural Language Models with a Continuous Cache". In: *arXiv preprint arXiv:1612.04426*.

Graves, Alex and Navdeep Jaitly (2014). "Towards end-to-end speech recognition with recurrent neural networks". In: *International conference on machine learning*, pp. 1764–1772.

Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton (2013). "Speech recognition with deep recurrent neural networks". In: *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, pp. 6645–6649.

Gretter, Roberto and Giuseppe Riccardi (2001). "On-line learning of language models with word error probability distributions". In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*. Vol. 1. IEEE, pp. 557–560.

Gururangan, Suchin, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith (July 2020). "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics. DOI: 10.18653/v1/2020.acl-main.740.

Gururangan, Suchin, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith (June 2018). "Annotation Artifacts in Natural Language Inference Data". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 107–112. DOI: 10.18653/v1/N18-2017.

Gutmann, Michael and Aapo Hyvärinen (2010). "Noise-Contrastive Estimation: A new Estimation Principle for Unnormalized Statistical Models". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, pp. 297–304.

Guu, Kelvin, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang (2020). "Realm: Retrieval-Augmented Language Model Pre-Training". In: *arXiv preprint arXiv:2002.08909*.

Harris, Zellig S (1954). "Distributional Structure". In: *Word* 10.2-3, pp. 146–162.

He, Tianxing and James Glass (2019). "Negative Training for Neural Dialogue Response Generation". In: *arXiv preprint arXiv:1903.02134*.

Health, National Institutes of et al. (1992). "National Advisory Board on Medical Rehabilitation Research, Draft V: Report and Plan for Medical Rehabilitation Research". In: *Bethesda, MD, National Institutes of Health*.

Hebb, Donald Olding (1949). *The organization of behavior: a neuropsychological theory*. J. Wiley; Chapman & Hall.

Herbelot, Aurélie and Marco Baroni (Sept. 2017). "High-Risk Learning: acquiring new word vectors from tiny data". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 304–309. DOI: 10.18653/v1/D17-1030.

Hewitt, John and Christopher D Manning (2019). "A structural probe for finding syntax in word representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138.

Hiemstra, Djoerd (2000). "A probabilistic justification for using tf× idf term weighting in information retrieval". In: *International Journal on Digital Libraries* 3.2, pp. 131–139.

Higger, Matt, Fernando Quivira, Murat Akcakaya, Mohammad Moghadamfalahi, Hooman Nezamfar, Mujdat Cetin, and Deniz Erdogmus (2016). "Recursive Bayesian Coding for BCIs". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25.6, pp. 704–714.

Hines, Elizabeth and Brandi Simonsen (2008). "The effects of picture icons on behavior for a young student with autism". In: *Beyond Behavior* 18.1, pp. 9–17.

Hinton, Geoffrey, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. (2012). "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups". In: *IEEE Signal processing magazine* 29.6, pp. 82–97.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long Short-Term Memory". In: *Neural computation* 9.8, pp. 1735–1780.

Holtzman, Ari, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi (2020). "The Curious Case of Neural Text Degeneration". In: *International Conference on Learning Representations*.

Holtzman, Ari, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi (2018). "Learning to Write with Cooperative Discriminators". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1638–1649.

Howard, Jeremy and Sebastian Ruder (2018). "Universal Language Model Fine-tuning for Text Classification". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339.

Huang, Fei and Alexander Yates (2009). "Distributional representations for handling sparsity in supervised sequence-labeling". In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 495–503.

Huang, Kexin, Abhishek Singh, Sitong Chen, Edward T Moseley, Chih-ying Deng, Naomi George, and Charlotta Lindvall (2019). "Clinical XLNet: Modeling Sequential Clinical Notes and Predicting Prolonged Mechanical Ventilation". In: *arXiv preprint arXiv:1912.11975*.

Huffman, David A (1952). "A Method for the Construction of Minimum-Redundancy Rodes". In: *Proceedings of the IRE* 40.9, pp. 1098–1101.

Huth, Alexander G, Wendy A De Heer, Thomas L Griffiths, Frédéric E Theunissen, and Jack L Gallant (2016). "Natural Speech Reveals the Semantic Maps that Tile Human Cerebral Cortex". In: *Nature* 532.7600, pp. 453–458.

Huth, Alexander G, Shinji Nishimoto, An T Vu, and Jack L Gallant (2012). "A Continuous Semantic Space Describes the Representation of Thousands of Object and Action Categories Across the Human Brain". In: *Neuron* 76.6, pp. 1210–1224.

Ippolito, Daphne, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch (July 2019). "Comparison of Diverse Decoding Methods from Conditional Language Models". In: *Proceedings of the 57th Annual Meeting of the Association*

for Computational Linguistics. Florence, Italy: Association for Computational Linguistics, pp. 3752–3762. DOI: 10.18653/v1/P19-1365.

Ito, Akinori, Masaki Kohda, and Mari Ostendorf (1999). "A new Metric for Stochastic Language Model Evaluation". In: *Sixth European Conference on Speech Communication and Technology*, pp. 1591–1594.

Iyer, R., M. Ostendorf, and M. Meteer (1997). "Analyzing and Predicting Language Model Improvements". In: *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pp. 254–261.

Jain, Sarthak and Byron C Wallace (2019). "Attention is not Explanation". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556.

Jeffreys, Harold (1998). *The theory of probability*. OUP Oxford.

Jelinek, Fred (1990). "Self-organized language modeling for speech recognition". In: *Readings in speech recognition*, pp. 450–506.

Jelinek, Fred and Robert L. Mercer (1980). "Interpolated Estimation of Markov Source Parameters from Sparse Data". In: *Proceedings, Workshop on Pattern Recognition in Practice*. Ed. by Edzard S. Gelsema and Laveen N. Kanal. Amsterdam: North Holland, pp. 381–397.

Jelinek, Fred, Robert Mercer, and Salim Roukos (1990). "Classifying words for improved statistical language models". In: *International Conference on Acoustics, Speech, and Signal Processing*. IEEE, pp. 621–624.

Johnson, Rie and Tong Zhang (2005). "A high-performance semi-supervised learning method for text chunking". In: *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 1–9.

Jozefowicz, Rafal, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu (2016). "Exploring the Limits of Language Modeling". In: *arXiv preprint arXiv:1602.02410*.

Kanai, Sekitoshi, Yasuhiro Fujiwara, Yuki Yamanaka, and Shuichi Adachi (2018). "Sigsoftmax: Reanalysis of the Softmax Bottleneck". In: *Advances in Neural Information Processing Systems*, pp. 286–296.

Katz, Slava (1987). "Estimation of probabilities from sparse data for the language model component of a speech recognizer". In: *IEEE transactions on acoustics, speech, and signal processing* 35.3, pp. 400–401.

Keskinen, Tuuli, Tomi Heimonen, Markku Turunen, Juha-Pekka Rajaniemi, and Sami Kauppinen (2012). "SymbolChat: A flexible picture-based communication platform for users with intellectual disabilities". In: *Interacting with Computers* 24.5, pp. 374–386.

Khandelwal, Urvashi, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis (2019). "Generalization through Memorization: Nearest Neighbor Language Models". In: *International Conference on Learning Representations*.

Kim, Suyoun, Takaaki Hori, and Shinji Watanabe (2017). "Joint CTC-attention based end-to-end speech recognition using multi-task learning". In: *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 4835–4839.

Kim, Young-Bum, Karl Stratos, and Ruhi Sarikaya (2016). "Frustratingly easy neural domain adaptation". In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 387–396.

Kindermans, Pieter-Jan, Hannes Verschore, and Benjamin Schrauwen (2013). "A unified probabilistic approach to improve spelling in an event-related potential-based brain–computer interface". In: *IEEE Transactions on Biomedical Engineering* 60.10, pp. 2696–2705.

Kingma, Diederik P and Max Welling (2014). "Auto-Encoding Variational Bayes". In: *stat* 1050, p. 1.

Kirkpatrick, James, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. (2017). "Overcoming Catastrophic Forgetting in Neural Networks". In: *Proceedings of the National Academy of Sciences of the United States of America* 114.13, p. 3521.

Kneser, Reinhard and Hermann Ney (1993). "Improved clustering techniques for class-based statistical language modelling". In: *Third European Conference on Speech Communication and Technology*.

Koehn, Philipp and Rebecca Knowles (2017). "Six Challenges for Neural Machine Translation". In: *ACL 2017*, p. 28.

Kompalli, Suryaprakash, Srirangaraj Setlur, and Venu Govindaraju (2009). "Devanagari OCR using a recognition driven segmentation framework and stochastic language models". In: *International Journal on Document Analysis and Recognition (IJDAR)* 12.2, pp. 123–138.

Koul, Rajinder and Robyn Harding (1998). "Identification and production of graphic symbols by individuals with aphasia: Efficacy of a software application". In: *Augmentative and Alternative Communication* 14.1, pp. 11–24.

Kravits, Tamara R, Debra M Kamps, Katie Kemmerer, and Jessica Potucek (2002). "Brief report: Increasing communication skills for an elementary-aged student with autism using the picture exchange communication system". In: *Journal of autism and developmental disorders* 32.3, pp. 225–230.

