

FRIEDRICH-ALEXANDER UNIVERSITÄT
ERLANGEN-NÜRNBERG

MASTER'S THESIS IN COMPUTER SCIENCE

Automated Encoding of Motives in Psychological Picture Story Exercises

Author:

Tilman ADLER
born 16.12.1989 in Nürnberg

Supervisors:

Dipl.-Inf. M. GROPP,
Prof. Dr.-Ing. habil. A. MAIER,
Prof. Dr. S. EVERT,
Prof. Dr. O. SCHULTHEISS

Started:

15.02.2017

Finished:

15.08.2017

Written at:

Chair for Pattern Recognition (CS 5)
Department of Computer Science
FAU Erlangen-Nürnberg



FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien des Lehrstuhls für Studien- und Diplomarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Erlangen, den 13. August 2017

Übersicht

In dieser Arbeit untersuchen wir, inwiefern Word Embeddings genutzt werden können, Texte aus psychologischen Bildgeschichtentests danach zu klassifizieren, ob in deren Autoren zuvor eine bestimmte implizite Motivation angeregt wurde oder nicht. Dazu vergleichen wir verschiedene Embeddings sowie einen Trainingskorpus, der vom Schreibstil den klassifizierten Geschichten gleicht, mit einem von Wikipediaartikeln und finden keine Vorteile. Wir erkennen außerdem keinen Vorteil darin, durch das Retrofitting von Synonymen aus zwei verschiedenen Synonymdatenbanken und das Erstellen eines ultradichten Embeddings bereits trainierte Word Embeddings zu verbessern. Wir betrachten ferner, ob das Gruppieren von Worten vor dem Erstellen eines Merkmalsvektors vorteilhaft für die Klassifizierung ist und verneinen dies. Durch Aufsummieren und Normalisieren berechnen wir einen Merkmalsvektor und sehen Erfolge darin, sowohl Stopworte zu entfernen sowie die Euklidische Norm zur Normalisierung zu verwenden. Die Ergebnisse der Klassifizierung dieses Vektors mit einer Support Vector Machine werden mit denen eines etablierten, manuellen Systems verglichen und können dieses an Genauigkeit übertreffen, wenn das Training auf dem gleichen Datensatz geschieht. Als Klassifizierer können Random Forest oder k -Nearest Neighbor mit der Word Mover's Distance diese nicht übertreffen.

Abstract

In this thesis we research how word embeddings can be used to classify texts from psychological picture story exercises regarding the separation of stories, in whose authors implicit motivation has either been aroused or not. For that different word embeddings are compared as well as different corpora to train them. We compare a corpus that is similar in writing style to the classified stories and one of Wikipedia articles and find no advantages. We neither find an advantage in improving word embeddings by retrofitting synonym information of two different synonym databases and by creating an ultradense word embedding. We also consider if clustering of words before creating a feature vector is beneficial for the classification results and negate that. We calculate a feature vector by summing and normalizing and find benefits in using stop word removal and the Euclidean norm for normalization. The results of classifying this vector using a support vector machine are compared to an established system for manual coding and outperform it in accuracy when trained on the same dataset. As classifiers random forest or k -nearest neighbor with the word mover's distance cannot outperform it.

Contents

1	Introduction	1
1.1	Implicit Motivation and Picture Story Exercise	1
1.2	Prior Work	3
1.2.1	On Coding by Human Experts	3
1.2.2	On Coding in an Automated Fashion	3
1.3	Research Goals	5
1.3.1	Motive Induction Studies	5
1.3.2	The “Bottom-Up” Approach	6
2	Datasets	7
2.1	Atkinson et al. 1954	8
2.2	McAdams 1980	8
2.3	McClelland et al. 1949	9
2.4	Veroff 1957	9
2.5	Winter 1973	10
2.6	Wirth and Schultheiss 2006	10
3	Methods	13
3.1	Word Embeddings and Modifications to Improve Them	13
3.1.1	Word2vec	14
3.1.2	GloVe	14
3.1.3	Retrofitting	15
3.1.4	Ultra Dense Word Embeddings	16
3.2	Word Embedding Training and General Preprocessing	17
3.2.1	Corpora and Training	17
3.2.2	Preprocessing	19

3.3	Classifiers and Features Tested	20
3.3.1	Bag of Words	20
3.3.2	Word Mover's Distance	20
3.3.3	Neural Network Based Approaches	21
3.3.4	Centroid Vector	23
3.3.5	Clustered Centroid Vectors	26
3.3.6	Classifiers Employed	30
4	Results	33
4.1	Training and Validation Procedure	33
4.2	Baselines	35
4.2.1	Winter Coding System	35
4.2.2	Bag of Words	37
4.3	Results for Individual Techniques	38
4.3.1	Word Mover's Distance	38
4.3.2	Centroid Vector and SVM	39
4.3.3	Centroid Vector and Random Forest with Ultradense Word Embeddings .	46
4.3.4	Clustered Centroids	48
4.3.5	Agglomerative Clustering by Cosine Dissimilarity	50
4.4	Comparison to Baselines	51
4.4.1	Accuracy per Story	51
4.4.2	Accuracy per Author	51
4.5	Results Across Datasets	53
5	Discussion	55
5.1	Remaining Issues	55
5.1.1	Low Accuracy	55
5.1.2	Possible Side Channels	57
5.1.3	Cross Study Validity	58
5.2	Word Embedding Training Corpus	59
6	Conclusion and Outlook	61
6.1	Our Contribution	61
6.2	Future Work	62
A	Example of a PSE Story	63

<i>CONTENTS</i>	ix
B Code Listing	65
C Larger Versions of Figures	67
List of Figures	73
List of Tables	75
Glossaries and Notation	77
Bibliography	79

Chapter 1

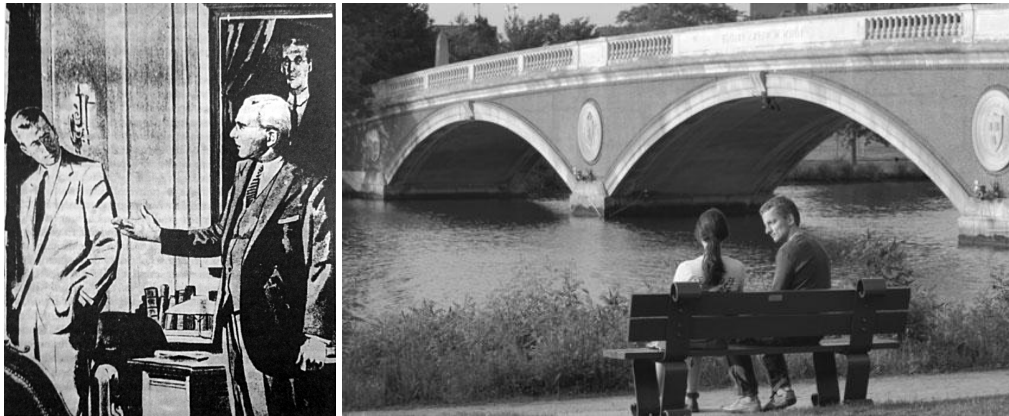
Introduction

The question of what animates us in our daily lives is an ancient one. There have been many explanations over hundreds of years – ranging from “the gods” of the ancients to Freud’s ideas of an unconscious driver in our mind [Fre89]. While in science the concept of “the gods” controlling our fate does not play a role any more, unconscious motivation is being researched actively. Today, Freud’s initial thoughts on psychoanalysis – colloquially known as “the talking cure” – have been challenged (at least partly) and developed further. Nevertheless, language is still the instrument of choice for taking a glimpse at the subconscious mind. The *picture story exercise (PSE)* does not involve talking but verbal communication in form of texts and those can be viewed from a natural language processing point of view.

1.1 Implicit Motivation and Picture Story Exercise

Before detailing how PSEs work, though, let us take a closer look at *implicit motivation*. The term itself stems from the observation that while people often *explicitly* state one thing in reality they act against their own statements. When plain deceptiveness cannot function as an explanation, it follows that people are *implicitly* guided by motives and values which they might not even be aware of [Sch10]. A person may therefore think that they derive pleasure from something, but the reality might be different. This can lead to unhappy and unfulfilled lives [Wei10a].

The observation of this disparity led to the conclusion that simply employing a questionnaire or interview to examine the motivation of a person is not going to reveal the truth reliably. Therefore McClelland and Atkinson were the first to perform an experiment in this context in



(a) Two men talking (b) A man and a woman sitting on a park bench by a river

Figure 1.1: A typical TAT and PSE image [Rhe08, Sch13a]

which subjects were shown images and were asked to write stories [Sch10]. At the time this was a *thematic apperception test (TAT)* which was later developed further to *picture story exercise*.

Both work in a similar fashion. A test person is shown a series of images, usually showing people in various situations and often interacting with each other. Depending on the setup, the image can be shown during the whole test or just a couple of seconds [Sch07]. Two exemplary images can be seen in fig. 1.1. The person has been given the instruction to write a short story inspired by the pictures. As the depictions are inconclusive, they can be interpreted in a variety of ways. Within a fixed time, usually around five minutes, the test subjects write stories of about 100 words. The proponents of employing TATs and PSEs for motivational research claim that unconscious states of mind are then reflected within the texts without the subject being aware of that [Rhe08].

The texts are finally analyzed by a specially trained psychologist in accordance with a manual. This process is called coding and ensures that results do not vary a lot between different coders. An example of a story written for fig. 1.1b and corresponding coding results can be read in appendix A. Coder training takes about 20 hours of work [Rin17]. Its goal is to reach about 85% consistency with expert coders. To further improve reliability often more than one coder will analyze a text. Usually they do produce similar results on the same story [Sch08].

The employment of human coders creates two problems, though. The first is an economic one: Human time costs money. This problem is amplified as one cannot simply use untrained laborers and optimally more than one coder is needed. For a typical research study about 16 to 40 hours of coding must be invested [Sch07]. That is not counting the training of coders and evaluation of results. Effectively this hinders progress in the field as researchers cannot afford

coding of large studies. The second problem is of academic and systemic nature. Even though inter-scorer reliability (i. e. correlation of coded motives) is relatively high when following suitable manuals and training coders well [Sch08], humans are not machines and therefore susceptible to (at least occasional) errors. This might introduce irregularities in tests.

Both problems could be solved by employing machine learning techniques. Cost would go down simply because less humans are required. And though computer programs are not perfect either, at least when they fail, they do so reliably and reproducibly.

1.2 Prior Work

The work on the subject can be viewed from two angles: The psychologist's and the computer scientist's angle. Let us first take a brief look on how psychologists have researched motivation coding in the past.

1.2.1 On Coding by Human Experts

The first problem, that of time consumption, has been tackled before. After McClelland et al. presented their findings in their book "The achievement motive" in 1953 [McC53], a number of other implicit motives have been researched. Usually a motive has been established in conjunction with a coding scheme for that motive. For example, Veroff developed a scoring system for need for power in 1957 [Ver57].

In 1991, though, David Winter proposed a coding system which can be used to code three motives at once: the achievement motive (*nAch*), the affiliation-intimacy motive (*nAff*) and the power motive (*nPow*) [Win91]. Of course, this saves a lot of time, as texts could now be scored in one pass, whereas before, scoring for these three motives would take three passes. Additionally coders need to be trained in only one scoring system instead of three. Since then, the Winter coding system has become very widespread and has replaced other, "original" systems for specific motives [Wei10b]. Therefore its discriminatory power is suitable as a baseline in our tests as we will explain in section 4.2.1.

1.2.2 On Coding in an Automated Fashion

While implicit motives and coding systems have been an active area of research in the last 60 years, little insight has been sought on how the coding process could be automated. Partly this is due to computers not being powerful enough for the task, of course. But since a couple

of years the question of how computers could be used to replace coders has been gaining traction.

The first experiments in the 1960s using only a couple of role-based word associations were not continued. They succeeded on older stories but could not be generalized [Hal15].

The research focus subsequently shifted to the marker-word hypothesis. According to this theory, stories can be coded only by looking at certain, meaningful words. Automated coding therefore involves manual creation of dictionaries, counting word co-occurrences and assigning words to categories [Bla10]. A comprehensive example of this is a software called “Linguistic Inquiry and Word Count” (LIWC). Words are categorized into one or more category per word and categories can have sub-categories, giving a broad overview of a text. Even though this software was not designed to be used to tap into TAT stories, Pennebaker and King could show a significant correlation of -0.33 of a LIWC factor and the achievement motive as scored by human scorers. No significant correlations were shown for any LIWC factors and *nAff* and *nPow* scored by experts in TAT stories [Pen99].

Similar research was performed by Schultheiss in 2013, using an updated version of LIWC. His regression analysis was based on LIWC category word count normalized by story length. He, too, found categories with weak correlations for *nAff* but also for *nPow* and *nAch*. When combining several categories by weighting, he could show higher correlations. Even further, he could demonstrate causal validity of his LIWC-derived score. To do so, he analyzed stories written by participants in a study where a specific motive was aroused, i. e. high motivation of one kind is artificially induced. We will describe this process later in section 1.3.1, but for now it suffices to know that this way the validity of a scoring system can be tested. His score changed with the arousal condition, indicating that it actually reflects changes in motivation [Sch13a].

For his PhD Halusic tried to create an automatic coding system for the implicit achievement motive. He compiled a list of synsets (a set of synonyms that can be used interchangeably in a context) possibly indicating achievement imagery from PSE stories that had been previously scored by humans. Then he computed the vector of the maximum relatedness of all words of each sentence of a story to these synsets, a process he calls Maximum Synset-to-Sentence Relatedness. This vector was finally classified by a neural network as containing *nAch* imagery or not. He achieved accuracy of about 80% - 83%, “but investigations of reliability between predicted values and human-coded values indicated that the model did not reach a level sufficient to consider the automated coding to be interchangeable with the human coding” [Hal15].

Ring and Pang are currently working on automating the coding process. They, too, are trying to predict presence or absence of motive imagery in PSE texts. They use a bag of words

after part-of-speech tagging (POS) on stories previously scored by humans. According to them, applying POS improves classification quality by about 4%. They reach about 68%, 84% and 76% accuracy on *nPow*, *nAch* and *nAff*, respectively [Rin17].

1.3 Research Goals

To our knowledge, nobody (except for the very recent, just described research by Halusic and Ring and Pang) has applied machine learning techniques to the problem. While their results look quite promising, both have taken a “top-down” approach. That is, they are using stories that have been coded by an expert as training data. They then try to replicate the findings of the expert by predicting absence or presence of coded imagery. This thesis, on the other hand, is trying to approach the problem “bottom-up”. To understand the difference we must take a step back and take a look at motive induction studies.

1.3.1 Motive Induction Studies

When McClelland and Atkinson conducted their studies on motivation they had two groups of test subjects: One was food deprived, the other one was not. As we said before they looked for a way to assess motivation changes without employing a questionnaire and from this stems the use of TAT and PSE for the assessment of implicit motivation. When they analyzed the stories written by the two groups, they found that they wrote about different themes [Sch10].

Usually these studies involve two groups of test subjects, one where the motive in question is artificially aroused (i. e. heightened) and one control group. This can be done in different ways. Veroff, when he researched the need for power, had stories written by students running for an office just before the announcement of poll results [Ver57]. Schultheiss showed participants either parts of the movie “The Godfather II”, “Bridges of Madison County” (a romantic movie) or a documentary to arouse, respectively, need for power, need for affiliation or no motive at all [Sch13a].

This kind of study was subsequently repeated when an implicit motive was researched. Winter describes the process of coming up with a coding system from stories written in a motive induction study in his book “The power motive”. He analyzed stories and extracted themes from all of them grouped by picture. Then he looked for themes showing up in stories written for different images, refining and broadening the definition of a theme. The presence of themes is then grouped into categories and formulated as a coding scheme. These schemes are then tested by having test coders code the stories blindly to the arousal condition. Should a

coding scheme, that was deduced in this way, not pass the blinded test, i. e. if a coder cannot differentiate between the two groups simply by applying the coding system, it is refined and re-formulated [Win73].

Assuming the arousal of a specific motive is effective, results should differ between the aroused and the non-aroused group. There are some problems with this study setup, though: Every participant in such a study is by nature already motivated in some way. If, by chance, someone with a high *nAff* took part in an affiliation arousal study and were part of the control group, he or she will decrease the average difference between the aroused and non aroused group. But *on average* the aroused group should display higher motivational needs than the non-aroused group.

1.3.2 The “Bottom-Up” Approach

This can be called the “bottom-up” approach, as one starts without any prior knowledge about the meaning of words and themes and simply learns from texts. Using this approach in a classifier has the distinct advantage that it does not take into account any preconceptions that might be embedded, for example, in manually curated word lists. One might suspect that, as the coding systems used for the “top-down” direction are also derived from these motive induction studies and experts have high inter-scorer-reliability, that their results are objective, too. However the scoring systems only separate the two groups to some extent and the human interpretation of text is guided by intuition.

This thesis tries to go the first step towards a reliable and objective system that takes stories in and gives scores out, without having been tweaked by human assessment. The questions it is trying to solve is: Can texts be classified as belonging to the aroused group or the non-aroused? If so, can this classifier be trained on the texts of one study and be used on those of another? In order to reduce this vast scope and make it more manageable we restrict ourselves to using word embeddings as the main tool in this work.

Chapter 2

Datasets

Before going into detail on the research at hand, let us take a short look at the datasets to be classified. All these datasets come from a motive arousal study. We describe in brief terms how each study was conducted and how the images used in the study are described in the original publication.

As we will see later in section 3.2.1 these do not form the corpus to train our word embeddings on: for this there are far too few stories. Instead, they were trained on Wikipedia articles and a specially compiled corpus chosen to mimic writing style and choice of words of the stories in these datasets.

We have stories from six motive induction studies available for analysis. Four studies contain about 340 stories, one has only 174 and one 482 stories, but this dataset had to be split into two with about 240 stories (see fig. 2.1). These numbers include both classes (aroused and non-aroused or control). In most studies, the number of stories of the aroused and control group are not equal, which could pose a problem in training and validation. This can be mitigated by using stratified cross validation and weighting, as will be described in section 4.1.

Another dataset on which we initially worked, labeled “Study2”, was later disregarded as arousal and control conditions were not clearly labeled for all images in the accompanying description documents and the remaining dataset was very small.

Most datasets are available in form of Microsoft Word documents. They were converted to text format and split so that one text file contains one story.

