Emotion and Affect Representation in Sentence Embeddings

Author:
Luis Alberto
Barradas Chacón
278183

Supervisors:
Prof. Dr. Christian Wartena
MSc. Rafael Drumond

9th June 2020

Thesis submited for
Master of Science in Data Analytics

Wirtschaftsinformatik und Maschinelles Lernen
Stiftung Universität Hildesheim
Universitatsplätz 1, 31141 Hildesheim

Statement as to the sole authorship of the thesis:

Emotion and Affect Representation in Sentence Embeddings.
I hereby certify that the master's thesis named above was solely written by me and that no assistance was used other than that cited. The passages in this thesis that were taken verbatim or with the same sense as that of other works have been identified in each individual case by the citation of the source or the origin, including the secondary sources used. This also applies for drawings, sketches, illustration as well as internet sources and other collections of electronic texts or data, etc. The submitted thesis has not been previously used for the fulfilment of a degree requirements and has not been published in English or any other language. I am aware of the fact that false declarations will be treated as fraud.

9th June 2020, Hildesheim

Abstract

Emotion detection and classification is a common task for Machine Learning in Natural Language Processing, but the theoretical bases that support it are weak, and in some cases inexistant. In this project we explore the effective representation of different emotions and affects from established labeled datasets in common word and sentence embeddings. These are compared to models of emotions from the field of Psychology, and suggestions are made on how to approach this problem in the future.

Acknowledgements

Acknowledgements Here

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

Why is it important to study emotion? Emotions are considered a human state that influences behaviour and decision making. Many times, when expressing thoughts in a written or spoken form, one or several emotions are present. Detecting these emotions is an important task for human interaction. Automatic emotion detection on text is thus a machine learning task required for comprehensive human-computer interaction.

1.1 Emotions and Affect

In the endless effort towards understanding human behaviour the phenomenon of Emotions has been recognized for centuries. It is after all, an experience that every human has. For many of us it represents a core variable in the representation of our biological, psychological, and social state. For such an important part of our lives, it is incredible how little we actually know about them. There is no scientific consensus on what an emotion is, and the term is often used to refer to mood, humor, temperament, personality, affect, character, and sentiment. This section has as a goal to point out relevant discoveries and conceptualizations in the history of the study of emotions, as a way of delimiting the current study, and as a mean of introduction to the topic for technology-focused readers, but also as a way of outlining a working definition of Emotion, differenciating it from Affect.

Historic Milestones

Hippocrates, the father of modern medicine, defined the theory of the Four Humors. This was based on the idea of humors, fluids, or chemicals that control human behaviour. The four humors should be in balance within the

body, and all diseases were caused by an inbalance of these.[**?**] Galen at took this theory and created what can be considered as the first theory of personality. He believed that human bodies had a predisposition for unbalance of the humors. This made some people have a tendency to have more or less of these, and in turn, this would have an effect on their baseline behaviour. He described four different characteristical behaviours: phlegmatic, choleric, sanguine and melancholic.[Irwin, 1947]

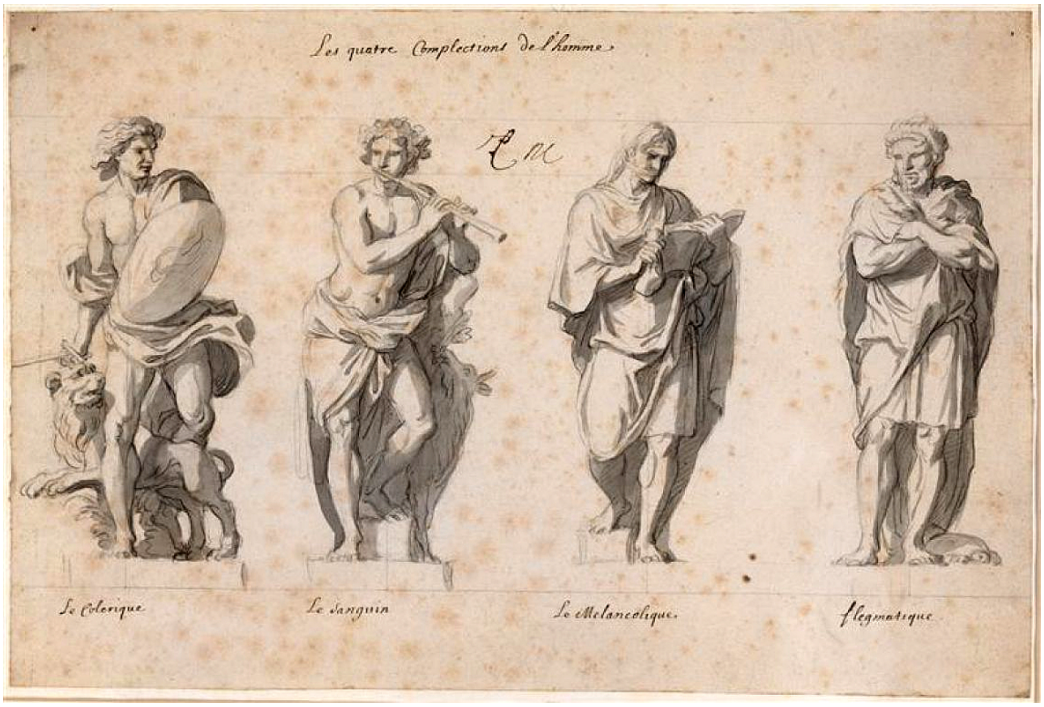

Figure 1.1: Wikicommons PD. The four temperaments by Charles Le Brun, part of the *Grande Commande*.