Kuhn, Thomas, Heinrich Niemann, and Ernst Günter Schukat-Talamazzini (1994). "Ergodic hidden markov models and polygrams for language modeling". In: *Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing*. Vol. 1. IEEE, pp. I–357.

Kulikov, Ilya, Alexander H Miller, Kyunghyun Cho, and Jason Weston (2018). "Importance of a Search Strategy in Neural Dialogue Modelling". In: *arXiv preprint arXiv:1811.00907*.

Kumar, Sachin and Yulia Tsvetkov (2019). "Von Mises-Fisher Loss for Training Sequence to Sequence Models with Continuous Outputs". In: *International Conference on Learning Representations*.

Kusner, Matt J and José Miguel Hernández-Lobato (2016). "Gans for sequences of discrete elements with the gumbel-softmax distribution". In: *arXiv preprint arXiv:1611.04051*.

Lamb, Alex M, Anirudh Goyal Alias Parth Goyal, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio (2016). "Professor forcing: A new algorithm for training recurrent networks". In: *Advances in neural information processing systems*, pp. 4601–4609.

Lample, Guillaume and Alexis Conneau (2019). "Cross-Lingual Language Model Pretraining". In: *arXiv preprint arXiv:1901.07291*.

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). "ALBERT: A Lite BERT for Self-Supervised Learning

of Language Representations". In: *International Conference on Learning Representations*.

Laplace, Pierre Simon marquis de (1902). *A philosophical essay on probabilities*. Wiley.

Lavelli, Alberto, Fabrizio Sebastiani, and Roberto Zanoli (2004). "Distributional term representations: an experimental comparison". In: *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pp. 615–624.

Lavie, Alon and Michael J Denkowski (2009). "The METEOR Metric for Automatic Evaluation of Machine Translation". In: *Machine translation* 23.2-3, pp. 105–115.

Lazaridou, Angeliki, Georgiana Dinu, and Marco Baroni (2015). "Hubness and Pollution: Delving into Cross-Space Mapping for Zero-Shot Learning". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Vol. 1, pp. 270–280.

Le, HaiSon, Ilya Oparin, Alexandre Allauzen, Jean-Luc Gauvain, and François Yvon (2011). "Structured Output Layer Neural Network Language Model". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5524–5527.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner (1998). "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Lee, Jinhyuk, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang (2020). "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4, pp. 1234–1240.

Lei, Tao, Regina Barzilay, and Tommi Jaakkola (2016). "Rationalizing Neural Predictions". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 107–117.

Lesher, Gregory W, Bryan J Moulton, D Jeffery Higginbotham, et al. (1999). "Effects of ngram order and training text size on word prediction". In: *Proceedings of the RESNA'99 Annual Conference*. Citeseer, pp. 52–54.

Levy, Omer and Yoav Goldberg (2014a). "Dependency-Based Word Wmbeddings". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Vol. 2, pp. 302–308.

— (2014b). "Neural word embedding as implicit matrix factorization". In: *Advances in neural information processing systems*, pp. 2177–2185.

Lewis, Patrick, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". In: *arXiv preprint arXiv:2005.11401*.

Li, Jiwei, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan (June 2016a). "A Diversity-Promoting Objective Function for Neural Conversation Models". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 110–119. DOI: 10.18653/v1/N16-1014.

Li, Jiwei, Will Monroe, and Dan Jurafsky (2016b). "A Simple, Fast Diverse Decoding Algorithm for Neural Generation". In: *arXiv preprint arXiv:1611.08562*.

Li, Liunian Harold, Patrick H Chen, Cho-Jui Hsieh, and Kai-Wei Chang (2019). "Efficient Contextual Representation Learning With Continuous Outputs". In: *Transactions of the Association for Computational Linguistics* 7, pp. 611–624.

Li, Zhizhong and Derek Hoiem (2017). "Learning without forgetting". In: *IEEE transactions on pattern analysis and machine intelligence* 40.12, pp. 2935–2947.

Light, Janice and Peter Lindsay (1992). "Message-encoding techniques for augmentative communication systems: The recall performances of adults with severe speech impairments". In: *Journal of Speech, Language, and Hearing Research* 35.4, pp. 853–864.

Ling, Wang, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Luis Marujo, and Tiago Luis (Sept. 2015). "Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 1520–1530. DOI: 10.18653/v1/D15-1176.

Linzen, Tal, Emmanuel Dupoux, and Yoav Goldberg (2016). "Assessing the Ability of LSTMs to learn syntax-sensitive dependencies". In: *Transactions of the Association for Computational Linguistics* 4, pp. 521–535.

Liu, Qi, Matt J Kusner, and Phil Blunsom (2020). "A Survey on Contextual Embeddings". In: *arXiv preprint arXiv:2003.07278*.

Liu, Xunying, Mark JF Gales, and Philip C Woodland (2008). "Context Dependent Language Model Adaptation". In: *Ninth Annual Conference of the International Speech Communication Association*.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692*.

Ljolje, Andre, Donald Hindle, Michael Riley, and Richard Sproat (2000). "The at&t lvcsr-2000 system". In: *Speech Transcription Workshop*. Citeseer.

Loper, Edward and Steven Bird (2002). "NLTK: The Natural Language Toolkit". In: *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pp. 63–70.

Luo, Ruotian and Gregory Shakhnarovich (2020). "Analysis of diversity-accuracy tradeoff in image captioning". In: *arXiv preprint arXiv:2002.11848*.

Ma, Teng, Hui Li, Lili Deng, Hao Yang, Xulin Lv, Peiyang Li, Fali Li, Rui Zhang, Tiejun Liu, Dezhong Yao, et al. (2017). "The Hybrid BCI system for Movement Control by Combining Motor Imagery and Moving Onset Visual Evoked Potential". In: *Journal of neural engineering* 14.2, p. 026015.

Mainsah, BO, LM Collins, KA Colwell, EW Sellers, DB Ryan, K Caves, and CS Throckmorton (2015). "Increasing BCI communication rates with dynamic stopping towards more practical use: an ALS study". In: *Journal of neural engineering* 12.1, p. 016013.

Manero, Albert, Peter Smith, John Sparkman, Matt Dombrowski, Dominique Courbin, Anna Kester, Isaac Womack, and Albert Chi (2019). "Implementation of 3D Printing Technology in the field of Prosthetics: Past, Present, and Future". In: *International journal of environmental research and public health* 16.9, p. 1641.

Marcus, Mitch, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger (1994). "The Penn Treebank: annotating predicate argument structure". In: *HUMAN LANGUAGE TECHNOLOGY: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Markov, Andrei (2006). "An example of statistical investigation of the text "Eugene Onegin"concerning the connection of samples in chains, trans. into English by G. Custance and D. Link". In: *Science in Context* 19.4, pp. 591–600.

Martin, Alex (2016). "GRAPES—Grounding representations in action, perception, and emotion systems: How object properties and categories are represented in the human brain". In: *Psychonomic bulletin & review* 23.4, pp. 979–990.

Martin, James (1965). *Programming Real-Time Computer Systems*. Tech. rep.

McCloskey, Michael and Neal J Cohen (1989). "Catastrophic interference in connectionist networks: The sequential learning problem". In: *Psychology of learning and motivation*. Vol. 24. Elsevier, pp. 109–165.

McCulloch, Gretchen (2020). *Because Internet: Understanding the New Rules of Language*. Riverhead Books.

Meltzer-Asscher, Aya, Julia Schuchard, Dirk-Bart den Ouden, and Cynthia K Thompson (2013). "The neural substrates of complex argument structure representations: Processing "alternating transitivity" verbs". In: *Language and cognitive processes* 28.8, pp. 1154–1168.

Merity, Stephen, Caiming Xiong, James Bradbury, and Richard Socher (2016). "Pointer Sentinel Mixture Models". In: *arXiv preprint arXiv:1609.07843*.

Mermillod, Martial, Aurélia Bugaiska, and Patrick Bonin (2013). "The stability-plasticity dilemma: Investigating the continuum from catastrophic forgetting to age-limited learning effects". In: *Frontiers in psychology* 4, p. 504.

Michel, Christoph M and Micah M Murray (2012). "Towards the utilization of EEG as a brain imaging tool". In: *Neuroimage* 61.2, pp. 371–385.

Mikolov, T., S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur (2011a). "Extensions of recurrent neural network language model". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5528–5531. DOI: 10.1109/ICASSP.2011.5947611.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013a). "Efficient Estimation of Word Representations in Vector Space". In: *arxiv*.

Mikolov, Tomáš, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černock' (2011b). "Empirical evaluation and combination of advanced language modeling techniques". In: *Twelfth annual conference of the international speech communication association*.

Mikolov, Tomáš, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model". In: *INTERSPEECH*, pp. 1045–1048.

Mikolov, Tomáš, Stefan Kombrink, Lukáš Burget, Jan Černock', and Sanjeev Khudanpur (2011c). "Extensions of Recurrent Neural Network Language Model". In: *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 5528–5531.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013b). "Distributed Representations of Words and Phrases and their Compositionality". In: *Advances in neural information processing systems*, pp. 3111–3119.

