University of York

Final year project

Word Embeddings

Joel Strouts

supervised by

Dr. Ben Powell

Abstract

In this report we attempt to provide a complete introduction to the core theories of word embeddings (numerical vector representations of words as they appear in text) sufficient to enable further reading of more advanced topics. This introduction covers briefly the linguistic context, theoretical development, most impactful implementations, and practical considerations of application of word embedding methods. In the first section we discuss the nature of the words, in the second we describe the cross disciplinary events which lead to development the key mathematical techniques for producing embeddings and explain their operation, and finally we discuss practical considerations of these algorithm's applications. As the primary relevance of these methods within research is their practical application we take a broader contextual approach opposed to a purely mathematical one and focus on the developments and methods most influential in the sphere of application. We assume familiarity with core mathematics at an undergraduate level as prerequisite for our descriptions of the selected implementation methods but otherwise assume no prior knowledge of the subject.

July 8, 2021

TABLE OF CONTENTS

1 INTRODUCTION 2
1.1 What are word embeddings? 2
1.2 How are they created? 5
1.3 What are they for? 7
1.4 Beyond word embeddings 9
1.5 Report Content & Structure 9

2 PRELIMINARIES 11
2.1 What are words? 11
2.2 Tokens and Types 12
2.3 Semantic relatedness 14
2.4 Distributional analysis 14
2.5 Terminology & Definitions 14

3 COUNT MODELS 18
3.1 Cooccurrence Matrix Types 18
3.2 Smoothing 18
3.3 Dimensionality Reduction 18

4 PREDICT MODELS 19
4.1 `word2vec`: Skipgram 19
4.2 `word2vec`: Continuous bag of words (CBOW) 19
4.3 Comparison to count models 19

5 PRACTICAL CONSIDERATIONS 20
5.1 Pre-processing 20
5.2 Bias 20

CONCLUSION 21

REFERENCES 22

CHAPTER 1

INTRODUCTION

1.1 What are word embeddings?

Word embeddings are numerical representations of words which are derived from the observed patterns of word usage in a body of example text. These representations take the form of position vectors within a *Word Space*. The resulting *Word Space Model* was first described by Schütze (1993) in the terms:

> " [The Word Space Model] uses feature vectors to represent words, but the features cannot be interpreted on their own. Vector similarity is the only information present in Word Space: semantically related words are close, unrelated words are distant. "

Example 1.1 *(Two Dimensional Word Space)*. The word space in Figure 1.1 illustrates both components of the above definition: first that related words are closer in space than less related words (the words *apple* and *orange* are closer in space than *apple* and *bicycle*), and second that the individual axis of variation are not meaningful considered in isolation from one another (there is no trait like 'size' which the x or y value for a word is descriptive of).

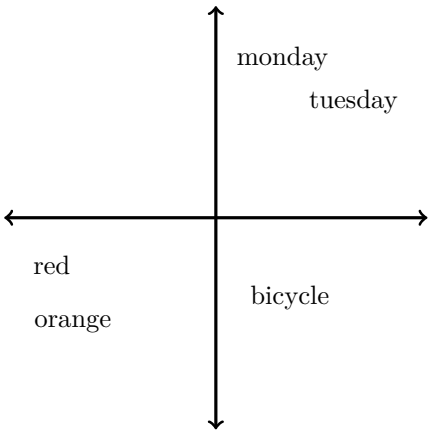

Figure 1.1: A two dimensional word space encoding semantic information about the five words; *red, orange, monday, tuesday & bicycle*

word	features	
	x	y
red	-1.55	-0.45
orange	-1.35	-1.0
monday	0.63	1.5
tuesday	1.3	1.1
bicycle	0.7	-0.75

Table 1.1: Feature vectors representing the words in Figure 1.1

More recent research has suggested that word spaces can in fact arrange word-representations with more order than Schütze's early description of these spaces accounted for. Notably, (Mikolov, Yih,

and Zweig 2013) demonstrated their *potential* for geometrically encoding semantic relationships between words (the direction in which one vector is distant from another can capture semantic information, not only the distance). Figure 1.2 illustrates the idea of geometrically encoded semantic information. Modern methodologies for assessing the quality of a word space consider the extent to which *multifaceted* linguistic information about the vocabulary (not just semantic, but; phonetic, morphological, and syntactic information about the encoded words) is recoverable by the embeddings (Yaghoobzadeh and Schütze 2016; B. Wang et al. 2019).