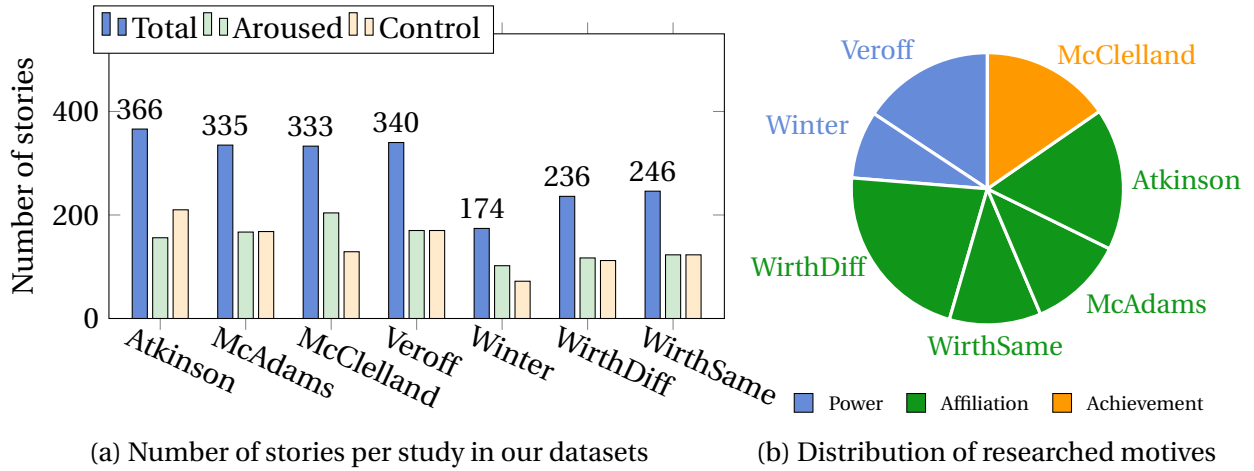


Figure 2.1: Dataset sizes in absolute and relative values.

2.1 Atkinson et al. 1954

The first dataset, labeled Atkinson, stems from the 1954 publication by Atkinson et al. on the affiliation motive.

The arousal condition was created by letting 31 members of a fraternity, having had an evening meal together, solve a task to rank traits making a person likable. The control condition was a class of 36 male psychology students having solved an anagram task.

The images used in this study are, as described in the paper: “(a) the heads of two young men facing each other in conversation; (b) a man standing in an open doorway looking away from the viewer at the landscape outside; (c) six men seated about a table as if in conference; (d) an older woman with a troubled expression looking away from a younger man; (e) four men sitting and lounging informally in a furnished room; and (f) a boy seated at a desk holding his head in one hand” [Atk54].

2.2 McAdams 1980

The McAdams dataset was created for the paper on the intimacy motive published in 1980 by McAdams. For our purposes we can see it as an affiliation dataset, though. Technically they are not the same, but for this thesis the distinction is not so important.

The participants were members of a fraternity and sorority, who were randomly assigned to the arousal and control groups. Both were formed by 21 men and 21 women. The arousal group members took the TATs after the initiation rituals at their respective association. These

festivities are reported to induce a feeling of closeness and brotherhood or sisterhood. The control group was administered the test in a classroom with minimal affiliative cues.

The pictures showed to the participants were “(a) two figures sitting on a park bench by a river, (b) a figure walking down a rain-covered street with his/her back to the camera, (c) three men talking in a log cabin, (d) a man sitting at a desk upon which sits a photograph of a family, (e) a group of people square dancing, and (f) an old man and a younger woman walking through a field with two horses and a dog” [McA80].

2.3 McClelland et al. 1949

The only dataset available to us comprising stories concerning need for achievement will be called McClelland. The texts were written for the 1949 paper by McClelland et al. on said motive. The original paper lists more than two groups, but for the purpose of this work we regroup the participants into a non-aroused group (the “relaxed” group mentioned in the paper) and an aroused group (consisting of the groups “failure” and “success-failure” from the paper).

The participants were all male psychology students and veterans. When assigned to the relaxed group, participants were given tasks and told that the test was to be analyzed, not the participants. That is, they could not fail or succeed, because they thought they were not examined. The other two groups were put under a lot of stress to succeed by giving them absurdly high goals to meet. They were furthermore told that another group of students excelled at these exercises and that they would be able to compare themselves to this excellent group. This resulted in them working harder and more concentrated. Both groups were first given some unrelated work before the TATs.

The images used in the study are described as “two men in overalls looking or working at a machine; a young man looking into space seated before an open book” and the TAT images TAT 7 BM (“‘father’ talking to ‘son’”) and TAT 8 BM (“boy and surgical operation”) [McC49].

2.4 Veroff 1957

We have already briefly touched on the Veroff dataset in the introduction in section 1.3.1. It was created for Veroff’s power motive research and published in 1957.

The arousal group were students running for an office, the holder of which was to be determined by election. They were administered the tests in a waiting period after the polls had closed and before the results were announced. To increase their *nPow*, they were asked

to rate their chances of winning. The non-aroused group was a class of psychology students who were not introduced to any cues which might heighten their need for power. Only male students participated in the experiment.

The images chosen for the study are described as: (a) “Two men in a library” (b) “Instructor in classroom” (c) “Group scene” (d) “Two men on campus” and (e) “Political scene” [Ver57].

2.5 Winter 1973

The Winter dataset also concerns the power motive. Winter created it for the development of his scoring system and described it in his book “The power motive”.

Participants in his study were all male master business administration students. To arouse the power motivation Winter showed one half of the test subjects a film of U.S. President Kennedy’s inauguration oath and speech. The neutral condition was showing the students a film depicting a businessman discussing science demonstration equipment. After having been shown the movies both groups were given a TAT comprising six images and a questionnaire for additional data collection.

The images used for the TAT are described as (a) a “group of soldiers in field dress; one is pointing to a map or chart” (b) a “boy lying on a bed reading a newspaper” (c) “ ‘Ship’s Captain’ talking to a man wearing a suit” (d) a “couple in a restaurant drinking beer; a guitarist is in the foreground” (e) “ ‘Homeland’: man and youth chatting outdoors” and (f) a “couple sitting with heads bowed on a marble bench” [Win73].

2.6 Wirth and Schultheiss 2006

For the last dataset, affiliation was aroused once again. Wirth and Schultheiss published about it in 2006, researching how motive arousal affects hormone levels. Participants were students of mixed sex. They were assigned to three groups, only two of which are interesting for our research. All participants were first administered a four picture PSE and then shown a movie. After the movie participants were asked to fill out a questionnaire and participate in another PSE (in random order).

The arousal group was shown an excerpt from the movie “The Bridges of Madison County”, a romantic movie. The control group watched “Amazon: Land of the Flooded Forest”, a National Geographic documentary. The images shown to the participants were (I): “ ‘trapeze artists’ (a man and woman about to catch each other on trapeze swings), ‘park bench’ (a

couple sitting together on a bench by a river), Thematic Apperception Test (TAT) card 3BM [...] (depicting from behind a person with head on arms, appearing distraught), and ‘excluded boy’ (depicting schoolgirls talking while an unhappy-looking boy stands apart)” and (II): “ ‘mountain’ (a woman helping a young girl mountain climbing), ‘horses’ (an older man and younger woman leading horses and talking), TAT card 13B (depicting a small boy sitting alone in a doorway), and ‘excluded girl’ (college-aged girls talking while a fourth girl stands apart with arms crossed)” [Wir06]. The image series I and II were shown in the pre- and post-movie condition to an equal number of participants in a randomized way.

We can look at this dataset in two ways: First, we can compare the pre-movie PSE stories from the arousal group to the stories written by the same group after the movie, which will be called $Wirth_{Same}$. Note that this is the only case where all authors were in both the aroused and non-aroused group. Second, we can compare the post-movie PSE stories from the aroused and the control group. We will call this view $Wirth_{Diff}$.

Chapter 3

Methods

Now that we have seen the datasets, we will look into the methods used to classify them. In this thesis word embeddings are used as the primary tool. The reasoning behind that is to increase detection ratio by including sentiment information contained in these vectors. Employing neural networks as classifiers is disregarded, as they typically require larger datasets to train on than are available to us.

We will now give an introduction to the techniques used and then continue to the classifiers in which they were applied.

3.1 Word Embeddings and Modifications to Improve Them

Word embeddings are, in short, high dimensional vectors representing words. As such they lie in a vector space, also called word embedding space (or briefly embedding space), where vectors of semantically and syntactically similar words are located in similar regions. Here, similar does not have to mean equivalent, though. “Fast” and “slow” may lie relatively closer to each other than “fast” and “mountain”, as they are both adjectives describing speed. The reason for this lies in the creation of word embeddings from word co-occurrence, as we will see.

First we will give an overview over the two different word embeddings that we trained and used. Thereafter we will introduce the techniques we applied to the already trained embeddings in order to improve them, in the hopes of thereby also improving the classification rates.

3.1.1 Word2vec

In 2013 Mikolov et al. published their paper on “continuous vector representations of words from very large data sets” [Mik13]. The word embeddings they proposed – known as word2vec – have since then been widely adopted as they are fast to learn on large corpora and delivered state-of-the-art performance on semantic tasks. Unfortunately, their original implementation is hard to find since Google ended the life of Google Code, so for this thesis we employ the implementation in gensim, a popular and widely used framework implementing natural language processing techniques.

Mikolov and his colleagues employed a neural network and trained it to predict a missing word from a given context. This is known as the continuous bag-of-words (CBOW) approach. They also proposed predicting the context from a given word, which they called (continuous) Skip-gram. They perform roughly the same for syntactic tests, while Skip-gram outperform CBOW on semantic tasks. However Skip-gram training takes longer and also requires larger corpora than CBOW [Mik13].

Due to limited machine time, we restricted ourselves to the continuous bag-of-words method. We trained word2vec embeddings on a context of 5 words. The resulting vectors were trained in a dimensionality of 400 and 600 entries to see if including more information made a difference in the classification process.

3.1.2 GloVe

While word2vec works by predicting words in a local context, the global vectors for word representation (GloVe) by Pennington et al. optimize word co-occurrence probability ratios on a global scale. The idea is that e. g. the word “ice” will (on average) appear as often together with the words “fashion” and “water” as the word “steam” does. This is due to the fact that both are unrelated to fashion and both are related to water, of course. However “ice” will appear more often in the context of “solid” whereas “steam” is more likely to be read next to “gas”. So, formulating this as the probability of one word occurring in the context of another word, the ratio $\frac{P(\text{fashion}|\text{ice})}{P(\text{fashion}|\text{steam})}$ will be roughly 1, while the ratio $\frac{P(\text{solid}|\text{ice})}{P(\text{solid}|\text{steam})}$ will become quite large. They transform this observation into an optimization problem. Its goal is to approximate the logarithm of the number of word co-occurrences of two words by the dot product of the vector representing these words.

This method also trains fast on large corpora and has better results in semantic and syntactic tests than word2vec [Pen14].

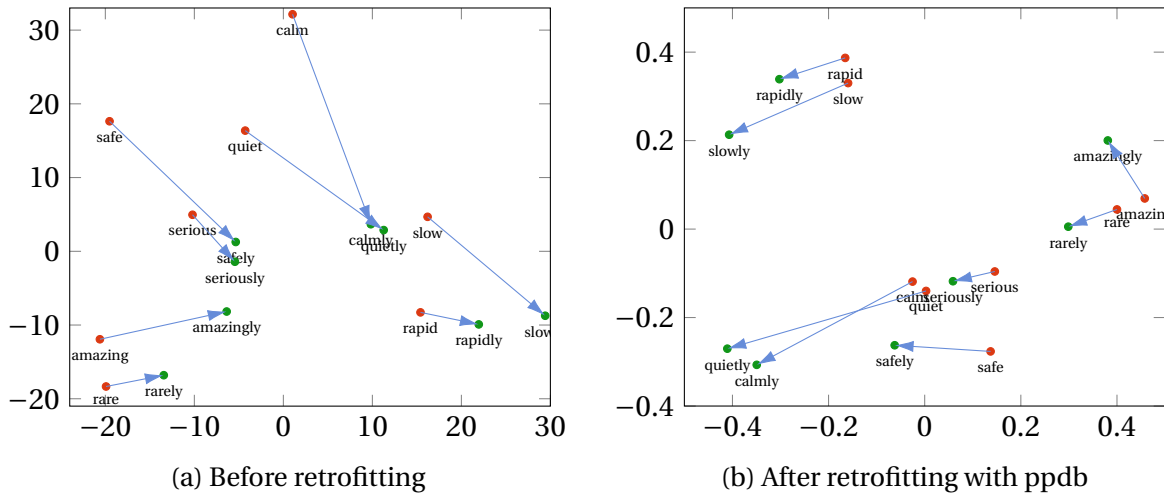


Figure 3.1: PCA of adjective to adverb relationship of different words in the 400 dimensional word2vec embedding space trained on ffcorpus

We trained GloVe embeddings using the implementation the authors provide on GitHub¹ and did not change the default parameters. We also trained vectors with 400 and 600 dimensions.

3.1.3 Retrofitting

There is not only research on how to generate good word embeddings, but also on how to improve pre-existing vectors. We evaluated two such methods regarding whether they improved classification results.

The first method adds information from semantic lexicons to the vectors. It was published in 2015 by Faruqui et al. [Far15], who call the process *retrofitting*. It works by taking pre-trained vectors and a list of synonyms as input. Then it optimizes the distance between words in such a manner that synonyms lie closer to each other in the vector space while not drifting too far away from their original representation.

One thing that this leads to is that the word vector space is more structured. This can be seen in fig. 3.1. Each data point in these principal components analysis (PCA) plots represents an adjective word vector and the corresponding adverb. The same words as in the original publication were chosen to allow better comparison. Before the retrofitting procedure, there is some tendency in the direction of the “adjective-to-adverb” vector as depicted by the arrows. However there are several outliers pointing in an almost perpendicular direction and the

¹<https://github.com/stanfordnlp/GloVe>

alignment of the vectors is quite poor. After retrofitting, the vectors of the relationship have a more consistent direction and are much closer to being parallel. This can be seen most clearly with the word pairs “rapid” and “slow” and their respective adverbs. With “quiet” and “calm” there is a visible improvement, even though the synonymous words were not drawn close enough together to make the relationship vectors perfectly parallel. Overall the method seems to work, although not as well as described by Faruqui et al. The “amazing” to “amazingly” vector, for example, has not been improved at all.

Faruqui and colleagues evaluated their method using four different synonym information sources. Retrofitting worked best with information from the *Paraphrase Database* [Gan13] and the *WordNet* lexical database [Mil95] (including synonyms, hypernyms and hyponyms). We therefore used these two databases in our experiments, too, and refer to them as *ppdb* and *wns+*, respectively.

We trained the retrofitted word embeddings using the code published on GitHub² and did not change the default hyperparameter settings.

3.1.4 Ultra Dense Word Embeddings

While retrofitting is “destructive” in a sense that it moves the word representations around in the vector space, *ultra dense word embeddings* try capturing information by rotating the vector space itself. Rothe et al. describe this in their 2016 paper.

Their method uses two sets of word vector representations as input. The words within the first set all have one semantic property in common, e. g. positive sentiment. The other set contains words with opposing semantic meaning, e. g. negative sentiment. They train a transformation matrix $\mathbf{Q}$ such that it rotates the vector space to minimize distance within the sets and maximizes the distance between vectors of different sets. The cost function is applied only to a subspace of the original embedding space, though. In the optimal case this leads to one dimension encoding the learned semantic meaning. Hence the name “ultra dense word embeddings”. In an optimal scenario the other dimensions can then be dropped, if they are not of interest any more [Rot16].

For this thesis a custom implementation had to be used, because there is no link to the source code in the paper. We tested the implementation by training on sentiment information from the databases listed in the publication. Afterwards words with positive sentiment had mostly positive numbers on the first dimension and words with negative sentiment mostly negative entries on it. This confirms that the implementation works as described in the paper.

²<https://github.com/mfaruqui/retrofitting>

Originally the method requires two sets of words of opposing meaning, such as sentiment words (e. g. “horrible“ and “lovable”). For this thesis, however, the goal was to train Q using the Pennebaker word lists taken from the LIWC software (see section 1.2.2). These contain words that do have some common semantical meaning, but in general can be very far apart (e. g. “comrade” and “exgirlfriend” from the “Friends” category). This poses two problems, though: First, it breaks with our intent to go “bottom-up”, as it introduces information not learned from the texts. This could be mitigated in a later step, for example by selecting words automatically from the given texts. The second is that the Pennebaker lists available to us do not contain words of opposing meaning. They are organized into 68 different categories with a varying number of words, some containing wildcards. To solve this we employed negative sampling, i. e. choosing random words that are not contained within the selected LIWC category. Rothe et al. stress in their paper that small, hand-collected word lists work better than automatically generated ones [Rot16]. While the LIWC category words are indeed hand-picked, the negative examples are not, but in general there should be a difference between the two groups.

The modified approach, in detail, is as follows: We load the word list and select the words of the chosen category or categories. We then expand the wildcards by matching them on the 80,000 most common words, as it is recommended in the original paper, because more infrequent words tend to be too noisy. Then we randomly sample an equal number of other words from the same pool and run the algorithm on these two sets.

The GloVe word embeddings are not sorted by word frequency and therefore the expansion of wildcards cannot be done simply on the resulting model alone. To fix this, the list of frequent words from the word2vec data can be exported and used for the GloVe embeddings, too. This is possible because both embeddings were trained on the same corpora.

3.2 Word Embedding Training and General Preprocessing

Now that we have looked into the different word embeddings that were employed for this work, the question remains how the word embeddings were trained and how texts were preprocessed.

3.2.1 Corpora and Training

Word embeddings should not be very sensitive to the corpus they have been trained on. After all, they are created by looking at word co-occurrence³. In most texts, “death” or “war” will not

³ For word2vec this might not seem obvious at first, but Pennington et al. devote a whole section of their paper to showing that GloVe and word2vec are closely related

be read next to the words “happiness” or “fun”. Therefore word embeddings are often trained on Wikipedia articles. The reason is simply that they are available, free and public domain. It surely helps, too, that gensim comes with an interface to read XML-Wikipedia dumps directly as offered by the Wikimedia and use them as a corpus.

Different corpora do have different distribution of words, though, of course. A simple example is that the words “love” and “you” will not often accompany each other in an encyclopedia but do in romantic novels. Using the same corpora for word embedding training as for classification would guarantee the same word distribution. We cannot train our embeddings on the texts from our dataset, though, as they are simply far too few and too short.

Therefore we looked for a class of comparable texts. At first, of course, Twitter comes to mind. Tweets are short casual texts and there is a sheer endless corpus available. However tweets seemed a little too short and they come with a large amount of abbreviations and more casual language than is common in PSE stories.

A good alternative option is fan fiction as a training corpus. Fan fiction texts are short stories, written by fans of another fictional work. There are stories unfolding in the worlds of TV series and shows, movies, books, games and much more. These texts come pretty close to the stories written by participants in a motive induction study: They are written by non-professional authors in a casual way and even are inspired by another work of art.

Of course, the authors have more time than five minutes for their texts and texts are much longer. But for our purpose that is not important. We are interested in having texts where the word co-occurrence is not altered by a specific writing style, like for example presenting facts in a neutral manner as the Wikipedia does. Therefore it is important to mention that fan fiction stories are not distributed equally over all genres. Manga and anime, for example, have a lot more authors and stories. So much in fact, that there is a special category for them on <http://fanfiction.net>, even though, strictly speaking, they are TV shows and cartoons. We tried to combat this by training on an equal number of stories from the categories “anime”, “book”, “movie” and “TV”.