Charles Darwin first described the importance of emotions in communication, and their relevance across cultures and even species. In his book ' *The Expression of the Emotions in Man and Animals* ' he writes ' . . . the young and the old of widely different races, both with man and animals, express the same state of mind by the same movements.' [Darwin and Prodger, 1872] He noticed that surprise was shown in humans across cultures, and even some mamals by raising the eyebrows. By framing emotions as a mean of communication, Darwin enabled the study of expression, and understanding of emotions as an evolutive advantage.

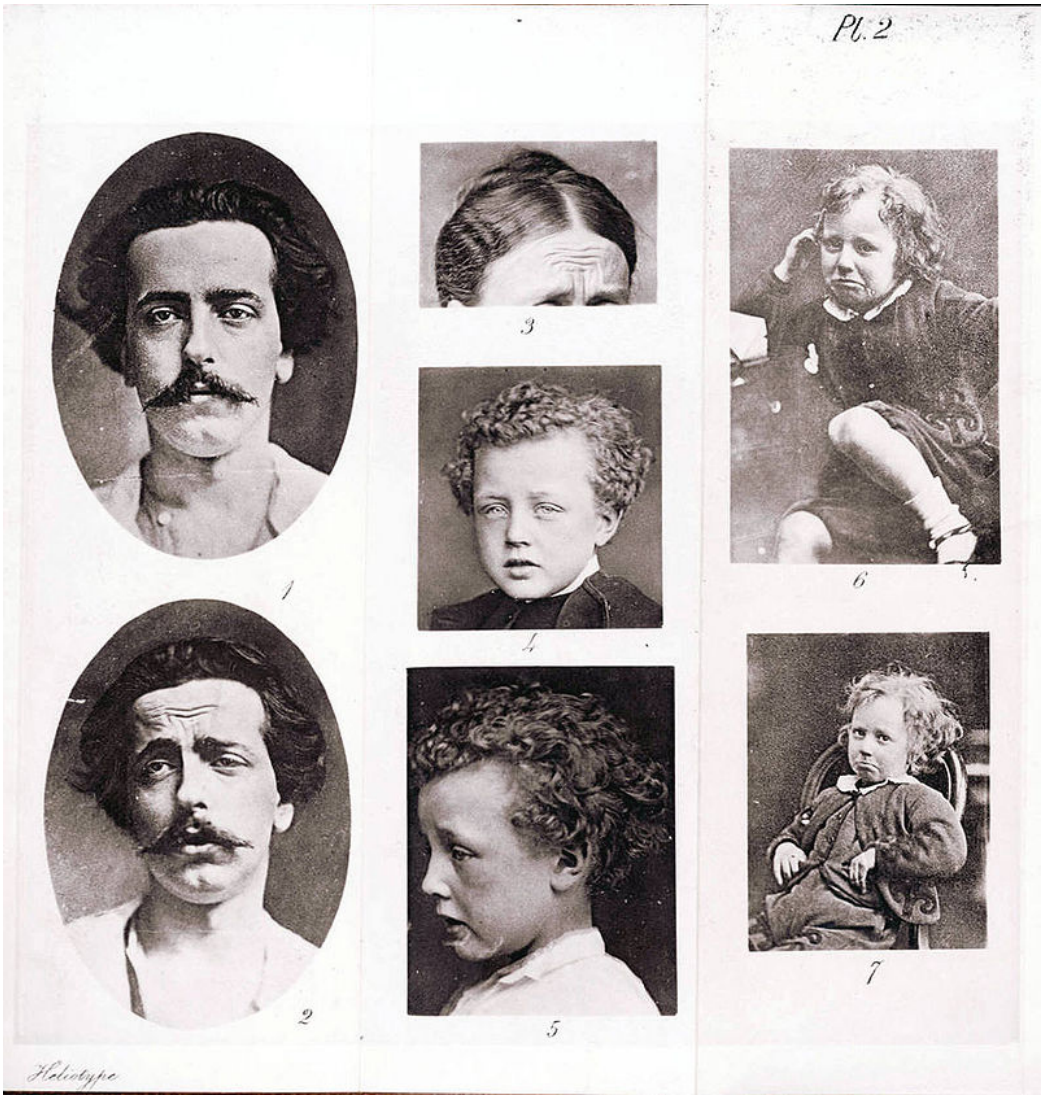

Figure 1.2: Wikicommons PD. Ilustration of grief from Darwin's *"The Expression of the Emotions in Man and Animals"*

Although Emotions were thought to be universal there was no measurement of it. The universality of emotions was first formalized by Paul Ekman in his 1997 paper: "Universal facial expressions of emotion". Ekman studied facial anatomy, and expressions of different populations and cultures across the globe. He arrived to the conclussion that there are seven universal facial expressions of emotion [Ekman and Keltner, 1997].

Figure 1.3: Ekman's photographs for cross-cultural research [Ekman, 1999].

While emotions are a human concept, the digital advances of thte milenium caught up and integrated with the field, to create the Affective Computing. The concept was first coined by Rosalind Picard, who not only created it, but is also a lead researcher in the field [Picard, 2000]. The concept was originaly related to Human-Computer interactions, and had the goal of computers expressing and recognizing emotions.

Concerned with the subjectivity of emotion messurement, and the lack of generalization of self-report affect scales, Robert Plutchik proposed a method of meassuring the baseic emotions in a systematic way. He also proposed a way of derivating new emotions from the universal set proposed by Ekman. This was to be done based on theory, with enough diversity, but systematically relatable to the universal emotions [Plutchik and Kellerman, 2013]. In

this way, the Plutchik model of emotions was created. This model is based on Ekman's universal emotions. Today, most Machine Learning tasks and datasets use this model of emotions. Different models will be discussed further in this chapter.