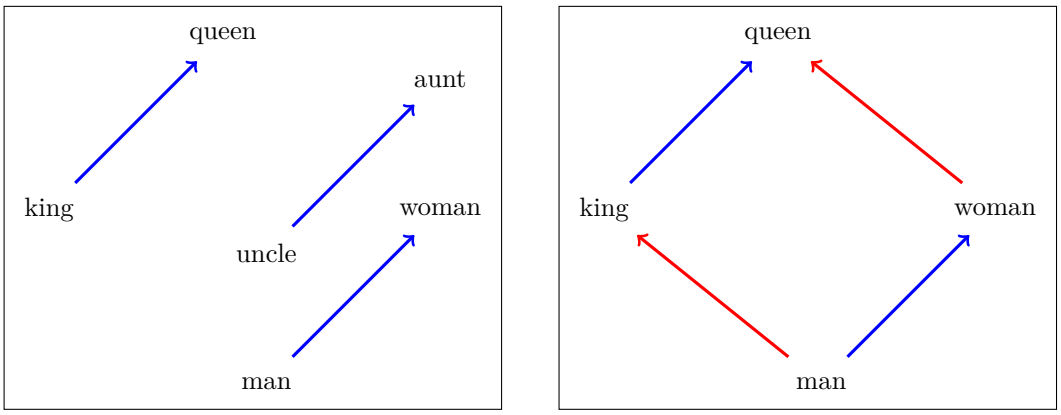

Figure 1.2: An idealised illustration of how semantic & syntactic information could be geometrically encoded by embeddings in a word space[1].

Typical word spaces, unlike the simplified form illustrated in example 1.1, are *high-dimensional*: it is common for the dimension of a word space to be in the order of tens, hundreds, or thousands. In a relatively low-dimensional pre-trained word embedding model[2] the word orange (which is represented in the above example by the vector $\langle -1.55, 0.35 \rangle$) is embedded as the 50-dimensional vector:

1. We use the term *idealised* here because although this figure is a reproduction of that which accompanies the famous $\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} = \overrightarrow{\text{queen}}$ example from (Mikolov, Yih, and Zweig 2013), more recent investigations (Linzen 2016; Rogers, Drozd, and Li 2017) have concluded that these analogical relationships are not the robust features of word space geometry which these examples suggest. Nevertheless, investigations into the geometry of word space have found some notable structure (Liu et al. 2018).

2. Model provided by Pennington, Socher, and Manning (2014). Available for download at (GloVe project website)

$$\text{orange} \mapsto$$

$$\begin{pmatrix} -0.42783, & 0.43089, & -0.50351, & 0.5776, & 0.097786, & 0.2608, & -0.68767, & -0.31936, & -0.25337, & -0.37255, \\ -0.045907, & -0.53688, & 0.97511, & -0.44595, & -0.50414, & -0.086751, & -1.0645, & 0.36625, & -0.52428, & -1.3413, \\ -0.2391, & -0.58808, & 0.56378, & -0.062501, & -1.7429, & -0.88077, & -0.27933, & 1.4705, & 0.50436, & -0.69174, \\ 2.0018, & 0.26663, & -0.85679, & -0.18893, & -0.021125, & -0.055118, & -0.50337, & -0.67157, & 0.55502, & -0.8009, \\ 0.10695, & 0.1459, & -0.55588, & -0.64971, & 0.22046, & 0.67415, & -0.45119, & -1.1462, & 0.16348, & -0.62946 \end{pmatrix} \tag{1.0}$$

High-dimensional vectors like (1.0) cannot be plotted on an x, y plane without loss of information, so we can't visualise points in a typical word space with the same approach as we used in example 1.1. An alternate method for visualising points in word space which still allows us to judge the similarity of vectors by their appearance, and is applicable higher dimensional vectors, is to map the magnitude of vector components onto a smooth colour gradient and then to compare the resulting colour-vectors, rather than the original numerical forms.

Example 1.2 *(50 dimensional Word Space).* Figure 1.3 applies the principle described above to visualise word vector outputs from a real word embedding model.

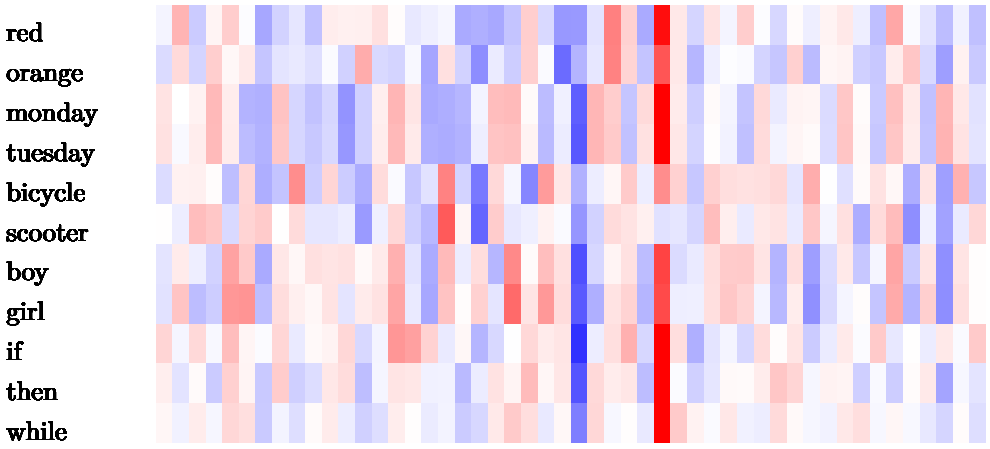

Figure 1.3: A selection of embeddings from the same pre-trained 50-dimensional *GloVe* word space model which (1.0) was taken from.