The corpus comprises a total number of 6,722,056 texts from these four categories from the aforementioned website. Texts were downloaded chapter-wise and stored together with additional information like author, genre, etc. for later reference. Stories and chapters were sampled randomly within each category. This probably introduced a bias towards popular topics like “Harry Potter” or “Naruto”. We do not think this poses be a problem for our application, though.

Authors often put content not belonging to the actual story in the first or last paragraph. They might excuse themselves for not updating for a while, tease the next chapter or simply thank their readers for their attention. Therefore we disregarded two paragraphs from the beginning and end of each text. We removed punctuation, lowercased the texts and removed words shorter than two characters. Finally we filtered out texts with less than 50 words and that left us with a total number of 5,862,972 (1,465,743 per category) texts. The training corpus has a size of 11.56 trillion (about 2000 per text) and a vocabulary of 4.35 million words. We will refer to it as *fan fiction corpus* or *ffcorpus*.

To compare against a corpus with more formal language we decided to train on a dump of the English Wikipedia. It contains a total number of 4,265,001 articles with 2.33 billion words. We will refer to this corpus as the *Wikipedia corpus* or *wikicorpus*.

The two corpora are quite different in size as we were unaware of the average length of fan fiction before having downloaded them. It seemed wasteful not to train on the complete *ffcorpus*, though, just to have a comparable corpus size. After all, the *wikicorpus* is still large enough to guarantee good word embeddings.

Before training, the same preprocessing was applied as for the stories when classifying them.

3.2.2 Preprocessing

Before passing the PSE stories to any classifier, the texts were preprocessed. That includes removing punctuation, lowercasing the texts and removing single character tokens as well as tokens longer than 15 characters. As we did the same when training the word embeddings, these words were not contained in the embeddings and would have been removed later either way. This process also ensures comparability, because all classifiers, including the bag-of-words one, receive the same input. A simple caching system decreased the time it took to look up a vector for a given word substantially.

For some methods it can be beneficial to remove stop words. Whenever this was done, we will indicate this in the following subsections or when discussing the results. The list of the 100 most frequent words from the Wikipedia corpus were used as stop words. However, we manually excluded some words that we deemed possibly influential from the list. This is a tiny deviation from our agenda as we try to minimize human interpretation of semantics in the process of coding implicit motives. But as we did not want to lose information possibly contained in these words we had to remove them from the stop word list. See code listing B.1 for the list of both included and excluded stop words. The most used words from the fanfiction

corpus contained more words that we think might have an influence, which is why we used the ones from Wikipedia.

3.3 Classifiers and Features Tested

We will now look into the classifiers and features we implemented and tested. All code was written in Python 3 and has been made available online⁴. For the classifiers and some natural language processing related tasks the frameworks *gensim*, *scipy* and *scikit-learn* were employed.

3.3.1 Bag of Words

The bag-of-words implementation from scikit-learn will function as a baseline. It works by applying term-frequency inverse-document-frequency (TF-IDF) to the word counts and allows varying the size of the n -grams considered. The document frequency factors were learned only on the training corpus. This, of course, implies that words only contained in the testing and validation sets are not taken into account with this classifier. With the resulting feature vectors we trained a support vector machine (SVM).

This approach is very similar to the experiments performed by Hiram and Pang as described in section 1.2.2 [Rin17].

3.3.2 Word Mover's Distance

When moving away from bag of words and implementing an approach using word embeddings we are faced with the problem of how to aggregate the words of a story into a single vector. How this could be done will be the topic of the upcoming subsections. But first we will describe a different approach where the vectors are not aggregated at all.

Instead, in this classifier, a distance between documents is calculated and classification is performed by employing a k -nearest neighbors (k -NN) classifier. As distance metric we apply the *Word Mover's Distance (WMD)* as proposed in 2015 by Kusner et al. [Kus15] Using the WMD has the advantage of leveraging the semantic information embedded in the word2vec and GloVe word embeddings. That is, in a bag of words, the words “like” and “fancy” will be distinct, while the WMD considers them as close as they are within the respective vector space.

⁴See <https://github.com/t-animal/Masterarbeit-Code>; All classifiers can be found in the `classifiers` subdirectory. They all extend the base classes from `util/classifiers`

To understand the Word Mover's Distance let us consider first, informally, the Earth Mover's Distance upon which it is built. Say there is large yard containing piles of soil which have to be moved to fill holes in the ground. Each hole has a specific size and location, which may differ from the size and location of the piles. The Earth Mover's Distance is the minimum cost it would take to move all piles to fill all holes, assuming a cost function of "amount of dirt" $\times$ "distance traveled" [Rub00]. Applying this distance to texts, we do not move dirt but word frequency, as we will now see.

In the case of the Word Mover's Distance the cost function is the Euclidean distance $c(i, j) = ||w_i - w_j||_2$ of two words in the word embedding space. The dimension of the piles or holes, respectively, is the normalized term frequency tf of a word $d_i = \frac{tf(w_i)}{\sum_{m=0}^n w_m}$ in a text of length n . The amount of dirt (in this case word frequency) moved from one word of one text of length n_1 to another word of another text of length n_2 is determined by a transformation matrix $\mathbf{T}$. As no more than d_i can be moved, $\mathbf{T}$ is constrained by

$$\sum_{j=1}^{n_1} \mathbf{T}_{ij} = d_i \quad \forall i \in 1, \dots, n_1.$$

Likewise a receiving word cannot get more frequency than it has in its own text, here denoted by d_j ,

$$\sum_{i=1}^{n_2} \mathbf{T}_{ij} = d_j \quad \forall j \in 1, \dots, n_2.$$

Now the final Word Mover's Distance of the two texts can be written as

$$\min_{\mathbf{T} \geq 0} \sum_{i,j} \mathbf{T}_{ij} c(i, j)$$

which is an optimization problem for which many efficient solvers exist (the WMD implementation in gensim uses pyEMD) [Kus15].

The WMD can be used to simply train a k -NN classifier. As the distance does not have hyperparameters, the number of neighbors is the only variable to be evaluated.

3.3.3 Neural Network Based Approaches

Except for the k -NN classifier of the last subsection and random forests as a side note, we will focus on SVMs for classification. Neural networks are very flexible and versatile but require a relatively large number of samples to learn from. A neural network is employed, for example, to learn the word2vec word embeddings, but this is only feasible because we do not train them

on our classification datasets but on the much larger Wikipedia and fan fiction corpora. Neural networks are not employed for the actual classification task as our datasets are probably too small for that.

However in the aggregation step it might be possible to aggregate word vectors to feature vectors as suggested by De Boomet al. in 2016 [Boo16]. They learned weights w_j such that the representation $t(C)$ of a short text of length n is given by

$$t(C) = \frac{1}{n} \sum_{j=1}^n w_j C_j$$

where C_j is the j th word of the (sorted) text. Before training, the words of a text are sorted from high to low inverse-document-frequency (IDF). The optimization is performed by a neural network. The training objective is to decrease the distance of related texts and increase the distance of unrelated texts. Texts were taken from the Wikipedia articles and tweets from news agencies. They were considered related when taken from the same article or tweets within a short timespan of each other containing the same hashtag [Boo16].

These findings can be applied to our problem by considering texts within (aroused and non-aroused) groups as related and across groups as unrelated. Unfortunately the implementation provided with the paper did not terminate on most datasets. We believe this is due to the large number of weights (average text length is about 90 words) and the small number of texts to train on. When the algorithm did terminate it did not improve results compared to other methods explored and we did not look into this method any further.

Nevertheless one takeaway from the paper is that weighting words according to their IDF value can be beneficial when using a centroid to aggregate them. The resulting weights as published in the paper decreased with word index j , indicating that words with lower IDF contribute less to the overall meaning of a text. We could not train word weights using the provided code, but we still tried weighting word vectors according to their IDF value when summing them up. We did not find an increase in accuracy due to this weighting and therefore dropped it again from our classifiers.

This experience strengthened our opinion that we should not explore neural networks in this thesis. It also led us to disregarding other neural network based approaches like skip-thought vectors [Kir15] or distributed representations of sentences and documents [Le14]. Nevertheless they should be mentioned here, as they might be useful when a large enough corpus of PSE stories can be compiled.

3.3.4 Centroid Vector

One of the first and at the same time simplest approaches that we explored sums up the vectors of a document to aggregate them, too, but they are not weighted.

The Feature Vector $\vec{d}$

We calculate the centroid vector $\vec{d}$ by simply summing the word vectors and normalizing by the number of word in a document D :

$$\vec{d}_{len} = \frac{1}{n} \sum_{\vec{v}_i \in D} \vec{v}_i.$$

Albeit technically not creating a centroid, different norms can be employed to normalize the feature vector. Another option that we will consider is the Euclidean norm

$$\vec{d}_{l2} = \frac{\sum_{\vec{v}_i \in D} \vec{v}_i}{\|\sum_{\vec{v}_i \in D} \vec{v}_i\|_2}.$$

Apart from the impact of changing the norm applied, we will explore also how stop word removal affects the classifier. However, while the removal of semantic noise superimposed by these relatively meaningless words might be desirable, it imposes the risk of losing information which could be contained in the frequency of these words. To counteract this, we count how often a word was removed, normalize the resulting vector independently and concatenate it with $\vec{d}$. One could say that we are bringing in the bag-of-words approach for the stop words, because we do not rely on their semantics but only on their frequency. An example of this can be seen in fig. 3.2b on the bottom (a plot that we will explain later, for now it suffices to see, that the right subplot without stop words has about 100 entries more representing the frequency). An additional “nostop” in the index of the feature vector will indicate that stop word removal was applied to it, such as $\vec{d}_{l2,nostop}$.

Another piece of information not contained in $\vec{d}$ so far is the image to which a story belongs. One might argue that it should not be relevant anyway, because we are trying to classify based on data which is independent of the image: clues regarding the motivational state contained in the semantics of the words used in a story. But on the other hand, these clues might manifest differently for different images. When trying to add a dimension indicating the image to the vectors we are faced with a problem, though. Namely the authors of the different studies mostly not specifying whether they randomized image order. When they did talk about it, they did not provide information on how the randomization could be undone. We could have

tried to match stories and images manually by content, but this seemed out of scope. Instead, we speculated that authors who did not mention randomization probably did not change image order across the study. Actually we do not see the necessity for randomization in all study setups but Wirth and Schultheiss' and therefore this assumption is probably sound. We proceed to add another dimension to $\vec{d}$. It stores an increasing number representing the image. This number is divided by a fixed value to fit it into the range of the other entries of the feature vector to prevent scaling effects.

As we could see in fig. 3.1, applying retrofitting to word embeddings improves their semantic and syntactic meaning in the sense that vectors are better aligned. We can hope to exploit this in this classifier. Assuming a structural semantic or syntactic difference is present between the aroused and non-aroused stories then this difference should accumulate when adding up the individual words. Of course there will be semantic tendencies imposed by the images themselves. For an image with a ship and a captain nautical words are naturally going to appear more often than in an image with two trapeze artists. But it is sensible to assume that this will be the case for the aroused and non-aroused group and that it can therefore be considered neutral when comparing both groups.

The last thing to be investigated is how using an ultradense subspace as described section 3.1.4 affects detection rates. For this, we train the rotation matrix Q using negative sampling on LIWC categories. For each motive we chose the two categories with the highest correlation to the content coded motive scores as determined by Schultheiss [Sch13a]. We did not choose categories, though, which were only significant after conversion to a dichotomous format, as this is just a boolean representing the presence or absence of any word of the category. For *nAff* the selected categories are "Other" (words like "her" or "himself") and "Friends" (for example words starting with "acquaintan"). For *nPow* they are "Space" (e.g. "above", "beyond") and (negatively correlated) "Tentative" (containing words such as "doubt" and "uncertain"). Lastly *nAch* correlates with the categories "Achievement" (e.g. "perfection", "try") and "Optimism" (words like "pride" or "bold"). We apply the ultradense subspace algorithm for each category independently and save the rotation matrix for later use. When an ultradense subspace transformation was applied to the vectors this will be shown by an additional index "ultradense" (e.g. $\vec{d}_{l2,ultradense}$).

Plotting $\vec{d}$

To get a better understanding of the classifier, we plotted the dataset using $\vec{d}_{l2}$. Two examples of this plot can be seen in fig. 3.2 (a larger, more detailed version can be found in figs. C.1

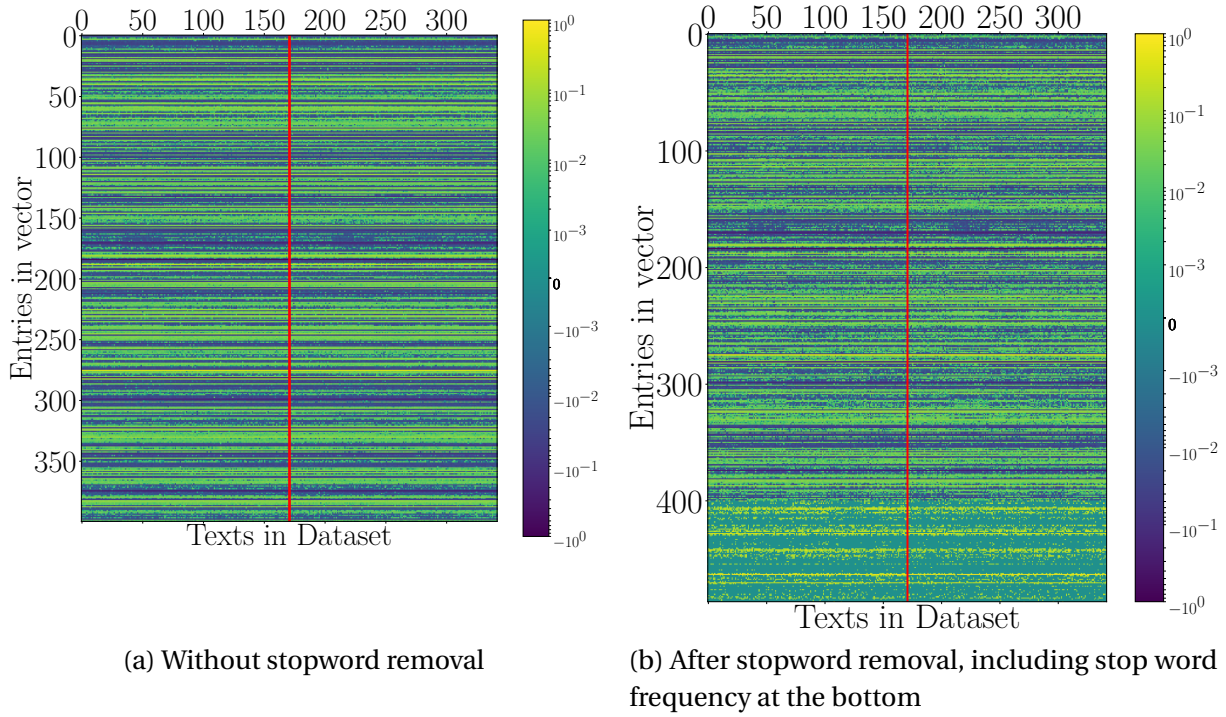


Figure 3.2: Matrix rendering of $\vec{d}_{l2}$ and $\vec{d}_{l2,nostop}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories

and C.2). In this depiction, one column represents the vector $\vec{d}_{l2}$ of one document. One row represents one entry of all $\vec{d}_{l2}$ across the dataset. The vertical red line in the middle separates the aroused stories on the right from the non-aroused stories on the left of it. Ideally we would hope that one or more rows correspond clearly with one of the two sides of the red line. In this case the word embedding would already capture the semantic difference between the two groups. That is, if one group used generally more words having to do with power than the other and if there was one row encoding this semantic meaning, it would accumulate on one side and be low on the other side.

Unfortunately we do not see such patterns. There are some darker spots around story 50 and 220 and vector index 160, though. These come from the fact that we sorted the documents according to their story number⁵, assuming – like we said before – that story order was not randomized. These spots confirm this assumption. However they were not as pronounced in all datasets if they were visible at all.

⁵The sorting was performed only for the purpose of generating these pictures, not when creating the training, test and validation sets of the cross validation phase.

When looking at fig. 3.2b one can see that the stop words do introduce a lot of noise to the classifier. After removing them, many previously hidden structures become visible. Some of these might actually discriminate between the two sets, for example in the noisy looking area around vector index 50. But these differences are hardly visible to the naked eye and only surface when subtracting the two halves of the image from each other, making it a faint hint that the classifier does capture a difference between the groups.

3.3.5 Clustered Centroid Vectors

Based on the centroid vector method we will now explore means to cluster information in order to improve detection rates. An initial approach might be inspired by manual coding. Humans, when coding PSE stories according to a manual, go through the story sentence by sentence and decide whether motive imagery is present or not. This could be emulated by computing a centroid vector within a window of text and thus creating several vectors for parts of the texts. With these vectors a majority vote could be done, possibly weighting votes using an SVM trained with probability estimates or a regression method.

During early testing of clustering we implemented this idea, but encountered two main problems with it. First, turning on probability estimates in the scikit-learn SVM implementation increased training times drastically, making it hard to work with. Instead, the distance to the hyperplane could be used, but we did not get as good results with that as with the probability estimates in preliminary tests. Second, the method increases noise in the already pretty noisy data. This is because probably not all sentences in a story contain motive imagery. In fact, the Winter coding data available to us suggest that imagery is present only in a minority of sentences. When considering multiple windows per document, most will not contain the necessary information and thus increase the problem. One possible approach to deal with this might be to reduce the cost of false negatives (i. e. feature vectors from an aroused story lying in the region of the non-aroused stories and vice-versa) during training.

Because of these problems we shifted our focus to other clustering methods. We still believe this solution has some merit, but did not have the time to come back to it.

Instead of building clusters within the documents based on a windowed approach, we will now explain approaches that cluster the words in their embedding space. The reason to do so lies in the simplification introduced by the centroid, which might just be an oversimplification. It might be, that the centroid captures only a too general sense of the story. There could be, for example, many words concerning love, friendship and harmony, possibly indicating a high *nAff*, which are simply overlaid by a lot of other words with no clear semantic meaning. But

similar words and synonyms should appear in similar contexts, as for example “I love chocolate” and “I like chocolate”. Therefore they should lie close to each other in the word embedding space. This property is amplified after applying retrofitting to the word embeddings. Thus we can hope to be able to find clusters of words within the stories that have similar meaning and thus to further “unclutter” the actually meaningful words and create a vector that is more representative of the actual story.