Mikolov, Tomáš, Ilya Sutskever, Anoop Deoras, Hai-Son Le, Stefan Kombrink, and Jan Cernocky (2012). "Subword Language Modeling with Neural Networks". In: *preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf)* 8.

Mikolov, Tomáš, Wen-tau Yih, and Geoffrey Zweig (2013c). "Linguistic regularities in continuous space word representations". In: *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746–751.

Miller, George A (1995). "WordNet: a Lexical Database for English". In: *Communications of the ACM* 38.11, pp. 39–41.

Miller, George A and Noam Chomsky (1963). "Finitary models of language users". In:

Minsky, Marvin and Seymour Papert (1969). "An introduction to computational geometry". In: *Cambridge tiass., HIT*.

Mirza, Mehdi and Simon Osindero (2014). "Conditional Generative Adversarial Nets". In: *arXiv preprint arXiv:1411.1784*.

Mnih, Andriy and Geoffrey Hinton (2007). "Three new graphical models for statistical language modelling". In: *Proceedings of the 24th international conference on Machine learning*, pp. 641–648.

Mnih, Andriy and Yee Whye Teh (2012). "A Fast and Simple Algorithm for Training Neural Probabilistic Language Models". In: *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*. Ed. by John Langford and Joelle Pineau. ICML '12. Edinburgh, Scotland, GB: Omnipress, pp. 1751–1758. ISBN: 978-1-4503-1285-1.

Mnih, Volodymyr, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu (2016). "Asynchronous Methods for Deep Reinforcement Learning". In: *International conference on machine learning*, pp. 1928–1937.

Morin, Frederic and Yoshua Bengio (2005). "Hierarchical Probabilistic Neural Network Language Model". In: *Proceedings of the International Workshop on Artificial Intelligence and Statistics*, pp. 246–252.

Müller-Frommeyer, Lena C, Niels AM Frommeyer, and Simone Kauffeld (2019). "Introducing rLSM: An integrated metric assessing temporal reciprocity in language style matching". In: *Behavior Research Methods* 51.3, pp. 1343–1359.

Murray, Christopher JL, Alan D Lopez, World Health Organization, et al. (1996). *The global burden of disease: a comprehensive assessment of mortality and disability from diseases, injuries, and risk factors in 1990 and projected to 2020: summary*. World Health Organization.

Nanjo, Hiroaki and Tatsuya Kawahara (2003). "Unsupervised language model adaptation for lecture speech recognition". In: *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*.

Napoles, Courtney, Matthew R Gormley, and Benjamin Van Durme (2012). "Annotated gigaword". In: *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pp. 95–100.

Neelakantan, Arvind, Jeevan Shankar, Alexandre Passos, and Andrew McCallum (2014). "Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1059–1069.

Neubig, Graham and Chris Dyer (2016). "Generalizing and Hybridizing Count-based and Neural Language Models". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1163–1172.

Neumann, Mark, Daniel King, Iz Beltagy, and Waleed Ammar (2019). "ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing". In: eprint: arXiv:1902.07669.

Nguyen, Thien Huu and Ralph Grishman (2014). "Employing word representations and regularization for domain adaptation of relation extraction". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 68–74.

Och, Franz Josef and Hermann Ney (2002). "Discriminative Training and Maximum Entropy Models for Statistical Machine Translation". In: *Proceedings of the 40th Annual meeting of the Association for Computational Linguistics*, pp. 295–302.

Oken, Barry S, Umut Orhan, Brian Roark, Deniz Erdogmus, Andrew Fowler, Aimee Mooney, Betts Peters, Meghan Miller, and Melanie B Fried-Oken (2014). "Brain–Computer Interface with Language Model–Electroencephalography Fusion for Locked-in Syndrome". In: *Neurorehabilitation and neural repair* 28.4, pp. 387–394.

Orhan, Umut, Deniz Erdogmus, Brian Roark, Shalini Purwar, Kenneth E Hild, Barry Oken, Hooman Nezamfar, and Melanie Fried-Oken (2011). "Fusion with language models improves spelling accuracy for ERP-based brain computer interface spellers". In: *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, pp. 5774–5777.

Orhan, Umut, Kenneth E Hild, Deniz Erdogmus, Brian Roark, Barry Oken, and Melanie Fried-Oken (2012). "RSVP Keyboard: An EEG Based Typing Interface". In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 645–648.

Österlund, Arvid, David Ödling, and Magnus Sahlgren (2015). "Factorization of latent variables in distributional semantic models". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 227–231.

Ott, Myle, Michael Auli, David Grangier, and Marc'Aurelio Ranzato (2018). "Analyzing Uncertainty in Neural Machine Translation". In: *ICML*.

Oxford (n.d.). In:

Öztürk, Seçil, Bülent Sankur, Tunga Güngör, Mustafa Berkay Yilmaz, Bilge Köroğlu, Onur Ağın, Mustafa İşbilen, Çağdas Ulaş, and Mehmet Ahat (2014). "Turkish Labeled Text Corpus". In: *2014 22nd Signal Processing and Communications Applications Conference (SIU)*. IEEE, pp. 1395–1398.

Palmini, Andre (2006). "The Concept of the Epileptogenic Zone: a Modern look at Penfield and Jasper's views on the role of Interictal Spikes". In: *Epileptic disorders* 8.2, pp. 10–15.

Pan, Sinno Jialin, Ivor W Tsang, James T Kwok, and Qiang Yang (2010). "Domain adaptation via transfer component analysis". In: *IEEE Transactions on Neural Networks* 22.2, pp. 199–210.

Pan, Sinno Jialin and Qiang Yang (2009). "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* 22.10, pp. 1345–1359.

Pang, Bo, Kevin Knight, and Daniel Marcu (2003). *Syntax-based Alignment of Multiple Translations: Extracting paraphrases and generating new sentences*. Tech. rep. CORNELL UNIV ITHACA NY DEPT OF COMPUTER SCIENCE.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu (2002). "BLEU: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318.

Parisi, German I, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter (2019). "Continual Lifelong Learning with Neural Networks: A Review". In: *Neural Networks* 113, pp. 54–71.

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). "On the difficulty of training recurrent neural networks". In: *International conference on machine learning*, pp. 1310–1318.

Passos, Alexandre, Vineet Kumar, and Andrew McCallum (2014). "Lexicon Infused Phrase Embeddings for Named Entity Resolution". In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pp. 78–86.

Patel, Rupal, Sam Pilato, and Deb Roy (2004). "Beyond Linear Syntax: An Image-Oriented Communication Aid." In: *Assistive Technology Outcomes and Benefits* 1.1, pp. 57–66.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global Vectors for Word Representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Pereyra, Gabriel, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey Hinton (2016). "Regularizing Neural Networks by Penalizing Confident Output Distributions". In:

Peters, Matthew, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018a). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237.

— (June 2018b). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202.

Pinter, Yuval, Robert Guthrie, and Jacob Eisenstein (2017). "Mimicking Word Embeddings using Subword RNNs". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 102–112.

Plank, Barbara and Gertjan van Noord (2010). "Dutch Dependency Parser Performance Across Domains". In: *LOT Occasional Series* 16, pp. 123–138.

Ponsford, Jennie L, JH Olver, and C Curran (1995). "A Profile of Outcome: 2 Years After Traumatic Brain Injury". In: *Brain injury* 9.1, pp. 1–10.

Potdar, Kedar, Taher S Pardawala, and Chinmay D Pai (2017). "A comparative study of categorical variable encoding techniques for neural network classifiers". In: *International journal of computer applications* 175.4, pp. 7–9.

Pradhan, Sameer, Alessandro Moschitti, and Nianwen Xue (2013). "Towards robust linguistic analysis using ontonotes". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 143–152.

Pratt, Lorien Y (1993). "Discriminability-based transfer between neural networks". In: *Advances in neural information processing systems*, pp. 204–211.

Pravdich-Neminsky, W (1912). "Ein versuch der registrierung der elektrischen gehirnerscheinungen". In: *Zentralbl Physiol* 27, pp. 951–960.

Premack, David and Guy Woodruff (1978). "Does the Chimpanzee Vave a Theory of Mind?" In: *Behavioral and brain sciences* 1.4, pp. 515–526.

Press, Ofir, Amir Bar, Ben Bogin, Jonathan Berant, and Lior Wolf (2017). "Language Generation with Recurrent Generative Adversarial Networks without Pre-Training". In: *Proceedings of the First Workshop on Learning to Generate Natural Language*. Sydney, Australia.

Radovanović, Miloš, Alexandros Nanopoulos, and Mirjana Ivanović (2009). "Nearest Neighbors in High-Dimensional Data: The Emergence and Influence of Hubs". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, pp. 865–872.

— (2010). "Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data". In: *Journal of Machine Learning Research* 11.Sep, pp. 2487–2531.