1.2 How are they created?

Word embeddings are the output of *word embedding algorithms*: Processes which take as their input large body of example language-usage and *automatically produce* (i.e., without need for directions given by the practitioner regarding the processing of the data) representations of the observed words *which encode a means of quantifying the semantic-relatedness of those words*.

> "What makes the word-space model unique in comparison with other geometrical models of meaning is that the space is constructed with no human intervention, and with no a priori knowledge or constraints about meaning similarities. In the word-space model, the similarities between words are automatically extracted from language data by looking at empirical evidence of real language use. "
>
> Sahlgren (2006)

How exactly is this information automatically extracted from a corpora of "real language usage" though? Sahlgren continues:

> "As data, the word-space model uses statistics about the distributional properties of words. The idea is to put words with similar distributional properties in similar regions of the word space, so that proximity reflects distributional similarity. The fundamental idea behind the use of distributional information is the so-called distributional hypothesis. "

Definition 1 – *The Distributional Hypothesis*

There are many variations of the statement of the distributional hypothesis; Sahlgren opts for the more general phrasing *"words with similar distributional properties have similar meanings"*, but with reference to its application in creating word spaces cites the more concrete *"words with similar meanings will occur with similar neighbours if enough text material is available"* of Schutze and Pedersen (1995) and the concise *"words which are similar in meaning occur in similar contexts"* of Rubenstein and Goodenough (1965). Firth (1957) notably captured the essence of this hypothesis with the much quoted statement *"You will know a word by the company it keeps"*.

Example 1.3 *(The distributional hypothesis).* In the figure from example 1.2, the vector representations of the words "Monday" and "Tuesday" are almost identical to one another. These representations were generated by analysing the contexts each word occurred in, so implied by this result is that the words "Monday" and "Tuesday" occur in very similar contexts.

Consider these partial quotations[a]:

"I'd be quite happy if I spent from Saturday night until _____ morning…" (1.1)

"It was a _____ and they walked on a tightrope to the sun." (1.2)

"I will love you at 4:15 pm next _____." (1.3)

"…the kind that blindside you at 4 pm on some idle _____." (1.4)

In each of these contexts it is clear that the *type* of word which is missing is likely to be a day of the week but it is difficult to guess with confidence any more specifically; the words "Monday" and "Tuesday" seem to be almost equally probable guesses -*their similarity of meaning is reflected in the similarity of their contexts.*

a. These quotations were found by searching for quotes containing the word "Monday" (in the case of (1.1) and (1.2)) or "Tuesday" (in the case of (**??**) and (1.4)) on the goodreads website

This approach to linguistics (distributional linguistics) was widespread in the 1950s (Jurafsky and Martin 2021), with key contributions to the theory made by Joos (1950), Harris (1954), and Firth (1957), and is related Wittgenstein's (1953) ideas about 'family resemblance' from the same period (Turney and Pantel 2010). The question of how word embeddings are created then can instead be phrased *How exactly is the distributional hypothesis employed by these (word embedding) algorithms?*. Below we paraphrase the descriptions of seven categories identified by B. Wang et al. (2019) into which word embedding methods fall:

Summary: Word Embedding Method Categorisation

***Co-occurrence Matrix** In this context we refer to either *word-document* or the *word-word* co-occurence matrices, though other formulations exist (Turney and Pantel (2010) provides a good overview of approaches). The co-occurence matrix X is such that the (i, j) entry, X_{ij} counts the number of times the word i occurred in the context j, where j could be the context of a document or the context of a near-by word.

***Continuous-Bag-of-Words and skip-gram** Two *iteration-based* methods proposed in the `word2vec` paper (Mikolov, K. Chen, et al. 2013) and widely used thereafter. The first model (CBOW) predicts the center word from its surrounding context and the second (SG) predicts the surrounding context words given a center word.

Neural Network Language Model The NNLM (Bengio et al. 2003) *jointly learns a word vector representation and a statistical language model*, and is *very computationally complex*.

FastText This method exists to address issues with other approaches which produce poor estimations for rarely used words. *FastText uses subword information explicitly so embedding for rare words can still be represented well*. It is based on the skip-gram model, where each word is represented as a bag of character

***N*-gram Model** n-grams are multi-word elements with n parts (for example "a man" is a 2-gram). n-gram models are methods which operate on n-grams in addition to, or in place of words. Zhao et al. (2017) incorporates the n-gram model in various baseline embedding models such as `word2vec`, GloVe, PPMI, and SVD.

Dictionary Model Dictionary models learn word representations from dictionary entries as well as large unlabelled corpus, both sources of information are incorporated into one representation. (Tissier, Gravier, and Habrard 2017) implements this method as `dict2vec`.

Deep Contextualised Model (Peters et al. 2018) introduced a model for *deep contextualised word representations which represent complex characteristics of words and word usage across different linguistic contexts*. An Embeddings from Language Models (ElMo) representation is generated with a function that takes an entire sentence as the input.

In this report we describe algorithms for generating embeddings from the *co-occurrence matrix* category (which we refer to as the "count models"), and the *continuous-bag-of-words / skip-gram* category (which we will refer to as the "predict models").