Furthermore a story might not be representable by only one vector. After all, it is made up by about one hundred word vectors pointing in possibly completely different directions. After summing them up, this diversity is gone. We hope to be able to restore parts of it by clustering the words in the embedding space and thus represent the story by several vectors. For example, a sad love story might be represented both by many vectors representing words of affection but also of grief. Summing up all these vectors leads to a vector not representing either. By clustering we hope to end up with, sticking with the example, one centroid vector for all the love related and one for all the grief related words.

k-Means

The *k*-means clustering algorithm is one of the most popular. It clusters vectors based on their Euclidean distance. However the results can vary depending on the initial starting points of the iterative algorithm. Furthermore, when choosing random starting points in each story and finding optimal clusters for each story, it is likely that the *n*-th cluster of each story does not correspond to that of the other stories. Some kind of sorting would be necessary to order the resulting clusters before concatenating them to a single feature vector so that the same clusters are compared in the classifier. Both problems can be circumvented by first pooling all word vectors of all training stories and applying a first stage of clustering. This can be done using randomly selected elements from the pool as starting vectors. The means of the clusters with minimal cost, or distortion as it is often called in this context, can then be chosen as starting point for the second stage clustering. It seems that 30 iterations of the first stage are enough find a good solution in most cases. This number was chosen empirically by starting out with a relatively low number of iterations and increasing it until the clusters with minimal distortion were usually created by less than the maximum of iterations. Another option would have been to start at the vector representation of words considered meaningful with regard to the current motive (e.g. “winning” for achievement) but this did not guarantee better results and also introduced human prejudice about semantics into the problem, which we were aiming to reduce.

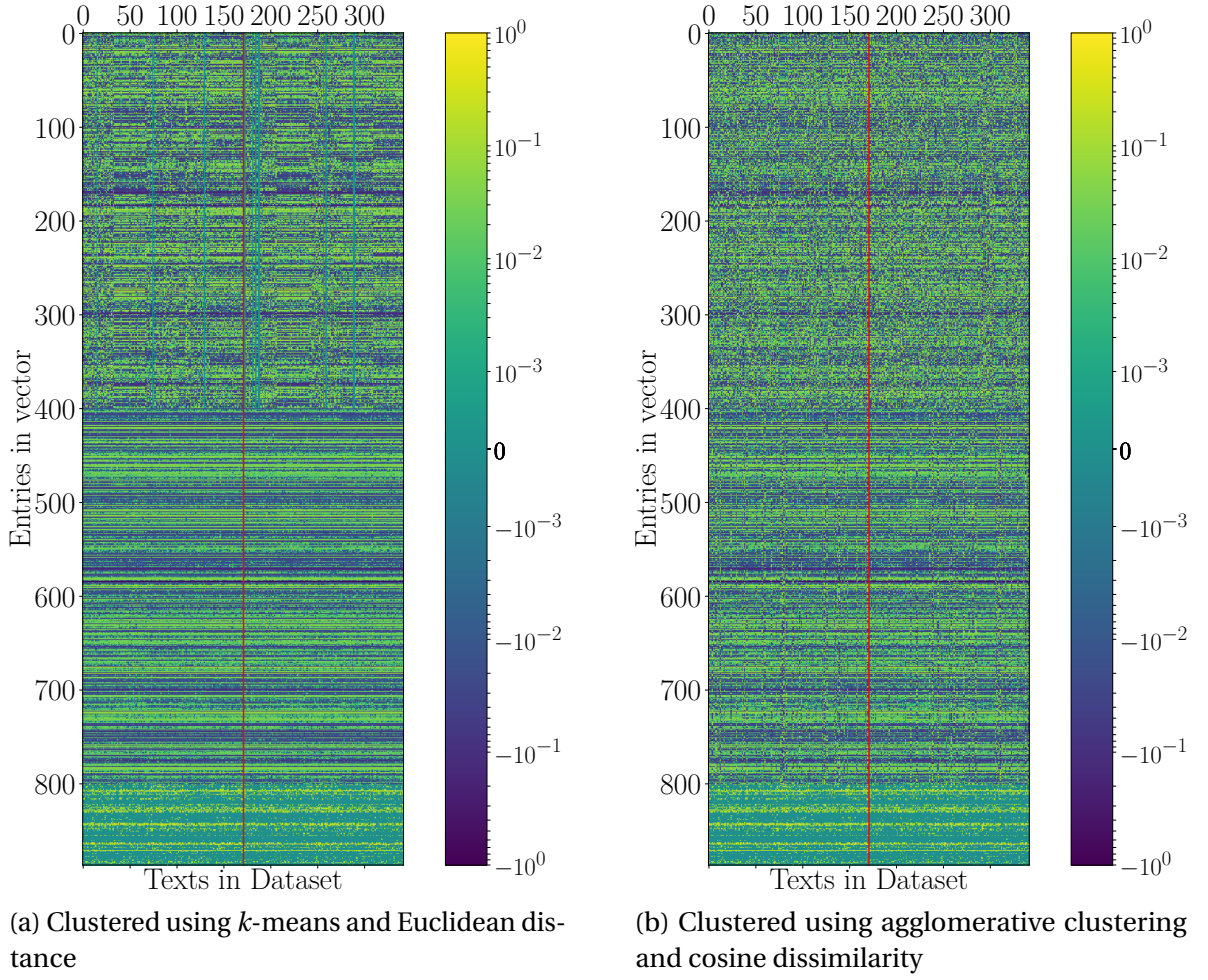


Figure 3.3: Matrix rendering of clustered $\vec{d}_{l2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories

Once the clusters on the pooled word vectors are acquired, we use their means as starting points for the second stage clustering. The second stage is applied to the word vectors of a document prior to creating its feature vector. One centroid $\vec{c}_k$ is then computed for the content of each cluster as it is returned by the second stage clustering. Note that there are two different uses of a norm here. First, the k -means algorithm uses the Euclidean norm to cluster the vectors. Second, either the size of the clusters is used to calculate the centroid of the cluster ($\vec{c}_{k,len}$) or the Euclidean norm of the vector sum of the cluster is used ($\vec{c}_{k,l2}$) to normalize it. The individual $\vec{c}_k$ are then concatenated to form the feature vector $\vec{d}$, which is used to train an SVM, in the same way as described in section 3.3.4.

Of course this introduces a new hyperparameter, k , the number of clusters to build. With higher k the risk of having empty clusters rises. When this happens, a zero vector is used for that cluster. This is not necessary most of the time, though, indicating that employing first stage clustering works well. As can be seen in fig. 3.3a, where empty clusters become visible as vertical lines, they do not occur distributed across the whole dataset but often lie directly next to each other, showing that a few authors' stories could not be clustered like the other authors'.

Figure 3.3a (for a larger version see fig. C.3) shows a typical result of this clustering method. Usually one or more clusters display a relatively regular pattern across the aroused and non-aroused condition and one or more clusters are relatively noisy. While the regular patterns can be seen in the simple centroid as well, the noisy regions might introduce an exploitable difference.

To indicate that k -means clustering was applied to a feature vector we will append k and the number of clusters to its index (e. g. $\vec{d}_{l2,k3}$).

Agglomerative Clustering

The Euclidean distance should work for word vectors, as similar words appear in similar contexts. Still, it might be too “precise” in that it also takes into account the distance from the origin. If the dimensions of the word embedding space encode semantical meaning, it might be beneficial to ignore how far a word is from the origin and only look at the general direction of its vector. This is captured by the cosine similarity, which is the cosine of the angle Φ between two vectors, $\cos(\Phi)$. Thus a value of 0 indicates that the two vectors are unrelated (very dissimilar), -1 means vectors exactly oppose each other and +1 means they lie in the exact same direction from the origin. The corresponding distance is called cosine dissimilarity: $1 - \cos(\Phi)$.

Looking at fig. 3.1b in the section about retrofitting, we can illustrate this in a simple two-dimensional plot. The words “rapidly” and “slow”, for example, lie in the same direction from the origin and therefore they have a low dissimilarity of only 3.855×10^{-2} . The wide angle between “rapidly” and “amazingly”, on the other hand, leads to a high dissimilarity of 0.9658.

When using this distance, vectors are clustered depending on their direction and therefore the resulting centroids $\vec{c}$ can be sorted. That means we can skip the two-stage clustering. To sort them, we measure the cosine of the angle between $\vec{c}$ and a vector of ones. As this calculation ($\frac{u \cdot v}{||u||_2 ||v||_2}$) requires the division by the product of the vector lengths, we cannot set empty clusters to zero. That would lead to a division by zero, which is why a vector of ones is used instead.

k -means clustering is not defined on other distances but the Euclidean distance, so another method must be used for the cosine dissimilarity. Agglomerative clustering, a hierarchical clustering method, is defined on any distance and will be employed for this method. We use average linkage to reduce the risk of single clusters taking in all vectors, leaving other clusters empty.

3.3.6 Classifiers Employed

We have already mentioned that the approach using Word Mover’s Distance employed a k -NN and that the bag-of-words baseline utilized an SVM for classification. For the other features, i. e. $\vec{d}$ and its variations like for example $\vec{d}_{l2}$, $\vec{d}_{l2,ultradense}$, $\vec{d}_{l2,k}$, etc. we tested two different classifiers.

First, we employed a support vector machine. The PCA of the vectors suggests that there are two regions that are quite entangled at the intersection (a depiction can be seen in fig. 5.1, where possible reasons for this entanglement are discussed). This could make it hard to separate the classes linearly – at least in the very reduced PCA view – an assumption that has been supported by initial experiments. Therefore we use a radial basis function (RBF) kernel. As there is an unequal number of aroused and non-aroused stories in most training sets, classes must be weighted inversely to their proportion to ensure that the results are not skewed towards the larger class. The hyperparameters to be varied are the penalty parameter C and influence parameter γ of the SVM.

We have not looked into ultradense word embeddings for this classifier. The reason for this is that document vectors created with them ($\vec{d}_{ultradense}$) probably do not improve the performance when using an SVM for classification. This is due to how the ultradense word embeddings are created and how this classifier works internally.

Rothe et al. aimed at reducing the number of possibly meaningful dimensions of word embeddings by applying an orthogonal transformation to the word embedding space. In doing so, they solved an optimization problem to minimize the distance between words of similar semantic meaning [Rot16]. An SVM on the other hand uses support vectors to define a hyperplane within the embedding space separating the two classes it was trained upon. Due to the distributive law, the orthogonal transformation *ortho* of the summed up vectors in $\vec{d}$ can be pulled out of the sum (it is just a multiplication by a rotation matrix $\mathbf{Q}$)

$$\vec{d}_{ultradense} = \frac{\sum \vec{v}_{i \in D} \text{ortho}(\vec{v}_i)}{\text{norm}(d)} = \text{ortho} \left(\frac{\sum \vec{v}_{i \in D} \vec{v}_i}{\text{norm}(d)} \right).$$

Therefore in order to transform an SVM trained on document vectors from normal word embeddings to one trained on $\vec{d}_{ultradense}$, merely the same orthogonal transformation to the support vectors has to be applied. This will happen automatically during the training process and thus the SVM will yield the same support vectors and the same hyperplane with respect to the orientation of the document vectors in the respective embedding space. Due to this observation we do not expect $\vec{d}_{ultradense}$ to perform any different on an SVM than a document vector built from the original word embeddings.

In order for the ultradense word embeddings to play out their strength, a dimensionality reduction must be performed on the word vectors. In the case of sentiment, where there are clear and lots of training words, this can be done up to one target dimension, as Rothe et al. showed [Rot16]. In the adaption employed in this thesis, though, we do not have such clear training words. First, the training words do not have one exact semantic property. Rather, they are a number of loosely connected words in a LIWC category like “Friends”. Second, the counterexamples are sampled randomly from the 80,000 most common words of the corpus that the embeddings were trained on and probably do not mean the exact opposite of the LIWC words. For these reasons we cannot simply reduce the number of dimensions to one, as Rothe et al. did. The dimensionality reduction process would have to be more complicated and more informed. This leads us to the random forest approach.

$\vec{d}$ itself is independent of the classification method applied to it. Hopes are that a random forest might be better suited for the ultradense subspaces. It has dimensionality reduction built into it, as only a subset of features are selected for one estimator. We weighted the classes the same way as with the SVM and varied the number of estimators and the size of the subset of features inspected when splitting.

Chapter 4

Results

Having explained how the classifiers and feature vectors were designed, we will now present our findings. First, we will take a look at the modus of training and validation. We will continue with an overview over the chosen baselines – the Winter coding system and a simple bag-of-words approach – and finish by detailing the results of the classifiers and the impact of the tested improvement strategies.

4.1 Training and Validation Procedure

For the evaluation of the methods we employed nested cross validation to optimize hyperparameters and test their prediction quality. When a random number generator was used, the seed was fixed to ensure comparability and reproducibility. That was necessary due to shuffling, for example before selecting the training, test and validation subsets and also in the SVM and random forest.

As we stated in chapter 2, there were two challenges to overcome with the datasets. First of all, the number of samples is relatively small to make meaningful predictions. If 10-fold nested cross validation had been applied, only a small number of stories would have been used in the final test set (17 at worst) per outer iteration. To counteract this, we employed 5-fold nested cross validation, effectively doubling the number of stories available for the set. After a completed iteration we reshuffled our dataset and ran another additional iteration to improve the prediction estimates of the measurement.

The aforementioned minimal test set size is actually still too high, because of the second problem: The number of aroused and non-aroused stories per dataset is usually not equal.

This means that were a test set mainly populated by one class and the classifier had a bias towards this class, the accuracy would be incorrectly heightened. Therefore it was necessary to employ stratified cross validation and fix the ratio of aroused to non-aroused stories in the test and validation sets to 50%. This ensures that the accuracy can be interpreted without further adjustments. The remaining stories were not disregarded, but included in the training set. In order to counter this imbalance during training, we weighted the two classes inversely to their proportions.

Hyperparameters were evaluated using a grid search approach, first on a logarithmic scale. In a second run we added more data points around the most promising hyper parameters of the inner cross validation giving it finer granularity. The final test results were stored as JSON including the hyperparameters and are available for download.¹ These files contain accuracy, precision and recall, even when not reported here. The significance of results has been calculated using a two-sided one-sample t-test for the null-hypothesis that the population mean is 50%, i. e. the classifier is guessing. We used the observed accuracies of the outer cross validation as sample population in the t-test. Throughout this chapter we use a significance level of 0.05.

When we are going to talk about an experiment that we ran, we refer to running a nested cross validation of one specific classifier using one specific feature vector on one specific dataset. That is, when we tested the influence of $\vec{d}_{len}$ against $\vec{d}_{l2}$ on all seven datasets for both types of word vectors, for example, we will report having run $2 \times 7 \times 2 = 28$ experiments. When we compare experiments we report on how many experiments improved due to a specific change. For example when comparing $\vec{d}_{len}$ against $\vec{d}_{l2}$, we have run 14 experiments for both. We would then report that using $\vec{d}_{l2}$ improves results in 5 cases and worsens them in 3 cases. The remaining 6 cases are what we call “neutral”. That is when either the results stayed the same or, more often, both compared experiments resulted in insignificant results (for the aforementioned significance level of $p < 0.05$) or results below 50% accuracy. An example of this can be seen in table 4.2, where the rightmost two results would be considered neutral, while the Winter dataset would be considered worse for the 1,2-gram version of the feature vector. It does not make sense to compare these as they are not reliable measurements and can be considered guessing. However, we count results as an improvement when they change from insignificant to significant even when the accuracy declines (because the variance went down, too) and vice-versa. As the tables on which we made these evaluations are quite numerous we have not included them in the appendix. Instead, we show excerpts from them when it is

¹The files can be downloaded at <https://t-animal.de/projekte/masterarbeit/>; see “Resourcen”

useful. They are, however, included with the JSON data and can be either obtained from the computer science chair to which this thesis was submitted or downloaded at the same address as the test results.

4.2 Baselines

Before we continue with the results from the classifiers, we will lay out the baselines to compare against. The Winter coding system serves as a baseline representing human coding and a simple bag-of-words classifier does so for automated systems.

4.2.1 Winter Coding System

The Winter coding system has de facto become the standard coding system for implicit motives, even though there are special coding systems for the three motives it is applicable for [Wei10b].

As Reported by the Author

In his publication Winter reports the results of his integrated running text scoring system in difference of coded imagery (i. e. instances of a hint to a specific motive) per 1000 words. That means he calculated the amount of imagery per story and averaged this over all stories for the aroused and neutral groups. He found significant differences for the McClelland, McAdams and Winter dataset with $p < 0.001$, $p < 0.02$ and $p < 0.005$, respectively. His scoring system did not discriminate well between the 2 groups of the Veroff and Atkinson datasets. The Wirth dataset used in this thesis was not available to him, of course, as it was created years after his publication.

He reports that this shows the validity of his scoring system, but admits that the results are not as conclusive as those of the original systems. This has to be kept in mind when comparing to Winter's results [Win91].

Classifying Individual Stories

While this information certainly shows that the Winter coding system can differentiate between the aroused and non-aroused groups of a motive induction study, it is rather unhelpful when comparing to a binary classifier. That is, in machine learning usually the accuracy of a classifier is reported.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
Accuracy per story	51.75% ± 4.90	50.74% ± 4.17	61.24% ± 4.59	57.35% ± 2.08	57.29% ± 3.04	N/A	N/A
Accuracy per author	52.33% ± 13.3	57.22% ± 6.05	67.41% ± 12.9	52.50% ± 12.5	62.50% ± 5.56	N/A	N/A

Table 4.1: Winter coding system baseline: Accuracy with standard deviation of the cross validation of said system classifying by a simple threshold; gray: non significant results for significance level $p < 0.05$

To make these results comparable, we apply a simple classifier that classifies each story based on a threshold. It uses the amount of imagery in a story according to the Winter coding system as the feature and trains the optimal threshold value (in steps of 0.5) to classify a story as not aroused (less imagery than the threshold) or aroused (more imagery than the threshold). As this classifier does not have hyperparameters we can perform a simple cross validation on it, again making sure that in the test set there is an equal ratio of aroused to non aroused stories. During training, results are weighed inversely proportional to group sizes, because, as mentioned before, the two groups usually are not of the same size.

The results can be seen in table 4.1. Except for Atkinson and McAdams the datasets are separated with about 57% to 61% accuracy. This is remarkable in so far as Winter reported the McAdams dataset to be clearly separable but the Veroff dataset as inseparable.

Classifying Authors

The reason for this disparity might lie in methodology. In psychology it is rather unusual to look at single stories. Usually the results of one individual study participant are aggregated and the sums are evaluated.

This can be emulated in the baseline classifier by averaging the number of stories per author and training a threshold on the resulting value. Doing so confirms the results by Winter pretty closely and the results are displayed in table 4.1, too. The Atkinson and Veroff dataset cannot be clearly separated, while the other three datasets are separable to some degree. This comes from the fact that sometimes a single story will contain several codeable instances of imagery while others of the same author might contain none. When averaging over the stories of an author and applying a new threshold, datasets can be classified better or worse.

While 57% to 67% may not seem high, it seems this is comparable to other coding systems. Unfortunately most authors of original coding systems did only describe the total difference

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
using	62.2%	51.92%	61.01%	62.35%	56.36%	54.78%	47.99%
1-grams	± 4.88	± 5.87	± 6.35	± 3.1	± 7.3	± 6.83	± 7.73
using	62.50%	54.75%	60.85%	63.09%	54.55%	52.5%	46.8%
1,2-grams	± 4.30	± 5.64	± 6.27	± 3.92	± 6.1	± 7.12	± 5.46

Table 4.2: Bag-of-words baseline: Accuracy with standard deviation of using a bag-of-words of n -grams classified by an SVM; gray: non significant results for significance level $p < 0.05$

in imagery, but at least Veroff reports the numbers of subjects above and below the median imagery count. He reports that 69.11% of authors are on the correct side of the median [Ver57]. To correctly compare this number one has to keep in mind that it was not created by cross validating the threshold and thus probably is a bit higher than it would have been when applying our baseline threshold classifier.

