universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Computational Linguistics Approaches to Historical Emotion Analysis: Evaluating Word Embeddings, Induction Algorithms, and Lexicon Choice.

verfasst von | submitted by

Jona Marie Hassenbach BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Arts (MA)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 647
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Digital Humanities
Betreut von | Supervisor:	Ass.-Prof. Mag. Mag. Dr. Andreas Baumann

Table of Contents

1. Introduction

In her 1871 novel Middlemarch, George Elliot explains the lack of social relations between two connected families, by stating *"for there were nice distinctions of rank in Middlemarch"*[1]. What is the modern reader to make of this sentence? The word 'nice' has undergone significant semantic change since the 1870s. To a modern reader the adjective is synonymous to 'pleasant' or 'lovely', however, to a contemporary of George Elliot, 'nice' had a more negative connotation, coming closer in meaning to 'refined' or 'dainty' (Wijaya and Yeniterzi, 2011). Herein lies the key to the correct understanding of the passage mentioned above: the "*distinctions of rank*" are not 'pleasant', they are 'refined': small differences in social standing, hence, provide an adequate reason to cease social interaction, even if two families are related. This example illustrates the importance of accounting for semantic change when interpreting historical documents.

In 2019, a study conducted by Hellrich et al. provided one possible solution to this exact problem: they developed a model for *historical emotion analysis* that computes a historical *Valence, Arousal,* and *Dominance* (VAD) score for every word in a given corpus. While the concepts of Valence, Arousal, and Dominance do not encapsulate the entirety of a word's meaning, emotions represent a crucial part of human nature, and *historical emotion analysis* can provide valuable insights to further our understanding of historical texts within the context of their time. The initial results of Hellrich et al. were promising, however, room for future improvement remained.

This master's thesis investigates the optimal setup of Hellrich et al.'s model. The initial model consists of three independent parts: a contemporary VAD lexicon, a historical word embedding, and an induction algorithm combining the two. This concept is built upon in three ways. First, I introduce three additional word embeddings – Positive Pointwise Mutual Information (PPMI; Church and Hanks, 1990), Continuous Bag of Words (CBOW; Mikolov et al., 2013), and FastText (Bojanowski et al. 2017) – and one additional induction algorithm, Linear Regression (Li et al., 2017). I then compare the performance of these model combinations with the original ones. Second, I replace the Warriner VAD

[1] George Elliot. 1871. Middlemarch. Page 231. Penguin Classics 1994 (reprinted 2003).

lexicon (Warriner et al., 2013) employed by Hellrich et al. with the NRC-VAD lexicon (Mohammad, 2020) and reevaluate all resulting model configurations. Third, I investigate optimal lexicon size across both lexica. All implementation is done using Python and the corresponding code can be found on my GitHub[2].

Results are as follows: The highest performance is achieved by the model combining the Skip-Gram with Negative Sampling (SGNS; Mikolov et al., 2013) embedding originally used by Hellrich et al., the Linear Regression induction algorithm, and the largest version of the Warriner VAD lexicon. Models relying on the FastText embedding yield similarly high results, both when using larger versions of Warriner and NRC-VAD. Notably, performance improves with increasing lexicon size, suggesting that there is no such thing as "too much contemporary influence." In addition, even the improved models underperform compared to contemporary methods of *emotion analysis*. Though there are possible practical reasons for these contradictory results, my findings may indicate that this model offers limited utility – at least for the historical period under investigation.

The general structure of this master's thesis is organized as follows: In the first part, I provide a brief introduction to the relevant topics. In the second part, I discuss the data and algorithms utilized in my experiments. In the third part, I describe the setup and results of all three experiments. In the fourth part, I discuss my findings and reflect on possible explanations for the model's limitations. In the final part, I draw a brief conclusion.

This work has been created with the support of generative AI, namely ChatGPT4 and DeepSeek. While coding, it was employed together with other resources that have been provided by previous research. To make this information accessible, a 'statement of origin' can be found at the beginning of every Python file.

[2] https://github.com/NotJona/HistoricalEmotionAnalysis

2. State of the Art

This chapter provides a brief introduction to the relevant topics. Starting with the term of *emotion analysis*, I examine its fundamental concepts, present the practical implementations that are relevant to this work, and provide a number of modern use-case examples. Next, I move to a brief introduction of word embeddings and their application in diachronic linguistics. Lastly, I present the most recent research conducted in the field of *historical emotion analysis*, the area that combines the concepts and tools of *emotion analysis* with word embeddings.

2.1. Emotion Analysis

The term *emotion recognition* refers to the task of identifying or predicting human emotions communicated through facial expressions, posture, gestures, tone, speech, or writing (Younis et al., 2024). The analysis is performed either by a human interpreter, through human-computer cooperation, or entirely by a computer – in which case it is referred to as *emotion analysis* and predominantly focuses on textual data. For its classification of emotions, *emotion analysis* borrows its theoretical framework from psychology. Depending on the specific goal or research question, two types of emotion classification models are commonly used:

On the one side, *discrete* or *categorical models*, define emotions as a set of discrete states: each emotional state arises from an independent psychological process and can be categorized clearly by its emotional expression. Hence, there are strict borders separating the different emotional states. This theory of emotions originates from evolutionary biology and is based on the assumption that a certain subset of emotions is universal to human nature. It was first proposed in 1872 by Charles Darwin, who conducted a study on facial expressions in humans and animals, and observed that similar circumstances cause similar behavior and expressions of emotion. He concluded that emotions are universal and a product of evolution: as conveyors of information, they affect an individual's chances of survival. Darwin's theory of emotions gave rise to different models for their identification and categorization. Ekman's model of the *Six*

Basic Emotions (1992) and Plutchik's *Wheel of Emotions* (2001) are two popular representatives. Ekman's model identifies the *Six Basic Emotions* as anger, surprise, disgust, enjoyment, fear, and sadness. In his studies on facial expressions, Ekman found that each base emotion exhibits a particular and recognizable set of characteristics independent of personal, societal, or cultural factors. Therefore, he considers them to be universal. This quality enables the basic emotions to be communicated verbally as well as nonverbally, and hence across different languages. In contrast, Plutchik's *Wheel of Emotions* (Figure 1a) provides a more nuanced classification model: it consists of eight primary emotions conceptualized as four opposing pairs: joy–sadness, anger–fear, trust–disgust, and anticipation–surprise. In addition, each primary emotion has both a higher- and lower-intensity secondary emotion attached to it. The emotions are arranged in three concentric circles, where the innermost circle consists of the high-intensity emotions, followed by the primary emotions in the middle circle and the low-intensity emotions on the outside. Additionally, similar to a color wheel, a third set of emotions can be obtained through the composition of primary emotions. The primary emotion joy, for example, exists on a scale between ecstasy as its high-intensity counterpart and serenity as its low-intensity equivalent. When combining joy with its two neighboring primary emotions, anticipation and trust, one obtains optimism and love respectively. While Plutchik's *Wheel of Emotions* considers emotions as discrete states, it functions as a link between discrete and dimensional models due to its intensity-based arrangement.

On the other side, *dimensional* or *continuous models*, represent a second type of emotion classification models. They are based on observations from clinical psychology, which suggest that emotions are not felt as discrete states, but rather as overlapping and ambiguous experiences. This theory of emotions proposes that all emotions arise from a limited number of fundamental psychological processes. Hence, dimensional models conceptualize emotions as either points or regions inside a vector space, whose dimensions are defined by these fundamental processes. This is especially advantageous, because it allows researchers to characterize emotions by their continuous relations to each other. Prominent representatives of dimensional models are the *Circumplex Model* by Posner, Russell, and Peterson (2005) (though its first version was published by Russell in 1980) and the *VAD Model* by Bradley and Lang (1994). The

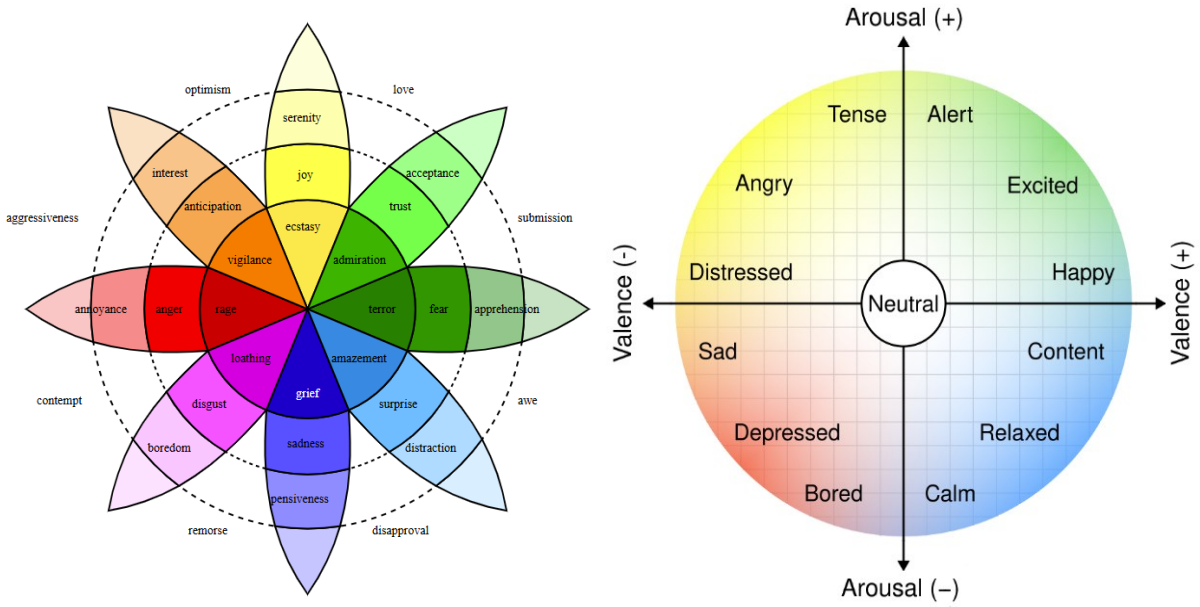


Figure 1a (left): Plutchik's Wheel of Emotions. Figure 1b (right): The Circumplex Model. (Anmol, 2023)

Circumplex Model consists of a two-dimensional vector space, the first dimension being defined by *Valence* (how positive the person expressing the emotion feels) and the second dimension being defined by *Arousal* (how activated the person feels). Thus, each emotion can be represented as a linear combination of these two concepts. Let us once again take the emotion joy as an example: in this model, joy (here referred to by 'happy') is represented as an emotional state with high Valence or pleasure and moderate Arousal (see Figure 1b). If the value of Arousal is increased or decreased while Valence stays unchanged, we move to excitement or content, respectively. If the value of Valence is reversed while Arousal stays low, the resulting emotional stage is upset. The model is called 'circumplex' because, within this system, the spectrum of human emotions naturally forms a circle. However, in their 1994 study, Bradley and Lang found that the dimensions of Valence and Arousal alone were not sufficient to explain the data they collected. Hence they proposed to add a third dimension, *Dominance*, to the *circumplex model,* which represents a person's perception of being in control. The three-dimensional model is called *Valence-Arousal-Dominance Model* (VAD).

Note that, due to their simplicity, one-dimensional models consisting only of a Valence (i.e., negative-positive) dimension are usually not regarded as *emotion analysis* models. Instead, they belong to the neighboring field *sentiment analysis* (Schmidt et al., 2021).

From a psychological perspective, the validity of the above-mentioned models is still a topic of discussion (Crawford, 2021), as is the conceptualization of emotions as discrete or dimensional stages in general. This work acknowledges both the complexity of human emotions and the ongoing debate surrounding emotion models. However, it does not contribute to this discussion nor advocate for the validity of any of the above-mentioned models. Instead, it focuses on the technical application, for which these models serve as important conceptual frameworks – as will be discussed in the following paragraphs.

The practical implementations of *emotion analysis* borrow from *natural language processing* (NLP) and take the form of either *machine learning*-based or *lexicon*-based approaches. Though data-driven approaches existed before 2020, machine learning-based methods and large language models (LLMs) have gained significant popularity over the last five years due to their high performance. However, these approaches are not the focus of my thesis, as they require large amounts of data and computing power, making them infeasible for a number of applications in- and outside the digital humanities. In contrast, lexicon-based approaches use preexisting *emotion lexica* for their classification of textual data: these lexica provide either a discrete label or a numerical score for each word in their list, depending on the underlying emotion model. The lexicon-based models then assign these values to the corresponding words in their data and compute the emotion expressed in the text. To assess a model's performance, model-generated labels are evaluated against human-annotated ones. Lexicon-based models are comparatively low-resource because, first, they rely on preexisting resources; second, they use relatively simple statistical methods for computation; and third, they require labeled data only for evaluation, not for training. Hence, lexicon-based approaches offer a valuable alternative when data quantity or computing power is limited. This is especially relevant when *emotion analysis* is applied to texts in low-resource languages or to historical documents.

The field of *emotion analysis* has existed for more than two decades. Examples of early works include the studies by Zhe and Boucouvalas (2002), Holzman and Pottenger (2003) and Ma et al. (2005), who built models to classify emotions in instant chat messages; Alm and Sproat (2005), who analyzed emotion expression in fairy tales; Mihalcea and Liu (2006), who examined expressions of happiness in blogposts; Genereux and Evans,

(2006), who used a binary classifier to distinguish between affective states in weblog posts; Strapparava and Mihalcea (2007), who explored different methods for emotion prediction in news headlines; and Kim et al. (2009) and Bollen et al. (2011), who analyzed the expression of negative emotions in Twitter posts. Mohammad analyzed gender-based differences in emotion expression in e-mails (2011), as well as genre-based differences between novels and fairy tales (2011), and developed models for emotion recognition in Twitter posts (Mohammad, 2012b; 2014; 2015; 2018; 2022). Social media posts were also the focus of Suttles and Ide (2013) as well as Meo and Sulis (2017), while Preoțiuc-Pietro et al. (2015) analyzed the expression of depression and PTSD in Twitter posts. Buechel and Hahn (2017a and 2017b) examined the impact of annotator perspective on emotion labeling. Sailunaz (2019) used an emotion prediction model to perform network analysis on Twitter communities. Since the publication of transformer-based models like BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), models targeting a wide range of tasks have been developed by fine-tuning these LLMs. While their application in *sentiment analysis* is significantly more common, they have also been used for emotion recognition. Examples include: Al-Omari et al. (2020), who developed a model to detect emotions in English text; Chiorrini et al. (2021), who fine-tuned a BERT model for *emotion analysis*; Kamath et al. (2022), who built EmoRoBERTa, a fine-tuned RoBERTa model capable of detecting 28 different emotion stages; and Aidam et al. (2024), who combined textual and visual emotion recognition.

To expand the number of resources necessary for *emotion analysis,* namely labeled datasets and, for lexicon-based approaches, emotion lexica, numerous annotation projects have been conducted, as well as studies aiming to automatically annotate unlabeled data, expand emotion lexica, or translate labeled data or emotion lexica into other languages. Bostan and Klinger (2018) provide a useful overview of available datasets for *emotion analysis*, as well as an aggregated dataset that unifies these resources in a common file and annotation format. Mohammad (2021) lists the largest and most influential manually created emotion lexica for the English language. Though impeded by cost or a limited number of speakers, efforts have been made to create emotion lexica in other languages – for example by Schmidtke et al. (2014) for German. In addition, both Leveau et al. (2012) and Warriner et al. (2013) observed that words'

emotion values remain stable across languages; hence, translating existing emotion lexica has been another focus of research. The *National Research Council Canada Valence, Arousal, and Dominance* lexicon (NRC-VAD*,* Mohammad, 2018), for example, has been machine-translated into 108 different languages. However, even the largest manually created emotion lexica cover only a fraction of a language's vocabulary. Therefore, another line of research has focused on automatically expanding them through different induction algorithms (Turney and Littman, 2003; Bestgen, 2008; Hamilton et al., 2016a; Li et al., 2017). Buechel and Hahn (2020) created emotion lexica for 91 languages utilizing both translation and induction methods. Their English emotion lexicon has a vocabulary of 2 million words. *Word emotion induction* is especially useful when labeled data is limited and translation is not a viable option. This applies particularly to *emotion analysis* of historical data, including both textual data written in an ancient language or an earlier language stage. The latter case forms the topic of this thesis and will be discussed in further detail in Section 2.3.