1.3 What are they for?

> "The task of representing words and documents is part and parcel of most, if not all, Natural Language Processing (NLP) tasks. In general, it has been found to be useful to represent them as vectors, which have an appealing, intuitive interpretation, can be the subject of useful operations (e.g. addition, subtraction, distance measures, etc) and lend themselves well to be used in many Machine Learning (ML) algorithms and strategies. "
>
> Almeida and Xexéo (2019)

Although there are a variety of different word embedding algorithms, each being suited to a different niche of application, they are all united by the following traits:

Summary: Shared Characteristics of Word Embedding Algorithms

Semantic Comprehension They provide a means of quantitatively assessing the semantic-relatedness of words;

Convenient Form The semantic information is encoded by vector representations of vocabulary words;

Automatic Generation Process The semantic information is *"automatically extracted from language data by looking at empirical evidence of real language use"*.

The *semantic comprehension* quality makes them appropriate for many tasks in natural language processing, and their *convenient form* (like Almeida and Xexéo notes above) in particular makes them work well as inputs to machine learning pipelines -Schütze (1993) notes: *"Neural networks perform best when similarity of targets corresponds to similarity of inputs; traditional symbolic representations do not have this property"*. Finally, their *automatic generation process* makes them preferable to alternative means of assessing word-relatedness which depend on extensive manual knowledge-input, like the structured lexical database: `wordnet` (Miller et al. 1990).

Example 1.4 *(Word Clustering Using Embeddings)***.** Figure 1.4 shows the result of using a point-clustering algorithm on the embeddings from example 1.2 to identify groups of related words. The resulting clusters, identified entirely computationally, correspond with the groupings an English speaker would naturally select given the same word choices.

Turney and Pantel (2010) include a thorough discussion of applications of *Vector Space Models* (the class of embedding algorithms referred to as *count models* in this report), listing the following examples of uses: word clustering, word classification, automatic thesaurus generation, word sense disambiguation, context sensitive spelling correction, semantic role labelling, query expansion, textual advertising & information extraction. example 1.4 illustrates the use of the embeddings from example 1.2 to perform the first task from this list of examples:

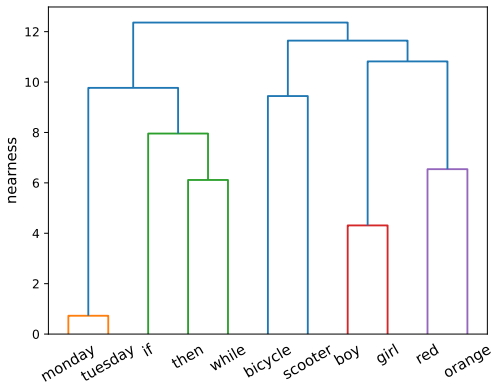

Figure 1.4: Word clusters identified by analysis of word vectors featured in Figure 1.3

word clustering.

1.4 Beyond word embeddings

Word embedding algorithms extract semantic information about language without any prior knowledge of underlying linguistic rules or significance. To these algorithms, language is just lists of discrete elements with underlying patterns to be identified. Due to this generality, word embedding algorithms can be used to encode the semantics of other types of ordered data. Many novel applications of the embedding process to capture semantic associations outside of the written language domain have been explored. Below we list some notable embedding applications we have come across in researching for this report:

Summary: Novel Applications of the Embedding Process

Graph Embeddings Grohe (2020) presents an overview of techniques which generalise the word embedding process to embed structured data -graphs (possibly labelled or weighted), or more generally relational structures, nodes of a large graph, or tuples appearing in a relational structure.

Recommendation Systems Grbovic and Cheng (2018) demonstrate the use of user and listing embeddings to improve real-time personalisation and similar listing recommendations at Airbnb, and J. Wang et al. (2018) employ item-graph embeddings generated from users' behaviour history to improve recommendation systems at Alibaba.

Music Embeddings Chuan, Agres, and Herremans (2018) model functional chord relationships and harmonic associations by embedding slices of music from a large corpus music across eight genres.

Customer Lifetime Value Prediction Chamberlain et al. (2017) propose a method to generate embeddings of customers which provide daily estimates of the future value of every customer, employed by the ASOS fashion retailer.

1.5 Report Content & Structure

In chapter 2 we discuss key linguistic concepts informing embedding-algorithm design: how we define 'word', what is meant by 'semantic relatedness' & core concepts from distributional semantics, and finally present a formal description of data these algorithms operate on, to be used

in the following chapters. In chapter 3 we discuss the generation of word embeddings using the vector-space-model methods following the framework laid out by Turney and Pantel (2010). In chapter 4 we discuss the two embedding algorithms packaged by the `word2vec` software, described by Mikolov, Sutskever, et al. (2013), and compare these methods to those already discussed. In the last chapter, chapter 5 we address practical considerations for the application of these methods: text preprocessing, model tuning & consideration of bias, before finally summarising observations made through the course of this research in the conclusion.