It is important to consider that for the last two decades, emotions have been studied based on Ekman's work on universal emotions. Although these do provide a framework for understanding how emotions came to be a part of the human experience, they do very little for their definition, or the description of emotions in language. Humans are complex, and even the most reliable model of universal emotions has exceptions. Psychologists interpret those differences with the help of the Theory of Constructed Emotions. This was first published by Lisa Feldman in 2014, and it describes the phenomenon of human emotions as a two-sided event: Affect and Emotion. Affect is a physiological phenomenon, the almost mechanical process that will enable behavioural response. The Emotional response is the cognitive contextualization of the former [Feldman Barrett and Russell, 2014]. Separation of physiological and cognitive responses allows the explanation of both, universality, and individual subjectivity.

Definition of Emotion

Within the context of this project it is important to distinguish between emotion and affect. Affect, in the context of this project will be treated as a term to associate predisposition towards stimuli. Thus, affect is in a sense, a general term that can be even used to describe animal, and other non-human entities. Emotions, on the other hand, are treated in this project as a state inherent to humans. This state is multidimensional, and every dimension, or emotion, can either be present in a certain amount, or not be present at all. Emotions present an affect value, but not necessarily otherwise.

Affect

In the context of this project the term Affect will be used to denote the autonomus physiological response of the human body to external stimuli, as well as the measure of these in terms of Valence, or Arousal. This definition complies with the Theory of Constructed Emotions, whithout invalidating the extended use of affect models of language.

The relevance of affect in text has increased since the popularization of text-based social networks, like twitter. There, individuals and organizations openly express their opinions. This creates an environment where implicit

feedback about entities is present. An easy way to abstract popular opinion about a named entity is learning the affect expressed in text, such as a tweet. Affect can be a multidimensional phenomenon, but the most important dimension of it is valence: whether a text expresses positive or negative emotion. This use of affect language models has proven useful to marketing, public relationship, and social sensing, but does not provide an insight into the human experience of emotion further than a signle dichotomical variable.

Affect is relevant to this project since emotions can be represented within the models of affect that include valence and arousal [Chacón, 2016].

Emotions in Communication

The main evolutive advantage of emotions is to be able to communicate an internal state with others. The communication, and therefore, the detection of emotions can be done through three different means:

- **Language or self report**: Using language, verbal, written, or otherwise, to express content with emotion, or explicitly declare an emotional state.
- **Facial Expressions**: The activation of different sets of facial muscles to express emotion. This must be measured visualy.
- **Biosignals**: The change of physiological states usually related to the limbic system can be measured through bio-sensors. This more accurately represent affect, but emotions can be measured through it.

This project is focused on language expressions of emotion. More specifically on expressions of emotion through text. This means that this project will not directly measure emotions on humans, but on text written by humans. These represent the expresion of a momentary emotional state, expressed through text, and stored for later analysis. Text analysis is a subset of the field of Natural Language Processing (NLP). This project is heavely based on NLP concetps. For this reason, different methods for emotion analysis in text are discussed next.

Models of Emotions

When searching for expressions of emotions in text, one must know what kind of emotions are being searched for. When asking a person what emotions do they know, a plethora of emotions can be named, but most of these are language, culture, or context dependant. Under certain cultures, or even subcultures, new emotion names can emerge, that describe a general emotion

under different context. Considering the theory of constructed emotions, there is as many emotions as context there are. Fortunately, the study of emotions being done here is restricted to the communication of emotions. For the communication of emotions a consensus must be made. A model of emotions is the selection and structure of categories in which the expression of an emotion can fall, commonly called 'Universal Emotions'. These are usually formed through the analysis of human (and some times animal) behaviour, with the framework of a theory of behaviour. In this section we introduce some relevant models of emotion for ML and NLP.

Ekman's model of Emotions

As mentioned in 1.1, the most widely recognized model of universal emotions in humans was created by Paul Ekman[Ekman, 1992]. This was created through the observation of facial expressions. Facial expressions are measured through Activation Units (AU's): Sets of muscles that, when contracted, deform the figure of the human face in specific ways. Emotions are then characterized by the activation of different sets of AU's. The current model of universal emotions contains seven emotions:

- Anger
- Disgust
- Fear
- Surprise
- Happiness
- Sadness
- Contempt

Although this model is based on facial expressions, most studies of emotion use this as the base model to explore and understand concepts of human behaviour. Notice that this model is based on the communication of emotions through facial expressons. These are said to be universal because facial expressions seem to have been forged through natural selection, and are independant of culture, language, or segregation of populations. In simple terms, humanity had a face for longer than most other human traits. Under this frame of reference, language as a mean of communication is a relatively new phenomenon, but since it allows for more detailed communication, it changes the way we communicate our internal states.

Plutchik's model of Emotions

Although Ekman's model provides a firm base for universal emotions, it lacks structure. Robert Plutchik created his model by removing Contempt, and adding two more emotions: anticipation and trust. By doing so, a three-dimentional model was createt that also provided dichotomical, or polar emotions, intensities, and derivatives through superposition.