Rajpurkar, Pranav, Robin Jia, and Percy Liang (July 2018). "Know What You Don't Know: Unanswerable Questions for SQuAD". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 784–789. DOI: 10.18653/v1/P18-2124.

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang (2016a). "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392.

— (Nov. 2016b). "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 2383–2392. DOI: 10.18653/v1/D16-1264.

Rasmy, Laila, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi (2020). "Med-BERT: pre-trained contextualized embeddings on large-scale structured electronic health records for disease prediction". In: *arXiv preprint arXiv:2005.12833*.

Raunak, Vikas, Siddharth Dalmia, Vivek Gupta, and Florian Metze (2020). "On Long-Tailed Phenomena in Neural Machine Translation". In: *arXiv preprint arXiv:2010.04924*.

Řehůřek, Radim and Petr Sojka (May 2010). "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, pp. 45–50.

Rezeika, Aya, Mihaly Benda, Piotr Stawicki, Felix Gembler, Abdul Saboor, and Ivan Volosyak (2018). "Brain–Computer Interface Spellers: A review". In: *Brain sciences* 8.4, p. 57.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic backpropagation and approximate inference in deep generative models". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning-Volume 32*, pp. II–1278.

Robins, Anthony (1993). "Catastrophic forgetting in neural networks: the role of rehearsal mechanisms". In: *Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*. IEEE, pp. 65–68.

Rogers, Anna, Olga Kovaleva, and Anna Rumshisky (2020). "A primer in bertology: What we know about how bert works". In: *arXiv preprint arXiv:2002.12327*.

Rolston, John D, Dario J Englot, Susannah Cornes, and Edward F Chang (2016). "Major and Minor Complications in extraoperative electrocorticography: a review of a national database". In: *Epilepsy research* 122, pp. 26–29.

Rosasco, Lorenzo, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri (2004). "Are Loss Functions All the Same?" In: *Neural Computation* 16.5, pp. 1063–1076.

Rosenblatt, Frank (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.

Rubin, Sue (2002). "One Size Does Not Fit All – Matching a Thinking System to a Student". In: *TASH Connections* 28.10.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986). "Learning Representations by Back-Propagating Errors". In: *nature* 323.6088, pp. 533–536.

Ryan, David B, GE Frye, George Townsend, DR Berry, S Mesa-G, Nathan A Gates, and Eric W Sellers (2010). "Predictive spelling with a P300-based brain–computer interface: increasing the rate of communication". In: *Intl. Journal of Human–Computer Interaction* 27.1, pp. 69–84.

Salton, Gerard, Anita Wong, and Chung-Shu Yang (1975). "A vector space model for automatic indexing". In: *Communications of the ACM* 18.11, pp. 613–620.

Schick, Timo and Hinrich Schütze (2019). "Attentive Mimicking: Better Word Embeddings by Attending to Informative Contexts". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 489–494.

Schmandt-Besserat, Denise (2010). *How Writing Came About*. University of Texas Press.

Schmandt-Besserat, Denise and Michael Erard (2008). "Origins and forms of writing". In: *Handbook of research on writing: History, society, school, individual, text*, pp. 7–22.

Schwartz, Roy, Jesse Dodge, Noah A Smith, and Oren Etzioni (2019). "Green AI". In: *arXiv preprint arXiv:1907.10597*.

Schwenk, Holger (July 2007). "Continuous Space Language Models". In: *Comput. Speech Lang.* 21.3, pp. 492–518. ISSN: 0885-2308. DOI: 10.1016/j.csl.2006.09.003.

Schwenk, Holger, Fethi Bougares, and Loic Barrault (2014). "Efficient training strategies for deep neural network language models". In: *NIPS workshop on Deep Learning and Representation Learning*.

See, Abigail, Stephen Roller, Douwe Kiela, and Jason Weston (June 2019). "What makes a good conversation? How controllable attributes affect human judgments". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (Aug. 2016). "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

Papers). Berlin, Germany: Association for Computational Linguistics, pp. 1715–1725. DOI: 10.18653/v1/P16-1162.

Serban, Iulian V, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau (2015). "Hierarchical Neural Network Generative Models for Movie Dialogues". In: *arXiv preprint arXiv:1507.04808* 7.8, pp. 434–441.

Serban, Iulian Vlad, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio (2016). "A hierarchical latent variable encoder-decoder model for generating dialogues". In: *arXiv preprint arXiv:1605.06069*.

Shah, Kashif, Raymond WM Ng, Fethi Bougares, and Lucia Specia (2015). "Investigating continuous space language models for machine translation quality estimation". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1073–1078.

Shane, Howard C, Sarah Blackstone, Gregg Vanderheiden, Michael Williams, and Frank DeRuyter (2012). "Using AAC technology to access the world". In: *Assistive technology* 24.1, pp. 3–13.

Shannon, Claude (July 1948a). "A Mathematical Model of Communication". In: *The Bell System Technical Journal* 27, pp. 379–423.

— (1951). "Prediction and Entropy of Printed English". In: *Bell System Technical Journal* 30, pp. 51–64.

Shannon, Claude E (1948b). "A mathematical theory of communication". In: *The Bell system technical journal* 27.3, pp. 379–423.

Shao, Louis, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil (2016). "Generating Long and Diverse Responses with Neural Conversation Models". In:

Sheu, Heng-Shiou and Sheng Li (2020). "Context-aware Graph Embedding for Session-based News Recommendation". In: *Fourteenth ACM Conference on Recommender Systems*, pp. 657–662.

Shin, Hanul, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim (2017). "Continual Learning with Deep Generative Replay". In: *Advances in Neural Information Processing Systems*, pp. 2990–2999.

Shrivastava, Anshumali and Ping Li (2014). "Asymmetric LSH (ALSH) for Sublinear Time Maximum Inner Product Search (MIPS)". In: *Advances in neural information processing systems*, pp. 2321–2329.

Smolensky, Paul (1988). "On the proper treatment of connectionism". In: *Behavioral and brain sciences* 11.1, pp. 1–23.

Speer, Robyn, Joshua Chin, and Catherine Havasi (2016). "Conceptnet 5.5: An Open Multilingual Graph of General Knowledge". In: *arXiv preprint arXiv:1612.03975*.

Speier, William, C Arnold, and Nader Pouratian (2016). "Integrating Language Models into Classifiers for BCI Communication: a review". In: *Journal of neural engineering* 13.3, p. 031002.

Speier, William, Corey Arnold, Jessica Lu, Ricky K Taira, and Nader Pouratian (2011). "Natural language processing with dynamic classification improves P300 speller accuracy and bit rate". In: *Journal of neural engineering* 9.1, p. 016004.

Srivastava, Akash, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton (2017). "Veegan: Reducing mode collapse in gans using implicit variational learning". In: *Advances in Neural Information Processing Systems*, pp. 3308–3318.

Stawicki, Piotr, Felix Gembler, Aya Rezeika, and Ivan Volosyak (2017). "A novel hybrid mental spelling application based on eye tracking and SSVEP-based BCI". In: *Brain sciences* 7.4, p. 35.

Strubell, Emma, Ananya Ganesh, and Andrew McCallum (2019). "Energy and Policy Considerations for Deep Learning in NLP". In: *arXiv preprint arXiv:1906.02243*.

Subramanian, Sandeep, Sai Rajeswar, Francis Dutil, Chris Pal, and Aaron Courville (Aug. 2017). "Adversarial Generation of Natural Language". In: *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada: Association for Computational Linguistics, pp. 241–251. DOI: 10.18653/v1/W17-2629.

Sun, Baochen, Jiashi Feng, and Kate Saenko (2015). "Return of frustratingly easy domain adaptation". In: *arXiv preprint arXiv:1511.05547*.

Sutskever, Ilya, James Martens, and Geoffrey E Hinton (2011). "Generating Text with Recurrent Neural Networks". In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1017–1024.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*, pp. 3104–3112.

Szafir, Daniel and Bilge Mutlu (2012). "Pay attention! Designing adaptive agents that monitor and improve user engagement". In: *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 11–20.

Tamari, Ronen, Chen Shani, Tom Hope, Miriam R L Petruck, Omri Abend, and Dafna Shahaf (July 2020). "Language (Re)modelling: Towards Embodied Language Understanding". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 6268–6281. DOI: 10.18653/v1/2020.acl-main.559.

Tannen, Deborah et al. (2005). *Conversational style: Analyzing talk among friends*. Oxford University Press.

Tevet, Guy and Jonathan Berant (2020). "Evaluating the Evaluation of Diversity in Natural Language Generation". In: *arXiv preprint arXiv:2004.02990*.

Thorndike, Edward L (1920). "Intelligence and Its Uses." In: *Harper's magazine*.

Thrun, Sebastian and Tom M Mitchell (1995). "Lifelong robot learning". In: *Robotics and autonomous systems* 15.1-2, pp. 25–46.

Todorov, Emanuel (2007). "Linearly-solvable Markov decision problems". In: *Advances in neural information processing systems*, pp. 1369–1376.