CHAPTER 2
PRELIMINARIES

2.1 What are words?

> "To many people the most obvious feature of a language is that it consists of words "
>
> Halliday and Teubert (2004)

> "There have been many definitions of the word, and if any had been successful I would have given it long ago, instead of dodging the issue until now "
>
> Matthews (1991)

The 'Word' workshop, held at the international research centre for linguistic typology 12-16 August 2000 consisted of 16 presentations discussing the answer to this question. Ten of those were published in the book *Word : a cross-linguistic typology*. The difficulty of the question is summarised in the book's conclusion as follows:

> "The problem of the word has worried general linguists for the best part of a century. In investigating any language, one can hardly fail to make divisions between units that are word-like by at least some of the relevant criteria. But these units may be simple or both long and complex, and other criteria may establish other units. It is therefore natural to ask if 'words' are universal, or what properties might define them as such. "
>
> P.H. Matthews, (Dixon 2002)

Some definitions of 'word' (like Bloomfield's (1926) *'A minimum free form is a word'*) are sufficient but not necessary, wheras others (like Lyons's (1968) *'a word may be defined as the unit of a particular meaning with a particular complex of sounds capable of a particular grammatical employment'*) is a potentially necessary but certainly not sufficient criterea. Is it possible to present a watertight definition? Our word-intuition is not watertight either, consider *so called* filler-words like "uh" or "um", or compound forms like "rock 'n' roll" and "New York" always used as complete units. These are examples of language elements that are sometimes treated as words and sometimes not. The boundaries we agree on now conform to ever changing conventions; the word "tomorrow" is listed in Johnson's 1755 dictionary of the English language as two distinct components "to", and "morrow" despite being indesputeably a word *to-day*. Halliday and Teubert (2004) asks "How do we decide about sequences like lunchtime (lunch-time, lunch time), dinner-time, breakfast time? How many words in isn't, pick-me-up, CD?". An understanding of 'word' formed within the English

language cannot blindly be applied beyond that scope either, Dixon 2002 notes "The idea of 'word' as a unit of language was developed for the familiar languages of Europe". Languages which are written without spaces, (like Chinese, Japanese, and Thai) present one particular example of the challenges in applying a local conception of 'word' beyond this scope.

Example 2.1. Chinese Sentence Segmentation X. Chen et al. (2017) gives the following example of the difficulty in agreeing on 'word' boundaries in Chinese: The sentence "Yao Ming reaches the finals" (姚明进入总决赛) May be divided up to yield either three words or five using the two segmentations methods; 'Chinese Treebank' segmentation, or 'Peking University' segmentation, respectively:

姚明 进入 总决赛

YaoMing reaches finals

姚 明 进入 总 决赛

Yao Ming reaches overall finals

Different languages present different difficulties -In the case of Arabic, expressions formed by adding consecutive suffixes to a single root word (like the sequence *establish* → *establishment* → *establishmentarianism* in English) are a more essential part of the language, such that complete sentences can be expressed as unbroken 'word' units. As a result the 'word' unit (identified by unbroken character sequences occuring between spaces) in Arabic will represent larger more complex chunks of meaning than same framework would identify for comparable sentences in English. What is the 'correct' amount of information that our word-units should signify?

2.2 Tokens and Types

"The limits of my language means the limits of my world"

Wittgenstein (1922)

Contains 11 word-tokens, and 7 word-classes (assuming the sentence is tokenized by identifying word boundaries at both ends and each blank space). **Lexicology** is the study of words, and **semantics** is the study of meaning. The 'embedding' in 'word embeddings' derives from the mathematical sense 'to embed in a space' (for example, to embed a graph in a coordinate space).

Word embedding algorithms are methods for embedding *lexicological objects* in *semantic space*, i.e., words a space where position relates to meaning somehow). This is achieved by statistical analysis of large quantities of written language (by extraction of word-tokens).

The conventions of written language, called **orthography**, make word-tokens an imperfect representation of 'words' in the lexicological sense. In particular, one problem is that distinct words do not always have distinct realisations in written language. Orthographic forms with this property are called **polysemus** (poly as in *many* and semus as in *meaning*, from the same root as semantics). Alternatively, the distinct words sharing this same form are referred to as **homonyms** (having the same name). The words *bank* as in money holding organisation and *bank* as in riverside are often used to illustrate this property. Indeed, some words, like *frequent* as in often, and *frequent* as in attend are examples of words distinct in spoken language despite being identical in written language; such words are said to be **homographs** (same orthography, different names).