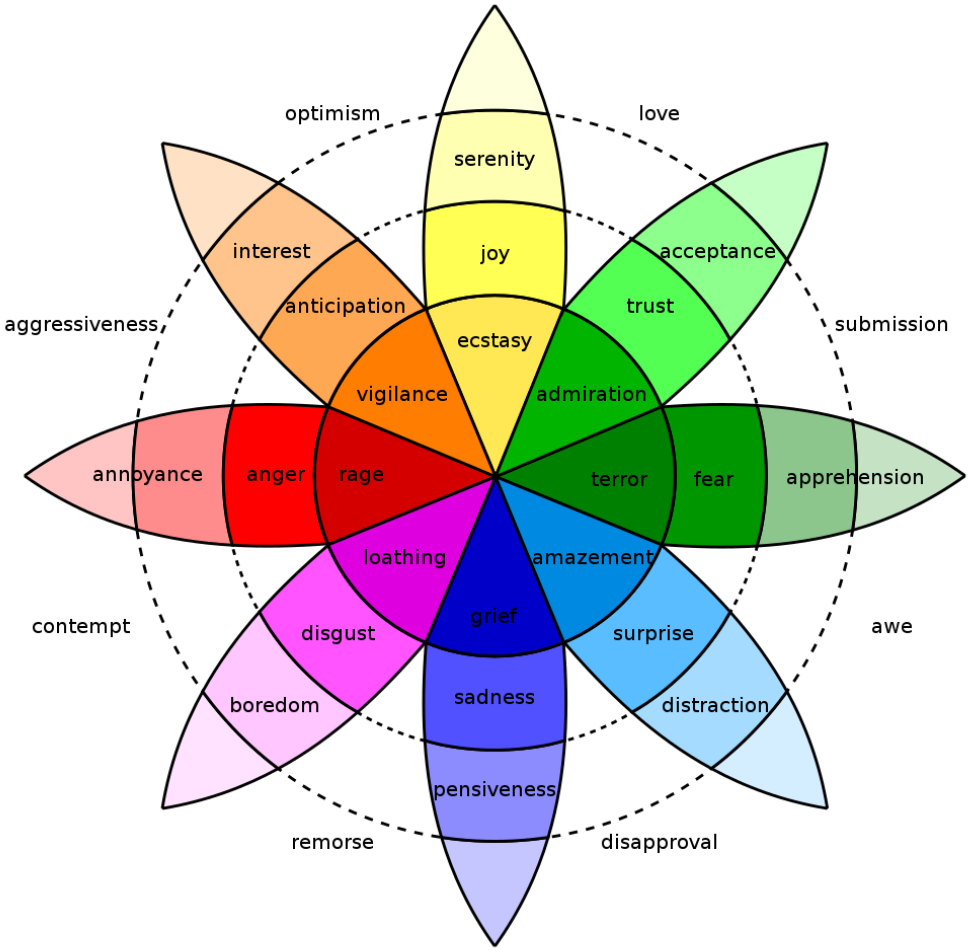

Figure 1.4: Wikicommons PD. Plutchik's wheel of emotions

The model shown in figure 1.1 shows these characteristics. The main emotions are expressed in different intensities, and named acordingly. The combination, or superposition of some emotions receives a new name. This

model is so well structured and colored, that it has called the attention of the scientific and engineering community. Unfortunately the premises of this model are faulty, and the scientific basis are questionable. The subjective nature of emotions doesn't allow for strict dichotomies. A simple poll asking 'What's the opposite of joy?' will turn in two answers: Anger and Sadness, deppending on the context.

Acording to the theory of Constructed Emotions, it is exactly context that creates emotions. This means that culture, language, learning, and many other environmental factors contribute to the characterization of an emotion. For this reason, when studying emotions in text, a mean of contextualizing an emotion word is necessary.

Emotions and Text

When it comes to contextualizing words, linguists have had tools since the 1930's [Corson, 1996]. This is the case of Lexical Fields. In it's simplest version, this tools ask people what words are most closely related to a concept. The result of a free assosiaton can be a network of interrelated words, or a set of words that relate to a concept. This set is also called a semantic field. By asking people what emotions are related to a specific word, a semantic field of emotions can be created.

This is exactly what Saif Mohammad did in 2013 [Mohammad and Turney, 2013], creating the Canadian National Research Counsil Emotional Lexicon (NRC EmoLex). This contains around 14 thousand words, and their association with the 8 basic emotions of the plutchik emotion model. This was an incredible effort, since several people must review every single word and emotion relationship, and evaluate them. All this work can actually be infered from large corpus. By creating a network of word relationships, WordNet provides a way to cluster into affective words [Strapparava et al., 2004]. This proves that a representation of emotions in text can be learned from text, without human intervention. Machine Learninng provides tools to optimize this representations.

1.2 Emotion and Machine Learning

The field of Natural Language Processing has seen great advances in recent years thanks to the use of Machine Learning for automatic inference of language models.

While there are many ways of using ML within NLP, the methodology this project focuses on is Word and Sentence Embedding. Embedding is the

process of giving a numeric representation to a word or sentence within a corpus. This allows for computational models to easily manipulate text data that would otherwise be an arbitrary encoding of text. With the advances in machine learning automatic word embedding became a possible solution to avoid crowdsourcing.

Several machine learning approaches try to automatically learn the best numeric representation for characters, words, or sentences given the context of a dataset. In the last decade there have been many efforts from research groups to generalize these embeddings though the use of powerful models, and bigger datasets.

One of the first models to call the community's attention was Google's Word2Vec[Mikolov et al., 2013]. A model trained on news articles that allowed for complex languge representation, like lexical artimetic. This model has the model of an Autoencoder, and is therefore unsupervised.

After Word2Vec, a series of language models were created. Word2Vec sccessors include ELMo, GloVe, which are similar but consider more or different context when creating the language model. With the creation of the transformer in ML [Vaswani et al., 2017], a second wave of ML language models were created. XLM, GPT, BERT and it's many iterations, all provide a language model that captures not only the representation of a word, but also it's context.