Tsarfaty, Reut, Dan Bareket, Stav Klein, and Amit Seker (2020). "From SPMRL to NMRL: What Did We Learn (and Unlearn) in a Decade of Parsing Morphologically-Rich Languages (MRLs)?" In: *arXiv preprint arXiv:2005.01330*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention Is All You Need". In: *Advances in neural information processing systems*, pp. 5998–6008.

Ven, Gido M van de and Andreas S Tolias (2019). "Three Scenarios for Continual Learning". In: *arXiv preprint arXiv:1904.07734*.

Vertanen, Keith and Per Ola Kristensson (2011). "A versatile dataset for text entry evaluations based on genuine mobile emails". In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, pp. 295–298.

Vidal, Jacques J (1973). "Toward Direct Brain-Computer Communication". In: *Annual review of Biophysics and Bioengineering* 2.1, pp. 157–180.

Vijayakumar, Ashwin K, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra (2016). "Diverse Beam Search: Decoding Diverse Solutions from Neural Sequence Models". In: *arXiv preprint arXiv:1610.02424*.

Waldert, Stephan (2016). "Invasive vs. Non-Invasive Neuronal Signals for Brain-Machine Interfaces: will one prevail?" In: *Frontiers in neuroscience* 10, p. 295.

Waldert, Stephan, Tobias Pistohl, Christoph Braun, Tonio Ball, Ad Aertsen, and Carsten Mehring (2009). "A review on directional information in neural signals for brain-machine interfaces". In: *Journal of Physiology-Paris* 103.3-5, pp. 244–254.

Walker, Christopher Bromhead Fleming (1987). *Cuneiform*. Vol. 3. Univ of California Press.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman (2018). "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355.

Wang, Yuxuan, Yutai Hou, Wanxiang Che, and Ting Liu (2020). "From static to dynamic word representations: a survey". In: *International Journal of Machine Learning and Cybernetics*, pp. 1–20.

Welleck, Sean, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston (2020). "Neural Text Generation With Unlikelihood Training". In: *International Conference on Learning Representations*.

Wen, Tsung-Hsien, Aaron Heidel, Hung-yi Lee, Yu Tsao, and Lin-Shan Lee (2013). "Recurrent neural network based language model personalization by social network crowdsourcing." In: *INTERSPEECH*, pp. 2703–2707.

Weston, Jason, Emily Dinan, and Alexander Miller (2018). "Retrieve and Refine: Improved Sequence Generation Models For Dialogue". In: *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pp. 87–92.

Wiegand, Karl and Rupal Patel (2012a). "Non-syntactic word prediction for AAC". In: *Proceedings of the Third Workshop on Speech and Language Processing for Assistive Technologies*, pp. 28–36.

— (2012b). "SymbolPath: a continuous motion overlay module for icon-based assistive communication". In: *Proceedings of the 14th international ACM SIGACCESS conference on Computers and accessibility*, pp. 209–210.

Williams, Ronald J and Jing Peng (1991). "Function Optimization Using Connectionist Reinforcement Learning Algorithms". In: *Connection Science* 3.3, pp. 241–268.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew (2019). "HuggingFace's Transformers: State-of-the-art Natural Language Processing". In: *ArXiv* abs/1910.03771.

Wolpaw, Jonathan R, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan (2002). "Brain–Computer Interfaces for Communication and Control". In: *Clinical neurophysiology* 113.6, pp. 767–791.

Woodland, PC, T Hain, GL Moore, TR Niesler, D Povey, A Tuerk, and EWD Whittaker (1999). "The 1998 HTK broadcast news transcription system: Development and results". In: *Proc. DARPA Broadcast News Workshop*. Morgan Kaufmann Los Altos, CA, pp. 265–270.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean (2016a). *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. arXiv: 1609.08144 [cs.CL].

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. (2016b). "Google's Neural Machine Translation System: Bridging the Gap Between Human and Machine Translation". In: *arXiv preprint arXiv:1609.08144*.

Xu, Jingjing, Xuancheng Ren, Junyang Lin, and Xu Sun (2018). "Diversity-promoting gan: A cross-entropy based generative adversarial network for diversified text generation". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3940–3949.

Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio (2015). "Show, attend and tell: Neural image caption generation with visual attention". In: *International conference on machine learning*, pp. 2048–2057.

Yang, Zhen, Wei Chen, Feng Wang, and Bo Xu (June 2018a). "Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 1346–1355. DOI: 10.18653/v1/N18-1122.

Yang, Zhilin, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen (2018b). "Breaking the Softmax Bottleneck: A High-Rank RNN Language Model". In: *International Conference on Learning Representations*.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le (2019). "Xlnet: Generalized Autoregressive Pretraining for Language Understanding". In: *Advances in neural information processing systems*, pp. 5753–5763.

Yao, Yuan, Lorenzo Rosasco, and Andrea Caponnetto (2007). "On early stopping in gradient descent learning". In: *Constructive Approximation* 26.2, pp. 289–315.

Yu, Lantao, Weinan Zhang, Jun Wang, and Yong Yu (2017). "Seqgan: Sequence generative adversarial nets with policy gradient". In: *Thirty-first AAAI conference on artificial intelligence*.

Yu, Zhiwei and Xiaojun Wan (2019). "How to Avoid Sentences Spelling Boring? Towards a Neural Approach to Unsupervised Metaphor Generation". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 861–871.

Yue, Yuguang, Yunhao Tang, Mingzhang Yin, and Mingyuan Yin (2020). "Discrete Action On-Policy Learning with Action-Value Critic". In: *arXiv preprint arXiv:2002.03534*.

Zafrir, Ofir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat (2019). "Q8bert: Quantized 8bit Bert". In: *arXiv preprint arXiv:1910.06188*.

Zaykovskiy, Dmitry (2006). "Survey of the speech recognition techniques for mobile devices". In: *Proc. of DS Publications*.

Zellers, Rowan, Yonatan Bisk, Roy Schwartz, and Yejin Choi (2018). "SWAG: A Large-Scale Adversarial Dataset for Grounded Commonsense Inference". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 93–104. DOI: 10.18653/v1/D18-1009.

Zhang, Tianyi, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi (2019). "BERTScore: Evaluating Text Generation with BERT". In: *International Conference on Learning Representations*.

Zhang, Yizhe, Zhe Gan, and Lawrence Carin (2016). "Generating text via adversarial training". In: *NIPS workshop on Adversarial Training*. Vol. 21.

Zhao, Bing, Matthias Eck, and Stephan Vogel (2004). "Language Model Adaptation for Statistical Machine Translation via Structured Query Models". In: *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pp. 411–417.

Zhao, Shengjia, Jiaming Song, and Stefano Ermon (2017a). "Towards deeper understanding of variational autoencoding models". In: *arXiv preprint arXiv:1702.08658*.

Zhao, Tiancheng, Ran Zhao, and Maxine Eskenazi (2017b). "Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 654–664.

Zipf, G. K. (1935). *The psycho-biology of language*. Oxford, England: Houghton, Mifflin, pp. ix, 336–ix, 336.

Zoph, Barret, Ashish Vaswani, Jonathan May, and Kevin Knight (2016). "Simple, Fast Noise-Contrastive Estimation for Large RNN Vocabularies". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1217–1222.

Appendix A

A.1 Chapter 3

A.1.1 Qualitative Examples

Following section 3.4.4, we provide a number of different qualitative examples for the mid- and low- bins predictions for the different models. In the examples below, the target token (that the model was required to predict) is in *italics*. Where appropriate, we include a portion of the sentence following the target token, to provide the reader with context; recall that the model did not have access to this information at the time that it was queried.

`G (NYT,low):` a report on japan trade was issued today by the labor-industry coalition on international trade a coalition that brings together the afl-cio and major u.s. steel textile and semiconductor workers along with the economic strategy institute an *industry-financed* research group ...

`ctg(NYT,low):` a report on japan trade was issued today by the labor-industry coalition on international trade a coalition that brings together the afl-cio and major u.s. steel textile and semiconductor workers along with the economic strategy institute an *international* research group ...

`G (PMC,low):` ... peptides were eluted by centrifugation followed by NUM additional *elutions* with NUM μl

`ctg(PMC,low):` ... peptides were eluted by centrifugation followed by NUM additional *NUM* with NUM μl

`G (NYT,mid):` among the NUM american women who have hysterectomies each year thousands may die prematurely of heart disease because doctors removed their *ovaries* along with ...

`ctg(NYT,mid):` among the NUM american women who have hysterectomies each year thousands may die prematurely of heart disease because doctors removed their *way* along with ...

`G (NYT,mid):` this mold covers the leaf in soot preventing the cells underneath from absorbing sunlight and conducting *photosynthesis* eventually ...

`ctg(NYT,mid):` this mold covers the leaf in soot preventing the cells underneath from absorbing sunlight and conducting *the* eventually ...

`G (PMC,mid):` in particular our objectives were to determine the association between NUM reported alcohol *misuse* and hiv sexual ...

`ctg(PMC,mid):` in particular our objectives were to determine the association between NUM reported alcohol *and* and hiv sexual ...