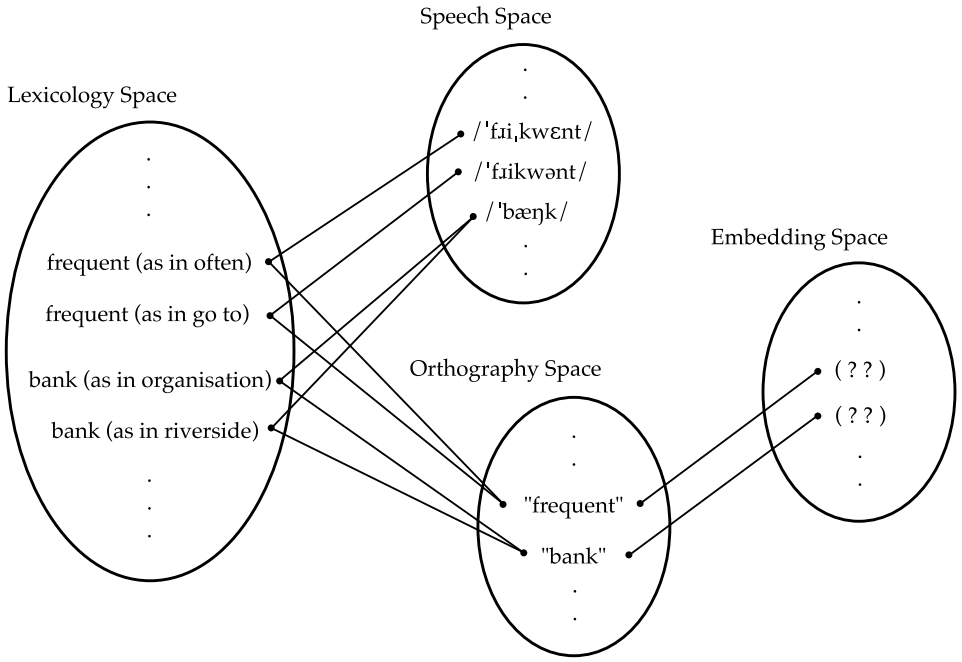

Figure 2.1: The *problem of polysemy* illustrated

Embedding algorithms which generate their representations without taking this *problem of polysemy* into account produce representations which conflate the multiple meanings of a word-form into a single point in the embedding space.

2.3 Semantic relatedness

2.4 Distributional analysis

2.5 Terminology & Definitions

We begin by defining terms already encountered in mathematical terms.

Definition 2 – *Orthography*

An orthography is a finite set of symbols.

Example 2.2. The sets:

$$\mathscr{O}_L = \{c \mid c \text{ is a symbol that can be copied and pasted the pdf of this report}\} \tag{2.1}$$

$$\mathscr{O}_P = \{c \mid c \text{ is the inscription of a regular polygon with eight sides or less}\} \tag{2.2}$$

$$\mathscr{O}_K = \{c \mid c \text{ is a typable character marked on Figure 2.2}\} \tag{2.3}$$

are all examples of orthographies.

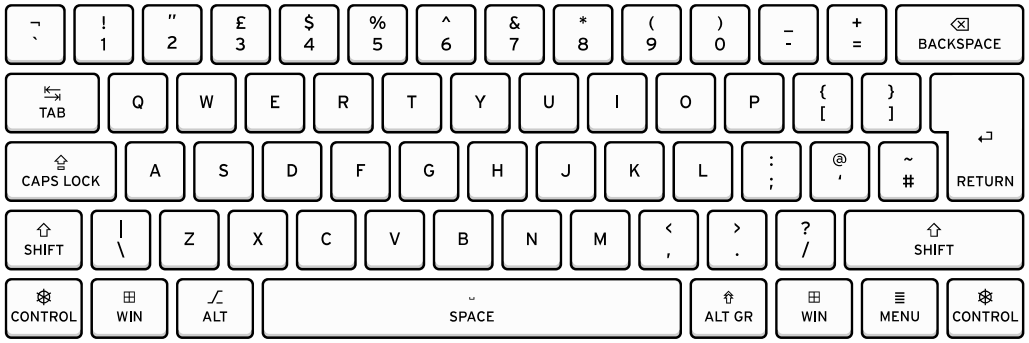

Figure 2.2: A standard uk keyboard layout

Note, the term 'orthography' in linguistics refers to the conventions of written language in general and is distinct from the use of 'an orthography' we use here.

Definition 3 – *Document*

A document d is a finite sequence $c_0, c_1, \ldots, c_n$ $(c \in \mathscr{O})$, where $\mathscr{O}$ is an orthography.

Example 2.3 *(This Report).* The sequence of symbols obtained by copy pasting this entire report into a text editor (ordered by starting with the cursor at the top of the file and moving the cursor with the arrow keys one position to the right at a time -including newlines) is a document using the orthography 2.1.

Example 2.4 *(Shapes Document).* The sequence of shapes: △△□△△⬠ is a document using the orthograpy 2.2.

Example 2.5 *(Plato Documents).* Any of the works of Plato available at the Project Gutenberg website, copied and pasted from their plain text formats and using the same procedure described above, are documents using the orthography 2.3.

Definition 4 – *Word-Token*

A word token w has the same attributes as a document; it is a finite sequence $c_0, c_1, \ldots, c_n$ $(c \in \mathscr{O})$, where $\mathscr{O}$ is an orthography. We only refer to sequences of symbols as word tokens when they are the output of a tokenizer, however.

Definition 5 – *Tokenizer*

A tokenizer T is a function that maps a document to a sequence of word tokens; $T(d) = w_1, w_2, \ldots, w_n$. The output is refered to as the 'tokenized' form of the document. We use the shorthand W_d to refer to the tokenized document $T(d)$ when it is clear which tokenization has been used.[1].