4.2.2 Bag of Words

When the Winter coding system is the psychologist’s baseline, the bag-of-words classifier is the computer scientist’s. As we described in section 3.3.1 the bag-of-words approach is the simplest classifier we tested, merely using the TF-IDF and classifying using an SVM.

With this classifier the size of the n -grams can be varied. We found that using both 1-grams and 2-grams yielded the best results with an average of 56.43%. Using only 2-grams decreased performance on all datasets but McAdams, where it increased accuracy by about 2 percentage points. The results for the 1-gram and 1-and-2-gram classifier are depicted in table 4.2.

This outcome is worth pausing a moment for. After all, this simple classifier seems to be on par with the Winter coding system and even beats it clearly on some datasets (this result is per story, so it has to be compared to the first line in table 4.1). This does correspond with the findings by Ring and Pang, who found that a bag of words on less noisy data reaches an accuracy of up to 84% [Rin17].

The poor performance on Wirth_{Same} raises the question, whether it only performs author attribution, a question we will come back to in chapter 5. This dataset is the only one where stories in the aroused and non-aroused group were written by the same authors. However, the performance on Wirth_{Diff} is not significant either, albeit better. It seems more probable that the problem is just the small dataset size. This explanation is supported by the fact that on the other small dataset, Winter, it does not have good results either.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
400d wiki-corpus	53.21% ± 3.95	56.85% ± 7.82	55.82% ± 5.18	52.94% ± 4.60	50.36% ± 8.75	53.15% ± 6.66	50.8% ± 4.78
600d wiki-corpus	53.54% ± 4.07	55.64% ± 8.83	56.0% ± 4.26	53.97% ± 3.36	51.71% ± 8.31	53.38% ± 8.96	53.23% ± 6.41
400d wiki-corpus _{ppdb}	55.46% ± 6.43	54.01% ± 6.26	53.49% ± 5.43	57.35% ± 4.65	51.12% ± 5.7	53.15% ± 5.93	52.17% ± 7.14

Table 4.3: Accuracy of classification with a k -NN classifier using the Word Mover’s Distance on GloVe vectors of different sizes; gray: non significant results for significance level $p < 0.05$

4.3 Results for Individual Techniques

Now that we have established baselines to compare against, we can evaluate how the methods that we have tested work in comparison.

4.3.1 Word Mover’s Distance

The Word Mover’s Distance compares documents based on the minimal distances words have to travel in their embedding space to become the words of the other document. As it is suitable only for comparing two documents with each other, the natural choice is a k -NN classifier.

Unfortunately it seems the WMD is not suitable for the task at hand. It does not yield higher accuracy than 57% on any dataset and only on five of the seven datasets does it produce any results that are significant with $p < 0.05$. We present these findings in table 4.3, where only the results using GloVe vectors trained on the Wikipedia corpus are displayed. The results using word2vec embeddings are not better, though, as shown by initial experiments using the pre-trained word vectors provided by Mikolov et al. As the classifier is one of the slowest we did not re-run the word2vec tests with our self-trained vectors in our final tests due to time constraints and machine availability.

This low performance probably comes from a combination of factors. The WMD might just be too rough a measure to capture the fine differences in semantics that are necessary to differentiate between the two groups. Albeit possible, this seems improbable, because the centroid feature vector is a pretty rough measure, too, and produces better results even in its simplest form. The bag-of-words classifier does not take any semantics into account and it also produces better results. This leads to the assumption that the training data is the underlying issue.

A k -NN classifier is by nature relatively prone to errors from noisy data sets, unless k is chosen high. As our baseline and the principal component analysis of the centroid vector shows, it seems that our dataset is pretty noisy (an example of the PCA of $\vec{d}_{l2,nostop}$ of a dataset can be seen in fig. 5.1). Intuitively this can be explained as some people in the control group might just naturally be very motivated by the motivational trait researched. They are not distinguishable from an averagely motivated person after arousal. The same goes for people with low natural motivation in the aroused group. We will go deeper into the issue of noisy data in chapter 5.

This reasoning does not make it seem probable that retrofitting the word vectors does improve the detection results. After all, this process cannot reduce the noise in the data but only shift the word vectors in the embedding space to improve semantic and syntactic relationships. However, this could lead to an improvement in the representativeness of the WMD and thus might cancel out the detrimental effects of noisiness. We tested word vectors retrofitted with both wns+ and ppdb. Both consistently improved the results on the Atkinson and Veroff dataset by up to 4.4 percentage points. However the accuracy on other datasets are mostly either improved only marginally or even negatively. An example is shown in table 4.3. We can see that the paraphrase database seemed to worked better on average than the wordnet synonyms, but this is probably due to chance on this level.

4.3.2 Centroid Vector and SVM

Let us now shift the focus from measuring the distance between documents and instead aggregating a document into a single document vector, that can be used for classification. This aggregation is done by summing up all word vectors of a document and normalizing the result. We will look into the findings on normalization using document length and the Euclidean norm, how stop word removal affects the results and whether they could be improved by training word embeddings on different corpora. Lastly we will see if there is a benefit to retrofitting with synonym information and building an ultradense word space using LIWC categories.

For this section we conducted experiments varying the removal of stop words, the norm applied, vector size, word embedding type (GloVe or word2vec), word embedding training corpus and whether they were retrofitted with a synonym database or not. For all experiments all these parameters were fixed to a specific value and other hyperparameters, like parameters for the SVM, were varied. For details on our training and validation scheme see section 4.1.

Influence of the Embeddings Training Corpus and Vector Size

First, we will answer the question whether there is an influence of the training corpus on which the word embeddings are trained. We have trained word embeddings on Wikipedia articles and a specially compiled corpus of fan fiction, which are stories written by nonprofessional authors. The reasoning behind this is that word embeddings are derived from word co-occurrence and this is probably different between encyclopedic articles and fictional stories.

To test whether there is a difference between the embeddings, we gathered the results from all experiments and grouped them by training corpus and by their vector size. We will call vectors of dimensionality 400 “400d” and those of dimensionality 600 “600d”. The results of these tests in detail and a more easily interpretable table form can be downloaded with the results’ JSON data. As they amount to about 50 tables we did not put them in the appendix of this thesis. We have performed a total of 224 experiments. We grouped them by dimensionality (112 experiments per group), both dimensionality and vector type (56 experiments per group) and training corpus (112 experiments per group). We then ran paired t-tests on the groups to find out if there was a difference in accuracy between them. This does not just allow for a more informed decision but also seemed the only feasible option due to the large number of experiments.

We found no significant difference between the 400d and 600d vectors ($p = 0.4159$). It seems that increasing the word vector size does not increase accuracy in our experiments. The mean accuracy of all experiments performed with the 400d vectors is 55.87% and 55.74% when using the 600d vectors. Seemingly very little additional useful information is contained in the word embeddings when going from 400 to 600 dimensions – at least no information improving our classifiers structurally.

We re-ran the t-test separately for the GloVe and word2vec embeddings, to see if the increase in dimensions yielded significantly different results on just one of them. This shows that the bad results before were mainly due to the word2vec embeddings, where we cannot find a significant difference ($p = 0.7995$). The GloVe embeddings might have a very weak tendency, but with $p = 0.1438$ these results are not reliable either.

The comparison of the word vectors grouped by training corpus revealed a difference with $p = 0.0461$. However, with the mean difference between the two groups being only 0.5313 percentage points (in favor of the Wikipedia corpus) they can be thought of practically equally useful for our task. Looking at the results from our experiments, one can come under the impression that the embeddings trained on the fanfiction corpus do a little better on the McAdams and Winter datasets, while the Wikipedia embeddings perform better on the Wirth datasets. But on a general level these differences seem to cancel out nearly completely.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
GloVe with stop words	63.45% ± 6.26	55.82% ± 5.95	60.24% ± 10.7	56.91% ± 3.72	50.26% ± 10.23	51.34% ± 2.05	49.36% ± 5.18
GloVe w/o stop words	60.43% ± 5.46	53.44% ± 4.12	61.96% ± 9.35	61.62% ± 3.97	44.9% ± 7.45	46.44% ± 3.59	46.79% ± 6.76
w2v with stop words	59.95% ± 5.5	52.37% ± 5.52	65.25% ± 8.74	59.85% ± 4.26	57.62% ± 8.71	47.52% ± 4.96	46.15% ± 3.56
w2v w/o stop words	56.94% ± 6.73	54.06% ± 5.85	62.39% ± 4.02	57.65% ± 3.53	52.79% ± 6.41	48.18% ± 5.0	49.57% ± 6.58

Table 4.4: Influence of stop word removal on the accuracy of classifying $\vec{d}_{len}$ of 400d vectors trained on ffcorpus using an SVM; gray: non significant results for significance level $p < 0.05$

This does not mean conclusively, however, that the assumption that training word embeddings on similar texts improves word embedding quality for a given task, is wrong. For this, more tests would be needed. But for the purpose of this thesis, it seems that training the word embeddings on the fanfiction corpus or the Wikipedia corpus produces very similar results. This could come from the fact that the fan fiction stories are not similar enough to the stories written in PSEs or TATs. Alternatively texts from the Wikipedia might have a similar enough writing style to our datasets that switching to another corpus does not produce any better results.

To reduce the amount of data in upcoming analysis we therefore disregard the 600d vectors and the results from the Wikipedia embeddings and focus on the 400d vectors based on the fanfiction corpus. The decision for the fanfiction embeddings was mainly a practical one: We had already computed some of the longer running jobs using them.

Influence of Stop Word Removal

One approach to improving the classification results is removing stop words. These words usually carry little meaning and do not contribute much to the meaning of a text. The 100 most used words in the Wikipedia make up the stop word list, except some that seemed meaningful. In order not to lose frequency information, the number of removed occurrences of each word was appended to the vector. The influence of stop word removal can be seen in table 4.4 and these results are a bit surprising.

Removing stop words actually decreases the performance on almost all datasets for the word2vec word embeddings. An exception are both Wirth datasets and the McAdams dataset.

The Wirth datasets, as the Winter dataset, are hard to classify due to their small number of stories – training sets are smaller and smaller test set size increases variance. This explains why removing stop words leads to seemingly random jumps in accuracy. While accuracy on the McAdams dataset seems to be positively influenced, that is not enough to shift the results over the 0.05 significance level.

GloVe word embeddings seem to profit at least partly from removing stop words, again leaving the Winter and Wirth datasets aside. Even the Atkinson and McAdams dataset, although losing in accuracy, at least have a reduced variance after removing stop words.

At first glance, an explanation of this phenomenon might lie in the research of Ring and Pang, who identified the most informative words in their bag-of-words classifier. They talked about them in their 2017 presentation in Erlangen and listed several words as most informative for nPow. Some of them are included in our stop word list [Rin17]. But this does not explain why the GloVe based feature vectors are not affected as much. Particularly as the performance on the Veroff dataset profits and it originates from *nPow* research.

Influence of the Norm Applied

We looked into two different norms used for compensating for the different lengths of stories. The first one to come to mind is normalizing by dividing by the number of words aggregated, yielding $\vec{d}_{len} = \frac{1}{n} \sum_{\vec{v}_i \in D} \vec{v}_i$. Usually for normalization of vectors the Euclidean norm is applied, though, leading to $\vec{d}_{l2} = \frac{\sum_{\vec{v}_i \in D} \vec{v}_i}{\|\sum_{\vec{v}_i \in D} \vec{v}_i\|_2}$. Results of experiments with the norm are shown in table 4.5.

The findings are similar to those of stop word removal. When using word2vec embeddings, results are affected in a mixed way, with accuracy improvements on two of the four large datasets. Dividing the sum of GloVe word vectors by the $l2$ norm increases accuracy on three of these datasets. In any case the standard deviation of the accuracy during the nested cross validation is going down for the GloVe embeddings and most of the time for the word2vec ones (for the significant results), which is positive for the quality of the classifier.

It may seem a bit confusing at first why applying the Euclidean norm outperforms the more natural choice of dividing by the number of summed up elements. The answer might be in the values of the individual word vectors. They typically lie in the range between $[-6; 6]$ and using the document length for normalization maintains that range. Dividing by the $l2$ norm on the other hand normalizes the feature vector into the range $[-1; 1]$, which may have a positive influence on the performance of the SVM.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
GloVe $\vec{d}_{l2}$	62.17% ± 4.37	57.48% ± 4.05	61.38% ± 10.17	57.35% ± 3.22	49.36% ± 12.07	51.5% ± 5.07	52.61% ± 8.99
GloVe $\vec{d}_{len}$	63.45% ± 6.26	55.82% ± 5.95	60.24% ± 10.70	56.91% ± 3.72	50.26% ± 10.23	51.34% ± 2.05	49.36% ± 5.18
w2v $\vec{d}_{l2}$	58.66% ± 3.17	54.49% ± 4.66	63.51% ± 5.78	60.44% ± 3.32	55.5% ± 10.44	47.5% ± 4.45	48.18% ± 6.92
w2v $\vec{d}_{len}$	59.95% ± 5.50	52.37% ± 5.52	65.25% ± 8.74	59.85% ± 4.26	57.62% ± 8.71	47.52% ± 4.96	46.15% ± 3.56

Table 4.5: Influence of using the document length ($\vec{d}_{len}$) versus the Euclidean norm ($\vec{d}_{l2}$) for normalization on 400d vectors trained on ffcorpus; gray: non significant results for significance level $p < 0.05$

Influence of Retrofitting

While stop word removal and normalization work with the individual word vectors of a document, retrofitting changes the actual word embeddings as a whole. For this thesis we retrofitted synonym information from two different databases which should align word vectors better, an effect that might accumulate when summing up the vectors and thus increase performance.

To evaluate if the retrofitted word vectors resulted in better accuracy, we collected all experiments with the centroid vector classifier. That is all experiments varying norm, stop word removal, embeddings training corpus, norm for normalization, word embeddings type and of course retrofitting with either the paraphrase database or the wordnet synonyms. In total that makes up 448 experiments (224 per retrofitted database). Then we compared the performance of the classifier using retrofitted word vectors with the results from the respective same experiment using non-retrofitted word vectors (another 224 experiments). Again, the tables that were evaluated are available with the experiments' JSON data.

The first observation is a confirmation of the findings in the paper by Faruqui et al. They found that the word2vec word embeddings profit more from retrofitting in semantic and syntactic evaluation benchmarks [Far15]. In our experiments, retrofitting improved results a total number of 96 times (42.86%) and deteriorated them 55 times (24.55%) when applied to word2vec. The experimental outcomes with GloVe embeddings were improved 84 times (37.50%) and worsened 65 times (29.02%)².

²Relative numbers do not add up to 100% due to about a third neutral, insignificant outcomes which we did not compare as they might just be guessing and do not justify definite statements

Second, we can observe that the effects of retrofitting are especially pronounced when also removing stop words. In 104 (46.43%) experiments with stop word removal retrofitting improves results in contrast to 76 (33.93%) experiments without stop word removal. This effect probably comes from the fact that retrofitting works using synonym information. Stop words, however, carry little meaning and often do not have synonyms at all. This means that the procedure probably does not work well on them or at least not in the sense of improving their semantical meaning. When they are kept in the stories they decrease the effect of the method on the classification rate, because they are also taken into account when creating the document vector but may not have been improved.

The last finding is that the Veroff dataset in particular is classified much better after applying retrofitting using either database: performance increases in 76.56% of experiments while for the other datasets the percentage is between 42.19% and 59.37%.

We could not find any other patterns in the data, though. All in all, the retrofitting procedure seems to lead to pretty random improvements. There can be no doubt that it does have an impact on the results. We were unable to make out, under which circumstances it is positive, though. On all datasets with all other configurations of our classifier we did not find a single configuration in combination with which the retrofitting guaranteed higher accuracy. Neither did we find strong evidence that using either of the two synonym databases has more benefits than the other: The paraphrase database increased performance in 93 (41.52%) of experiments while the wordnet synonyms improved it in 87 (38.84%) of cases.

Combining These Findings

In essence, the GloVe vectors were, at least in parts, positively affected both by removing stop words and by using the Euclidean norm when normalizing and also by retrofitting. We can therefore hope that applying all techniques increases the accuracy for these vectors even further. As removing stop words did not work well for the word2vec embeddings we will not consider them here. The results of our experiments in this regard are shown in table 4.6.

It is remarkable that the combination of removing stop words and normalizing by the l_2 norm improves the detection rates on almost all datasets, especially on the small ones Winter and Wirth_{Diff}. Wirth_{Same} appears to be influenced positively, too, just not quite enough to make the results significant ($p = 0.0844$). The Atkinson dataset is classified better than with any of two methods applied alone. For the McClelland and Veroff datasets the improvements lie below applying stop word removal only, but the result is still better than for $\vec{d}_{len}$, i. e. including stop words and not using the l_2 norm. The McAdams dataset is the opposite of that, meaning

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
GloVe _{400d}	64.73% ± 4.32	54.34% ± 4.94	61.62% ± 7.5	60.44% ± 4.98	56.95% ± 4.66	57.63% ± 8.35	53.61% ± 5.58
GloVe _{600d,wns+}	63.61% ± 3.78	54.64% ± 3.53	61.02% ± 7.17	61.62% ± 3.57	56.93% ± 6.47	56.50% ± 7.10	53.83% ± 4.04

Table 4.6: Accuracy of classifying $\vec{d}_{l2,nostop}$ of GloVe vectors trained on ffcorpus using an SVM; gray: non significant results for significance level $p < 0.05$

it performs less well than when only removing stop words, but better than the starting point. We cannot be entirely sure, however, that the good results on the McAdams dataset are not just due to chance, as they do not appear when using the wikipedia embeddings and not using the 600d fanfiction vectors, either.

The general improvement is not a random effect, though. The combination of stop word removal and normalization by the Euclidean norm improves classification rates on the smaller datasets in almost all experiments we performed.

Applying retrofitting to this specific combination of methods does not improve overall accuracy, which is why we do not display it in the table.

After we had discussed the influence of the dimensionality of the word embeddings we disregarded the 600d word vectors for the rest of this section and the findings we discussed so far apply to them, as well. But we must come back to it here, because using $\vec{d}_{l2,nostop,wns+}$ of 600d word vectors is the only combination of methods that we found, where our classifier had significant results on all datasets. We added these results to table 4.6. However, we believe the slightly better results on Wirth_{Same} might merely be due to chance and not due to structural differences between the 400d and 600d vectors. As it is hard to tell this for sure, though, we will compare both to our baselines.