`G (PMC,mid):` NUM patients underwent cardiopulmonary resuscitation including NUM patients with respiratory arrest and pronounced *bradycardia*

`ctg(PMC,mid):` NUM patients underwent cardiopulmonary resuscitation including NUM patients with respiratory arrest and pronounced *NUM*

A.2 Chapter 5

A.2.1 A Kernel Transform

hypothesis

We hypothesize that if dense regions hurdle the performance, then if the region in particular, and the space in general, would be warped to reduce similarities and congestion in some highly populated embedding regions, it will improve performance. For that reason, we are to find a transform that warps the space such that it 'spreads' the embeddings (making those regions and the space more spacious), while maintaining the original relationships among them as much as possible. We presume that there exist a kernel that warps the initial space while maintaining the relationships and distance proportions, and sufficiently make the clamped regions in particular, and the space in general, more spacious.

Kernel Search

A kernel that qualifies for that purpose will show that the relationships in cosine similarity among several predefined embedding vectors in that space are maintained after applying the transform. Otherwise, we resume to search for a suitable kernel.

transform type

The chosen kernel is a power of 0.75 which is be applied element-wise on each of the vectors. The reason for its choice is described in section A.2.1.

measuring success

The means to measure the ability of the kernel to maintain the original relationship proportions constructed in the embedding space are by comparing cosine similarity proportions over a sample of different types of vectors (representing known terms in the space). It is important to emphasize to the reader that we describe the protocol to follow in order to find a suitable transform and this is the main contribution in that regard, rather than the specific number we found to be suitable to our embedding space.

distance proportion

To evaluate whether the transform maintained the original relationships proportions, Figure A.1 describes the cosine similarity between different terms, pre- and post-transform from left to right respectively. Comparing the color gradients in the figure shows that both images follow a similar pattern of gradients but of a darker color scale on the post transform image, suggesting that the relationships proportions maintained but were weakened.

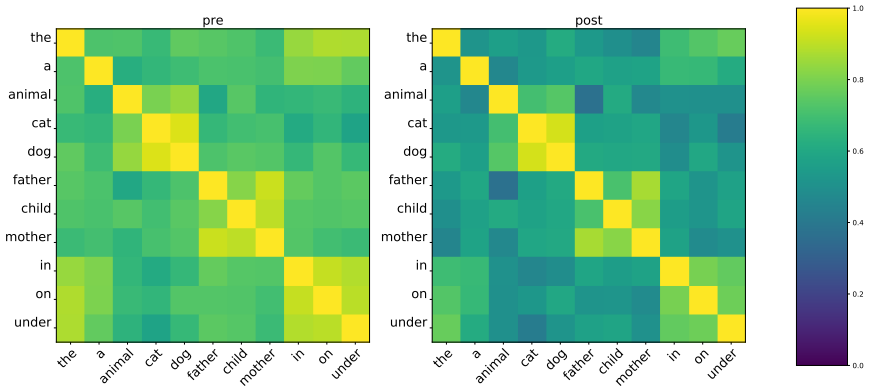


Figure A.1: cosine similarity among a sample set of embedding

In Figure A.2 there are additional plots to describe the corresponding division and subtraction of the pre- and post- transform seen in Figure A.1. Both plots show that a similar color is maintained across of term combinations after division and subtraction, suggesting that to a reasonable degree the terms have gone through a similar change after the transform.

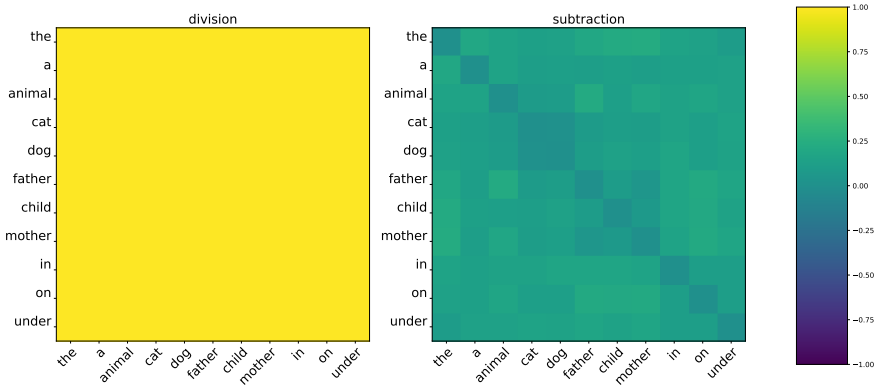


Figure A.2: left: dividing pre with post, right: subtracting post from pre

Therefore, we conclude that this transform maintains the proportions of the pre-transform relationships established among the terms in the embedding space.

A.3 Chapter 6

A.3.1 Protocol to elicit paraphrase pairs

We provide here the guidelines to elicit rare/common paraphrase sentences. First, a simple routine extracts possible triples that constructs the linguistic relationship desired of hypo/hypernym and a hypernym-sibling ((x_r, x_h, x_a)). We begin by taking the intersection of the vocabulary seen in the training partition of the Wiki-103 corpus with that found in WordNet (using the NLTK package (Loper et al., 2002)), and then further filtered for vocabulary items with WordNet entries exhibiting the desired linguistic relationship. (A synonym/antonym construction could also have been chosen alternatively).

The frequency dynamic for the rare/common triples ($x_r/x_c, x_h, x_a$) was (low, mid/high,mid/high) and for the common (mid/high, mid/high, mid/high) respectively. Words occurring fewer than 50 times in the Wiki-103 training partition were categorized as "low," and were categorized as "mid/high" otherwise. Finally, a human annotator manually identified an appropriate context sentence for each target word via online search across the following web-based dictionaries: `merriam-webster.com`, `thesaurus.com`, `sentencedict.com`, and `dictionary.cambridge.org`.

A.3.2 The Rare-Word Triples

Here are the 50 rare-word triples following (x_r, x_h, x_a) order

(a) The `afterdamp` occurring in such situations is a mixture of carbon dioxide and carbon monoxide.

(b) The `liquid` occurring in such situations is a mixture of carbon dioxide and carbon monoxide.

(c) The `gas` occurring in such situations is a mixture of carbon dioxide and carbon monoxide.

(a) The gradual `accretion` of terror over many years left hundred dead and thousands wounded

(b) The gradual `increase` of terror over many years left hundred dead and thousands wounded

(c) The gradual `decrease` of terror over many years left hundred dead and thousands wounded

(a) From 1894 to 1902 he was at the university of Texas as `adjunct` professor of political science, professor (after 1900), and dean of the faculty (after 1899).

(b) From 1894 to 1902 he was at the university of Texas as `affiliate` professor of political science, professor (after 1900), and dean of the faculty (after 1899).

(c) From 1894 to 1902 he was at the university of Texas as `successor` professor of political science, professor (after 1900), and dean of the faculty (after 1899).

(a) Very often, the letters one might be comfortable and familiar with are `allographs` of quite different sounds in the second language.

(b) Very often, the letters one might be comfortable and familiar with are `characters` of quite different sounds in the second language.

(c) Very often, the letters one might be comfortable and familiar with are `marks` of quite different sounds in the second language.

(a) Actual surface `amphibole` deposits in residential areas were ignored for testing purposes.

(b) Actual surface `mineral` deposits in residential areas were ignored for testing purposes.

(c) Actual surface `atom` deposits in residential areas were ignored for testing purposes.

(a) Like all serious `anglers,` she never keeps what she catches.

(b) Like all serious `fishermen,` she never keeps what she catches.

(c) Like all serious `crewmen,` she never keeps what she catches.

(a) If you are a serious `angler,` bring your own equipment.

(b) If you are a serious `fisherman,` bring your own equipment.

(c) If you are a serious `cook,` bring your own equipment.

(a) nonetheless, the roofline makes an elegant night-time `apparition.`

(b) nonetheless, the roofline makes an elegant night-time `appearance.`

(c) nonetheless, the roofline makes an elegant night-time `disappearance.`

(a) Only a shallow `arroyo` and barrel cactus break the monotony.

(b) Only a shallow `gully` and barrel cactus break the monotony.

(c) Only a shallow `hollow` and barrel cactus break the monotony.

(a) That's when your pants are pulled up your `arse.`

(b) That's when your pants are pulled up your `anus.`

(c) That's when your pants are pulled up your `cervix.`

(a) He's wearing his famous `astrakhan` hat.

(b) He's wearing his famous `fur` hat.

(c) He's wearing his famous `leather` hat.

(a) For women undergoing breast `augmentation,` the rate of complications was lower.

(b) For women undergoing breast `increase,` the rate of complications was lower.

(c) For women undergoing breast `decrease,` the rate of complications was lower.

(a) This is not a guy pursuing a romantic, outdoorsy `avocation.`

(b) This is not a guy pursuing a romantic, outdoorsy `pastime.`

(c) This is not a guy pursuing a romantic, outdoorsy `nightlife.`

(a) Since then, talk shows have become a national `avocation.`

(b) Since then, talk shows have become a national `pastime.`

(c) Since then, talk shows have become a national `sport.`

(a) There, in an elegant casino, well-heeled patrons would play roulette at green `baize` tables.