These models learn to represent words from context: By looking at the words and sentences in a document, a contextualization is given to the use of a specific word, in every document. These language representations are far from a lexicon. They are numerical representations that, due to their automatically learned creation, are difficult for humans to interpret. Once these models are trained on large amounts of data, the trained model can be stored, and re-distributed to be used in different language tasks. This is called a pre-trained model. For this reason, such a context-dependant model can be used in short sentences, or even single words: The model is in itself the abstraction of a large amount of language data, learned through text documents. There exist also models trained not on documents, but on dialogues. These models will not be used in the context of this project.

By observing the abstract representations of these models, one can learn how specific concepts are learned by pretrained ML models from text.

1.3 Problem setting

Given pre-trained language models, and corpus of labeled text with single emotions, can we find the similitude between the structure of the represent-

ation of said text and their emotions in the abstract space created by the language model and a model of emotions backed up by scientific research?

1.4 Objective

To quantifiably and objectively analyze the representation of emotions in Machine Learning pre-trained Language Models, while presenting a human-understandable cualitative description to promote the discussion of models of emotions and atuomatic learning of human concepts in Natural Language Processing, and Machine Learning.

As consequence of this objective, the methodology to analyze a dataset of labeled emotions based on pre-trained language models is to be established. The intuitions and qualitative results of this project are to be developed into actual analysis methods to be used in the understanding of language models.

1.5 Justification

Be it in dialogue, local or international media, or even entretainment, current events have proven that the incorrect understanding of the emotional response of the general population can lead to severe social problems. Subcultures, minorities, and opressed populations are expressing their problems and difficulties in plataforms all around the internet. People need to express themselves,be heard and understood, as well as understand other points of view, but untrained emotional reaction is being weaponized to duscourage discussion, dialogue, and democracy. Be it as a political or military action, or simply by lack of self conciousness, the emotions of every technology user and media consumer can turn against themselves. Justified pacific protests can turn into meaningless riots. Rational arguments can turn into senseless discussions, that separate populations and capture us in our subjective realities. I believe that the understanding, awareness, and acceptance of our emotions and others' can lead towards the path to dialogue, comprehension and peace. To do so at a scale as large as the one presented by the internet, and global media, we must use the same tools that have created such a vast field. The understanding of our data, ML and AI models, and their representations of our reality can help us understand ourselves. This project is at it's core, an atempt at understanding human nature.

Chapter 2

Related Work

There have been many atempts at understanding automatically learned language models, but none has been focused on understanding emotions as we do here. For this reason, similar methods from close fields of research have been used to establish a methodology. In this chapter we talk about those methods and their contribution to this project.

The embedding of emotional words is ML Language Models can be compared directly to Emotion Lexicons. In this project we make use of that one created by Mohamad and Turney through crowdsourcing [Mohammad and Turney, 2013].

Since creating an Emotion Lexicon through crowdsourcing is a costly taks, the rest of the models use are automatic aproaches to do so. Vo and Zhang proved that an automatic approach to learning sentiment lexicons for short texts can be done through the use of emojis [Vo and Zhang, 2016]. This method uses the intrinsic usage of emojis to express positive or negative valence in a sentence, and exploded this to expand that valence to words used in the same context.

Maas et alċreated a method to learn word vectors for sentiment analysis in 2011 [Maas et al., 2011].

By applying machine learned automatic embeddings, the creation of word embeddings based only on text data was open as a possibility. This is also a method that later became the popular Word2Vec model [Mikolov et al., 2013]. The methods used in this paper were used to create word embeddings specific for the used datasets. More about this can be read on this same chapter 2.1. The Word2Vec model proved that concepts can be abstracted by providing semantic arthimetic. This allows for functions like the substraction of concepts to obtain their root meaning.

Although Word2Vec was inarguably useful when it was created, it was also proved redundant by experiments like the one by Rothe et al; who suggested an orthogonal transformation to word embeddings used on Se-

mEval2015 [Rothe et al., 2016]. This yielded ultradense word embeddings for affect concepts. This first example of a linear transformation on an abstract space opened the posibility of transforming the vector space to understand it.

As a way to reduce redundancies, and understand the absraction of valence, affect, and other similar concepts, Hollis et alfurther explored transformations of a word vector space by means of component analysis, thus creating models of semantics from text [Hollis and Westbury, 2016]. The current project heavely relies on this specific research. Under our theoretical framework, affect is a superset of emotion, and since Hollis et alhave already found abstractions of affect in Word2Vec, we expect to find similar results.

The mentioned research has only been done with affect. Research in the field of emotion detection is scarce, and generaly doesn't fulfill the prestige or quality requirements suggested by the Masters of Science in Data Analytics. A reason for this might be the abstract nature of emotions, the outdated emotion model, the lack of scientific foundations in the creation of datasets of emotion in text, or simply the overwhelming usefulness of affect in comparison with emotion analysis.

2.1 Language Models

There is an incredible amount of pre-trained Machine Learned Language models. For this project we have selected models based on the following criteria:

- The model was trained with a large amount of general purpose language corpora.
- It represented a breakthrough in NLP tasks at the moment of its publication.
- The model has been reproduced, implemented, and tested in many ML language tasks.

Under this criteria, four models have been spotted as candidates for the experiment:

- **Word2Vec**: Words to Vectors
- **GloVe**: Global Vectors for Word Representation
- **ELMo**: Embeddings from Language Models

- **BERT**: Bidirectional Encoder Representations from Transformers

Word2Vec is the result of converting large corpus into itself, by using an auto-encoder method, with help of a one-hot encoding of the corpus vocabulary [Mikolov et al., 2013]. At the time of its publication it captured much atention, mostly due to the posibility of semantic artimetic. This was tipified by the 'King - Man + Woman = Queen' example. Due to the one-hot encoding step in the algorithm, it does not solve the problem of words with multiple meanings.