2.2. Word Embeddings and Their Role in Diachronic Linguistics

Before we can move to the application of *emotion analysis* to historical text, i.e., the topic of *historical emotion analysis*, we have to familiarize ourselves with *word embeddings* and their role in *diachronic linguistics*. In NLP, a word embedding is a numerical representation of a word, typically in the form of a real-valued vector (Nielbo et al, 2024). It captures a word's meaning by assigning similar words vector representations that are closer in the vector space, while words with different meanings have vector representations that are further apart. As one may already infer from this definition, word embeddings operate on the principle that a word's meaning is defined by its context, that is, by the other words it commonly occurs with. As the linguist John Rupert Firth put it in 1957: "You shall know a word by the company it keeps."[3] This idea not only lies at the heart of NLP but is also a fundamental concept of distributional semantics, including its sub-field, *diachronic linguistics*.

[3] Firth, J. R.1957. Studies in Linguistic Analysis. Wiley-Blackwell, page 11.

Word embeddings can be broadly categorized into two main types depending on the methods they are based upon: statistical methods or neural network-based methods. In their most basic form, word embeddings based on statistical methods rely on the so-called *co-occurrence matrix*. For each word in a document, this matrix counts which words occur in its context, with the context being a fixed window of the *L* words before and after the word in question. Since each row represents one word with its entries denoting the frequency of the context words, it already constitutes a simple word embedding. To improve upon this base concept on both a theoretical and a practical level, various word embedding algorithms have been developed. These use statistical methods to transform the co-occurrence matrix, either by reducing its size, weighting its entries differently, or both. In contrast, word embeddings based on neural networks rely on the vector representations the network learns during training. To process textual data, the neural network first assigns each word in its vocabulary a random vector. Then, during its training process, the neural network adjusts both the network's weights and the vector representations to correctly predict words in their respective context. The resulting word vectors encode the semantic relationships between the words in the vocabulary and, as such, are word embeddings. A special case of neural network-based word embeddings consists of algorithms that process subword information, that is, the individual character n-grams that make up words, rather than the words themselves. In this case, word embeddings are obtained by combining the subword representations of a word. Examples of word embedding algorithms based on statistical methods are *Positive Pointwise Mutual Information* (PPMI; Church and Hanks, 1990) and *Singular Value Decomposition* (SVD) on a PPMI matrix (SVD_{PPMI}; Levy et al., 2015). Examples of word embedding algorithms based on neural networks include *Skip-Gram with Negative Sampling* (SGNS; Mikolov et al., 2013), *Continuous Bag of Words* (CBOW; Mikolov et al., 2013), and *FastText* (Bojanowski et al. 2017). I employ all of these in my work and will hence discuss them in further detail in Section 3.

Word embeddings have been applied in a wide range of fields, both in scientific research and beyond it. One such field of study – and the one relevant to this work – is *diachronic linguistics*, i.e., the scientific study of linguistic change. *Diachronic linguistics* investigates the conditions under which language change occurs and strives to

understand the underlying mechanisms (Tahmasebi et al. 2021). To this end, different approaches are employed, including the analysis of word embeddings trained on historical or diachronic corpora. Traditional methods to investigate semantic change involves selecting a small set of words, collecting examples of their occurrence spanning several decades or centuries, and evaluating these instances manually. Hence, establishing a result's objectivity is difficult, as it is tied to the specific researcher's expertise and judgment. In addition, this approach is hard to quantify, due to its time-consuming nature. Thus, the traditional method is rather unsuitable for the study of large-scale linguistic change. Here, word embedding models offer a valuable alternative: to evaluate a word's meaning in a specific time period, a model is trained on the respective historical corpus, and the word's vector representation and its neighborhood are analyzed. On the other hand, to investigate a word's semantic change over time, a model is trained for each time period in question, and the resulting word embeddings are compared. Note that, since word embeddings are not deterministic, they need to be aligned to enable meaningful comparison. This approach is more suitable for large-scale investigations, as word embedding models can process vast amounts of data and provide a representation for every word in their vocabulary. In addition, this approach provides a quantitative method to measure linguistic similarity, namely the distance between word embeddings. Hence, establishing a result's objectivity is simpler, as results are easier to reproduce – provided one has access to the historical corpora, preprocessing choices, and model configurations.

Early work in this area was conducted by Sagi et al. (2011), Wijaya and Yeniterzi (2011) and Gulordava and Baroni (2011): Sagi et al. successfully traced traditionally recognized categories of semantic change using an early form of word embedding, *Latent Semantic Analysis* (LSA; Landauer, Foltz, and Laham, 1998), and by measuring the density of the embedding vectors of the words in question. Wijaya and Yeniterzi (2011), on the other hand, analyzed the semantic change of a small set of words by extracting all 5-grams in which a word occurred from the *Google Books Corpus* (Michel et al., 2010), organizing the data by year, and then performing a cluster analysis using a word embedding model and k-nearest neighbors. Gulordava and Baroni (2011) investigated semantic change between the 1960s and 1990s on a significantly larger scale by computing the weighted

co-occurrence vectors of 10,000 words for both time periods and calculating their cosine similarity. In their 2014 study of semantic change, Jatowt and Duh compared the performance of various early word embedding algorithms, including weighted co-occurrence matrices and LSA. Interestingly, they measured semantic change on different levels, including a sentiment based one. In the same year, Kulkarni et al. (2014) employed *skip-gram* word embedding models – among other approaches – to construct a time series for large-scale linguistic change-point detection. Similarly, Hamilton et al. (2016b) utilized PPMI, SVD_{PPMI}, and SGNS to construct time-series for semantic change analysis in four different languages. In addition, the word embeddings were used as a basis for statistical analyses, resulting in the proposition of two statistical laws of semantic change. A study of semantic shifts conducted by Rosenfeld and Erk (2018) also employed SGNS, comparing time-series-based models with a continuous time-based one. As part of *SemEval-2020 Task 1*, Schlechtweg et al. (2020) compared the performance of various models for the task of unsupervised lexical semantic change detection in four languages, including Latin. Interestingly, contextualized embedding models did not yet outperform word embedding ones – most of which were based on SGNS. More recently, Liétard et al. (2023) employed both unsupervised and supervised models, including SGNS and complex combinations of SGNS with other models, to evaluate two well-known laws of synonym change. Of personal interest to me, a study by Stopponi et al. (2024) tested several word embedding models, including CBOW and PPMI, to assess their usefulness for the analysis of semantic change in Ancient Greek and obtained promising results.

Note that recent studies in *diachronic linguistics* (Giulianelli et al., 2020; Fourrier and Montariol, 2022; Lenci et al., 2022; Wen and Xu, 2022; Schlechtweg et al.; 2025) also employ the successors of word embedding models – namely, token embedding models or LLMs like ELMo (Peters et al., 2018) and BERT – often comparing the performance of these two model types with varying results. However, as was the case with *emotion analysis*, these token embedding models are not the focus of my thesis, as they require large amounts of training data – a prerequisite not met by the two corpora I use.

2.3. Historical Emotion Analysis

After familiarizing ourselves with both *emotion analysis* and *diachronic linguistics*, we can now move to the main topic of my thesis – *historical emotion analysis*. The term *historical emotion analysis* refers to the application of *emotion analysis* methods to historical data. This method is employed to measure emotional arcs in literary texts, analyze emotional changes over different time periods and even serve as another indicator of semantic change.

Theoretically, any *emotion analysis* of documents written before the contemporary period – however one chooses define it – can be considered an example of *historical emotion analysis*. This applies to several studies presented in Section 2.1, namely those on fairy tales by Alm and Sproat (2005) and Mohammad (2011). To avoid unnecessary repetition, let us take a closer look at a few additional examples: The study by Acerbi et al. (2013) examines the expression of emotion in English books written between 1900 and 2000. They analyzed the frequency of words that carry emotional content, relying on the word lists of WordNet Affect (Strapparava and Valitutti, 2004), which provide representative terms for each of the *Six Basic Emotions*, and successfully evaluated their results against real historical data. Interestingly, they found an overall decrease of emotion-related words from 1900 to 2000, with the exception of fear-related words, which have increased since the 1970s. Within this trend, they observed a growing divergence in emotion between American and British books, with American books being notably more emotional than British ones since the 1980s. Bentley et al. (2014) utilized both the same time frame and the same word list-based approach to compute their so-called *Literary Misery Index* and found a strong correlation to the *U.S. Economic Misery Index*—though they found that the *Literary Misery Index* typically lagged seven years behind. Hence, they were able to demonstrate that the emotional expression in literary texts reflects the economic conditions of their time, only delayed by a few years due to writing and publishing processes. Another example worth mentioning is the 'Emotions of London' study by Heuser et al. (2016), who investigated the depiction of London in English novels between 1700 and 1900. Notably, they collected 15,000 literary passages mentioning 382 different locations within London, annotated their emotional content, and then plotted the locations along with their respective emotions on four city maps,

one for each 50-year period included in their study. Though the study was intended to include a wide spectrum of emotions, Heuser et al. reduced them to “happiness”, “fear”, and “neutral” during the annotation process, due to inter-annotator disagreement. Hence, this study can also be interpreted as an example of the much broader field of *historical sentiment analysis*. However, the historical maps still merit its mention. In 2017, Leemans et al. annotated emotion expressions in a corpus of 29 Dutch-language theatre plays written between 1600 and 1800, both to investigate changes in historical emotion expression and to test existing emotion recognition tools, namely the word-list dictionaries LIWC (Pennebaker et al., 2015) and WordNet Affect. By expanding the Dutch versions of these dictionaries with historical terms, they improved results, thereby demonstrating the necessity of adapting contemporary *emotion analysis* methods for historical documents. Schmidt et al. conducted several studies on sentiment and emotions in historical German plays (2018b, 2021b, 2021c, 2021d). In their most recent study (Schmidt et al., 2022), they compared different models for *sentiment* and *emotion analysis*, including lexicon-based approaches (used only for *sentiment analysis*), data-driven models (CBOW and FastText), and both contemporary transformer-based models, namely BERT and ELECTRA (Clark et al., 2019), as well as their historical counterparts. For both training and evaluation, they relied on their own annotated corpus of historical German plays. Transformer-based models outperformed traditional approaches on both tasks, with overall performance being significantly better on the task of *sentiment analysis* than *emotion analysis*. Notably, contemporary LLMs yielded slightly better results than historical transformer-based models, though even in this case, accuracy never exceeded 57% on the *emotion analysis* tasks.

This last example illustrates the potential of transformer-based models. However, as mentioned above, these models are not always practical. First, for some languages, such as Ancient Greek, there is simply not enough textual data to train a reliable LLM. For other languages, an LLM may exist, but only in its contemporary form. While this may be sufficient for 19th-century German, the same may not apply to other languages or earlier language stages. In such cases, one might fine-tune a contemporary model – which is possible but requires a significant amount of historical training data and computing power. Lastly, employing an LLM for *emotion analysis* necessitates fine-tuning for the

specific task, which, in turn, requires large amounts of labeled data. This requirement is especially difficult to meet when accounting for semantic change, as one has to rely on expertly annotated data in the absence of historical witnesses. While this work does not aim to discourage the use of LLMs, its focus is on exploring an alternative approach for areas where LLMs are not applicable:

In their 2016 and 2017 studies Hellrich et al. presented their own methodology for historically accurate *emotion analysis*, combining a lexicon-based approach with tools from *diachronic linguistics* to account for semantic change: First, for a given time period, they trained a word embedding model on a representative historical corpus. Based on the resulting historical word embeddings, they then induced a historical VAD lexicon for the period in question. To this end, they relied on a contemporary VAD lexicon, namely the German *ANGST* lexicon (Schmidtke et al., 2014), as their seed words – that is, the starting values of the induction algorithm. Based on the historical word embeddings, the induction algorithm then assigned VAD scores to the previously unlabeled words in the historical corpus by computing a word's distance to the seed words and weighting their VAD scores accordingly. Hence, words with similar meanings in a given time period received similar VAD scores. The resulting historical VAD lexicon was evaluated against data annotated by experts of the specific time period, who were tasked to assign scores 'as if' they were contemporary readers from the respective period. To demonstrate the potential of this method, Hellrich et al. applied it to the following two examples: Relying on the core corpus of the *Deutsches Textarchiv* (DTA; Geyken, 2013), they generated a historical VAD lexicon for every 30-year period between the years 1690 and 1899. They then demonstrated how their method could help illustrate semantic change over time by tracking the VAD scores of different words. For instance, the German term Sünde ('sin') showed an increase in Valence between 1690 and 1899, while Arousal decreased and Dominance remained constant. This coincides with its most common collocations shifting from a predominantly religious context to one that also includes a moral-bourgeois meaning. Both observations align with the fact that the term 'Sünde' gained a less negative connotation over the course of the Age of Enlightenment and secularization. The second example illustrates a more large-scale application: Utilizing the different historical lexica to assign a weighted VAD score to each document in the three literary

genres – narrative, lyric, and drama – Hellrich et al. were able to track the emotional expression of these genres over time. They observed the most distinct emotional patterns between 1780 and 1809 (approximately covering Weimar Classicism) and between 1870 and 1899 (covering late German Realism), with both Valence and Dominance decreasing while Arousal remained constant. In 2019, Hellrich et al. expanded upon their initial concept by applying it to 1830s English. They focused on comparing different induction and word embedding algorithms to identify the optimal implementation of their method for historical VAD lexicon induction. As their gold standard for evaluation, they created expertly annotated VAD lexicons for both 1830s English and German of the 1810s to 1830s, which they published alongside their study. Though the initial results of Hellrich et al. were promising, room for future improvement remained.

As mentioned in Section 1, this is the topic of this thesis. After reproducing the English part of Hellrich et al.'s initial results, I will investigate the model's optimal implementation by comparing additional induction and word embedding algorithms, as well as examining the ideal seed word lexicon size. For evaluation, I will rely on the gold-standard lexicon provided by Hellrich et al. Before this can be done, however, we must first examine the data and algorithms that will be employed in detail. To this end, let us now move to Section 3.

3. Data and Algorithms

In this chapter, I discuss the resources and tools utilized in Sections 4 and 5. I start with the relevant data, and then move on to word embedding as well as induction algorithms.

3.1. Data

3.1.1. Training Data

As training data for my historical word embeddings, I rely on the 1830s section of the *Corpus of Historical American English* (COHA; Davies, 2012). The COHA contains over 100,000 texts from the 1810s to the 2000s, which are organized by decades, each decade since the 1830s consisting of equally sized and genre-balanced data. Aside from textual data, the COHA includes metadata in the form of automatically generated POS annotations and lemmatizations. This, combined with genre balance, makes it well-suited for training historically accurate word embeddings.

In accordance with Hellrich et al., I rely on all texts from the years 1830 to 1839 of the COHA. This results in 612 texts: 181 fictional and 431 non-fictional (including 367 texts from the 'magazine' category). Note that genre balance is considered with respect to word count, not document count. Hence, before pre-processing, this results in 9,018,621 words for fiction, 7,091,846 words for non-fiction, and 16,110,467 words in total.