Influence of Including Image Information

The last bit of information that we did not include in the feature vector until now is for which image a story was written. This was possible only under the assumption that the images were given to all participants of a study in the same order. Then simply another dimension with an increasing number representing the image id could be added to the feature vector. However, we found this had almost no effect, leaving the accuracy on most datasets the same.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
$\vec{d}_{l2}$	55.61% ± 4.29	56.7% ± 6.38	57.01% ± 6.58	57.35% ± 5.05	46.52% ± 8.52	50.63% ± 5.19	50.84% ± 6.11
$\vec{d}_{l2,ultradense}$ category 1	54.65% ± 5.16	56.73% ± 4.2	58.08% ± 6.12	56.76% ± 7.7	53.17% ± 6.43	48.64% ± 7.85	50.53% ± 7.62
$\vec{d}_{l2,ultradense}$ category 2	56.08% ± 5.7	54.35% ± 3.83	55.95% ± 6.54	57.79% ± 6.07	53.79% ± 8.14	49.1% ± 6.75	50.53% ± 7.63

Table 4.7: Accuracy of classifying $\vec{d}_{l2}$ of 400d GloVe word embeddings trained on ffcorpus using a random forest; with and without applying ultradense word embeddings transformation; gray: non significant results for significance level $p < 0.05$

Influence of Ultradense Word Embeddings

Due to the orthogonal transformation being transparent to the classification by an SVM we did not expect improvements due to ultradense word embeddings. We still ran some short experiments with them to check this assumption and could confirm it. Instead, we focus on random forests in combination with the ultradense subspaces.

4.3.3 Centroid Vector and Random Forest with Ultradense Word Embeddings

Generally speaking the random forest classifier performs worse (about two to five percentage points) than the SVM which is why we will not go into as much detail here as before. The classifier did not perform well on the small datasets Winter, Wirth_{Diff} and Wirth_{Same}. On these three datasets it produced a significant result with more than 50% accuracy in only three experiments out of 672 (that is varying datasets, retrofitting, word embeddings, training corpus, etc). A typical result can be seen in the first line of table 4.7.

Stop word removal did not structurally improve the classification rates. This is especially true for the word2vec embeddings where it decreased performance twice as often as it increased it. Using the Euclidean norm for normalization has a positive effect in general, improving results in 62.50% of non-neutral experiments. While, when using an SVM, the combination of the $l2$ norm and stop word removal improved results more consistently than each on its own, we could not see such an effect with the random forests. The influence of retrofitting is mixed and depends a lot on the datasets. The results on the McClelland and Veroff datasets tend to gain more and suffer less from retrofitting than on the Atkinson and McAdams datasets. However the negative effects on the latter two are so pronounced that using retrofitting with the random forest classifier is not an option.

The main question we wanted to answer with this classifier, though, is whether the ultradense subspace modifications are beneficial or not. For each motivational trait we selected the two LIWC categories with the highest correlation according to the research by Schultheiss [Sch13a]. Then we trained the rotation matrix for the ultradense word embeddings on them and multiplied the feature vector with the resulting rotation matrix $\mathbf{Q}$.

Before we go into detail on the results let us quickly look at whether the subspace optimization succeeded. The original implementation as proposed by Rothe et al. worked on words that could be clearly marked as positive or negative [Rot16]. For this thesis, though, the algorithm is used to train on LIWC categories and a set of randomly sampled words from the 80,000 most common words in the wikicorpus. Therefore it is not easy to test whether the rotation actually optimized the semantic meaning. To do so one needs a certain number of words with the same semantic meaning as the words from the LIWC category, which are not contained in it. We tried this by training the rotation matrix on the category “Family” and manually tested several synonyms (as suggested by an online service) for words of the category. We could see that the rotation had seemed to work. However as this process is very tedious we did not repeat it for the categories used in our classifier. But as we have seen that the method works for sentiment and could verify it on this LIWC category, we are confident that the optimization works.

The results of the experiment are shown in table 4.7, containing the classification results of the random forest classifier without applying the ultradense transformation and with the rotation applied for both chosen LIWC categories. The ultradense word embeddings did not change the results of the classifier much. The McClelland dataset is the only one improved by more than half a percentage point and that only by one of the two categories (“Optimism”). With word2vec embeddings results are similar to those displayed in the table. All in all, using the ultradense subspace does not seem to be beneficial.

This could have a number of reasons. First, the optimization on the LIWC categories might not have worked. As we saw an effect with the “Family” category we believe they have, though. Possibly the optimization would have to be performed on more than one axis, so that it is more likely that the random forest selects one of the optimized dimensions in an estimator. We optimized only one axis to represent the semantic content of the category, which might just not be enough. The most probable explanation, though, is that the corresponding axis of the feature vector was not very discriminatory. After all, the correlations in Schultheiss’ research were mostly weak to moderate and discriminated well between the two classes only in combination with each other [Sch13a]. One solution might be to train for more than one category, but this introduces even more prejudice into the system, which we tried to avoid.

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
No clustering	64.73% ± 4.32	54.34% ± 4.94	61.62% ± 7.5	60.44% ± 4.98	56.95% ± 4.66	57.63% ± 8.35	53.61% ± 5.58
$k = 2$	63.45% ± 4.46	58.10% ± 3.48	59.46% ± 7.60	57.50% ± 5.03	56.60% ± 8.22	54.97% ± 5.75	50.18% ± 6.92
$k = 3$	62.64% ± 4.34	56.75% ± 5.24	60.45% ± 4.70	56.62% ± 4.80	53.74% ± 9.08	53.75% ± 4.89	48.99% ± 5.88
$k = 4$	62.02% ± 4.88	55.23% ± 4.46	61.59% ± 7.42	56.18% ± 6.16	51.60% ± 9.21	52.03% ± 6.40	52.82% ± 6.54

Table 4.8: Accuracy of classifying $\vec{d}_{l2,nostop}$ of 400d GloVe vectors trained on ffcorpus using an SVM; with and without applying k -means clustering before creating the centroids; gray: non significant results for significance level $p < 0.05$

4.3.4 Clustered Centroids

The feature vector that we looked at so far is a very rough representation of a document. The benefits of word embeddings get a bit lost in the process of summing up the vectors. Before constructing the centroid we have a number of word vectors pointing into the embedding space in possibly completely different directions. Thereafter we are left with only one vector that quite probably points into a direction not representing any of the vectors it was calculated from. To properly represent a story, more than one vector might be needed, which might be achievable by clustering the word vectors before summing them up.

K-Means Clustering

The first method we researched is k -means clustering. Starting from our experiments without clustering we ran the classifier using the 400d vectors trained on the fanfiction corpus and varied removing stop words and the norm applied. We also varied the number of clusters from two to four.

As expected, removing stop words and using the Euclidean norm ($\vec{d}_{l2,nostop}$) for normalization performs best with k -means clustering, too. Results using this feature vector are displayed in table 4.8. While this confirms the findings from section 4.3.2, the more interesting question is how the number of clusters formed influences classification rates. We found that increasing the number of clusters does not necessarily increase performance and we could not see a clear pattern when more clusters lead to better accuracy. In any case the best-performing number

of clusters in almost all cases does not beat the classifier without clustering. This can also be seen in the table.

The idea behind the clustering method was to find dense regions of words from a story in the embedding space. As synonyms or synonymous information should be closer together in it, they should be clustered together. Therefore one reason why clustering does not seem to improve performance might be that synonyms are not close enough together. However, the retrofitting procedure is designed to shift synonymous words closer together in the word embedding space, so we re-ran our experiments using the retrofitted word vectors.

Comparing the results from clustered feature vectors before and after retrofitting we can observe that using the paraphrase database works a lot better, with 42 experiments improving performance against 26 when using the wordnet synonyms. In relative numbers (again counting experiments where neither resulted in significant classification rates or less than 50% accuracy as neutral) that is an increase in 50.00% and 30.95% of experiments and a decrease in 21.43% and 36.90%, respectively. Once again we can see that the accuracy on the Veroff dataset seems to profit more than the others and again more clusters generally do not mean higher accuracy. Although more than two thirds of the non-neutral experimental outcomes gained from using word embeddings retrofitted with the paraphrase database, there was no case in which enough datasets were improved in comparison to the unclustered feature vector to actually improve upon it regarding mean accuracy across all datasets.

For all datasets there is one combination of stop word removal, norm and number of clusters that improves the accuracy of the classifier. For example, classifying $\vec{d}_{len,nostop,ppdb,k4}$ gets accuracy on the McClelland dataset up to 64.49%. For the Veroff dataset, however, the best combination is $\vec{d}_{l2,nostop,wns+,k2}$, yielding 61.03% accuracy. As these peaks do not seem to follow a pattern, we conclude that they are not due to structural improvement through clustering, but rather due to chance.

It seems that the performance is still dominated by stop word removal and the norm used for normalization. In general the accuracy increased and decreased similarly as without clustering.

One might be inclined to attribute the general drop in performance to the curse of dimensionality. After all, the feature vector size is doubled to quadrupled compared to the unclustered one. The SVM might just overfit in the inner cross validation and thus the accuracy in the outer cross validation might suffer. In this case, though, the average accuracy in the inner cross validation should have risen compared to the unclustered classifier and this is not the case. For $k = 4$ it sank from 61.96% to 59.81%.

The clustering reduces the number of words going into one of the concatenated centroids $\vec{c}$ (in comparison to using all words at once to calculate $\vec{d}$), though. Our goal was to find words having a similar meaning by looking for clusters in the embedding space and thus to reduce the amount of overlaid semantic information. Fewer words mean, however, that noise is less likely to cancel out. Possibly this leads to a deterioration of the feature vector that cannot be compensated for by the clustering.

4.3.5 Agglomerative Clustering by Cosine Dissimilarity

Another possible explanation could be that clustering by the Euclidean distance is simply not adequate. Maybe a coarser measure is necessary, such as the general direction of the vectors, as the cosine similarity provides. The k -means algorithm is not defined for other distance metrics than l_2 , though, which is why we turned to agglomerative clustering.

Using the length of a cluster for normalization seems to be an unsuitable norm in combination with this clustering method. Applying it reduced the accuracy to the chance of guessing in most cases. We therefore focused on the Euclidean norm. The cosine similarity and Euclidean distance are very similar on vectors that have been l_2 -normalized. However, the normalization is performed after the clustering, so that observation does not apply.

Unfortunately clustering by the cosine similarity does not work well either. It hardly ever improves upon the classifier without clustering and when it does, these improvements mostly seem to be random. The only pattern that we could make out is that the McClelland dataset benefits more from this clustering method than the others. Most datasets, in fact, are impacted negatively.

As with the k -means approach it might be possible that retrofitting improves the results. As the results are still far worse than the unclustered method we will not go into detail, though. We found that using the wordnet synonym database improves results more often than the paraphrase database. Interestingly, the Wirth_{same} dataset is classified with 58.23% accuracy when clustering into four clusters and applying retrofitting with wns+. However, as other datasets are impacted very negatively, we did not pursue this method further.

It is quite probable that in the pursuit of applying a coarser measure, we chose one that is simply too rough. This leads, in combination with the postulated effect of fewer words adding up in one cluster, to worse results.

4.4 Comparison to Baselines

Now that we have detailed the different approaches that we tried out, we will compare the best-performing two to our baselines. We compare the results from both the 400 dimensional $\vec{d}_{l2,nostop}$ and the 600 dimensional $\vec{d}_{l2,nostop,wns+}$ trained on the fanfiction corpus and classified by an SVM. For the Wirth datasets we do not have human coded data available, so a comparison is not possible against the Winter coding system. However, we can compare them to the bag-of-words classifier, or the computer scientist’s baseline, as we called it earlier.

4.4.1 Accuracy per Story

First, we are going to evaluate the performance per story, as we did throughout this chapter. As table 4.9 shows, both feature vectors beat the Winter coding system clearly on the Atkinson, McAdams and Veroff dataset. They perform about the same on the stories compiled by McClelland and are beaten narrowly on the stories from Winter’s own research.

The advantage becomes smaller when comparing to the bag-of-words approach, though. There are some improvements of the results on the Atkinson dataset and minor ones on the McClelland dataset. It is beaten on the Veroff dataset and narrowly on McAdams’. However turning to the smaller datasets Winter, Wirth_{Diff} and in parts Wirth_{Same}, the use of word embeddings pays off.

Viewed from this perspective, we can say that we have created a method that is eye to eye with the Winter coding system and improves upon the simplest approach of counting words at least for small datasets.

4.4.2 Accuracy per Author

As we stated before, though, in the field of psychology a different perspective is common. Usually results are not assessed per story, but the results of all stories of an author are aggregated. Only when we applied a very simple thresholding classifier to the image counts per author we could confirm the observations Winter stated in his paper. He found a significant difference in coded motive imagery in all datasets except Atkinson and Veroff.

To compare to this view we aggregated the results of the classifiers per author, too. As we used an SVM for classification we could simply assign a confidence to the classification result of each story by taking the distance of its feature vector to the separating hyperplane. We took the results of the outer cross validation (i. e. the results that make up the accuracy in table 4.9)

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
Winter co-coding system	51.75% ± 4.90	50.74% ± 4.17	61.24% ± 4.59	57.35% ± 2.08	57.29% ± 3.04	N/A	N/A
Bag of words 1,2-gram	62.66% ± 4.16	54.75% ± 5.64	60.85% ± 6.27	62.94% ± 3.65	54.55% ± 6.1	52.5% ± 7.12	50.63% ± 6.82
$\vec{d}_{l2,nostop}$	<u>64.73%</u> ± 4.32	<u>54.34%</u> ± 4.94	<u>61.62%</u> ± 7.5	<u>60.44%</u> ± 4.98	<u>56.95%</u> ± 4.66	<u>57.63%</u> ± 8.35	53.61% ± 5.58
$\vec{d}_{l2,nostop,wns+}$	<u>63.61%</u> ± 3.78	<u>54.64%</u> ± 3.53	<u>61.02%</u> ± 7.17	<u>61.62%</u> ± 3.57	<u>56.93%</u> ± 6.47	<u>56.50%</u> ± 7.10	<u>53.83%</u> ± 4.04

Table 4.9: Comparison of classifying a centroid vector using an SVM against baselines; results are per story; gray: non significant results for significance level $p < 0.05$; **bold**: beats the Winter baseline; underlined: beats the bag-of-words baselines

of our validation procedure as described in section 4.1 and summed up the signed distances of all stories of one author. The result is either a positive or negative floating point number representing that the author was part of the aroused or non-aroused group. As there are no parameters to optimize, there is no variance to the result that could be reported. This might seem unfair, as cross validated results typically are worse than when using a fixed value, but the aggregated distances come from a cross validation themselves and therefore do not have an advantage.

The accuracy of this aggregation is shown in table 4.10³. All three computer-based methods are clearly in the lead now. The word embeddings based classifier beats the Winter coding system on all datasets, at times by over twenty percentage points. It manages to classify authors even on those datasets that are opaque to the Winter coding system. The advantage over the bag-of-words approach increases, too, with only the authors of the McAdams dataset being classified better by it.

The drop in percentage on Wirth_{Same} when using the 600d $\vec{d}_{l2,nostop,wns+}$ feature vector feeds the suspicion that the good results on this dataset on the per story level were only due to chance and not a result of improvements due to retrofitting, as we said before.

³The results on the McClelland dataset are not a typo

	Atkinson	McAdams	McClelland	Veroff	Winter	Wirth _{Diff}	Wirth _{Same}
Winter coding system	52.33% ± 13.3	57.22% ± 6.05	67.41% ± 12.9	52.50% ± 12.5	62.50% ± 5.56	N/A	N/A
Bag of words 1,2-gram	70.49%	59.52%	68.47%	76.47%	58.62 %	52.54%	45.16%
Centroid $\vec{d}_{l2,nostop}$	<u>75.41%</u>	<u>58.33%</u>	<u>68.47%</u>	<u>77.94%</u>	<u>68.97%</u>	<u>59.32%</u>	<u>61.29%</u>
Centroid $\vec{d}_{l2,nostop,wns+}$	<u>73.77%</u>	<u>61.90%</u>	64.87%	<u>75.00%</u>	<u>65.51%</u>	<u>55.93%</u>	48.39%

Table 4.10: Comparison of classifying a centroid vector using an SVM against baselines; results are per author; gray: non significant results for significance level $p < 0.05$; **bold**: beats the Winter baseline; underlined: beats the bag-of-words baselines

4.5 Results Across Datasets

Apart from the question if we can classify texts of a dataset, we also asked “If so, can this classifier be trained on the texts of one study and be used on those of another?” when stating our research goals. As we were not contempt with the performance of our classifier within datasets, we did not explore how to modify it so that it works across datasets - we wanted to get into reasonably high accuracy ranges within a dataset first.

Still we can apply the classifiers and see if they work across datasets, too. To do so we optimized the hyperparameters on a dataset using cross validation. Then we used these hyperparameters, trained on the whole dataset and tested on all others concerning the same motivational trait. As we have only one *nAch* dataset (McClelland) we could not test that. However we trained on Veroff and tested on Winter and vice-versa (for *nPow*) and also on all 2-tuples of Atkinson, McAdams, Wirth_{Same} and Wirth_{Diff} (for *nAff*). To evaluate the significance of the results we compared to the one-sided binomial 5% confidence interval.

No combination of training and test dataset yielded significant results, except the combinations of Wirth_{Same} and Wirth_{Diff}. However these cannot be taken into consideration, as the aroused group was the same for both datasets. Training on the stories compiled by Atkinson and testing on McAdams’ resulted in 54.10% accuracy having a p-value of exactly 0.10, which could be interpreted as a trend. But as all other results were insignificant and thus basically reduced to guessing, this one result is probably rather due to chance.

Chapter 5

Discussion

Looking at the results and the comparison against the baseline, the question is: was the goal of classifying stories according to their author's arousal or non-arousal met? We will now look into this question and also discuss other observations that have come up in the previous chapters. As we have already discussed the implications of applying different methods when detailing their results, we will focus on the preeminent and recurring points in this chapter.

5.1 Remaining Issues

When merely looking at the data as presented in section 4.4, one could say that we have devised a method that is better than the Winter coding system at the task of separating the two groups of a motive induction study. However, these results must be taken with a grain of salt and a couple of issues remain.

5.1.1 Low Accuracy

First of all, while in comparison with the Winter baseline the accuracy is higher, it is still far from the reliable system that ideally could be aimed for, with only about 60% accuracy when applied per story.