Example 2.6 *(This Report Tokenized).* The document from example 2.3 if tokenized by treating everything occuring between two spaces (or between a space and the start or end of a line) yields a sequence of word tokens beginning *"THE" "UNIVERSITY" "OF"*.

Example 2.7 *(Shapes Document Tokenized).* The document from example 2.4 if tokenized

using the function:

$T(d)$: `Group repeated symbols in the sequence` d `into word-tokens (in the order of occurrence).`

produces the output: "△ △", "□", "△ △", "⬠"

Example 2.8 *(Simple Text Tokenizer).* Given a document d using the orthography $\mathscr{O}_K$ from example 2.2 apply the following algorithm:

Definition 6 – *Corpus*

A corpus C is a finite set of documents $\{d_1, d_2, \ldots, d_n\}$.

Example 2.9 *(Plato Corpus).* All the of Plato works referred to in example 2.5, if converted to documents using the same procedure described there, considered together form a corpus of documents. We will refer to this corpus as C_p.

Definition 7 – *Vocabulary*

The vocabulary of a tokenized document W_d is the set of unique word tokens in the tokenization of that document; $V(W_d) = \{w \mid w \in \text{ the sequence } W_d\}$. The vocabulary of a corpus is the union of the vocabularies of each document in that corpus; $V(C) = V(d_1) \cup V(d_2) \cup \cdots \cup V(d_n)$.

Example 2.10 *(Shapes Document Vocabulary).* Using the same tokenization as before, the document from example 2.4 has the vocabulary:

$$V(\text{"△ △", "□", "△ △", "⬠"})$$
$$= \{\text{"△ △", "□", "⬠"}\}$$

Definition 8 – # *Operator*

The # symbol is used to denote the number of elements of the object it precedes. Before sequences it counts the number of elements in the sequence: $\#W_d = n$ (for $W_d = w_1, \ldots, w_n$) [2]. Before sets it is a shorthand for the size of that set: $\#V(d) = |V(d)| =$ the number of words in the vocabulary $V(d)$. Before a corpus it counts the number of documents in that corpus.

Example 2.11 *(Shapes Document Sizes).* The word count of the tokenized shapes document

$$W_d = \text{“}\triangle\ \triangle\text{”, “}\square\text{”, “}\triangle\ \triangle\text{”, “}\pentagon\text{”}$$

is $\#W_d = 4$ and the vocabulary size is $\#V(W_d) = 3$.

Example 2.12 *(Plato Document Sizes).* Word token counts and vocabulary sizes for a selection Plato documents from 2.5, tokenized using the simple method: [a] have

a. the code used to create these examples and others examples on the same corpus is available in the appendix

Example 2.13 *(Plato Corpus Size).* The complete Plato corpus (2.9) consists of 25 documents and, each tokenized the same way as before, has the word count $\sum_{d \in C_P} \#W_d = 670,936$ and the vocabulary size $\#V(C_P) = 19,432$.

CHAPTER 3

COUNT MODELS

3.1 Cooccurrence Matrix Types

3.2 Smoothing

3.3 Dimensionality Reduction

CHAPTER 4

PREDICT MODELS

4.1 `word2vec`: Skipgram

4.2 `word2vec`: Continuous bag of words (CBOW)

4.3 Comparison to count models

CHAPTER 5

PRACTICAL CONSIDERATIONS

5.1 Pre-processing

5.2 Bias

CONCLUSION

BIBLIOGRAPHY

Almeida, Felipe and Geraldo Xexéo (2019). "Word Embeddings: a Survey". In: *CoRR*.

Bengio, Yoshua et al. (Mar. 2003). "A Neural Probabilistic Language Model". In: *J. Mach. Learn. Res.* 3.null, pp. 1137–1155. ISSN: 1532-4435.

Bloomfield, Leonard (1926). "A Set of Postulates for the Science of Language". In: *Language* 2.3, pp. 153–164. ISSN: 00978507, 15350665.

Chamberlain, Benjamin Paul et al. (Aug. 2017). "Customer Lifetime Value Prediction Using Embeddings". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, nil. DOI: 10.1145/3097983.3098123.

Chen, Xinchi et al. (Apr. 2017). "Adversarial Multi-Criteria Learning for Chinese Word Segmentation". In: *arXiv*.

Chuan, Ching-Hua, Kat Agres, and Dorien Herremans (Dec. 2018). "From context to concept: exploring semantic relationships in music with word2vec". In: *Neural Computing and Applications* 32.4, pp. 1023–1036. ISSN: 1433-3058. DOI: 10.1007/s00521-018-3923-1.

Dixon, Robert (2002). *Word : a cross-linguistic typology*. Cambridge: Cambridge University Press. ISBN: 052104605X.

Firth, J. (1957). "A Synopsis of Linguistic Theory, 1930-1955". In: Studies in Linguistic Analysis, Philological. Longman.

Grbovic, Mihajlo and Haibin Cheng (2018). "Real-Time Personalization Using Embeddings for Search Ranking at Airbnb". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*. KDD '18. London, United Kingdom: Association for Computing Machinery, pp. 311–320. ISBN: 9781450355520. DOI: 10.1145/3219819.3219885.