3.1.2. Seed Word Lexica

For my seed word lexica, I rely on three VAD lexica: the *Affective Norms for English Words* lexicon (ANEW; Bradley and Lang, 1999), the VAD lexicon by Warriner (2013), and the NRC-VAD lexicon (Mohammad, 2018).

The ANEW is the smallest English VAD lexicon, comprising 1,034 entries rated by a class of psychology students. The Warriner lexicon is a widely used extension of ANEW, including 1,033 of ANEW's 1,034 terms – though not its VAD scores. Warriner contains 13,915 words whose VAD scores were obtained via crowdsourcing. A total of 1,827 participants took part in the rating process, with a male-to-female ratio of approximately

40 to 60. Lastly, the NRC-VAD lexicon, with 19,971 entries, is the largest of the three. It includes 1,024 entries from ANEW and 13,855 from Warriner – though, again, not their VAD scores. It too was created via crowdsourcing, with approximately 1,800 participants and a male-to-female ratio of 63 to 37. Note that the VAD scores for entries shared by Warriner and NRC-VAD differ significantly, with Pearson correlations of 0.814 for Valence, 0.615 for Arousal, and only 0.326 for Dominance (for further details, see Mohammad, 2018). This thesis is agnostic with regard to any potential superiority of one lexicon over the other; instead, it focuses on investigating which lexicon yields better results in the specific context of my study. Both ANEW and Warriner rate their VAD scores on a scale of 1 to 9, while NRC-VAD represents its values on a 0 to 1 interval and was therefore scaled up to align with the other two lexica.

3.1.3. Gold Standard

For my evaluation, I rely on the VAD lexicon for 1830s English created by Hellrich et al. in 2019. In their study, Hellrich et al. closely followed the methodology developed by Bradley and Lang, with one crucial difference: They instructed participants to evaluate the provided terms as if they were living in the respective time period. The participants were two doctoral students specialized in historical linguistics and experienced in interpreting 19th-century texts. In the absence of native speakers of 19th-century English expert opinion is the best available alternative for obtaining labeled emotion data. The historical lexicon consists of 100 words, randomly selected from the 1,000 most frequent nouns, adjectives, and lexical verbs in the COHA, and rated on a scale of 1 to 9.

Note that VAD scores between historical and contemporary lexica differ substantially. The historical scores for the word ‘divine’, for example, are 7.0 for Valence, 7.0 for Arousal, and 2.0 for Dominance, while Warriner assigns it contemporary scores of 7.2, 3.0, and 6.0, respectively. As a reason for the differing scores, annotators cited the yet-to-occur secularization, while the stable value of Valence may, in part, be due to the increased popularity of the second, non-spiritual meaning of ‘divine’ as something ‘supremely good’. For the 97 words shared by the English gold standard lexicon and Warriner, the correlations are 0.66 for Valence, 0.51 for Arousal, and 0.31 for Dominance. These

differences between expert-annotated historical data and contemporary data illustrate both the reality of linguistic change and the need for historically accurate tools when performing *historical emotion analysis*.

3.2. Word Embedding Algorithms

Let us first consider notation: For their implementation of word embedding algorithms – though they only use SVD_{PPMI} and SGNS – Hellrich et al. rely on the work of Levy et al. (2015). Accordingly, I adopted Levy et al.'s notation as the standard and adapted differing notations where necessary, specifically for CBOW and FastText, which are not included in Levy et al.

For a corpus K, V_W denotes the set of all words occurring in K, i.e., the vocabulary of K, while V_C denotes the set of all co-occurring words in K, i.e., the context vocabulary of K. Note that in our case, every word in K is an element of both V_W and V_C, since every word is considered both as a word that has a context and as a word that occurs in other words' contexts. Hence, V_W and V_C are identical, though they are utilized for different purposes. The collection of observed word-context pairs is denoted by D, with a context word $c \in V_C$ being defined as a context word of $w \in V_W$ if it appears in the $L \in \mathbb{N}$ words before or after w, the so-called L-sized context window. For every word $w \in V_W$ and context word $c \in V_C$, its word-context pair is represented by $(w, c) \in D$, while $\#(w, c)$ denotes the number of times the pair occurs in D. Similarly, $\#(w) = \sum_{c' \in V_C} \#(w, c')$ and $\#(c) = \sum_{w' \in V_W} \#(w', c)$ are the number of times w and c appear in D. For a sequence of words $c_{-L}, c_{-L+1}, \dots, c_{-1}, w, c_1, \dots, c_{L-1}, c_L$ occurring in the corpus K, where $w \in V_W$ and each $c_i \in V_C$, the context $\bar{c} := \{c_{-L}, c_{-L+1}, \dots, c_{-1}, c_1, \dots, c_{L-1}, c_L\}$ denotes the set of surrounding words, the set $\bar{C}$ the collection of all contexts $\bar{c}$, and the set $\bar{D}$ all pairs $(w, \bar{c})$ of words w occurring in contexts $\bar{c}$. Additionally, the vectors $\vec{w} \in \mathbb{R}^d$ and $\vec{c} \in \mathbb{R}^d$ denote the d-dimensional vector representations of $w \in V_W$ and $c \in V_C$. The set of all vectors $\vec{w}$ forms the rows of the $|V_W| \times d$ matrix W, i.e., the word embedding matrix, and similarly, the set of all vectors $\vec{c}$ the rows of the $|V_C| \times d$ matrix C, i.e., the context embedding matrix. Finally, an embedding matrix obtained by a specific embedding algorithm a is referred to as W^a and C^a (e.g., W^{PPMI} or W^{SGNS}).

3.2.1. Positive Pointwise Mutual Information

As mentioned in Section 2.2, traditional statistical approaches are based on what is referred to as *co-occurrence matrix*, a $|V_W| \times |V_C|$ matrix M that is constructed as follows: for a corpus K, a vocabulary V_W, a context vocabulary V_C and a context window of size L, every row of M represents a word $w \in V_W$ and every column of M a context word $c \in V_C$. The specific entry of M that corresponds to a word w's row and a context word c's column is given by the number of times w and c occur together, that is, by $\#(w, c)$. M is one of the simplest versions of a word embedding, with its rows serving as vector representations $\vec{w}$ of the vocabulary words $w \in V_W$ and its columns serving as the vector representations $\vec{c}$ of the context words $c \in V_C$. Hence, in this case, $M = W$ and case $M^\top = C$.

Since most words occur in only a limited number of contexts, $\#(w, c)$ often reduces to 0. Therefore, M is usually a high-dimensional sparse matrix. However, since the entries of M consist of absolute values, M provides no relative contextual information. To address this issue, different methods have been designed to normalize M, a popular one being *Positive Pointwise Mutual Information* (PPMI) and its predecessor, *Pointwise Mutual Information* (PMI; Church and Hanks, 1990). PMI is defined as the log ratio between the joint probability of w and c and the product of their individual probabilities. Thus, it replaces the value of $\#(w, c)$ with:

$$PMI(w, c) = \log\left(\frac{P(w,c)}{P(w)P(c)}\right) = \log\left(\frac{\#(w,c)\cdot|D|}{\#(w)\cdot\#(c)}\right).$$

However, since $M$is a sparse matrix, the respective M^{PMI}matrix consists of many entries equal to $\log(0) = -\infty$. This information, though, is neither interpretable in a meaningful way nor computationally feasible. Hence, PMI is often replaced by either PMI_0, which defines $PMI(w, c) = 0$ whenever $\#(w, c) = 0$, or PPMI, which replaces all negative values with 0:

$$PPMI(w, c) = \max\left(PMI(w, c), 0\right).$$

PPMI is the more common approach, as it has been shown that M^{PPMI} outperforms M^{PMI_0} on semantic similarity tasks (Bullinaria and Levy, 2007). Keeping in line with our notation, the word-embedding matrix W^{PPMI} obtained from PPMI is given by M^{PPMI}, and the context-embedding matrix C^{PPMI} by its transpose, $(M^{PPMI})^\top$.

Note that all PMI-based approaches tend to be biased towards infrequent events (Turney and Pantel, 2010): if a rare context word c co-occurs with a target word w even once, $PPMI(w, c)$ usually yields a relatively high value, due to $P(c)$ being close to 0 while being a factor in $PPMI(w, c)$'s denominator. Hence, the values of $\vec{w}$ tend to be skewed towards rate context words. Nonetheless, PPMI remains a widely used method, having produced many valuable results across various contexts.

3.2.2. Singular Value Decomposition on a PPMI Matrix

An issue that neither PPMI nor any other PMI-based approach resolves is that the resulting word embedding matrix M^{PPMI} is a high-dimensional sparse matrix. This negatively affects computational efficiency, especially when working with large, diverse corpora. To address this problem, methods of dimensionality reduction are commonly employed, with truncated Singular Value Decomposition (SVD[4]) being a popular choice: Given a matrix $M \in \mathbb{R}^{m \times n}$, SVD uses methods from linear algebra to decompose M into three matrices $M = U\Sigma V^{\top}$, where $U \in \mathbb{R}^{m \times m}$, $V^{\top} \in \mathbb{R}^{n \times n}$, and $\Sigma \in \mathbb{R}^{m \times n}$. U and V are orthogonal matrices and Σ is a diagonal matrix containing the singular values of M in decreasing order. In a second step, to simplify the matrix while keeping its essential structure, all but the first d singular values of Σ are set to zero. This effectively reduces Σ to a smaller diagonal matrix $\Sigma_d \in \mathbb{R}^{d \times d}$ – i.e., Σ has been *truncated*. Similarly, U and $V^{\top}$ can be reduced to $U_d \in \mathbb{R}^{m \times d}$ and ${V^{\top}}_d \in \mathbb{R}^{d \times n}$, as their remaining rows and columns would evaluate to zero in a multiplication with the modified Σ. The resulting matrix $\bar{M}=U_d\Sigma_d{V^{\top}}_d$ is the "optimal"[5] approximation of M, but with lower rank. It preserves the original structure of the data while removing noise – which is the statistical goal of applying SVD.

How can this be interpreted in terms of word embeddings? Suppose the matrix M is a co-occurrence matrix, representing the relationship between m vocabulary words and n context words. Then, conceptually, the matrix U can be understood as a word embedding

[4] For information on the origins of SVD see: G. W. Stewart, *On the early history of the singular value decomposition*.

[5] "Optimal" here refers to being optimal with respect to the Frobenius norm, a mathematical norm used to measure the distance between two matrices through elementwise comparison.

of the vocabulary, since every row in U corresponds to a word $w \in V_W$. Similarly, the matrix V represents the context, with every row in V corresponding to a context word $c \in V_C$. The values of Σ, finally, indicate the importance of the underlying semantic dimensions, which capture the relationships between the vocabulary and the context words. Hence, the matrices U_d and V_d can be understood as low-dimensional, dense word and context embeddings that reduce complexity while preserving structural information. Note that, if V_W=V_C, U_d is commonly replaced by the average of word and context embeddings, that is, $U_d := \frac{1}{2}(U_d + V_d)$.

SVD_{PPMI}, finally, combines the advantages of both PPMI and SVD by performing SVD on a PPMI matrix instead of a simple co-occurrence matrix. Hence, following our notation, the word and context embeddings $W^{SVD_{PPMI}}$ and $C^{SVD_{PPMI}}$ obtained from SVD_{PPMI} are given by the truncated matrices U_d^{PPMI} and V_d^{PPMI}, respectively. Compared to PPMI, SVD-based representations are more robust and have been shown to perform on par with neural network-based methods such as SGNS (Levy et al., 2015).

3.2.3. Skip-Gram with Negative Sampling

In contrast to count-based statistical approaches like PPMI and SVD_{PPMI}, Skip-Gram with Negative Sampling (SGNS) is based on a probabilistic model. It is one of the two models developed within the Word2Vec framework (Mikolov et al., 2013), alongside Continuous Bag of Words (CBOW).

For a vocabulary V_W, a context vocabulary V_C a context window of size L, and a set of word-context pairs D, SGNS's main objective is to accurately predict a word w's surrounding context words. To this end, SGNS aims to maximize a function of the dot product $\vec{w} \cdot \vec{c}$ for all word-context pairs $(w, c) \in D$, while minimizing it for negative examples, that is, word-context pairs (w, c_N) that are not necessarily elements of D. The function applied to the dot product is typically the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$, which outputs values close to 1 for large positive inputs and close to 0 for large negative inputs. This objective is based on the idea that words occurring in similar contexts should have similar vector representations, resulting in high dot products, while unrelated words should have dissimilar representations and low dot products. The sigmoid function then

transforms the dot products into a probability-like scores, approximating the likelihood that two words co-occur.

For mathematical reasons, the logarithm is applied to the sigmoid function of the dot product, and hence the overall optimization problem takes the following form:

$$argmax_\theta \left[\sum_{(w,c)\in D} log\, \sigma(\vec{w} \cdot \vec{c}) + \beta \sum_{(w,c)\in V_W \times V_C} \log \sigma(-\vec{w} \cdot \vec{c}) \right],$$

where θ denotes all parameters to be optimized – i.e., all entries of all vectors $\vec{w}$ and $\vec{c}$ corresponding to a word $w \in V_W$ or a context word $c \in V_C$ – and β is a real-valued multiplier. Note that the k negative examples are not chosen randomly, but probabilistically, according to the unigram distribution $P_D(c) = \frac{\#(c)}{D}$, meaning SGNS is more likely to draw negative examples from frequent context words. The unigram distribution is typically smoothed by raising it to the power of $\alpha = 0.75$, as this has been found to improve SGNS's performance (Levy et al., 2015).

In its practical implementation, SGNS solves the optimization problem described above by employing a two-layer neural network trained via gradient descent. For all words $w \in V_W$, the network's weights are optimized to predict the most likely context words $c \in V_C$, and the learned input and output weights serve as the dense, low-dimensional word and context embeddings, respectively. Following our established notation, they are denoted by W^{SGNS} and C^{SGNS}.

SGNS has been shown to outperform SVD_{PPMI} on word analogy tasks, while the opposite holds for word similarity tasks (Levy et al, 2015; Hamilton, 2016b). Nonetheless, SGNS is considered a robust baseline model due to its consistent performance, while being fast and computationally efficient to train.

3.2.4. Continuous Bag of Words

In contrast to SGNS, CBOW follows a different approach. While SGNS aims to predict the most likely context word $c \in V_C$ for a given word $w \in V_W$, CBOW aims to predict the most likely word $w \in V_W$ given a context window of size L and a context $\bar{c} :=$

$\{c_{-L}, c_{-L+1}, \dots, c_{-1}, c_1, \dots, c_{L-1}, c_L\}$. To this end, CBOW aims to maximize $\sigma(\vec{w} \cdot \vec{\bar{c}})$ for all averaged context vectors $\vec{\bar{c}} = \frac{1}{2L}\sum_{i \neq 0} \vec{c}_i$ corresponding to a context $\bar{c} \in \bar{C}$ and their associated words $w \in V_W$, while minimizing $\sigma(\vec{w}_N \cdot \vec{\bar{c}})$ for negative samples w_N that do not co-occur within the given context $\bar{c}$. Despite this difference in direction, CBOW relies on the same key concepts as SGNS: as a "*word is defined by the company it keeps*"[6], a context $\bar{c}$ and the words w that occur in it should have similar vector representations (i.e., high dot products), while words w_N unrelated to the context $\bar{c}$ should have dissimilar representations and low dot products. The sigmoid function, again, translates this score into a probability-like measure of co-occurrence.