Glove is recognizable between other language models, for its linear substructures of meaning. Since it was trained on aggregated co-occurrence statistics, it captures semantic structure better than Word2Vec [Pennington et al., 2014]. It still asigns a one-to-one representation of words and embedded vectors, so it does not solve ambiguities.

ELMo solved this last mentioned problem by analyzing context [Peters et al., 2018]. This was acheived by training on prediction of words in forwards and backwards passes. Even though this model solved the problem of context-dependant meaning, it was created with the premise that context in text is secuential, and it's architecture dependant on LSTMs showed this.

BERT was the first algorithm to solve this problem, by implementing a context-dependant learning, that is not based on the secuential structures. This was done with the use of Transformers. A deep learning architecture based on the atention model, that does not depend on secuential structures.

Both BERT and ELMo give different embeddings to words in different contexts, but BERT has proven better at solving language tasks. For this reason, only BERT will be used in this project.

One last model will be used as a mean of comparing results between the different models. This is FastText [Joulin et al., 2017]. FastText is very similar to the algorithm with which word2vec was created. It creates a one hot encoding of a corpus, and creates a latent dimension through training either an autoencoder, for an unsupervised approach, or a classifier, for a supervised one. This algorithm requires traning on the corpus. Since the corpus selected on this project are relatively small, FastText provides a way to create a baseline for pretrained models, by analyzing what a basic model trained only on the corpus would look like.

Selected Language Models

All candidate language models have been used in this project. Following are some of the peculiarities about them.

FastText

Python's FastText library[Joulin et al., 2017] is used in this project. This provides two approaches for training the model: an unsupervised, and a supervised. The unsupervised requires a text file with one sentence per line. The algorithm is in charge of the tokenization. This of course only works in english. The supervised approach requires a similar file for the corpus, but at the end of every line, two underscores most be followed by the label of the given sentence.

Word2Vec

Since this pre-trained model has a one-to-one correspondance between word and embedding, a dictionary can be downloaded and imported via the gensim python library [Mikolov et al., 2013]. This model has been trained with the Google News corpus. It weights about 1.5 Gb, and has a latent space of 300 dimensions. It is supposed to be located at the url `https://code.google.com/archive/p/word2vec/`, but the file is not to be found. Forums on google groups for the word2vec (`https://groups.google.com/forum/#!topic/word2vec-toolkit/z0Aw5powUco`) point several urls where the model can be found.

GloVe

This model, provided by the Standford University, is of easy access, and as Word2Vec, can be imported as a dictionary [Pennington et al., 2014]. The download can be found under `https://nlp.stanford.edu/projects/glove/`. This specific version selected was trained on the Wikipedia corpus, contains 6 billion words, uses 300 latent dimensions, and weights less than 1Gb.

BERT

The BERT model is trained not with one, but two types of tasks. The first one is masked word or sentence prediction, and a second one requires extra layers on the architecture and a fine-tunning training for task specific performance [Devlin et al., 2019]. The pre-trained model that one can get is the language model trained with the masked-language task. This model is not as easy to get, since the defoult python libraries to import BERT, require training and fine-tunning. For this reason, the bert-embeddings python library has been selected for this task.

2.2 Analysis Algorithms

Two algorithms have been chosen for dimensionality reduction:

- PCA: Principal Component Analysis
- TSNE: T-distributed Stochastic Neighbor Embedding

PCA can be interpreted as a linear transformation on the input space, that yields the maximum explainability by the least amount of dimensions. While TSNE uses statistical information to maximize the distribution of information of groups, while minimizing the distribution of information within groups.

2.3 Datasets

There are many datasets of 'emotion in text' on the internet, and finding them is not a new problem. Unfortunately, the methodology and rigor for their creation cannot be easily tested. A heavy use of the paper 'An analysis of annotated corpora for emotion classification in text' by Klinger in 2018 [Klinger et al., 2018] was done. This paper not only collects information about the datasets, but also tests their valididty in the context of a text cliassifier.

Inclusion Criteria

To be included into these experiments, the following criteria must be met by a dataset:

- The dataset must contain short labeled texts, in english.
- The label must be a single emotion, from an eckman-analogous emotional model.
- The labels must not be a reference to valence, arousal, dominance, or other affect models.

The text to be analyzed must be in english, since the methods and language models that we are testing will not all be available in other languages. The single-single label criterion has been chosen due to the restriction of two-dimensional projections, and their visualizations as scatter plots. The label is to be exrpessed as a single color on scatter plots, and a multi-label problem would not present the effect desired when developing the desired intuitions.

Candidate datasets

For the datasets included in Klinger's original paper [Klinger et al., 2018] the naming in the paper was not followed. This is due to the inconsistencies between the paper and their github repository, which (as the moment of writing this thesis) was last updated on Dec 17 2019 (commit e58d676). The dataset naming conventions used here is the same as in the document called ' *unified dataset of emotion in text* ': `https://github.com/sarnthil/unify-emotion-datasets/tree/master/datasets`.