Grohe, Martin (2020). "Word2vec, Node2vec, Graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data". In: *CoRR*.

Halliday, M.A.K. and W. Teubert (2004). *Lexicology and Corpus Linguistics*. Open Linguistics. Bloomsbury Academic. ISBN: 9780826448620.

Harris, Zellig S. (Aug. 1954). "Distributional Structure". In: *WORD* 10.2-3, pp. 146–162. ISSN: 2373-5112. DOI: 10.1080/00437956.1954.11659520.

Joos, Martin (Nov. 1950). "Description of Language Design". In: *The Journal of the Acoustical Society of America* 22.6, pp. 701–707. ISSN: 0001-4966. DOI: 10.1121/1.1906674.

Jurafsky, Dan and James H. Martin (2021). "Speech and language processing : an introduction to natural language processing, computational linguistics, and speech recognition". Draft of third edition of the book.

Linzen, Tal (Aug. 2016). "Issues in evaluating semantic spaces using word analogies". In: *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. Berlin, Germany: Association for Computational Linguistics, pp. 13–18. DOI: 10.18653/v1/W16-2503.

Liu, Shusen et al. (Jan. 2018). "Visual Exploration of Semantic Relationships in Neural Word Embeddings". In: *IEEE Transactions on Visualization and Computer Graphics* 24.1, pp. 553–562. ISSN: 2160-9306. DOI: 10.1109/tvcg.2017.2745141.

Lyons, J. (1968). *Introduction to Theoretical Linguistics*. Introduction to Theoretical Linguistics. Cambridge University Press. ISBN: 9780521095105.

Matthews, P. H (1991). *Morphology*. Cambridge England New York: Cambridge University Press. ISBN: 9780521410434.

Mikolov, Tomas, Kai Chen, et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *CoRR*.

Mikolov, Tomas, Ilya Sutskever, et al. (2013). "Distributed Representations of Words and Phrases and Their Compositionality". In: *CoRR*.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (June 2013). "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*. Atlanta, Georgia: Association for Computational Linguistics, pp. 746–751.

Miller, George A. et al. (1990). "Introduction to WordNet: An On-line Lexical Database*". In: *International Journal of Lexicography* 3.4, pp. 235–244. ISSN: 1477-4577. DOI: 10.1093/ijl/3.4.235.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

Peters, Matthew E. et al. (2018). "Deep Contextualized Word Representations". In: *CoRR*.

Rogers, Anna, Aleksandr Drozd, and Bofang Li (Aug. 2017). "The (too Many) Problems of Analogical Reasoning with Word Vectors". In: *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 135–148. DOI: 10.18653/v1/S17-1017.

Rubenstein, Herbert and John B. Goodenough (Oct. 1965). "Contextual Correlates of Synonymy". In: *Commun. ACM* 8.10, pp. 627–633. ISSN: 0001-0782. DOI: 10.1145/365628.365657.

Sahlgren, Magnus (2006). *The word-space model : using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Stockholm: Dep. of Linguistics, Stockholm Univ. ISBN: 9171552812.

Schutze, Hinrich and Jan O. Pedersen (1995). "Information Retrieval Based on Word Senses". In: *In Proceedings of the 4th Annual Symposium on Document Analysis and Information Retrieval*, pp. 161–175.

Schütze, Hinrich (1993). "Word Space". In: *Advances in Neural Information Processing Systems*. Ed. by S. Hanson, J. Cowan, and C. Giles. Vol. 5. Morgan-Kaufmann.

Tissier, Julien, Christophe Gravier, and Amaury Habrard (Sept. 2017). "Dict2vec : Learning Word Embeddings using Lexical Dictionaries". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 254–263. DOI: 10.18653/v1/D17-1024.

Turney, P. D. and P. Pantel (2010). "From Frequency To Meaning: Vector Space Models of Semantics". In: *Journal of Artificial Intelligence Research* 37.nil, pp. 141–188. DOI: 10.1613/jair.2934.

Wang, Bin et al. (2019). "Evaluating word embedding models: methods and experimental results". In: *APSIPA Transactions on Signal and Information Processing* 8, e19. DOI: 10.1017/ATSIP.2019.12.

Wang, Jizhe et al. (2018). "Billion-Scale Commodity Embedding for E-Commerce Recommendation in Alibaba". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*. KDD '18. London, United Kingdom: Association for Computing Machinery, pp. 839–848. ISBN: 9781450355520. DOI: 10.1145/3219819.3219869.

Wittgenstein, Ludwig (1922). *Tractatus logico-philosophicus*. London : Routledge Kegan Paul.

— (1953). *Philosophical investigations*. Oxford: B. Blackwell. ISBN: 0631146709.

Yaghoobzadeh, Yadollah and Hinrich Schütze (Aug. 2016). "Intrinsic Subspace Evaluation of Word Embedding Representations". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 236–246. DOI: 10.18653/v1/P16-1023.

Zhao, Zhe et al. (Sept. 2017). "Ngram2vec: Learning Improved Word Representations from Ngram Co-occurrence Statistics". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 244–253. DOI: 10.18653/v1/D17-1023.