Hence, CBOW's optimization problem takes the following form, again employing the logarithm for mathematical convenience:

$$argmax_\theta \left[\sum_{(w,\bar{c}) \in \bar{D}} log\, \sigma(\vec{w} \cdot \vec{\bar{c}}) + \beta \sum_{(w,\bar{c}) \in V_W \times \bar{C}} \log \sigma(-\vec{w} \cdot \vec{\bar{c}}) \right],$$

where θ, again, denotes all parameters to be optimized – i.e., all entries of all vectors $\vec{w}$ and $\vec{c}$ corresponding to a word $w \in V_W$ or a context word $c \in V_C$, as $\vec{\bar{c}}$, per definition, is a linear combination of context vectors $\vec{c}_i$. As before, β is a real-valued multiplier. Note that for selecting its negative examples, CBOW employs the same procedure as SGNS.

In its practical implementation too, CBOW follows along the same lines as SGNS by employing a two-layer neural network trained via gradient descent. However, instead of encoding a word $w \in V_W$ to predict its context word $c \in V_C$, CBOW encodes the surrounding context words $c_i \in \bar{c}$, averages their embeddings, and uses the resulting vector representation to predict the center word w. The learned input and output weight matrices serve as the dense, low-dimensional word and context embeddings, respectively. Following our established notation, they are denoted by W^{CBOW} and C^{CBOW}.

Theoretically, CBOW's context-based approach enables it to capture word meanings on a deeper level. However, in practice, this potential has not been fully realized. In a 2019 study, Wang et al. compared various word embedding models, including SGNS and

[6] Again Firth, see footnote [3]

CBOW, and found that SGNS outperformed CBOW on tasks such as part-of-speech tagging, chunking, and named entity recognition. Results for sentiment analysis, however, were mixed, depending on the dataset and evaluation method.

3.2.5. FastText

In contrast to all previously described algorithms, FastText (Bojanowski et al., 2017) does not operate at the word level. Instead, it represents a word as the sum of its n-grams, capturing meaning at the sub-word level. This approach has been shown to be particularly beneficial for languages with rich morphology and a large number of rare words, as FastText can provide reliable vector representations even for words unseen during training.

Apart from considering the subword units of a word $w \in V_W$ rather than w alone, FastText operates within the same framework as SGNS. A word w is typically represented as the set Z_w of its 3- to 6-character n-grams (including word boundary symbols) along with the word itself. Each element $z_i \in Z_w$ is associated with a vector $\vec{z}_i$. The dot product used in SGNS is then replaced with $\sum_{z_i \in Z_w} \vec{z}_i \cdot \vec{c}$, while the optimization objective and the implementation remain unchanged. To obtain the final embedding for a word w, FastText simply sums the vector representations of all its n-grams. Following our notation, the word and context embedding matrices obtained through FastText are denoted by W^{Fast} and C^{Fast}.

3.3. Induction Algorithms

Lastly, we must familiarize ourselves with the different emotion induction algorithms, the first three of which are also employed by Hellrich et al. Here, too, let us first consider the notation: For a given word embedding W, a corresponding vocabulary V_W, and a VAD lexicon L, let $V_{Seed} := \{s_1, \dots, s_n\} \subseteq V_W$ denote the set of seed words, $W_{Seed} := \{\vec{s}_1, \dots, \vec{s}_n\}$ the corresponding set of word embedding vectors, and $E_{Seed} := \{\vec{e}_{s_1}, \dots, \vec{e}_{s_n}\}$ the corresponding set of VAD scores. The aim of each emotion induction algorithm is to

assign a VAD score $\vec{e}_w$ to any word $w \in V_W \backslash V_{Seed}$ by utilizing its corresponding word embedding vector $\vec{w}$, along with W_{Seed}, and E_{Seed}.

3.3.1. k-Nearest-Neighbor

This algorithm is a modified version of the k-Nearest-Neighbor-based algorithm by Bestgen (2008). While the original assigns one-dimensional sentiment scores, the modification by Hellrich et al. replaces these with three-dimensional VAD scores. For each word $w \in V_W \backslash V_{Seed}$, w's VAD score is set to the average score of the k seed words in $nearest(k, w) := \{s_1, \dots, s_k\}$, which are the k seed words most similar to w. Similarity is determined using cosine similarity, calculated as $\cos(\vec{w}, \vec{s}_i) := \frac{\vec{w} \cdot \vec{s}_i}{\|\vec{w}\|^7 \| \vec{s}_i \|}$, where $\vec{w} \in W$ and $\vec{s}_i \in W_{Seed}$ are the corresponding word embedding vectors. Cosine similarity ranges from 1 to -1, yielding values close to 1 if two vectors point in the same direction and values close to -1 if they point in opposite directions. Since semantically similar words are expected to have similar vector representations, selecting the k seed words most similar to w corresponds to choosing those with the highest cosine similarity values as elements of $nearest(k, w)$. Having determined $nearest(k, w)$, kNN then assigns the VAD score $\vec{e}_w$ for a word w as:

$$\vec{e}_w := \frac{1}{k} \sum_{s_i \in nearest(k,w)} \vec{e}_{s_i}$$

3.3.2. PaRaSimNum

This algorithm is an adaptation of the PaRaSim algorithm introduced by Turney and Littman (2003). The original approach assigns one-dimensional sentiment scores to a word w based on its similarity to two opposing sets of paradigm words, whereas the modification proposed by Hellrich et al. extends this to three-dimensional VAD scores. Specifically, the set of seed words, V_{Seed}, is defined as the union of the two paradigm sets. Then, for each word $w \in V_W \backslash V_{Seed}$ with word embedding vector $\vec{w}$, the algorithm computes a weighted sum of the VAD vectors $\{\vec{e}_{s_1}, \dots, \vec{e}_{s_n}\}$ corresponding to the seed

[7] The notation $\| \ \|$ refers to the Euclidean norm. For a vector $\vec{x} = (x_1, \dots, x_n)$, it is defined as: $\| \vec{x} \| := \sqrt{{x_1}^2 + \cdots + {x_n}^2}$.

words. Each VAD vector is weighted by the cosine similarity between $\vec{w}$ and the embedding of the respective seed word $\vec{s}_i$. Lastly, a normalization factor is applied. Hence, PaRaSimNum assigns the VAD score $\vec{e}_w$ for a word w as follows:

$$\vec{e}_w := \frac{\sum_{s_i \in S} \cos(\vec{w}, \vec{s}_i) \vec{e}_{s_i}}{\sum_{s_i \in S} \cos(\vec{w}, \vec{s}_i)}$$

3.3.3. Random Walk

This algorithm is an adaptation of the RandomWalk algorithm proposed by Hamilton et al. (2016a). The original approach propagates sentiment scores through a random walk over a weighted lexical graph. The modification introduced by Hellrich et al. again extends this to three-dimensional VAD scores. The weighted lexical graph is constructed by connecting each word $w \in V_W \backslash V_{Seed}$ to the k seed words in $nearest(k, w)$ (as defined in Section 3.3.1.). The weights of the edges are defined as follows:

$$E_{w,s_i} = \arccos\left(-\frac{\vec{w} \cdot \vec{s}_i}{\|\vec{w}\| \|\vec{s}_i\|}\right).$$

Next, the random walk method by Zhou et al. (2004) is employed to propagate VAD scores. The intuition is that the VAD score of a word w should be proportional to the combined probabilities of all random walks starting from any seed word $s_i \in V_{Seed}$ and reaching w. That is, the VAD scores of seed words more closely connected to w should have a greater influence on its VAD score than those further away in the lexical graph.

For the implementation, let $|V_W|$ denote the number of all words $w \in V_W$. Let $T \in \mathbb{R}^{|V_W| \times |V_W|}$ be the transition matrix, initialized as described by Zhou et al., and let $S \in \mathbb{R}^{|V_W| \times 3}$represent the VAD scores of the seed words: Each row in S corresponding to a seed word is set to its VAD score, while all other rows are set to zero. Lastly, let $P \in \mathbb{R}^{|V_W| \times 3}$ represent the induced VAD score for each word $w \in V_W$. P is initialized by setting all its entries to $\frac{1}{|V_W|}$, and then updated iteratively using the following formula:

$$P^{(t+1)} = \beta T P^{(t)} + (1 - \beta) S.$$

The factor $\beta \in [0,1]$ balances local and global consistency, that is, whether the algorithm favors similar scores for neighboring words (with higher β) or correct scores for seed words (with lower β).

This process is run twice: once with S populated by the original VAD scores of the seed words, yielding a final result denoted byP^+, and once with S populated by the inverted VAD scores (each score mirrored around 5), yielding P^-. Both runs are executed until they converge. The final VAD scores are then obtained as

$$P^{final} := {P^+}/{(P^+ - P^-)},$$

where the i-th row of P^{final} corresponds to the VAD score $\vec{e}_{w_i}$ of the word $w_i \in V_W$.

3.3.4. Linear Regression

This algorithm is taken directly from Li et al. (2017), who employed linear regression to induce a contemporary VAD lexicon. Specifically, they found the Ridge regression model (Hoerl and Kennard, 1970) to be among the most suitable models. Hence, this is the one I proceeded with, the only modification being that I utilized historical rather than of contemporary word embeddings.

For each seed word $s_i \in V_{Seed}$ with corresponding word embedding vector $\vec{s}_i = (s_1^i, \dots, s_n^i)$ and VAD vector $\vec{e}_{s_i} = (e_{s_i}^V, e_{s_i}^A, e_{s_i}^D)$, the task of linear regression is to learn the mapping functions f_V, f_A, and f_D, where

$$f_X(\vec{s}_i) := a_1^X s_1^i + \cdots + a_n^X s_n^i \; for \; X \in \{V, A, D\},$$

such that the predicted values closely match the corresponding VAD scores $e_{s_i}^V$, $e_{s_i}^A$, and $e_{s_i}^D$. To this end, the Ridge regression model optimizes the following objective for each dimension $X \in \{V, A, D\}$:

$$min_{\vec{a}^X} \sum_{s_i \in V_{Seed}} \| f_X(\vec{s}_i) - e_{s_i}^X \|_2^2 + \beta \| \vec{a}^X \|_2^2,$$ [8]

where $\vec{a}^X = (a_1^X, \ldots, a_n^X)$, and β is the regularization weight.

The learned mapping functions f_V, f_A, and f_D provide the optimal explanation of our given data. Hence, they can subsequently be used to compute the VAD score of any word $w \in V_W \backslash V_{Seed}$ with vector representation $\vec{w}$. The predicted VAD score vector $\vec{e}_w$ is then given by:

$$\vec{e}_w := (f_V(\vec{w}), f_A(\vec{w}), f_D(\vec{w})).$$

[8] The notation $\| \|_2^2$ refers to the squared Euclidean norm (also called the squared L^2 norm). For a vector $\vec{x} = (x_1, \ldots, x_n)$, it is defined as: $\| \vec{x} \|_2^2 := {x_1}^2 + \cdots + {x_n}^2$.

4. Experiments

Hellrich et al.'s model consists of three independent components: a historical word embedding trained on the 1830s section of the COHA, a contemporary VAD lexicon providing seed words and their corresponding VAD scores, and an induction algorithm that combines the two to infer historical VAD scores for the remaining vocabulary. Since each component can be realized using different concrete implementations, a wide range of model configurations can be compared. In their experiments, Hellrich et al. evaluate the performance of 12 models: they use SVD_{PPMI} and SGNS as word embedding algorithms; a limited version of the ANEW and the full ANEW (with VAD scores taken from Warriner in both cases) as seed lexica; and kNN, PaRaSimNum, and RandomWalk as induction algorithms. To evaluate performance, the results are compared against their historical gold standard lexicon.

Hellrich et al. found that the six models trained with the full ANEW as the seed lexicon consistently outperformed those trained with the limited ANEW. Among the former six models, however, performance differences were not statistically significant, and the average correlation across all three affective dimensions between predicted VAD scores and the gold standard never exceeded 0.365. Thus, there is both room for improvement and an open question regarding the optimal number of seed words: while the models benefited from using the full ANEW, too much contemporary influence may negatively affect performance. Hence, after recreating the experiments of Hellrich et al., I will investigate these two areas by expanding the original setup in three ways: by employing three additional word embedding algorithms – PPMI, CBOW, and FastText; by incorporating an additional VAD lexicon, NRC-VAD; and by introducing an additional induction algorithm, Linear Regression.

I proceed as follows: In Section 4.1, I preprocess the COHA and train the five historical word embeddings. In Section 4.2, I examine every possible combination of historical word embedding and induction algorithm, while, in concurrence with Hellrich et al., using the ANEW lexicon for seed words and the Warriner lexicon for their corresponding VAD scores. I too evaluate performance by comparing the predicted scores against the historical gold standard. Thus, in this section, I both replicate Hellrich et al.'s original

experiments and explore an initial set of improvement options. In Section 4.3, I replace the VAD scores from the Warriner lexicon with those from the NRC-VAD lexicon and repeat the process described in Section 4.2. Since the VAD scores in Warriner and NRC-VAD differ significantly, this allows me to investigate whether NRC-VAD offers additional potential for improvement. Lastly, in Section 4.4, I investigate the impact of seed lexicon size. For both Warriner and NRC-VAD, I select subsets ranging from 2,000 to 8,000 words (and up to 10,000 for NRC-VAD) with a step size of 1,000 words. To ensure that differences in performance are indeed a result of changes in lexicon size, I randomly generate 50 lexica for each size. I then select the three highest-performing models from Sections 4.2. and 4.3., and run each of them on all 50 randomly generated lexica for every size step. To evaluate a selected model's performance at a specific seed lexicon size, I compute the average score across all 50 runs for each affective dimension.

4.1. Preprocessing and Training

In concurrence with Hellrich et al., the 1830s subsection of the COHA was preprocessed using the lemmatization provided, punctuation was removed, all text converted to lowercase, and words with a token frequency lower than 10 were removed. The resulting training corpus consists of 12,873,074 words with 21,784 unique lemmata.

Following Hellrich et al., historical word embeddings were trained using a context window of four words, limited by document boundaries but not sentence boundaries. For SVD_{PPMI}, SGNS, CBOW, and FastText, the dimensionality of the word vectors was set to 300. For PPMI, dimensionality is not an adjustable parameter, as it is determined by the number of unique lemmata in the training corpus; in this case, PPMI yielded word vectors with 21,784 dimensions. For SVD_{PPMI}, word and context embedding were combined. For SGNS, hyperparameter settings follow Hellrich et al., who in turn follow Hamilton et al. (2016a). For CBOW and FastText, the same hyperparameters were used, except for the number of training epochs, which was set to eight for SGNS, 20 for CBOW, and 35 for FastText. For FastText, the n-gram size was set to range from three to six, following Bojanowski et al. (2017). The Gensim (Řehůřek, 2010) implementations of SGNS, CBOW, and FastText were used to train the respective historical embeddings.

For both Warriner, and NRC-VAD, 97 words that also appear in the gold standard had to be excluded, as gold standard entries cannot be used as seed words. In addition, words not present in the COHA were removed, as they do not have historical word embeddings and therefore cannot be used in our models. This filtering resulted in 8,519 words for Warriner and 11,003 words for NRC-VAD. For the experiments in Sections 4.2 and 4.3, the lexica had to be further restricted to words also occurring in ANEW, as – following Hellrich et al. – seed word lexica must consist of ANEW words paired with VAD scores from either Warriner or NRC-VAD. This second filtering resulted in 8,422 words for Warriner and 10,906 words for NRC-VAD, denoted by Warriner-ANEW and NRC-VAD-ANEW, respectively. Note that all NRC-VAD lexica were rescaled to match with the 1-9 scale used by the other three lexica.