(b) There, in an elegant casino, well-heeled patrons would play roulette at green `fabric` tables.

(c) There, in an elegant casino, well-heeled patrons would play roulette at green `covering` tables.

(a) We then repaired to a conference room in the guesthouse and faced each other across a green `baize` table.

(b) We then repaired to a conference room in the guesthouse and faced each other across a green `fabric` table.

(c) We then repaired to a conference room in the guesthouse and faced each other across a green `decorated` table.

(a) In 1970 Vukobratovi proposed a theoretical model to explain and control `biped` locomotion.

(b) In 1970 Vukobratovi proposed a theoretical model to explain and control `animal` locomotion.

(c) In 1970 Vukobratovi proposed a theoretical model to explain and control `plant` locomotion.

(a) They are large `biped` robots and they have a significant amount of strength.

(b) They are large `animal` robots and they have a significant amount of strength.

(c) They are large `plant` robots and they have a significant amount of strength.

(a) `blanch` grape leaves in boiling water with half the lemon juice.

(b) `cook` grape leaves in boiling water with half the lemon juice.

(c) `blend` grape leaves in boiling water with half the lemon juice.

(a) Williecake is like a giant, fine chocolate-frosted `brownie.`

(b) Williecake is like a giant, fine chocolate-frosted `cookie.`

(c) Williecake is like a giant, fine chocolate-frosted `pancake.`

(a) Soak the `bulgur` in warm water to cover for about an hour.

(b) Soak the `wheat` in warm water to cover for about an hour.

(c) Soak the `rice` in warm water to cover for about an hour.

(a) Like most women in Kabul, she still wore her `burqa.`

(b) Like most women in Kabul, she still wore her `garment.`

(c) Like most women in Kabul, she still wore her `footwear.`

(a) The garden features rhododendrons, roses, canna, hostas, shade plants and varieties of shrubs, topiaries and `trees.`

(b) The garden features rhododendrons, roses, a herb, hostas, shade plants and varieties of shrubs, topiaries `and` trees.

(c) The garden features rhododendrons, roses, a weed, hostas, shade plants and varieties of shrubs, topiaries `and` trees.

(a) They form a `canopy` that blocks most of the harsh sunlight.

(b) They form a `shelter` that blocks most of the harsh sunlight.

(c) They form a `housing` that blocks most of the harsh sunlight.

(a) The concrete `carport` was empty but for two bare metal chairs.

(b) The concrete `garage` was empty but for two bare metal chairs.

(c) The concrete `shed` was empty but for two bare metal chairs.

(a) Some birds nest in tree `cavities.`

(b) Some birds nest in tree `holes.`

(c) Some birds nest in tree `basins.`

(a) We take in the view as a `cerulean` butterfly flits by.

(b) We take in the view as a `blue` butterfly flits by.

(c) We take in the view as a `green` butterfly flits by.

(a) Between shows, she will change to a strapless orange `chiffon` gown.

(b) Between shows, she will change to a strapless orange `fabric` gown.

(c) Between shows, she will change to a strapless orange `sheet` gown.

(a) As the woman's `chiffon` scarf blew in the breeze, I realized I could see straight through it.

(b) As the woman's `fabric` scarf blew in the breeze, I realized I could see straight through it.

(c) As the woman's `covering` scarf blew in the breeze, I realized I could see straight through it.

(a) When used in this way, each `cirrocumulus` element is referred to as a " cloudlet ".

(b) When used in this way, each `cloud` element is referred to as a " cloudlet ".

(c) When used in this way, each `storm` element is referred to as a " cloudlet ".

(a) A `cirrocumulus` is filled with a large group of tiny water droplets that can be seen in the air.

(b) A `cloud` is filled with a large group of tiny water droplets that can be seen in the air.

(c) A `halo` is filled with a large group of tiny water droplets that can be seen in the air.

(a) when preparing indian-style chai adding `cloves` is optional

(b) when preparing indian-style chai adding `spice` is optional

(c) when preparing indian-style chai adding `herbs` is optional

(a) two stars thought to be `coeval` because they have nearly the same mass and brightness

(b) two stars thought to be `contemporary` because they have nearly the same mass and brightness

(c) two stars thought to be `friends` because they have nearly the same mass and brightness

(a) order `coleslaw` made with vinaigrette instead of choosing potato salad.

(b) order `salad` made with vinaigrette instead of choosing potato salad.

(c) order `pizza` made with vinaigrette instead of choosing potato salad.

(a) And you will never have to wear a `turquoise` jacket again.

(b) And you will never have to wear a `green` jacket again.

(c) And you will never have to wear a `blue` jacket again.

(a) Every Frenchman, and woman, seems to own a `poodle.`

(b) Every Frenchman, and woman, seems to own a `dog.`

(c) Every Frenchman, and woman, seems to own a `cat.`

(a) The mouth, with its jaws, forms a conical outgrowth which projects backwards, so that its apex lies beneath the `prothorax.`

(b) The mouth, with its jaws, forms a conical outgrowth which projects backwards, so that its apex lies beneath the `thorax.`

(c) The mouth, with its jaws, forms a conical outgrowth which projects backwards, so that its apex lies beneath the `buttocks.`

(a) He turned and stalked away, disappearing with a `puff` of cool breeze.

(b) He turned and stalked away, disappearing with a `smoke` of cool breeze.

(c) He turned and stalked away, disappearing with a `eat` of cool breeze.

(a) He drinks to `quench` his thirst, and that is all.

(b) He drinks to `meet` his thirst, and that is all.

(c) He drinks to `feed` his thirst, and that is all.

(a) He drinks to `quench` his thirst, and that is all.

(b) He drinks to `meet` his thirst, and that is all.

(c) He drinks to `serve` his thirst, and that is all.

(a) Right in the middle stood a cute little `redhead` about to make her shot

(b) Right in the middle stood a cute little `person` about to make her shot

(c) Right in the middle stood a cute little `plant` about to make her shot

(a) 4 large portobello mushroom caps, 4 to 6 inches in `diameter`

(b) 4 large portobello mushroom caps, 4 to 6 inches in `length`

(c) 4 large portobello mushroom caps, 4 to 6 inches in `mass`

(a) The decision to `discontinue` the game was done as preventive medicine

(b) The decision to `stop` the game was done as preventive medicine

(c) The decision to `continue` the game was done as preventive medicine

(a) The `shipowner` shall be liable for compensation for any losses suffered by the charterer thereby.

(b) The `owner` shall be liable for compensation for any losses suffered by the charterer thereby.

(c) The `male` shall be liable for compensation for any losses suffered by the charterer thereby.

(a) The glowing embers of the pyrotechnics `singe` the roofs of the estate.

(b) The glowing embers of the pyrotechnics `burn` the roofs of the estate.

(c) The glowing embers of the pyrotechnics `wound` the roofs of the estate.

(a) As a `sinologist` , Stephen Owen is famous for his works on classical Chinese literature.

(b) As a `scholar` , Stephen Owen is famous for his works on classical Chinese literature.

(c) As a `genius` , Stephen Owen is famous for his works on classical Chinese literature.

(a) Under conditions of obligatory isolation and social distancing, common people invented new kinds of `sociality` and new genres of epidemic expressions.

(b) Under conditions of obligatory isolation and social distancing, common people invented new kinds of `nature` and new genres of epidemic expressions.

(c) Under conditions of obligatory isolation and social distancing, common people invented new kinds of `discipline` and new genres of epidemic expressions.

(a) Hamlet's `soliloquy,` and the Navy Seal copypasta, have drawn the ire of the star's entertainment company Roc Nation LLC, which has asked YouTube to remove the material for copyright violation

(b) Hamlet's `speech,` and the Navy Seal copypasta, have drawn the ire of the star's entertainment company Roc Nation LLC, which has asked YouTube to remove the material for copyright violation

(c) Hamlet's `audio,` and the Navy Seal copypasta, have drawn the ire of the star's entertainment company Roc Nation LLC, which has asked YouTube to remove the material for copyright violation

(a) The `sower` has planted you there for a purpose.

(b) The `farmer` has planted you there for a purpose.

(c) The `builder` has planted you there for a purpose.

(a) The `speedup` in the rate of change, especially within the past two decades, is well known.

(b) The `speed` in the rate of change, especially within the past two decades, is well known.

(c) The `travel` in the rate of change, especially within the past two decades, is well known.

(a) The clouds moved behind the white `steeple.`

(b) The clouds moved behind the white `tower.`

(c) The clouds moved behind the white `platform.`

A.3.3 The Common-Word Triples

Here are the 50 common-word triples following (x_c, x_h, x_a) order

(a) 4 large portobello mushroom caps, 4 to 6 inches in `diameter`

(b) 4 large portobello mushroom caps, 4 to 6 inches in `length`

(c) 4 large portobello mushroom caps, 4 to 6 inches in `mass`

(a) Nike decided to `discontinue` the new brand of shoes

(b) Nike decided to `stop` the new brand of shoes

(c) Nike decided to `continue` the new brand of shoes

(a) Because of the poor economy, the factory will immediately `discontinue` operations.

(b) Because of the poor economy, the factory will immediately `cease` operations.