Lastly, the candidate list includes the datasets mentioned, but is not restricted to them:

- AffectiveText [Strapparava and Mihalcea, 2007]
- AIT-2018 [Mohammad et al., 2018]
- CrowdFlower
- DailyDialogs [Li et al., 2017]
- Emotion-Cause [Ghazi et al., 2015]
- Emotiondata-Aman [Aman, 2007]
- EmotionPush [Huang and Ku, 2018]
- EmoBank [Buechel and Hahn, 2017]
- fb-valence-arousal [Preoţiuc-Pietro et al., 2016]
- Friends [Chen et al., 2018]
- Grounded-Emotions [Liu et al., 2017]
- ISEAR International Survey On Emotion Antecedents And Reactions [Scherer and Wallbott, 1990]
- Tales [Alm et al., 2005]
- EmoInt [Mohammad and Bravo-Marquez, 2017]
- TEC The Twitter Emotion Corpus published [Mohammad, 2012]
- Electoral-Tweets [Mohammad et al., 2014]
- SSEC The Stance Sentiment Emotion Corpus published [Schuff et al., 2017]

The link to these datasets can be found under the github repository for the unified emotion datasets. https://github.com/sarnthil/unify-emotion-datasets/tree/master/datasets From this list, several datasets use an affective model of valence, arousal or dominance. Removing the datasets that do not explicitly comply with the inclusion criteria leaves the following:

- CrowdFlower
- DailyDialogs [Li et al., 2017]
- Emotion-Cause [Ghazi et al., 2015]
- EmotionPush [Huang and Ku, 2018]
- Friends [Chen et al., 2018]
- EmoInt [Mohammad and Bravo-Marquez, 2017]
- TEC The Twitter Emotion Corpus published [Mohammad, 2012]
- Electoral-Tweets [Mohammad et al., 2014]
- SSEC The Stance Sentiment
  Emotion Corpus published [Schuff et al., 2017]

Selected Datasets

Due to availability, the selected datasets are CrowdFlower, EmotionPush, and Friends. The analysis has been done on the three datasets, but this document only presents the visualizations of the first dataset. The Crowd-Flower dataset has been selected due to it's internal structur, and it's best demonstration of the methods, used in this project. Nontheless, the three datasets have been analyzed, and the results can be examined on the project repository.

CrowdFlower has a total of 40000 tweets, each tagged with one of 14 emotions from the following list:

- empty
- sadness
- enthusiasm
- neutral

- worry
- sadness
- love
- fun
- hate
- happiness
- relief
- boredom
- surprise
- anger

From these, empty, and neutral were taken off the dataset. More about this on 3. Tweets are for the most part self-contained, and are thus considered as containing the context necesary to analyze.

For both Friends and EmotionPush, a thousand dialogues are included, where every line is labeled with one of the six emotions from the Ekman model, or a 'neutral' label.

2.4 EmoLex

The NRC EmoLex has been a useful reference. Due to the fact that it relates words with emotions, it can be seen both as a dataset and as a very simple language model of emotions. In this project it will only be used as a dataset.

The EmoLex was created with the Plutchik model of emotions, but two other variables were included. Words can be related to two other concepts: positive and negative. These are considered emotions, and have been removed form our analysis for two reasons. The first one is that, as it has been delimited, this project's goal is to analyze emotions, and not valence. The second one is that valence is in all models of valence a dichotomical variable, and some words in the EmoLex contain both relationships with the positive and negative concepts. This is a conflicting find worth mentioning, but one that falls out of the scope of this thesis.

The lexicon is also filled with words that do not relate to any variable. This effectively reduces the size of the dataset from 14181 words to a few

thousands. A further reduction of the selected words has been made, by selecting only words thar relate to a single emotion. The total number of words that fullfills these criteria is 2344.

2.5 Research Question

When using pre-trained models for word and sentence embedding, **is the information about the emotional and affective content or context of the word or sentence represented in the vector space?**

This representation can be an abstract concept, so to formalize it, the research question can be approached in three different ways:

- Is there a direct correlation between any of the dimensions of the vector space and human-labeled emotions and affect?
- Is there a linear transformation that will yield a direct correlation to the same human-labeled emotions?
- Is there a hierarchical structure that accurately represents the embedding of said labels?

In the next chapter we explain the methodology applied to answer these questions.

Chapter 3

Methodology

To analyze the representation of emotions in different word embeddings, this project has been divided in two main parts: Embedding and Analysis. The embedding part includes selecting the intermediate representation of the dataset, and the usage of the language model to do so. The Analysis is focused on finding the information contained in the language models, through the exploration of the mentioned intermediate representation. A high emphasis on dimensionality-reducing visualizations was done in this last part. These allow for the development of intuitions that can be further explored through statistical tests.

The process for finding information on the desired embedded structure has been divided in consecutive steps, that represent progressive steps into finding structure in a dataset. The steps are the following:

1. **Correlation**: A correlation between embedded dimensions and labels.
2. **Linear Transformation**: A correlation between the labels and a linear transformation of the embedded dimensions.
3. **Non-Linear Transformation**: A non-linear transformation for obtaining separable clusters.
4. **Hierarchical clusterinng**: Clustering the embedded dimensions, or linear transformations of these.

These steps also correspond to the complexity of a theoretical classifier on the embedded dimension. The first one answering to the question: Can the output of the embeddings be used directly to classify the labels? The second step corresponds to asking if a linear transformation of the embedded space could yield a classifier. For example, can an LDA preform on

this dataset and embedding? The third question relates to the preformance of a non-linear classifier. This is the case of a Single Layer Perceptron. Considering the embeddings can be used as an intermediate step, for a further clasification, this step should already yield the results of baseline emotion classifiers. The fourth question relates to the relation of emotion and valence. Emotions tend to fit hierarchicaly into affective models of arousal and valence [Chacón, 2016]. By searching for a hierarchy in the representation of emotions, we should be able to trace back the representation of valence.