Following Hellrich et al., k was set to 16 for both kNN and PaRaSimNum. For RandomWalk, β was set to 0.9 in accordance with Hamilton et al. (2016a), while, for linear regression, β was set to 1, as this is the default value.

4.2. Word Embeddings & Induction Algorithms in Combination with Warriner

With the seed word lexicon set as Warriner-ANEW, experiments were conducted for all possible combinations of word embedding and induction algorithms, resulting in 20 different model configurations including the six also computed by Hellrich et al. Results were evaluated against the gold standard using Pearson's r to measure the correlation between predicted and actual values for each affective dimension (Figure 2). To compare model performance with Hellrich et al. (Figure 3), the mean correlation across all affective dimensions was also computed, as this is the only metric reported in their study.

Correlation results vary by affective dimension: Valence shows the highest correlations, ranging from 0.12 to 0.54, with a median of 0.475. This is followed by Arousal, with correlations between -0.05 to 0.48 and a median of 0.38. For Dominance, correlations are considerably lower, ranging from 0.07 to 0.26 and a median of 0.20. This variation in performance aligns with the findings of Hellrich et al. (2019, 2020), who observed that Valence is typically the easiest dimension to predict. Note that for both Valence and Arousal nearly all correlations are statistically significant, with only one and two cases

Model	Valence	Arousal	Dominance	Mean
kNN_CBOW	0.47	0.35	0.17†	0.3291
kNN_FastText	**0.54**	**0.37**	**0.24**	**0.3850**
kNN_PPMI	0.45	0.32	0.23	0.3369
kNN_SGNS	0.48	0.42	0.20†	0.3644
kNN_SVD_{PPMI}	0.42	0.33	0.15†	0.2997
PaRaSimNum_CBOW	0.25	0.19†	0.20	0.2145
PaRaSimNum_FastText	0.48	0.38	0.17†	0.3452
PaRaSimNum_PPMI	0.47	0.38	0.18†	0.3473
PaRaSimNum_SGNS	0.52	0.40	0.20	0.3759
PaRaSimNum_SVD_{PPMI}	0.12†	-0.05†	0.09†	0.0545
RandomWalk_CBOW	0.42	0.37	0.10†	0.2955
RandomWalk_FastText	0.51	0.39	0.21	0.3691
RandomWalk_PPMI	0.52	0.36	0.21	0.3669
RandomWalk_SGNS	0.48	0.41	0.16†	0.3483
RandomWalk_SVD_{PPMI}	0.50	0.41	0.18†	0.3591
LinReg_CBOW	0.47	0.39	0.25	0.3712
LinReg_FastText	**0.49**	**0.38**	**0.26**	**0.3782**
LinReg_PPMI	0.43	0.48	0.20†	0.3681
LinReg_SGNS	**0.50**	**0.36**	**0.26**	**0.3771**
LinReg_SVD_{PPMI}	0.41	0.47	0.07	0.3164

Figure 2: Correlation between model predictions and the gold standard, measured by Pearson's *r*. The three highest-performing models are shown in bold. Entries marked with an asterisk (†) are not statistically significant ($p > 0.05$).

respectively yielding a p-value greater than 0.05. In contrast, for Dominance, 50% of the correlations are not statistically significant.

Following Hellrich et al., the mean correlation across all affective dimensions is used as measure of overall model performance. By this metric, the combination of FastText historical word embedding and kNN induction algorithm yields the best results, closely followed by the combination of FastText embedding and linear regression, and the model using SGNS embedding and linear regression. On average, models utilizing the FastText embedding perform best, followed by those using SGNS. Models based on the PPMI embedding often perform well in one affective dimension but yield lower results in the

remaining two. Models using either the CBOW or SVD_{PPMI} embedding show the weakest overall performance – the only exception being the model combination of CBOW and linear regression. On average, models using linear regression achieve the highest performance, with four of the eight top models relying on this induction algorithm. Results for models employing kNN and RandomWalk are slightly more mixed, while most PaRaSimNum-based models perform below average, including the two lowest-ranking ones.

Model	Hellrich et al.	Mean
kNN_SGNS	0.365	0.3644
kNN_SVD_{PPMI}	0.307	0.2997
PaRaSimNum_SGNS	0.361	0.3759
PaRaSimNum_SVD_{PPMI}	**0.348**	**0.0545**
RandomWalk_SGNS	0.361	0.3483
RandomWalk_SVD_{PPMI}	0.351	0.3591

Figure 3: Mean Correlation reported by Hellrich et al. with corresponding mean correlation taken from Figure 2. The one diverging model is shown in bold.

Comparing the results with those of Hellrich et al., five out of six models yield very similar outcomes. However, one model, the combination of the SVD_{PPMI} embedding and the PaRaSimNum induction algorithm, diverges significantly. Further testing revealed no evidence of a faulty implementation of PaRaSimNum or any issues with the embedding itself. While the performance of most PaRaSimNum-based models is below average, not all underperform drastically. The combinations of PaRaSimNum with PPMI and with FastText rank 12th and 13th, respectively. The combination of PaRaSimNum and SGNS even performs well, achieving 4th place. Other models using the SVD_{PPMI} embedding also perform as expected, suggesting that the problem lies specifically in the combination of PaRaSimNum and SVD_{PPMI}. Notably, both kNN and RandomWalk consider only the k nearest neighbors, while PaRaSimNum computes the weighted sum of all similarities between the target word and the seed words. Upon closer inspection, it was found that the SVD_{PPMI} embedding often yields similarity values close to zero. Consequently, PaRaSimNum's denominator is close to zero too, which I suspected to cause the highly divergent VAD scores this model produced. However, when replacing the SVD_{PPMI}

embedding with the simpler PPMI embedding, the similarity values are also close to zero, yet the resulting model performs normally – suggesting that the small denominator is effectively balanced by an equally small numerator. Hence, I suspect that the singular value decomposition of the PPMI matrix introduces a form of nonlinearity into the SVD_{PPMI} embedding, which in turn disrupts this balance between numerator and denominator. disrupts this balance between numerator and denominator. As SVD is not deterministic, the diverging results may be explained by Hellrich et al. using a different SVD implementation, potentially yielding an embedding that is equally correct but more suitable for the PaRaSimNum algorithm.

4.3. Word Embeddings & Induction Algorithms in Combination with NRC-VAD

After replacing the Warriner-ANEW seed word lexicon with the NRC-VAD-ANEW lexicon, the experiments described in Section 4.2 were repeated. The predictions of the resulting 20 models were again evaluated against the gold standard using Pearson's r to measure correlation (Figure 4).

Compared to Section 4.2, the correlations of both Valence and Arousal are more varied, ranging from 0.04 to 0.54 and -0.08 to 0.49, respectively, while the median correlation increased in both cases – to 0.49 for Valence and 0.405 for Arousal. For Dominance, correlations are considerably lower, ranging from -0.01 to 0.15, with a median of just 0.05. Notably, while all but one correlation for Valence and Arousal are statistically significant, none of the Dominance correlations reach statistical significance. This indicates that the models have slightly improved in predicting Valence and Arousal but have become significantly less effective at predicting Dominance. This trend is also reflected in the mean correlation across all three affective dimensions, which ranges from -0.0157 to 0.3823 with a median of 0.316. This is lower than the mean correlation of Section 4.2, which ranged from 0.0545 to 0.385 with a median of 0.353. However, I suspected that this drop in performance would not persist when increasing seed word lexicon size – and this was partly confirmed by the results presented in Section 4.4.

Again using mean correlation across all affective dimensions as measure of overall model performance, the combination of FastText historical word embedding and linear

Model	Valence	Arousal	Dominance	Mean
kNN_CBOW	0.48	0.36	0.07†	0.3034
kNN_FastText	0.49	0.46	0.03†	0.3281
kNN_PPMI	0.51	0.38	0.00†	0.2966
kNN_SGNS	0.52	0.40	0.11†	0.3442
kNN_SVD_{PPMI}	0.46	0.42	0.01†	0.2973
PaRaSimNum_CBOW	0.22	0.22	0.04†	0.1591
PaRaSimNum_FastText	0.48	0.40	0.06†	0.3137
PaRaSimNum_PPMI	0.49	0.38	0.04†	0.3063
PaRaSimNum_SGNS	**0.54**	**0.40**	**0.10†**	**0.3468**
PaRaSimNum_SVD_{PPMI}	0.04†	-0.08†	-0.01†	-0.0157
RandomWalk_CBOW	0.44	0.38	0.04†	0.2861
RandomWalk_FastText	0.48	0.43	0.03†	0.3160
RandomWalk_PPMI	0.51	0.39	0.01†	0.3029
RandomWalk_SGNS	0.51	0.41	0.03†	0.3181
RandomWalk_SVD_{PPMI}	0.50	0.45	0.00†	0.3177
LinReg_CBOW	0.49	0.40	0.06†	0.3185
LinReg_FastText	**0.52**	**0.47**	**0.15†**	**0.3823**
LinReg_PPMI	0.47	0.45	0.05†	0.3233
LinReg_SGNS	**0.52**	**0.41**	**0.11†**	**0.3472**
LinReg_SVD_{PPMI}	0.43	0.49	0.05†	0.3251

Figure 4: Correlation between model predictions and the gold standard, measured by Pearson's *r*. The three highest-performing models are shown in bold. Entries marked with an asterisk (†) are not statistically significant ($p > 0.05$).

regression induction algorithm yields the best results, followed by the combinations of SGNS embedding with linear regression and PaRaSimNum. On average, models utilizing the SGNS embedding perform the best, followed by those using FastText. Models based on the PPMI, CBOW, or SVD_{PPMI} embedding perform well when paired with linear regression, but below average otherwise. Models utilizing linear regression perform the best by far, with five of the top eight models relying on this induction algorithm. In contrast, results for models employing kNN, RandomWalk, and PaRaSimNum are mixed, they often perform well with certain historical embeddings, typically SGNS, but tend to yield

Warriner-ANEW

	kNN	PaRaSimNum	RandomWalk	LinReg	Average
CBOW	15	19	18	5	14.25
FastText	1	13	6	2	5.5
PPMI	14	12	8	7	10.25
SGNS	9	4	11	3	6.75
SVD_{PPMI}	17	20	10	16	15.75
Average	11.2	13.6	10.6	6.6	

NRC-VAD-ANEW

	kNN	PaRaSimNum	RandomWalk	LinReg	Average
CBOW	14	19	18	8	14.75
FastText	5	12	11	1	7.25
PPMI	17	13	15	7	13
SGNS	4	3	9	2	4.5
SVD_{PPMI}	16	20	10	6	13
Average	11.2	13.4	12.6	4.8	

Figure 5 (above): Ranks of the different model configurations with Warriner-ANEW as seed word lexicon. Figure 6 (below): Ranks of the different model configurations with NRC-VAD-ANEW as seed word lexicon. Both figures include average ranks per embedding (last column) and per induction algorithm (last row).

lower performance on average. For a detailed comparison of the model rank distributions in Sections 4.2 and 4.3, see Figures 5 and 6.

4.4. Investigating Optimal Lexicon Size

I selected the three highest-performing model configurations from Sections 4.2 and 4.3, and proceeded as described above. For each lexicon, I created subsets ranging from 2,000 to 8,000 words (and up to 10,000 for NRC-VAD) with a step size of 1,000 words. To ensure that differences in performance were indeed a result of changes in lexicon size, I randomly generated 50 lexica for each size and seed word lexicon. Replacing Warriner-ANEW and NRC-VAD-ANEW with the corresponding randomly generated seed lexica, I evaluated each selected model's performance at a given lexicon size by computing the average score across all 50 runs for each affective dimension (see Figures 7 and 15).

Warriner					
Model	**Size**	**Valence**	**Arousal**	**Dominance**	**Mean**
kNN_FastText	2,000	0.43	0.32	0.17†	0.31
kNN_FastText	3,000	0.45	0.35	0.19†	0.33
kNN_FastText	4,000	0.47	0.36	0.21†	0.35
kNN_FastText	5,000	0.50	0.39	0.24	0.38
kNN_FastText	6,000	0.49	0.42	0.26	0.39
kNN_FastText	**7,000**	**0.49**	**0.43**	**0.27**	**0.40**
kNN_FastText	8,000	0.48	0.44	0.25	0.39
LinReg_FastText	2,000	0.49	0.38	0.23	0.37
LinReg_FastText	3,000	0.52	0.42	0.24	0.39
LinReg_FastText	4,000	0.53	0.43	0.24	0.40
LinReg_FastText	5,000	0.53	0.45	0.24	0.41
LinReg_FastText	6,000	0.54	0.45	0.24	0.41
LinReg_FastText	**7,000**	**0.54**	**0.46**	**0.25**	**0.42**
LinReg_FastText	**8,000**	**0.54**	**0.46**	**0.25**	**0.42**
LinReg_SGNS	2,000	0.51	0.38	0.25	0.38
LinReg_SGNS	3,000	0.52	0.38	0.27	0.39
LinReg_SGNS	4,000	0.53	0.40	0.29	0.41
LinReg_SGNS	5,000	0.53	0.41	0.30	0.41
LinReg_SGNS	6,000	0.54	0.42	0.30	0.42
LinReg_SGNS	7,000	0.54	0.42	0.31	0.42
LinReg_SGNS	**8,000**	**0.54**	**0.43**	**0.31**	**0.43**

Figure 7: Correlation between model predictions and the gold standard, measured by Pearson's *r* and averaged for each lexicon size. For each model configuration, its highest-performing variation is shown in bold. Entries marked with an asterisk (†) are not statistically significant ($p > 0.05$).