(c) Because of the poor economy, the factory will immediately `continue` operations.

(a) He was educated in a school at Jesmond, kept by Mr Ivison, a `cleric` of the church of England.

(b) He was educated in a school at Jesmond, kept by Mr Ivison, a `clergyman` of the church of England.

(c) He was educated in a school at Jesmond, kept by Mr Ivison, a `rabbi` of the church of England.

(a) In 1878 he was elected a Foreign `affiliate` of the Royal Astronomical Society.

(b) In 1878 he was elected a Foreign `associate` of the Royal Astronomical Society.

(c) In 1878 he was elected a Foreign `successor` of the Royal Astronomical Society.

(a) Because my cat has four feet and not two, it is definitely not a `biped.`

(b) Because my cat has four feet and not two, it is definitely not a `bipedal.`

(c) Because my cat has four feet and not two, it is definitely not a `quadrupedal.`

(a) Becoming `cloudy` and dry from the north.

(b) Becoming `opaque` and dry from the north.

(c) Becoming `clearer` and dry from the north.

(a) `posterior` fossa neoplasms or multiple sclerosis may rarely cause vertigo or hearing loss.

(b) `backside` fossa neoplasms or multiple sclerosis may rarely cause vertigo or hearing loss.

(c) `anterior` fossa neoplasms or multiple sclerosis may rarely cause vertigo or hearing loss.

(a) The UN declared it a `safe` area.

(b) The UN declared it a `guarded` area.

(c) The UN declared it a `unsafe` area.

(a) Here we `reward` performance, not face time.

(b) Here we `reinforce` performance, not face time.

(c) Here we `penalize` performance, not face time.

(a) a capital `M`

(b) a capital `letter`

(c) a capital `space`

(a) Between them they `ate` an entire cake.

(b) Between them they `eat` an entire cake.

(c) Between them they `swallow` an entire cake.

(a) A `peace` accord was reached on 26 March.

(b) A `treaty` accord was reached on 26 March.

(c) A `contract` accord was reached on 26 March.

(a) The information is transmitted electronically to the central `computer.`

(b) The information is transmitted electronically to the central `machine.`

(c) The information is transmitted electronically to the central `toy.`

(a) She suggested an `analogy` between the human heart and a pump.

(b) She suggested an `comparison` between the human heart and a pump.

(c) She suggested an `search` between the human heart and a pump.

(a) It does not say anything explicit about the `legitimacy` of corporate power in relation to society generally.

(b) It does not say anything explicit about the `credibility` of corporate power in relation to society generally.

(c) It does not say anything explicit about the `complexity` of corporate power in relation to society generally.

(a) The captain gave the `order` to abandon ship.

(b) The captain gave the `command` to abandon ship.

(c) The captain gave the `proposal` to abandon ship.

(a) The price of cigarettes is set to `rise` again.

(b) The price of cigarettes is set to `increase` again.

(c) The price of cigarettes is set to `decrease` again.

(a) The `mean` of the three numbers 7, 12 and 20 is 13, because the total of 7, 12 and 20 is 39, and 39 divided by 3 is 13.

(b) The `expectation` of the three numbers 7, 12 and 20 is 13, because the total of 7, 12 and 20 is 39, and 39 divided by 3 is 13.

(c) The `median` of the three numbers 7, 12 and 20 is 13, because the total of 7, 12 and 20 is 39, and 39 divided by 3 is 13.

(a) When he entered the kitchen, bringing a great `gust` of cold air with him, he was all smiles.

(b) When he entered the kitchen, bringing a great `wind` of cold air with him, he was all smiles.

(c) When he entered the kitchen, bringing a great `wave` of cold air with him, he was all smiles.

(a) They departed Texas on a three-year `odyssey` that took them as far as Japan.

(b) They departed Texas on a three-year `journey` that took them as far as Japan.

(c) They departed Texas on a three-year `driving` that took them as far as Japan.

(a) This restriction creates a barrier for `global` electronic commerce.

(b) This restriction creates a barrier for `global` electronic commerce.

(c) This restriction creates a barrier for `global` electronic commerce.

(a) This `restriction` creates a barrier for global electronic commerce.

(b) This `regulation` creates a barrier for global electronic commerce.

(c) This `possession` creates a barrier for global electronic commerce.

(a) The `tent` protected us from the worst of the weather.

(b) The `shelter` protected us from the worst of the weather.

(c) The `housing` protected us from the worst of the weather.

(a) Chew your `fish` well before swallow.

(b) Chew your `food` well before swallow.

(c) Chew your `milk` well before swallow.

(a) Maria looks set to `overrun` the music scene with her style and image.

(b) Maria looks set to `invade` the music scene with her style and image.

(c) Maria looks set to `bombard` the music scene with her style and image.

(a) He went to work on a `ranch.`

(b) He went to work on a `farm.`

(c) He went to work on a `bank.`

(a) Better to `reign` in hell than serve in heaven.

(b) Better to `govern` in hell than serve in heaven.

(c) Better to `manage` in hell than serve in heaven.

(a) The sample of people questioned was `drawn` from the university's student register and stratified by age and gender.

(b) The sample of people questioned was `derived` from the university's student register and stratified by age and gender.

(c) The sample of people questioned was `sourced` from the university's student register and stratified by age and gender.

(a) She can't accept even mild `criticism` of her work.

(b) She can't accept even mild `disapproval` of her work.

(c) She can't accept even mild `refusal` of her work.

(a) She's waiting for you in the `conference` room upstairs.

(b) She's waiting for you in the `meeting` room upstairs.

(c) She's waiting for you in the `audience` room upstairs.

(a) Eventually the `session` came to a merciful end.

(b) Eventually the `term` came to a merciful end.

(c) Eventually the `duration` came to a merciful end.

(a) The cooperative `wave` started in Britain in the 19th century.

(b) The cooperative `movement` started in Britain in the 19th century.

(c) The cooperative `contact` started in Britain in the 19th century.

(a) He guffawed with `delight` when he heard the news.

(b) He guffawed with `pleasure` when he heard the news.

(c) He guffawed with `affection` when he heard the news.

(a) This `booklet` provides useful information about local services.

(b) This `book` provides useful information about local services.

(c) This `read` provides useful information about local services.

(a) The `chorale` hurriedly got ready and sang, and as they did, the son of the man who had died heard them.

(b) The `hymn` hurriedly got ready and sang, and as they did, the son of the man who had died heard them.

(c) The `ritual` hurriedly got ready and sang, and as they did, the son of the man who had died heard them.

(a) You can freeze any leftover `chili` for another meal.

(b) You can freeze any leftover `dish` for another meal.

(c) You can freeze any leftover `milk` for another meal.

(a) One of them, doubtless the Substitute `prosecutor,` was on his feet, playing with an unlit pipe.

(b) One of them, doubtless the Substitute `lawyer,` was on his feet, playing with an unlit pipe.

(c) One of them, doubtless the Substitute `critic,` was on his feet, playing with an unlit pipe.

(a) The judge ordered him to purge his `contempt` by apologizing to the court.

(b) The judge ordered him to purge his `dislike` by apologizing to the court.

(c) The judge ordered him to purge his `depair` by apologizing to the court.

(a) We'll `flag` the records of interest in the database and then we can give you a print-out.

(b) We'll `mark` the records of interest in the database and then we can give you a print-out.

(c) We'll `touch` the records of interest in the database and then we can give you a print-out.

(a) I'm not strong enough to `carry` him.

(b) I'm not strong enough to `bring` him.

(c) I'm not strong enough to `fly` him.

(a) That `sale` precludes further development on this site.

(b) That `agreement` precludes further development on this site.

(c) That `announcement` precludes further development on this site.

(a) However, if the `researcher` had such skills it would be very helpful to the writing process.

(b) However, if the `scientist` had such skills it would be very helpful to the writing process.

(c) However, if the `advocate` had such skills it would be very helpful to the writing process.

(a) Some couples see single women as a `threat` to their relationships.

(b) Some couples see single women as a `warning` to their relationships.

(c) Some couples see single women as a `telling` to their relationships.

(a) Where did you `bury` the cat's body?

(b) Where did you `lay` the cat's body?

(c) Where did you `park` the cat's body?

(a) They were the first expedition to `scale` the heights of Everest.

(b) They were the first expedition to `measure` the heights of Everest.

(c) They were the first expedition to `balance` the heights of Everest.

(a) The war brought infinite harm to the `nation.`

(b) The war brought infinite harm to the `people.`

(c) The war brought infinite harm to the `system.`

(a) She served a two-year stint as an `aide` to Congressman Jim McNulty.

(b) She served a two-year stint as an `assistant` to Congressman Jim McNulty.

(c) She served a two-year stint as an `volunteer` to Congressman Jim McNulty.

(a) I carry memories of my `homeland` around with me.

(b) I carry memories of my `country` around with me.

(c) I carry memories of my `borough` around with me.

(a) Damage was confined to a small portion of the `castle.`

(b) Damage was confined to a small portion of the `mansion.`

(c) Damage was confined to a small portion of the `lodge.`