There are a couple of reasons that make the classification task a rather hard one. We have already touched upon the problem that is inherent to motive induction studies: there are people who, by nature, are highly motivated by, for example, *nAff*. When they take part in a motive induction study and are assigned to the non-aroused group, they diminish the difference between the two groups. This is, of course, not a problem specific to *nAff*, but to

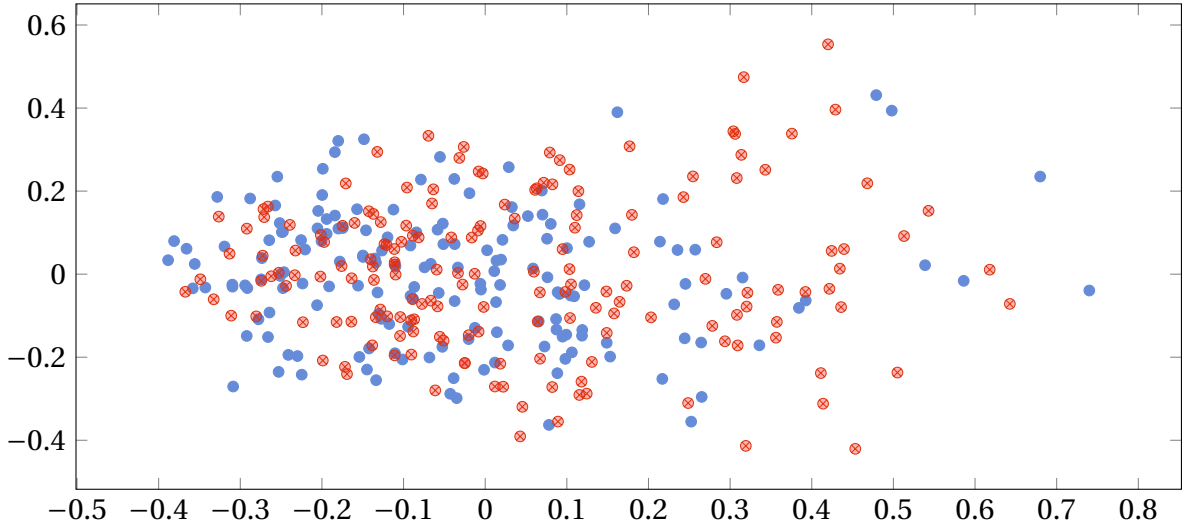


Figure 5.1: PCA of $\vec{d}_{l2,nostop}$ on the Veroff dataset from 400d GloVe vectors trained on the fanfiction corpus; Red, with cross: aroused, blue: non-aroused

all motivational needs. It can also be applied the other way round: If there is a person in the aroused group with unusually low $nAff$, the arousal might just bring him or her to “normal” levels. Veroff’s study design might mitigate this problem a bit, as he did not just try to arouse the need for power, but also chose a group of students for his aroused group, where he could reasonably assume high power motivation. He touches on another problem, though, writing “perhaps [subjects] in the aroused group have had peculiar learning experiences predisposing them to write power-related imagery for reasons having nothing to do with power motivation as defined” [Ver57]. Furthermore, there might be participants for whom the arousal did just not work. They might, for example, watch a romantic movie and not have a heightened need for affiliation afterwards. These problems have the same effect, which is that there are stories in the groups that do not belong there (or are there for the wrong reason).

This could be described as “noise” in the datasets and looking at the principal component analysis of $\vec{d}_{l2,nostop}$ from 400d GloVe vectors in fig. 5.1 the effect is evident. There is a clear tendency of the aroused vectors to lie on the right side of the plot but on the left side there is a large region where the non-aroused and aroused vectors are quite entangled.

It might be, that $\vec{d}_{l2,nostop}$ is simply a bad measure and does not discriminate well between the two classes. A more sophisticated feature vector might be necessary to capture it. But as the Winter coding system does not do a much better job at discriminating the two classes either, it seems probable that the entanglement comes from the noisiness of the data. After all, the noise is increased by the fact that a story typically does not contain only sentences

which one would consider a marker for arousal, but also neutral sentences that might appear in the same way in the non-aroused group. This could be mitigated by an approach as we described it in the beginning of section 3.3.5, where the feature vector is not built on a whole story but in a window shifted across it. In any case there is the need to somehow filter out the noise.

This explanation is supported by the findings of Ring and Pang who work on the problem in the “top down” approach and therefore can use only stories that are considered high in motivation according to the Winter coding system. They reach higher accuracy with a simpler approach evidently because they are working on less noisy data [Rin17]. Presumably if they had single sentences to work on and not whole stories they could improve their results even further. But, as was already mentioned in the introduction, the “top down” approach introduces preconceptions and possible errors into the classifier, as it only strives to emulate expert coding.

If this is true, the issue might not so much be to improve the classifier, but the preprocessing. The question then is, how and at what stages to filter out these texts. Filtering could occur right at the time when a story is written. Or it could be performed on the stories before creating the feature vector from them. Another option could be to filter the feature vectors before classification. Albeit this turns the issue into a chicken-and-egg problem. A second kind of measure for the motivational needs of a person would greatly improve this situation. An example might be hormone levels of participants of a motive induction study.

One might be inclined to think that a classifier should be able to deal with the noise in the natural distribution of implicit motivation. But as the final goal is to have an objective measurement tool that is easy and fast to use, cleaner training data would lead to cleaner results. Sticking with the metaphor of a measurement tool: cleaner training data means a more precisely tuned dial.

5.1.2 Possible Side Channels

While the low accuracy might come from noisy data, another question is whether the accuracy achieved is actually measuring the discriminatory power regarding the aroused and non-aroused groups. We already speculated while introducing the bag-of-words baseline that it might simply perform authorship attribution. This was because it classified correctly to some degree on most datasets but completely failed on Wirth_{Same} (see table 4.2). This dataset, however, is the only one where the aroused and non-aroused groups comprise the same authors.

The obvious conclusion is that it is not actually the arousal state that is classified but the

authorship of documents. This becomes seemingly more likely when looking at the story wise results in table 4.9. Here, too, $Wirth_{Same}$ performs worst. This is an effect that can be seen across most experiments.

We have no means to disprove the assumption, because we only have this one dataset where authors were the same in both groups. The only other dataset where this would have been the case, initially labeled Study2, had only 30 stories per group. However, there are some hints making it seem improbable.

The first is merely another explanation for the low accuracy. While the datasets Atkinson, McAdams, McClelland and Veroff all have more than 300 stories, both Wirth datasets contain only about 240 stories. The Winter dataset is even smaller. It is no surprise, then, that the training is a lot worse on these throughout most experiments and classifiers.

The second argument is more substantial. When looking at the accuracy per author, the $Wirth_{Same}$ dataset is right in the middle performance wise. For this measure, the same author in the aroused and non-aroused group was virtually split into two. What this large disparity means is that the wrongly classified stories in $Wirth_{Same}$ were lying very close to the separating hyperplane, while the correctly classified ones were farther away. If the classification were on authors and not on arousal state, this would not have been the case.

Besides authorship of a document, there is another side channel that might incorrectly raise accuracy. For example, when a person has seen “The Godfather” they might choose words in their stories that do not indicate a high $nPow$ but are still discriminatory. Men might be described as mafiosi or wearing suits. This would simply come from the act of watching the specific movie and does not represent the current motivational state of that author. The non-aroused group on the other hand might use words that were planted in their head by the control experiment. When applying a coding system these words would be considered non informative and not counted. When using a bag of words or any other automated measure these words could be incorrectly taken into account. To tell if this is a real problem the stories would have to be examined linguistically. It is only a minor issue, though, as not all study setups bear the risk of introducing such an error. It strongly depends on the way motives were aroused and the chosen images.

5.1.3 Cross Study Validity

A major issue that has not been solved within this thesis is how to train a classifier on one dataset and use it on a different one. It is quite probable that the feature vector that is employed mostly captures the general theme of the story. To compare two studies, it would be necessary

that they have the same images. Unfortunately from the description of the images used in the studies (see chapter 2) it seems that the only image that has been used in multiple studies is one showing a couple sitting on a bench by a river. It came to use in the McAdams and Wirth datasets which have another five different images each.

It might be possible to subtract the feature vector of the non-aroused group from the aroused group, giving in essence the difference between both groups and then work with that. Another option might be to somehow iteratively reduce the dimensions of the feature vector, eliminating the dimensions that merely describe the overall theme (a similar approach to that which we hoped employing random forests would utilize).

5.2 Word Embedding Training Corpus

Apart from the observations regarding the research goals of this thesis, a side-note can be made. We were not able to confirm the assumption that training word embeddings on a corpus that is similar to the classified texts improves the accuracy. As we said before, this assumption is not disproved, though. The only thing we could show is that there is no significant difference in the accuracy of the experiments using our feature vectors on our datasets. There are a number of reasons why the improvements might not have manifested in the task at hand.

It might just be that the writing style of the Wikipedia and the PSE stories are not as far apart as we had thought. While from reading PSE stories and encyclopedic articles this impression must be come to, it might be a false one. The vocabulary that makes up the PSE stories might in general co-occur with the same words in both training corpuses. We have not found publications on this subject and it might be worth investigating.

Chapter 6

Conclusion and Outlook

This thesis set out to go the first steps toward automating the coding process of implicit motives in psychological picture story exercise. It sought the more challenging path of starting at motive induction studies rather than beginning with already coded stories by human experts.

6.1 Our Contribution

Our contribution is a classifier that separates aroused from non-aroused stories as good as or better than the Winter coding system, when trained and evaluated on stories of the same study. In comparison to a simple bag-of-words classifier it performs better on small datasets. We have shown that it is possible to separate stories of motive induction studies using word embeddings with about 54% to 64% accuracy. When the results are aggregated per author, accuracy increases to 58% to 78%. For this we have evaluated different methods of classifying stories and creating feature vectors.

We can see that using an SVM for classification appears to be superior to random forests and have found hints that using neural networks is not a feasible alternative. Neither does employing a k -NN classifier in combination with the Word Mover's Distance yield better results.

While stop word removal and normalization by the Euclidean distance each on their own do not reliably improve upon a centroid vector, in combination they do so for GloVe based vectors in most of our experiments. Clustering words in the embeddings space by their Euclidean distance and cosine dissimilarity appears to not improve results.

We have also evaluated a number of possibilities of improving word embeddings and their influence on the feature vector and thus classification rates. For this, we have compiled

a large corpus of fan fiction for word embedding training and compared the results to a corpus formed by Wikipedia articles. We found that using different training corpora and 400 or 600 dimensional vectors does not seem to have an influence on the accuracy of the classification. Neither did we find evidence that retrofitting word embeddings using the paraphrase database or wordnet synonyms helps much for the classification task. Lastly ultradense word embeddings seemingly also do not lead to better performance.

6.2 Future Work

There are some areas where more research is needed. It is unclear how to handle the noise within datasets that probably comes from people who are predisposed towards a specific group. Removing that noise would probably improve classification rates by a large margin.

Furthermore the question remains how to classify stories across study boundaries when different images have been used.

The next step would be to not just classify stories as aroused or non-aroused but assign a numerical value representing how strong the motivational themes within the story are. A good starting point could be the distance to the hyperplane in an SVM, as using it to aggregate results per author works well. This could then be compared closer to existing coding systems and one day replace the labor intensive process of manual coding.

Appendix A

Example of a PSE Story

Sample of a PSE story and human expert coding as published by Schultheiss [Sch13b] (underline in original, cursive added by us) for fig. 1.1b.

“These are two lovers spending time at their favorite bridge (*n Affiliation*). The guy is trying to convince his girlfriend (*n Power*) to follow him to California, where he wants to enter graduate school. The woman already has a successful career (*n Achievement*) as a ballet dancer in Boston and has become quite famous (*n Power*). The two try to find a solution that will allow them to continue to stay together as a couple (*n Affiliation*). Eventually, they will find a compromise that enables both of them to have successful careers (*n Achievement*) and be in each other’s company as much as possible (*n Affiliation*).”

Appendix B

Code Listing

Code Listing B.1: Excerpt from the util.classifiers module showing set of stopwords

```
class Classifier(BaseEstimator, ClassifierMixin) :
    """ A classifier base class. It mainly just specifies the interface """

    """ Stopwords are the 100 most frequent words from the wikipedia,
    except some words which we think might have an impact on detection rate """
    stopwords = {"the", "of", "and", "in", "to", "was", "is", "for", "on", "as",
                "by", "with", "he", "at", "from", "that", "his", "it", "an",
                "were", "are", "also", "which", "this", "or", "be", "first",
                "new", "has", "had", "one", "their", "after", "not", "who",
                "its", "but", "two", "her", "they", "she", "references", "have",
                "th", "all", "other", "been", "time", "when", "school",
                "during", "may", "year", "into", "there", "world", "city",
                "up", "de", "university", "no", "state", "more", "national",
                "years", "united", "external", "over", "links", "only",
                "american", "most", "team", "three", "film", "out", "between",
                "would", "later", "where", "can", "some", "st", "season",
                "about", "south", "born", "used", "states", "under", "him",
                "then", "second", "part", "such", "made", "john", "war",
                "known", "while"} - \
    {"he", "his", "first", "new", "her", "they", "she", "city",
     "world", "university", "united", "him", "second"}
```


Appendix C

Larger Versions of Figures

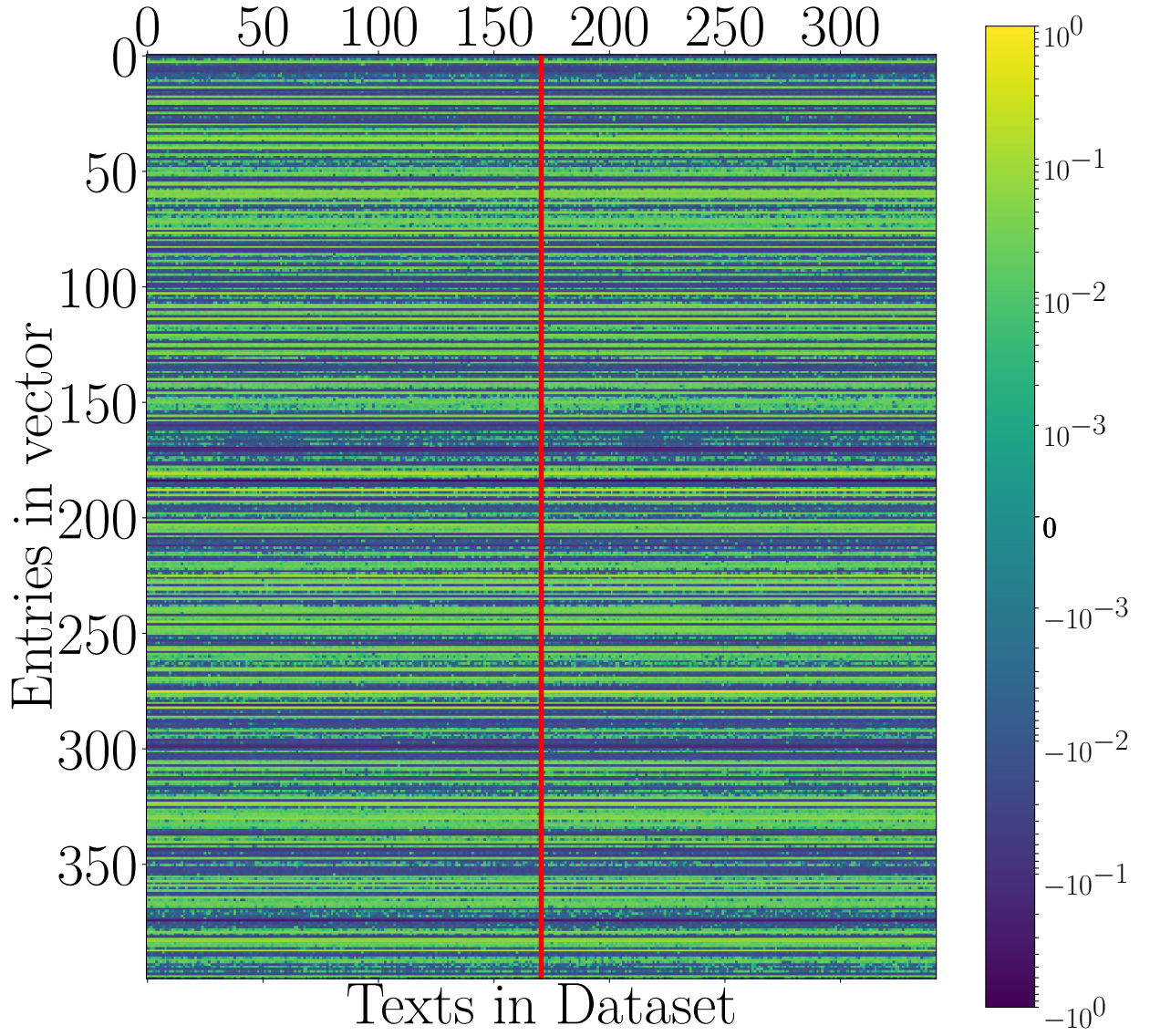


Figure C.1: Matrix rendering of $\vec{d}_{l2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories(cf. fig. 3.2a)

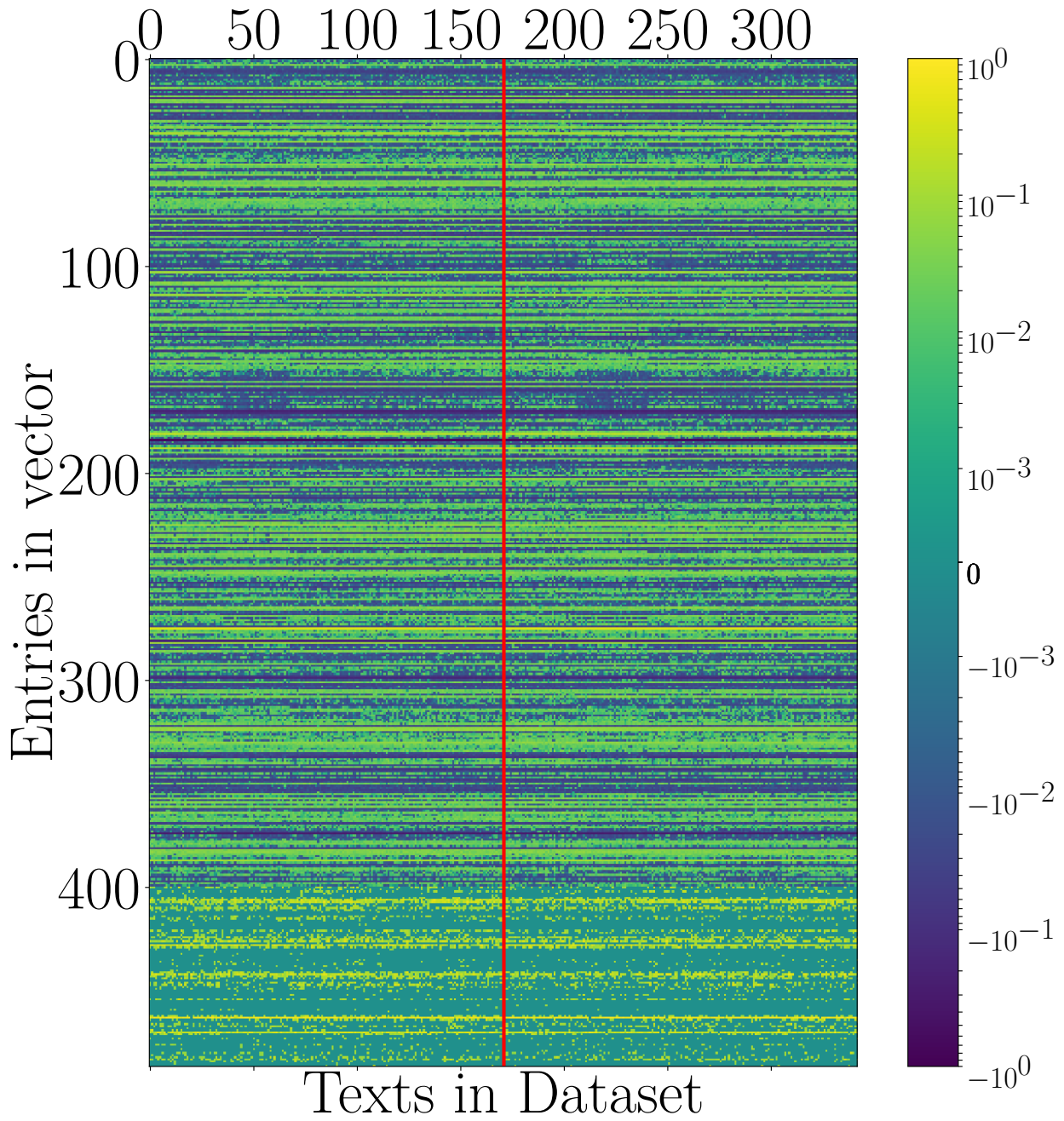


Figure C.2: Matrix rendering of $\vec{d}_{l2,nostop}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories; including stop word frequency at the bottom (cf. fig. 3.2b)