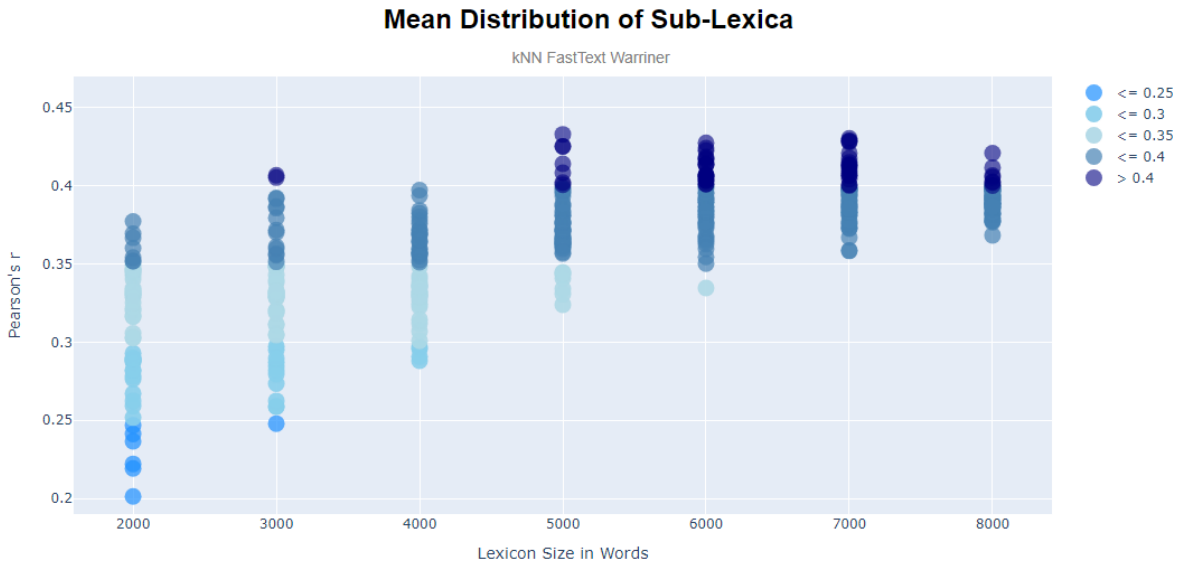
Concerning the models based on Warriner, increasing the size of the seed word lexica led to improved performance for all three models. Measured by the mean correlation across all three dimensions (Figure 8), the two models employing linear regression outperform the model based on the kNN algorithm, with the model using the SGNS embedding yielding slightly better results. This also holds true for the Valence dimension (Figure 9), which achieved the highest correlations values among all dimensions. In contrast, in the

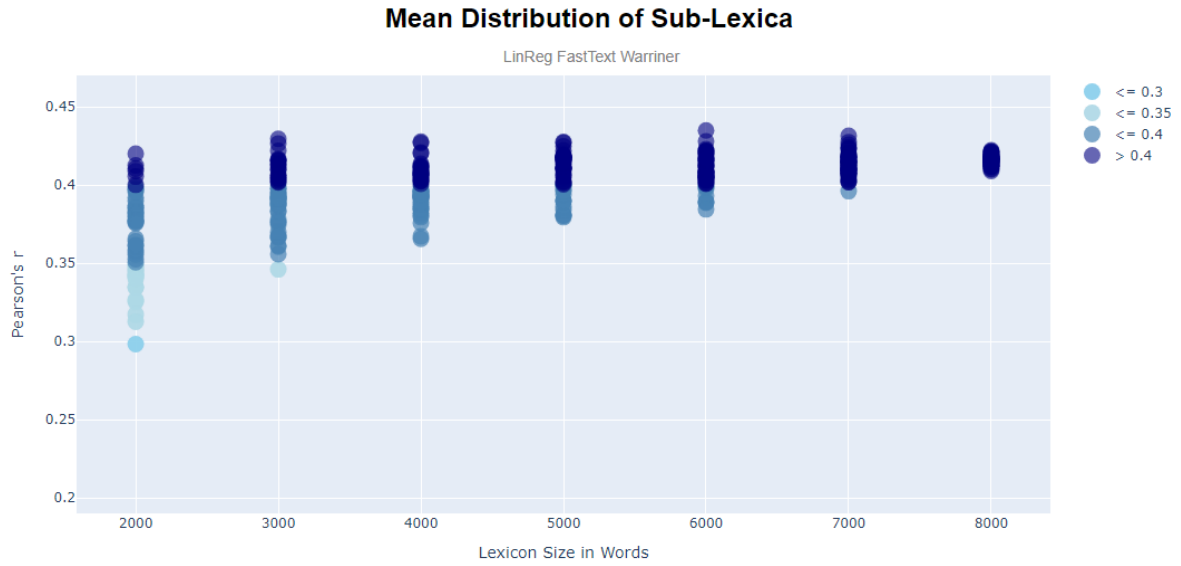
Figures 8 to 11 (from top to bottom): Performance development of the Warriner-based models with increasing lexicon size: mean over all three dimensions (Figure 8), Valence (Figure 9), Arousal (Figure 10), and Dominance (Figure 11).

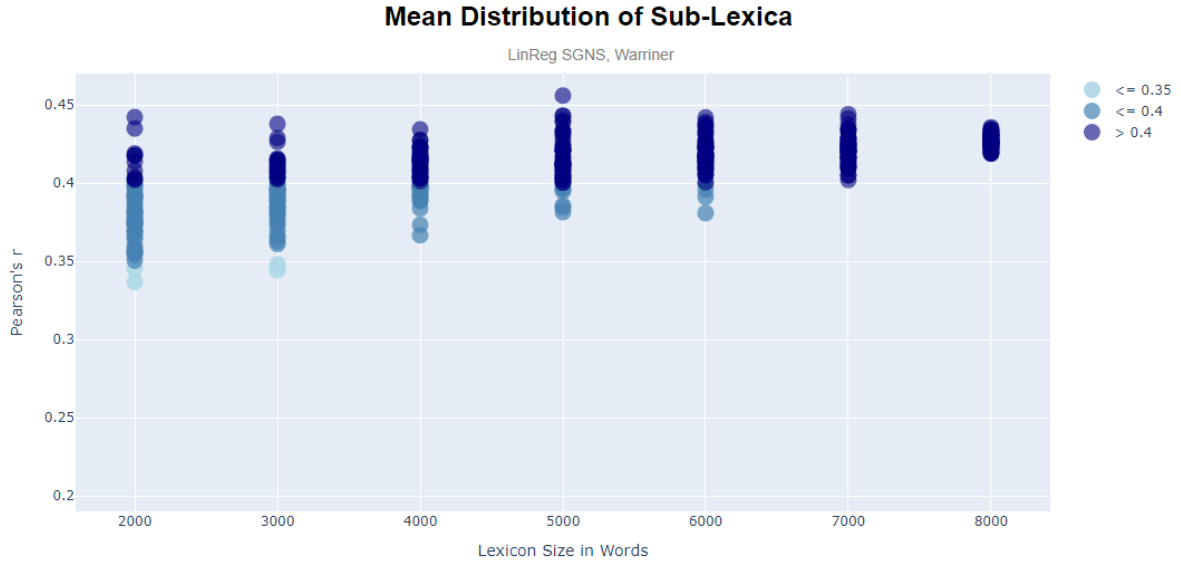
Arousal dimension (Figure 10), the model employing linear regression and the FastText embedding outperforms the other two models. Notably, this is the only dimension in which the model combining linear regression and the SGNS embedding yields lower results than the others, and only for lexicon sizes of 6,000 and above. Also worth noting is that the largest performance jump occurs in this dimension, specifically for the model combining the kNN algorithm with the FastText embedding, with correlation increasing by 0.12. In the Dominance dimension (Figure 11), all models yield by far the lowest correlation values, with the combination of linear regression and FastText performing considerably better than the others. This explains the overall advantage of this model in the mean performance across all three dimensions. Consistent with the findings in Sections 4.2 and 4.3, the variation in performance across the VAD dimensions persists, with Valence remaining the easiest to predict.

Considering that all values shown in Figure 7 and visualized in Figures 8 to 11 are means computed over the correlation values of the 50 sub-lexica, I also examined the variation within these sub-lexica, at least with respect to their mean correlation across all three dimensions (Figures 12 to 14). Notably, the performance increase observed in Figure 7 appears to result from the sub-lexica's mean correlation values converging as lexicon size grows. For all three models there are sub-lexica of size 2,000 or 3,000 that already achieve a mean correlation above 0.4. For the combination of kNN and FastText, a total of 53 out of 350 sub-lexica achieved a mean correlation above 0.4; for linear regression with FastText, 225 sub-lexica did; and for linear regression with SGNS, 250.

I investigated whether these 'successful' sub-lexica share a common set of words. This is not the case: neither overall nor within any single model is there even one word shared by all successful sub-lexica. However, for all three models, there are words that appear in more than 80% of the corresponding successful sub-lexica. When comparing, for each model, the 100 most frequent words across the successful sub-lexica, the overlap (Figure 22) between the model employing the kNN algorithm and the two models using linear regression is relatively small – 11 words for the FastText-based model and 9 for the SGNS-based model – while the overlap between the two linear regression models is much larger, with 51 shared words.







Figures 12 to 14 (from top to bottom): Distribution of sub-lexica performance (measured by mean correlation across all three dimensions) for the Warriner-based models: kNN-FastText model (Figure 12), LinReg-FastText model (Figure 13), and LinReg-SGNS model (Figure 14).

NRC-VAD

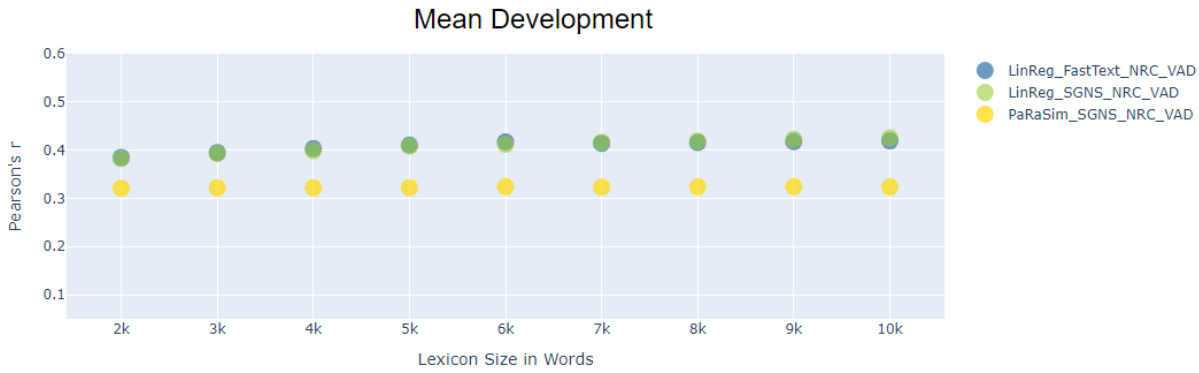
Model	Size	Valence	Arousal	Dominance	Mean
LinReg_FastText	2,000	0.54	0.41	0.21	0.39
LinReg_FastText	3,000	0.55	0.42	0.21	0.39
LinReg_FastText	4,000	0.56	0.43	0.22	0.40
LinReg_FastText	5,000	0.57	0.44	0.22	0.41
LinReg_FastText	6,000	0.58	0.45	0.22	0.42
LinReg_FastText	7,000	0.57	0.45	0.22	0.41
LinReg_FastText	8,000	0.58	0.45	0.22	0.42
LinReg_FastText	9,000	0.58	0.45	0.22	0.42
LinReg_FastText	**10,000**	**0.58**	**0.45**	**0.23**	**0.42**
LinReg_SGNS	2,000	0.54	0.43	0.18†	0.38
LinReg_SGNS	3,000	0.55	0.45	0.18†	0.39
LinReg_SGNS	4,000	0.55	0.46	0.19†	0.40
LinReg_SGNS	5,000	0.56	0.47	0.20†	0.41
LinReg_SGNS	6,000	0.56	0.48	0.20†	0.41
LinReg_SGNS	7,000	0.57	0.48	0.20	0.42
LinReg_SGNS	8,000	0.57	0.48	0.20	0.42
LinReg_SGNS	**9,000**	**0.57**	**0.49**	**0.21**	**0.42**
LinReg_SGNS	**10,000**	**0.57**	**0.49**	**0.21**	**0.42**
PaRaSimNum_SGNS	2,000	0.49	0.38	0.09†	0.32
PaRaSimNum_SGNS	3,000	0.49	0.38	0.09†	0.32
PaRaSimNum_SGNS	4,000	0.49	0.38	0.09†	0.32
PaRaSimNum_SGNS	5,000	0.50	0.38	0.09†	0.32
PaRaSimNum_SGNS	6,000	0.50	0.38	0.09†	0.32
PaRaSimNum_SGNS	7,000	0.50	0.38	0.09†	0.32
PaRaSimNum_SGNS	8,000	0.50	0.38	0.09†	0.32
PaRaSimNum_SGNS	9,000	0.50	0.38	0.09†	0.32
PaRaSimNum_SGNS	10,000	0.50	0.38	0.09†	0.32

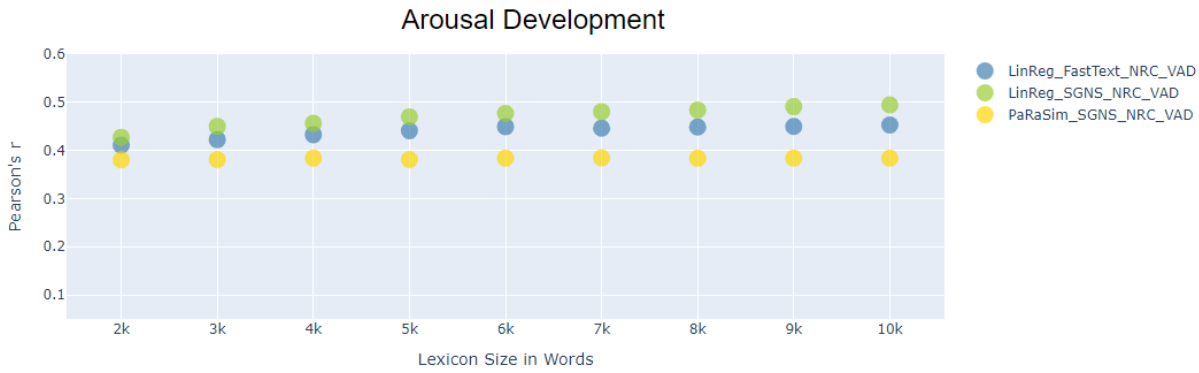
Figure 15: Correlation between model predictions and the gold standard, measured by Pearson's *r* and averaged for each lexicon size. For each model configuration, its highest-performing variation is shown in bold. Entries marked with an asterisk (†) are not statistically significant ($p > 0.05$).

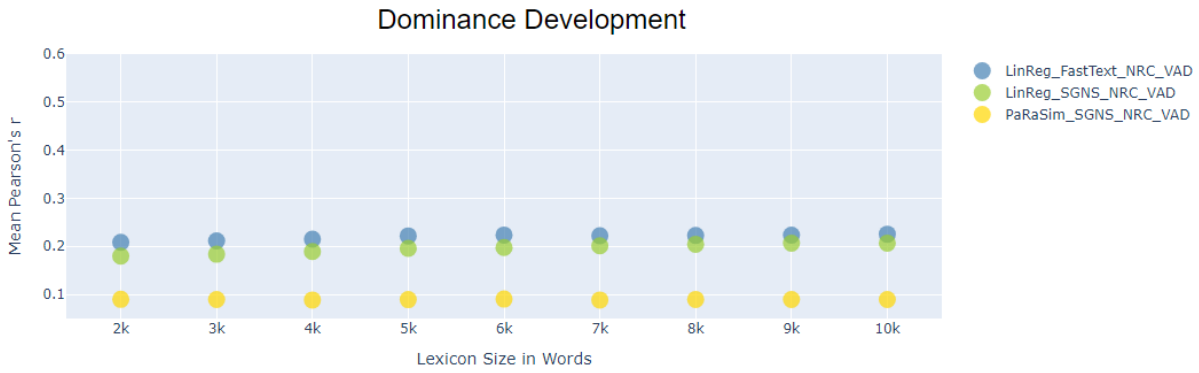
Concerning the models based on NRC-VAD (Figure 15), increasing the size of the seed word lexica led to improved performance in two of the three models – specifically, the combinations of linear regression with the FastText embedding and with the SGNS embedding. IN contrast, the third model, which combines the PaRaSim algorithm with the SGNS embedding, performed best when limited to NRC-VAD-ANEW (see Figure 4 for comparison). With increased lexicon size, correlation values initially declined across all three dimensions, and then either showed only minimal improvement (Valence) or remained stagnant (Arousal and Dominance).

Both when measured by the mean correlation across all three dimensions (Figure 16) and when considering the individual affective dimensions (Figures 17–19), the two models employing linear regression outperform the model based on the PaRaSim algorithm. The linear regression models yield almost identical results for both the mean correlation and the Valence dimension, with the model based on the SGNS embedding performing slightly better in the Arousal dimension and slightly worse in the Dominance dimension. Consistent with the findings in Sections 4.2 and 4.3, as well as earlier in this Section, the variation in performance across the VAD dimensions persists, with Valence remaining the easiest to predict. Note that, as suspected in Section 4.3, statistically significant correlations across all three dimensions were achieved – at least for both models employing linear regression: for the FastText-based model at all seed word lexicon sizes, and for the SGNS-based model at all seed lexica of size 7,000 and above. This does not hold true for the model employing the PaRaSim algorithm. Since this model never achieved statistical significance in the Dominance dimension, I have excluded it from any further discussion or analysis.

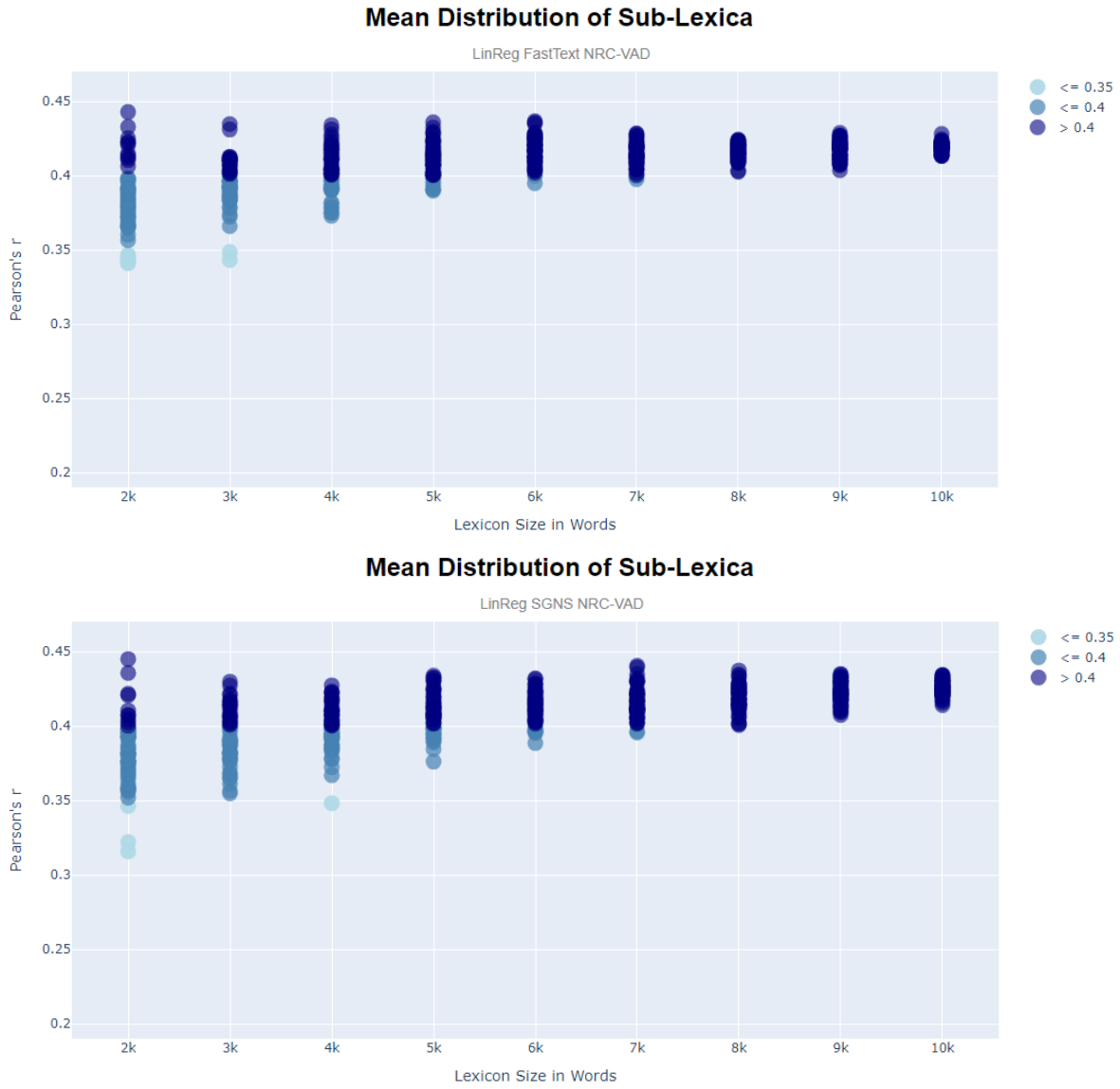
For the two remaining models, I investigated sub-lexica variation and found results similar to those of the Warriner-based models (Figures 20 and 21). The performance increase observed in Figure 15 again appears to result from the mean correlation values of the sub-lexica converging as lexicon size increases. As before, there are sub-lexica of size 2,000 that already achieve a mean correlation above 0.4 (nearly 0.45). Again, the successful sub-lexica – 348 (out of 450) for the FastText-based model and 333 for the SGNS-based model – do not share any common words, neither overall nor within a single







Figures 16 to 19 (from top to bottom): Performance development of the NRC-VAD-based models with increasing lexicon size: mean over all three dimensions (Figure 16), Valence (Figure 17), Arousal (Figure 18), and Dominance (Figure 19).



Figures 20 and 21: Distribution of sub-lexica performance (measured by mean correlation across all three dimensions) for the NRC-VAD-based models: LinReg-FastText model (top), SGNS-FastText model (bottom).

model. Notably, for both models, there are no words that appear in 80% of the successful sub-lexica. Nonetheless, when comparing the 100 most frequent words across the successful sub-lexica of each model, there is an overlap of 53 words (Figure 22).

In conclusion, for five of the six model configurations, increasing the size of the seed word lexicon led to improved model performance. Across all models, variation in performance between the VAD dimensions persisted, with Valence remaining the easiest to predict. Most models using larger seed lexica achieved statistically significant correlations across all three dimensions. The highest overall performance was achieved by the model

combining the SGNS historical embedding, the linear regression induction algorithm, and the largest version of the Warriner seed word lexicon. However, all five successful model configurations paired with the largest version of their corresponding seed word lexicon yielded similarly high results.

	kNN FastText Warriner	LinReg FastText Warriner	LinReg SGNS Warriner	LinReg FastText NRC-VAD	LinReg SGNS NRC-VAD
kNN FastText Warriner		11	9	1	0
LinReg FastText Warriner	11		51	0	1
LinReg SGNS Warriner	9	51		0	0
LinReg FastText NRC-VAD	1	0	0		53
LinReg SGNS NRC-VAD	0	1	0	53	

Figures 22: Overlap of 100 most frequent words across the successful sub-lexica of each model.

Considering sub-lexica variation, all five models examined show convergence with increasing lexicon size, and all have high-performing sub-lexica at sizes of 2,000 or 3,000. Notably, when comparing the 100 most frequent words across the successful sub-lexica of each model, the observed overlaps (Figure 22) appear to be driven more by the choice of induction algorithm and seed word lexicon than by any inherent properties of the words themselves. There is almost no overlap between Warriner-based and NRC-VAD-based models, while the two substantial overlaps occur between models that use the same seed word lexicon and the same induction algorithm. Hence, these findings do not – as one might have suspected – indicate the existence of a universal set of words that consistently enhances any model's performance when included in a (sub-)lexicon.

In conclusion, the outcomes described above indicate that, in general, performance improves with increasing lexicon size – regardless of whether the kNN algorithm or linear regression is employed, whether SGNS or FastText embedding is used, or whether the seed-word lexicon is based on Warriner or NRC-VAD. Additionally, it is possible to achieve equally high performance with much smaller seed word lexica, but this requires

additional testing to identify the optimal combination of induction algorithm and seed word lexicon.

Notably, these results suggest another interesting conclusion: There seems to be no such thing as “too much contemporary influence”. In general, the larger the seed word lexicon – that is, the greater the contemporary influence – the higher the model's performance. The broader implications of this finding will be discussed in the following chapter.

5. Discussion

The results of the previous chapter indicate that the greater the contemporary influence, the higher a model's performance. Combined with the fact that the correlations between both contemporary VAD lexica and the gold standard – 0.66 and 0.65 for Valence, 0.51 and 0.60 for Arousal, and 0.31 and 0.33 for Dominance, for Warriner and NRC-VAD respectively – are higher than any correlation values achieved in Chapter 4, this leads us to an interesting conundrum: While linguistic change is an indisputable fact, attempting to account for it in our *historical emotion analysis* produced worse results than simply applying contemporary *emotion analysis*. How can this apparent contradiction be accounted for?

First, we must consider the limitations of this experiment. Our training corpus – the 1830s section of the COHA – contains only 16 million words, which is relatively small by NLP standards. While the resulting word embeddings are capable of solving basic similarity and analogy tasks, they may lack the nuance necessary for *emotion analysis*. The gold standard lexicon poses another limitation for our experiment: it consists of only 100 words and may lack representativeness, as it was annotated by just two persons. While we are fortunate to have annotated historical data like this, it might simply be too small to serve as a reliable gold standard. Both more textual data for training and more annotated data for evaluation are needed to draw a more definitive conclusion. However, even if further experiments confirm my findings, they might only be indicative of that time period and its specific stage of the English language. I highly doubt they apply to earlier language stages, and any possible generalizability to another language would require further experiments.

Second, we must examine the limitations of our method. Combining a seed word lexicon with word embeddings to induce VAD scores has proven effective in contemporary settings, that is, when both the seed lexicon and embeddings are based on modern English (Hellrich et al., 2019; see Section 6.2). Correlations as high as 0.679 for Valence, 0.445 for Arousal and 0.574 for Dominance have been achieved. Combining a contemporary seed lexicon with historical embeddings yielded promising initial results, indicating that this adaptation of the contemporary approach might be a viable option for

historical emotion analysis. However, upon further investigation, my findings suggest otherwise. While I was able to improve model performance to some extent, the results consistently fell short of those achieved in the contemporary setting. In addition, contemporary *emotion analysis* outperforms the models developed in this study. It is possible that a model combining a historical VAD lexicon with historical word embeddings would yield better results. However, this would require substantially more historically annotated data – which is, at present, unavailable. Given our current resources, my results indicate that the method of *historical emotion analysis* proposed by Hellrich et al., and implemented in this paper, may not be a viable alternative.

Finally, we must examine the limitations of our underlying concept. The fact that even the contemporary models achieve correlation values no higher than 0.679 may indicate another limitation – namely, that our approach is word-based. Word-based models do not distinguish between different word senses; instead, they apply the same procedure regardless of context. In addition, it has been shown that in concreteness labeling tasks, annotators tend to consider only a word's most common sense (Reijnierse, 2019). If applicable to *emotion analysis*, these results indicate that word-based VAD lexica do not accurately represent the emotional meaning of a word's less common senses. As word-based approaches can oversimplify natural language and result in a non-negligible loss of information, accounting for a word's different senses, its tokens, may offer a valuable alternative.

For the task of *historical emotion analysis*, however, the token-based approach poses several challenges. Token-based models, that is, LLMs, require substantial amounts of both labeled and unlabeled training data, a requirement that is rarely met in historical contexts. To a lesser extent, this limitation also affects the fine-tuning of contemporary LLMs. Nevertheless, Wen and Xu (2022) successfully fine-tuned a BERT model on early 20th-century English. Note that this approach is only suitable for historical language stages that are not too distant from their modern counterparts. In an effort to circumvent these LLM-related issues – though issues related to evaluation data persist – alternative methods rely on computationally extracting labels from appropriate data. Tiessler et al. (2025, submitted), for example, combine extracted sense-level VAD scores with

diachronic sense proportions to compute historical VAD scores, though they must rely on the same gold standard as I do, which, as mentioned above, is not optimal.

In conclusion, whether the limitations of my model stem from its training or evaluation data, the method of its implementation, or its underlying concept, the challenges described in this section ultimately underscore the need for more historically annotated data, which is crucial for model evaluation and the advancement of robust methods for *historical emotion analysis*.

6. Conclusion

This thesis investigated the optimal setup of Hellrich et al.'s model for *historical emotion analysis* by comparing five word embedding algorithms – PPMI, SVD_{PPMI}, SGNS, CBOW, and FastText – combined with four induction algorithms – kNN, PaRaSimNum, Random Walk, and Linear Regression – and examining the effects of different seed word lexica, including Warriner VAD versus NRC-VAD, as well as lexicon size.

The highest performance is achieved by the model combining SGNS embedding, Linear Regression, and the largest version of the Warriner VAD lexicon. Models relying on the FastText embedding yield similarly high results, both when using larger versions of Warriner and NRC-VAD. Equally high performance can be achieved with much smaller seed word lexica, but identifying the optimal combination of induction algorithm and seed lexicon requires additional testing.

Notably, regardless of the choice of word embedding, induction algorithm, or seed word lexicon, model performance improves with increasing lexicon size, indicating that there is no such thing as "too much contemporary influence." Combined with the observation that even my improved models underperform compared to contemporary methods of *emotion analysis*, these findings indicate that this particular model offers limited utility for the historical period under investigation.

Ultimately, further research is required to clarify the sources of the model's limitations, to identify contexts in which it may still prove useful, and to determine suitable alternative methods when it does not. However, such research is severely constrained by the current lack of expertly annotated historical VAD data. The creation of such data is essential for both reliable evaluation and the development of more robust models for *historical emotion analysis* – models that play a critical role in advancing our understanding of historical texts and the diachronic development of emotional meaning.

Bibliography

Alberto Acerbi, Vasileios Lampos, Philip Garnett, and R. Alexander Bentley. 2013. The expression of emotions in 20th century books. In *PLoS ONE*, 8(3):e59030.

Davidson K. Aidam, Ben Benuwa, Stephen O. Oppong. 2024. Sentiment and emotion analysis using pretrained deep learning models. In *Journal of Data, Information and Management,* 6, 277–295.

Hani Al-Omari, Malak Abdullah, Samira Shaikh. 2020. EmoDet2: Emotion Detection in English Textual Dialogue using BERT and BiLSTM Models. 11th International Conference on Information and Communication Systems.

Cecilia Ovesdotter Alm and Richard Sproat. Emotional Sequencing and Development in Fairy Tales. In *Jianhua Tao, Tieniu Tan, and Rosalind W. Picard, editors, Affective Computing and Intelligent Interaction, Lecture Notes in Computer Science*, 668–674.

Anmol. 2023. Circumplex model of emotion. Own work. Licensed under CC BY-SA 4.0.

Alexander R. Bentley, Alberto Acerbi, Paul Ormerod, and Vasileios Lampos. 2014. Books average previous decade of economic misery. In *PLoS ONE*, 9(1):e83147.

Yves Bestgen. 2008. Building affective lexicons from specific corpora for automatic sentiment analysis. In *LREC 2008 — Proceedings of the 6th International Conference on Language Resources and Evaluation*, 496–500.

Johan Bollen, Alberto Pepe, and Huina Mao. 2011. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In *Proceedings of the International AAAI Conference on Web and Social Media*, *5*(1), 450-453.

Piotr Bojanowski, Edouard Grave, Armand Joulin, Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. In *Transactions of the Association for Computational Linguistics*, 8:135–146.

Laura-Ana-Maria Bostan and Roman Klinger. 2018. An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, 2104–2119.

Margaret M. Bradley and Peter J. Lang. 1994. Measuring emotion: The self-assessment manikin and the semantic differential. In *Journal of Behavior Therapy and Experimental Psychiatry*, 25(1):49–59.

Margaret M. Bradley and Peter J. Lang. 1999. Affective norms for English words (ANEW): Stimuli, instruction manual and affective ratings. Technical Report C-1, The Center for Research in Psychophysiology, University of Florida.

Sven Buechel, Johannes Hellrich, Udo Hahn. 2016b. Feelings from the past - Adapting affective lexicons for historical emotion analysis. In *Proceedings of the Workshop on Language Technology Resources and Tools for Digital Humanities*. 54–61.

Sven Buechel and Udo Hahn. 2017a. Emobank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*: Volume 2, Short Papers, 578–585.