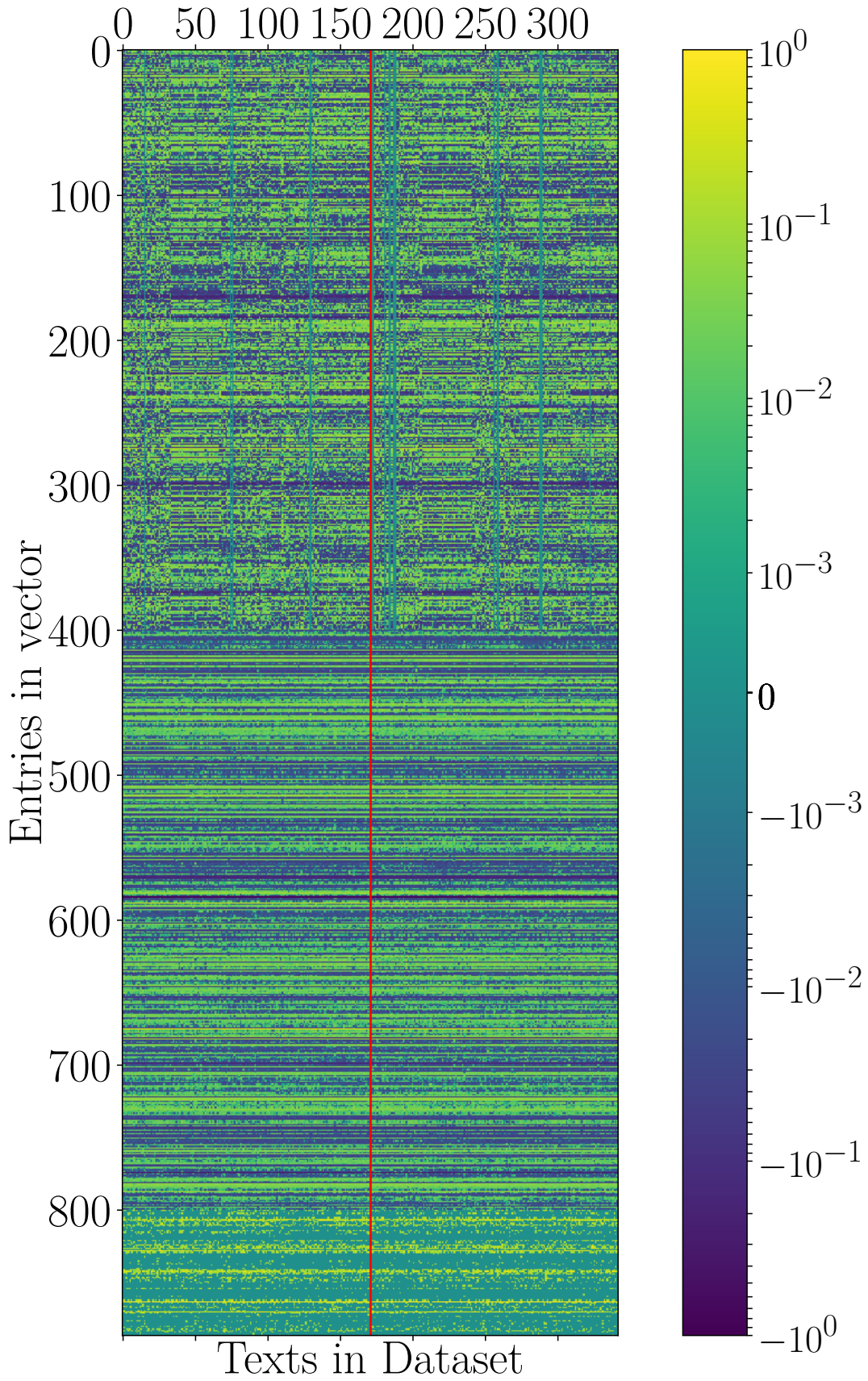


Figure C.3: Matrix rendering of $\vec{d}_{l2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories: Clustered using k -means and Euclidean distance (cf. fig. 3.3a)

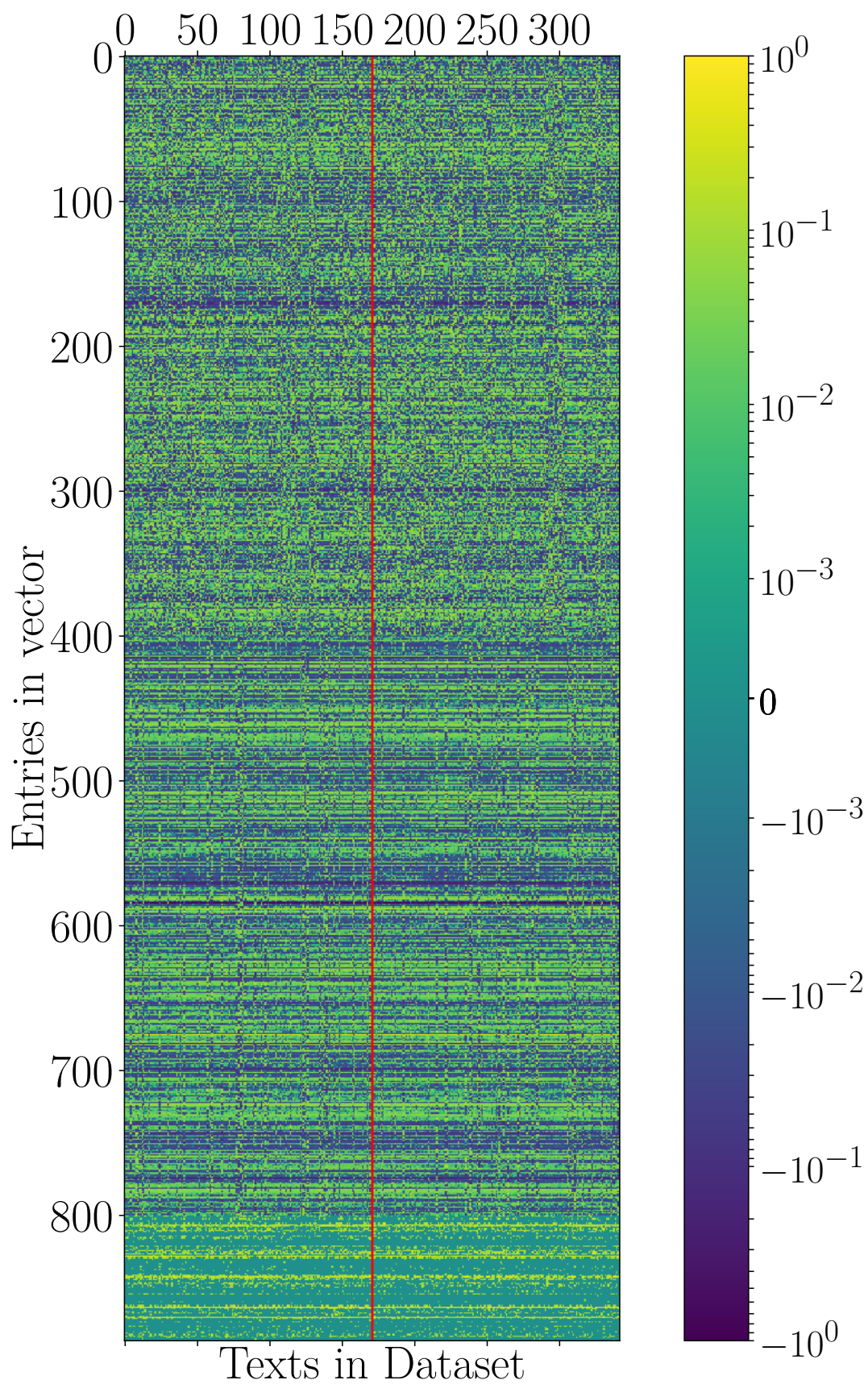


Figure C.4: Matrix rendering of $\vec{d}_{l2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories: Clustered using agglomerative clustering and cosine dissimilarity (cf. fig. 3.3b)

List of Figures

1.1	A typical TAT and PSE image	2
2.1	Dataset sizes in absolute and relative values.	8
3.1	PCA of adjective to adverb relationship of different words in the 400 dimensional word2vec embedding space trained on ffcorpus	15
3.2	Matrix rendering of $\vec{d}_{l_2}$ and $\vec{d}_{l_2, nostop}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories	25
3.3	Matrix rendering of clustered $\vec{d}_{l_2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories	28
5.1	PCA of $\vec{d}_{l_2, nostop}$ on the Veroff dataset from 400d GloVe vectors trained on the fanfiction corpus	56
C.1	Matrix rendering of $\vec{d}_{l_2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories(cf. fig. 3.2a)	68
C.2	Matrix rendering of $\vec{d}_{l_2, nostop}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories; including stop word frequency at the bottom (cf. fig. 3.2b)	69
C.3	Matrix rendering of $\vec{d}_{l_2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories: Clustered using k -means and Euclidean distance (cf. fig. 3.3a)	70

C.4	Matrix rendering of $\vec{d}_{l2}$ on the Veroff dataset using the 400d GloVe vectors trained on the fanfiction corpus. Left of the red line are non-aroused stories, right of it aroused stories: Clustered using agglomerative clustering and cosine dissimilarity (cf. fig. 3.3b)	71
-----	--	----

List of Tables

4.1	Winter coding system baseline: Accuracy with standard deviation of the cross validation of said system classifying by a simple threshold	36
4.2	Bag-of-words baseline: Accuracy with standard deviation of using a bag-of-words of n -grams classified by an SVM	37
4.3	Accuracy of classification with a k -NN classifier using the Word Mover's Distance on GloVe vectors of different sizes	38
4.4	Influence of stop word removal on the accuracy of classifying $\vec{d}_{len}$ of 400d vectors trained on ffcorpus using an SVM	41
4.5	Influence of using the document length ($\vec{d}_{len}$) versus the Euclidean norm ($\vec{d}_{l2}$) for normalization on 400d vectors trained on ffcorpus	43
4.6	Accuracy of classifying $\vec{d}_{l2,nostop}$ of GloVe vectors trained on ffcorpus using an SVM	45
4.7	Accuracy of classifying $\vec{d}_{l2}$ of 400d GloVe word embeddings trained on ffcorpus using a random forest; with and without applying ultradense word embeddings transformation	46
4.8	Accuracy of classifying $\vec{d}_{l2,nostop}$ of 400d GloVe vectors trained on ffcorpus using an SVM; with and without applying k -means clustering before creating the centroids	48
4.9	Comparison of classifying a centroid vector using an SVM against baselines; results are per story	52
4.10	Comparison of classifying a centroid vector using an SVM against baselines; results are per author;	53

Glossaries and Notation

Acronyms

<i>k</i>-NN	<i>k</i> -nearest neighbors
CBOW	continuous bag-of-words
IDF	inverse-document-frequency
JSON	Javascript Object Notation
LIWC	Linguistic Inquiry and Word Count, a software used to automatically analyze a text based on word categories
PCA	principal components analysis
POS	part-of-speech tagging
PSE	picture story exercise
RBF	radial basis function
SVM	support vector machine
TAT	thematic apperception test
TF-IDF	term-frequency inverse-document-frequency
WMD	Word Mover's Distance

Glossary

400d	vector built from 400 dimensional word embeddings
600d	vector built from 600 dimensional word embeddings
ffcorpus	corpus of fan fiction stories for word embedding training

GloVe	global vectors for word representation, a kind of word embedding learned by optimizing on word co-occurrence counts on a global level
nAch	need for achievement
nAff	need for affiliation
nPow	need for power
ppdb	paraphrase database
w2v	see word2vec
wikicorpus	corpus of wikipedia articles for word embedding training
wns+	wordnet synonyms database, including synonyms, hypernyms and hyponyms
word2vec	a kind of word embedding learned by predicting a word from context or vice-versa using a neuronal network

Recurring Mathematical Notation

$\vec{c}$	Feature vector similar to $\vec{d}$, except it is created for a single cluster of vectors after clustering
$\vec{d}$	Feature vector created by summing up word vectors of a document and normalizing it
$\vec{d}_{k2}, \vec{d}_{k3}, \dots$	this index of $\vec{d}$ indicates that k -means clustering was performed, the number indicates the number of clusters
$\vec{d}_{l2}$	this index of $\vec{d}$ indicates that the Euclidean norm was used for normalization
$\vec{d}_{len}$	this index of $\vec{d}$ indicates that the document length was used for normalization
$\vec{d}_{nostop}$	this index of $\vec{d}$ indicates that stop word removal was applied
$\vec{d}_{ultradense}$	this index of $\vec{d}$ indicates that an ultradense subspace transformation was applied
Q	Orthogonal rotation matrix trained by the ultradense word embeddings method to rotate the embedding space so that it represents a specific semantical meaning in a subspace

Bibliography

- [Atk54] John W. Atkinson, Roger W. Heyns, and Joseph Veroff. The effect of experimental arousal of the affiliation motive on thematic apperception. *The Journal of Abnormal and Social Psychology*, 49(3):405–410, 1954.
- [Bla10] Virginia Blankenship. Computer-based modeling, assessment, and coding of implicit motives. In *Implicit Motives*, pages 186–208. Oxford University Press, 2010.
- [Boo16] Cedric De Boom, Steven Van Canneyt, Thomas Demeester, and Bart Dhoedt. Representation learning for very short texts using weighted word embedding aggregation. *Computing Research Repository*, abs/1607.00570, 2016.
- [Far15] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado, 2015.
- [Fre89] Sigmund Freud. Das Ich und das Es. In *Psychologie des Unbewußten. (Studienausgabe) Bd. 3 von 10*. S. Fischer Verlag, Frankfurt, 1989.
- [Gan13] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764, 2013.
- [Hal15] Marc Halusic. *Developing a computer coding scheme for the implicit achievement motive*. PhD thesis, University of Missouri–Columbia, 2015.
- [Jon01] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001. [Online; accessed 2017-07-20].

- [Kir15] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *Computing Research Repository*, abs/1506.06726, 2015.
- [Kus15] Matt J Kusner, Yu Sun, Nicholas I Kolkin, Kilian Q Weinberger, et al. From word embeddings to document distances. In *International Conference on Machine Learning*, volume 15, pages 957–966, 2015.
- [Le14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, pages 1188–1196, Beijing, China, 2014.
- [McA80] Dan P McAdams. A thematic coding system for the intimacy motive. *Journal of research in personality*, 14(4):413–432, 1980.
- [McC49] David C McClelland, Russell A Clark, Thornton B Roby, and John W Atkinson. The projective expression of needs. iv. the effect of the need for achievement on thematic apperception. *Journal of Experimental Psychology*, 39(2):242, 1949.
- [McC53] D. C. McClelland, J. W. Atkinson, R. A. Clark, and E. L. Lowell. *The achievement motive*. New York, Appleton-Century-Crofts, 1953.
- [Mik13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Workshop Proceedings of the International Conference on Learning Representations 2013*, 2013.
- [Mil95] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- [Ped11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [Pel08] Ofir Pele and Michael Werman. A linear time histogram metric for improved sift matching. In *Computer Vision–ECCV 2008*, pages 495–508. Springer, October 2008.
- [Pel09] Ofir Pele and Michael Werman. Fast and robust earth mover’s distances. In *2009 IEEE 12th International Conference on Computer Vision*, pages 460–467. IEEE, September 2009.

- [Pen99] James W Pennebaker and Laura A King. Linguistic styles: language use as an individual difference. *Journal of personality and social psychology*, 77(6):1296, 1999.
- [Pen14] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *Proceedings of EMNLP 2014*, 2014.
- [Řeh10] Radim Řehůřek and Petr Sojka. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA.
- [Rhe08] Falko Rheinberg. *Motivation*. Kohlhammer W., 2008.
- [Rin17] Hiram Ring and Joyce Shu Min Pang. Automating motive identification in texts. Slides from a presentation in Erlangen, 2017.
- [Rot16] Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. Ultradense word embeddings by orthogonal transformation. *Computing Research Repository*, abs/1602.07572, 2016.
- [Rub00] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. The earth mover’s distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.
- [Sch07] Oliver C Schultheiss and Joyce S Pang. Measuring implicit motives. In *Handbook of research methods in personality psychology*, pages 322–344. Guilford Publications, 2007.
- [Sch08] Oliver C. Schultheiss, Scott H. Liening, and Daniel Schad. The reliability of a picture story exercise measure of implicit motives: Estimates of internal consistency, retest reliability, and ipsative stability. *Journal of Research in Personality*, 42(6):1560–1571, 2008.
- [Sch10] Oliver Schultheiss and Joachim Brunstein. Introduction. In *Implicit Motives*, pages ix–xxvii. Oxford University Press New York, NY, 2010.
- [Sch13a] Oliver C Schultheiss. Are implicit motives revealed in mere words? Testing the marker-word hypothesis with computer-based text analysis. *Frontiers in Psychology*, 4(748), 2013.

- [Sch13b] Oliver C. Schultheiss. The hormonal correlates of implicit motives. *Social and Personality Psychology Compass*, 7(1):52–65, 2013.
- [Ver57] Joseph Veroff. Development and validation of a projective measure of power motivation. *The Journal of Abnormal and Social Psychology*, 54(1):1, 1957.
- [Wei10a] Joel Weinberger, Tanya Cotler, and Daniel Fishman. Clinical implications of implicit motives. In *Implicit motives*, pages 468–487. Oxford University Press New York, NY, 2010.
- [Wei10b] Joel Weinberger, Tanya Cotler, and Daniel Fishman. The duality of affiliative motivation. In *Implicit motives*, pages 71–89. Oxford University Press New York, NY, 2010.
- [Win73] D.G. Winter. *The power motive*. Free Press, 1973.
- [Win91] David G Winter. Measuring personality at a distance: Development of an integrated system for scoring motives in running text. *Perspectives in Personality*, 1991.
- [Wir06] Michelle M. Wirth and Oliver C. Schultheiss. Effects of affiliation arousal (hope of closeness) and affiliation stress (fear of rejection) on progesterone and cortisol. *Hormones and Behavior*, 50(5):786–795, 2006.