Sven Buechel and Udo Hahn. 2017b. Readers vs. writers vs. texts: Coping with different perspectives of text understanding in emotion annotation. In *Proceedings of the 11th Linguistic Annotation Workshop*, 1–12.

Sven Buechel, Johannes Hellrich, Udo Hahn. 2017. The Course of Emotion in Three Centuries of German Text—A Methodological Framework. In *Digital Humanities*, 176-179.

Sven Buechel, Susanna Rücker, and Udo Hahn. 2020. Learning and Evaluating Emotion Lexicons for 91 Languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1202–1217.

John A. Bullinaria and Joseph P. Levy. 2007. Extracting semantic representations from word co-occurrence statistics: a computational study. In *Behavior Research* Methods, 39(3):510–526.

Andrea Chiorrini, Claudia Diamantini, Alex Mircoli, Domenico Potena. 2021. Emotion and sentiment analysis of tweets using BERT.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. In *Computational Linguistics*, 16(1): 22–29.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In: *arXiv preprint arXiv:2003.10555*

Kate Crawford. 2021. The Atlas of AI : Power, Politics, and the Planetary Costs of Artificial Intelligence. Yale University Press, chapter 5, “Affect”.

Charles Darwin. 1998. The expression of the emotions in man and animals. Oxford University Press, Oxford.

Mark Davies. 2012. Expanding horizons in historical linguistics with the 400-million word Corpus of Historical American English. In *Corpora*, 7:121–157.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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), 4171–4186.

Paul Ekman. 1992. An argument for basic emotions. In *Cognition & Emotion*, 6(3–4):169–200.

John Rupert Firth. 1957. Studies in Linguistic Analysis. Wiley-Blackwell, page 11.

Clémentine Fourrier and Syrielle Montariol. 2022. Caveats of measuring semantic change of cognates and borrowings using multilingual word embeddings. In *Proceedings of the 3rd Workshop on Computational Approaches to Historical Language Change*, 97–112.

Michel Genereux and Roger Evans. 2006. Distinguishing affective states in weblogs. In: *AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs*, 27–29.

Mario Giulianelli, Marco Del Tredici, and Raquel Fernandez. 2020. Analysing Lexical Semantic Change with Contextualised Word Representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3960–3973.

Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In *Proc. GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics*, 67–71.

William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016a. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *EMNLP 2016*, 595–605.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016b. Diachronic word embeddings reveal statistical laws of semantic change. In *ACL 2016, Long Papers*, 1489–1501.

Johannes Hellrich, Sven Buechel, and Udo Hahn. 2019. Modeling Word Emotion in Historical Language: Quantity Beats Supposed Stability in Seed Word Selection. In *Proceedings of the 3rd Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, 1–11.

Ryan Heuser, Franco Moretti, Erik Steiner. 2016. The emotions of London. In *Stanford Literary Lab Pamphlets* 13.

Arthur E. Hoerl, Robert W. Kennard. 1970. "Ridge Regression: Biased Estimation for Nonorthogonal Problems". Technometrics. 12 (1): 55–67.

Lars Holzman and William Pottenger. 2003. Classification of Emotions in Internet Chat: An Application of Machine Learning Using Speech Phonemes. Technical Report. Leigh University.

Rohan Kamath, Arpan Ghoshal, Sivaraman Eswaran and Prasad Honnavalli. 2022. An Enhanced Context-based Emotion Detection Model using RoBERTa. In *IEEE International Conference on Electronics, Computing and Communication Technologies*, 1-6.

Elsa Kim, Sam Gilbert, Michael J. Edwards, Erhardt Graef. 2009. Detecting sadness in 140 characters: Sentiment analysis of mourning Michael Jackson on twitter. Tech. Rep. 3, Web Ecology Project.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2014. Statistically significant detection of linguistic change. In *Proc. WWW*, pages 625–635.

Adam Jatowt and Kevin Duh. 2014. A framework for analyzing semantic change of words across time. In *JCDL 2014*, 229–238.

Thomas K. Landauer, and Susan T. Dumais. 1997. A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge. In *Psychological Review*, 104: 211–240.

Inger Leemans, Janneke M. van der Zwaan, Isa Maks, Erika Kuijpers, Kristine Steenbergh. 2017. Mining embodied emotions: a comparative analysis of sentiment and emotion in dutch texts, 1600–1800. In *Digital Humanities Quaterly*, 11, H. 4.

Alessandro Lenci, Magnus Sahlgren, Patrick Jeuniaux, Amaru Cuba Gyllensten, Martina Miliani. 2022. A comparative evaluation and analysis of three generations of Distributional Semantic Models. In *Lang Resources & Evaluation,* 56, 1269–1313.

Nicolas Leveau, Sandra Jhean-Larose, Guy Denhière, and Ba-Linh Nguyen. 2012. Validating an interlingual metanorm for emotional analysis of texts. In *Behavior Research Methods*, 44(4):1007–1014.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. In *Transactions of the Association for Computational Linguistics*, 3:211– 225.

Minglei Li, Qin Lu, Yunfei Long, and Lin Gui. 2017. Inferring Affective Meanings of Words from Word Embedding. In *IEEE Transactions on Affective Computing*, 8(4):443-456.

Bastien Liétard, Mikaela Keller, Pascal Denis. 2023. A Tale of Two Laws of Semantic Change: Predicting Synonym Changes with Distributional Semantic Models.

Yinhan Liu, 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* abs/1907.11692: n. pag.

Chunling Ma, Helmut Prendinger, and Mitsuru Ishizuka. 2005. Emotion estimation and reasoning based on affective textual interaction. First International Conference on Affective Computing and Intelligent Interaction, 622–628.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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)*, 2227–2237.

Rosa Meo and Emilio Sulis. 2017. Processing Affect in Social Media: A Comparison of Methods to Distinguish Emotions in Tweets. In *ACM Transactions on Internet Technology*, 17(1):7:1–7:25.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, William Brockman, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. 2010. Quantitative Analysis of Culture Using Millions of Digitized Books. In *Science* (Published online ahead of print: 12/16/2010).

Rada Mihalcea and Hugo Liu. 2006. A corpus-based approach to finding happiness. In *AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs*, 139–144.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations*.

Saif M. Mohammad. 2011. From Once Upon a Time to Happily Ever After: Tracking Emotions in Novels and Fairy Tales. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, 105–114.

Saif M. Mohammad and Tony (Wenda) Yang. 2011. Tracking sentiment in mail: how genders differ on emotional axes. In *Proceedings of the ACL 2011 Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, 70-79.

Saif M. Mohammad, 2012b. #emotional tweets. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, 246–255.

Saif M. Mohammad, Xiaodan Zhu, and Joel Martin. 2014. Semantic Role Labeling of Emotions in Tweets. In *Proceedings of the ACL 2014 Workshop on Computational Approaches to Subjectivity, Sentiment, and Social Media*, 32-41.

Saif M. Mohammad, Svetlana Kiritchenko, Xiaodan Zhu, and Joel Martin. 2015. Sentiment, Emotion, Purpose, and Style in Electoral Tweets. In *Information Processing and Management*, 51 (4): 480–499.

Saif M. Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. 2018. Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*,1-17.

Saif Mohammad. 2018. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *ACL 2018 — Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, volume 1*: Long Papers, 174–184.

Kristoffer Nielbo, Folgert Karsdorp, Melvin Wevers, Alie Lassche, Rebekah Baglini, Mike Kestemont, Nina Tahmasebi. 2024. Quantitative text analysis. In *Nature Reviews Methods Primers,* 4(1), 25.

James Pennebaker, Ryan L. Boyd, Kayla Jordan, Kate Blackburn. 2015. The development and psychometric properties of LIWC2015. Austin, TX: Univ. of Texas.

Robert, Plutchik. 2001. The nature of emotions. In *American Scientist*, 89(4): 344–350.

Jonathan Posner, James Russell, Bradley Peterson. 2005. The circumplex model of affect: An integrative approach to affective neuroscience, cognitive development, and psychopathology. In *Development and Psychopathology*, 17 (3): 715–734.

Daniel Preoţiuc-Pietro, Johannes Eichstaedt, Gregory Park, Maarten Sap, Laura Smith, Victoria Tobolsky, H. Andrew Schwartz, and Lyle Ungar. 2015. The role of personality, age, and gender in tweeting about mental illness. In *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, 21–30.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50.

Gudrun Reijnierse, Christian Burgers, Marianna Bolognesi, Tina Krennmayr. 2019.How polysemy affects concreteness ratings: The case of metaphor. In *Cognitive Science*, 43(8), e12779

Julia Rodina and Andrey Kutuzov. 2020. RuSemShift: a dataset of historical lexical semantic change in Russian. In *Proceedings of the 28th International Conference on Computational Linguistics*, 1037– 1047.

Alex Rosenfeld and Katrin Erk. 2018. Deep Neural Models of Semantic Shift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1 (Long Papers), 474–484.

James Russell. 1980. A circumplex model of affect. In *Journal of Personality and Social Psychology*, 39 (6): 1161–1178.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2011. Tracing semantic change with latent semantic analysis. In *Current Methods in Historical Semantics*, page 161.

Kashfia Sailunaz and Reda Alhajj. 2019. Emotion and sentiment analysis from Twitter text. In *Journal of Computational Science*, Volume 36, 101003.

Thomas Schmidt, Manuel Burghardt, Katrin Dennerlein. 2018b. Sentiment Annotation of Historic German Plays: An Empirical Study on Annotation Behavior. In *Proceedings of the Workshop on Annotation in Digital Humanities*, 47-52.

Thomas Schmidt, Katrin Dennerlein, Christian Wolff. 2021b. Emotion Classification in German Plays with Transformer-based Language Models Pretrained on Historical and Contemporary Language. *In Proceedings of the 5th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, 67-79.

Thomas Schmidt, Katrin Dennerlein, Christian Wolff. 2021c. Towards a Corpus of Historical German Plays with Emotion Annotations. In *3rd Conference on Language, Data and Knowledge* (LDK 2021).

Thomas Schmidt, Katrin Dennerlein, Christian Wolff. 2021c. Using Deep Neural Networks for Emotion Analysis of 18th and 19th century German Plays. In *Fabrikation von Erkenntnis: Experimente in den Digital Humanities*.

Thomas Schmidt, Katrin Dennerlein, Christian Wolff. 2022. Evaluation computergestützter Verfahren der Emotionsklassifikation für deutschsprachige Dramen um 1800. In *DHd 2022 Kulturen des digitalen Gedächtnisses. 8. Tagung des Verbands "Digital Humanities im deutschsprachigen Raum"*.

David S. Schmidtke, Tobias Schröder, Arthur M. Jacobs, and Markus Conrad. 2014. ANGST: Affective norms for German sentiment terms, derived from the affective norms for English words. In *Behavior Research Methods*, 46(4):1108–1118.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, 1–23.

Dominik Schlechtweg, Frank D. Zamora-Reina, Felipe Bravo-Marquez, Nikolay Arefyev. 2025. Sense through time: diachronic word sense annotations for word sense induction and Lexical Semantic Change Detection. In: *Language Resources and Evaluation*, 59(2): 1431–1465.

Gilbert W. Stewart: On the early history of the singular value decomposition. 1993. In *SIAM Review*. 35 (4), 551–566.

Carlo Strapparava and Alessandro Valitutti. 2004. Wordnet-affect: an affective extension of wordnet. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, 1413–1418.

Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *Proceedings of the Fourth International Workshop on Semantic Evaluations* (SemEval-2007), 70–74.

Silvia Stopponi, Nilo Pedrazzini, Saskia Peels-Matthey, Barbara McGillivray, Malvina Nissim. 2024. Natural Language Processing for Ancient Greek: Design, advantages and challenges of language models. In *Diachronica*, 41(1).

Jared Suttles and Nancy Ide. 2013. Distant Supervision for Emotion Classification with Discrete Binary Values. In *Computational Linguistics and Intelligent Text Processing*, 7817. In *Lecture Notes in Computer Science*, 121–136.

Nina Tahmasebi, Lars Borin, Adam Jatowt. 2021. Computational approaches to semantic change. Language Science Press.

Max Tiessler, Joaquim Motger, Fiorina Piroi, Andreas Baumann. 2025 (Submitted) . EmoTracker - A New Framework for Modeling and Forecasting Diachronic Emotion Dynamics.

Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. In *ACM Transactions on Information Systems*, 21(4):315–346.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. In *Journal of Artificial Intelligence Research*, 37(1):141–188.

Krishnapriya Vishnubhotla and Saif M. Mohammad. 2022. Tweet Emotion Dynamics: Emotion Word Usage in Tweets from US and Canada. In *Proceedings of the 13th Language Resources and Evaluation Conference*, 4162–4176.

Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, C.-C. Jay Kuo. 2019. Evaluating word embedding models: methods and experimental results. In *APSIPA Transactions on Signal and Information Processing*. 8.10.

Amy Beth Warriner, Victor Kuperman, and Marc Brysbært. 2013. Norms of valence, arousal, and dominance for 13,915 English lemmas. In *Behavior Research Methods*, 45(4):1191–1207.

Qiu Wen and Yang Xu. 2022. HistBERT: A Pre-trained Language Model for Diachronic Lexical Semantic Analysis. In *ArXiv* abs/2202.03612: n. pag.

Derry Tanti Wijaya and Reyyan Yeniterzi. 2011. Understanding semantic change of words over centuries. In *Proc. Workshop on Detecting and Exploiting Cultural Diversity on the Social Web*, 35– 40.

Eman M. G. Younis, Someya Mohsen, Essam H. Houssein, Osman Ali Sadek Ibrahim. 2024. Machine learning for human emotion recognition: a comprehensive review. In *Neural Computing and Applications, 36*, 8901–8947.

Xu Zhe and Anthony Boucouvalas. 2002. Text-to-Emotion Engine for Real Time Internet Communication. In *International Journal of Communication Systems*, pp. 164–168.

Denny Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *NIPS 2004*, 321–328.

Abstract

This master's thesis investigates potential extensions of the model for historical emotion analysis proposed by Hellrich et al. (2019). For this purpose, the model – based on the combination of historical word embedding, a VAD lexicon, and an induction algorithm – is extended by three word embeddings (PPMI, CBOW, FastText), an induction algorithm (linear regression), and a VAD lexicon (NRC-VAD). In addition, the influence of lexicon size is examined. Model performance can be successfully improved: the best model combines SGNS embedding, linear regression as induction algorithm, and the largest version of the Warriner VAD lexicon used in the original model. However, the results indicate that a stronger contemporary influence has a positive effect on historical emotion analysis. Possible causes for this contradictory outcome are discussed.

Abstract

Die vorliegende Masterarbeit untersucht Erweiterungsmöglichkeiten des von Hellrich et al. (2019) vorgelegten Modells zur historischen Emotionsanalyse. Zu diesem Zweck wird das aus der Kombination von historischem Word Embedding, VAD Lexikon und Induktionsalgorithmus bestehende Modell um drei Word Embeddings (PPMI, CBOW, FastText), einen Induktionsalgorithmus (lineare Regression) und ein VAD Lexikon (NRC-VAD) erweitert. Zudem wird der Einfluss von Lexikongröße untersucht. Modellleistung kann erfolgreich gesteigert werden: das beste Modell kombiniert SGNS Embedding, lineare Regression als Induktionsalgorithmus und die größte Version des im ursprünglichen Modell verwendeten Warriner VAD Lexikons. Die resultierenden Daten zeigen allerdings, dass stärkerer zeitgenössischer Einfluss sich positiv auf die historische Emotionsanalyse auswirkt. Mögliche Ursachen für dieses widersprüchliche Ergebnis werden untersucht.