Separating the methodology in this way, enables a progressive apporximatons approach to answering the research question. It is highly unlikely that a general lenguage model in machine learning represents emotions into a single dimension in a linear manney, but it is increasingly more likely that some correlation is found with a linear transformation of the aforementioned. In case these two approaches present no information about emotions, a hierarchical clustering can extract the intrinsic information of affect in emotions. Since previous works have already shown that affect can be represented in vector spaces, created with a linear transformation of word embeddings [Hollis and Westbury, 2016], it would be contradictory to not find a hierarchical structure of emotions in this last step. If this were to happen, it would be reasonable to question the dataset and it's methodology, or the contextual information lost in the embedding process.

3.1 Preliminaries

This research was managed as both, a research project, and a software development project. With scientific rigor, order, and reproducibility in mind, a git repository has been setup, where not only the working environment is provided, but also the history of the project development.

Environment Setup

The environment is all required hardware and software necessary to execute this computational experiment. Here it is presented how to reproduce the same environment, to be able to reproduce the results presented. A short organizational note is also included, to keep track of the project management.

Organizational

The project planning was layed out throught three months: March, April and May of 2020. A total of twelve weeks were divided into four equal sprints,

where the four main tasks in the project were equaly separated in time: Exploration and Preparation, Programming, Experiments, and Writing. The four sprints were described by tasks, further divided by sub-tasks. These were kept in track and followed by me, and both Supervisors through the Asana application.

The repository is accessible through github:
`https://github.com/abcsds/MasterThesis`

Hardware

This project was implemented and executed in my personal computer: A Manjaro Linux x86˙64, with Kernel 5.6.11-1-MANJARO. The available CPU is an Intel i7-8700K (12) @ 5.000GHz, and the GPU is an NVIDIA GeForce GTX 1080 Ti. A total of 15937MB of RAM memory were available for experiments, as well as 16GB of swap disk. Although much of the technology available for these experiments is more than necesary, the execution of some BERT models is not possible with these technical specifications. This influenced the selection of the BERT model to be used, and played a big part in selecting a pre-embedding of models.

Software

As mentioned, the development and execution were on a Linux Operating Systetm (OS). The Distribution used was Manjaro. This OS is a rolling release distribution, so the version used changed allong the development. This is one of the reasons why virtual development and execution environments were used: to keep reproductability, and ensure a stable testing. The only reason for this OS to be used is that it is my personal computer. As a Version Control System (VCS), git was added to the repository. This enables distributed access and historical revision for anyone trying to reproduce or supervise the project. Several development tools were used. For text editing and script execution, Atom 1.46.0 was used. Within the Atom environment, community packages were used to simplify the workflow: Hydrogen 2.14.1, for example, allows the execution of python code from within the text editor, and can even show output of the lines executed. For some exploratory analysis, Jupyter Notebooks [Kluyver et al., 2016] were used. To run these, a specific virtual environment was created with Docker 19.03 and NVIDIA-Docker. A docker image for theses notebooks was created. The dockerfile of this image contains the libraries used for data exploration. The downloading of the BERT models ran in TensorFlow is also contained in this dockerfile. The description and an initialization script for the virtual container are in-

cluded in the project folder called 'TF'. While the notebooks provided were used for data exploration, and visualization. Most of the development was done on the text editor. For this, python virtual environments were created with the help of the `virtualenv` and `virtualenvwrapper` python libraries. For these, a 'requirements.txt' file was provided with the libraries used, and their versions. When developping, the desired virtualenvironment was activated. After this, the atom editor is open on the desired folder. By doing so, the Hydrogen library takes the virtual environment for the execution of the code in the project. By developing in this way, the whole project is available from the folder view on Atom. Code can be executed, and tested on the run, as if it were a Jupyter notebook, but changes are inmediately integrated into the code repository. This specific development environment was seleted to avoid conflicts between Jupyter Notebooks, and the VCS. The explorations are stored as notebooks, but cannot really represent the development of the project. The Python programming language was used for the programming of the current project. This is due to it's incredible flexibility, access to the main ML libraries, and the predisposition of the Wirtschaftsinformatik und Maschinelles Lernen Institut. Under Python's umbrella of libraries, several were specifically added to enable this study. A List of the used libraries is provided in the appendix 6.3.

Two main ML frameworks were selected for the current project: TensorFlow 2.1.0 [Abadi et al., 2015] (TF), and PyTorch 1.4.0 [Paszke et al., 2019] (Also called Torch, for simplicity.). TF was selected specifically for it's access to a pre-trained BERT library [Lai, 2019] for embedding sentences. This was very usefull, since, compared to the Transformers library [Wolf et al., 2019], it must not be fine-tunned. TF confronts developers with two main compatibility issues:

- The cuda library being used most be a specific version. Most TF libraries will only work under CUDA library 9.2. Some might run under 10.1, but not under 10.2. Since the development environment is a rolling release linux distribution, the latest version of libraries is provided. Installing multiple versions brings problems to the day-to-day usage. Since the environment is also my personal computer, a virtual environment with containers were used instead, and for these, NVIDIA-Docker.

- at the moment of the development of this project, TF is undergoing a major version change, from 1.x to 2.x. Many reference libraries, and all code I have creted, used, or studied in my masters is depricated. The techniques learned during my studies need to be updated, and in many cases, re-learned. This is not an uncommon problem in technology, but it opens the opportunity for changing the work framework.

For all ML programming requirements that did not use the pre-trained BERT library, PyTorch was used. Certain algorithms were not programmed, but simply integrated from their implementation on python:

- FastText: This algorithm was not implemented. It's python library from the implementaiton of Facebook Research was used [Joulin et al., 2017].
- MulticoreTSNE: The TSNE algorithm was not implemented. Since it has heavy requirements on hardware, its implementation using distributed computing was used [Ulyanov, 2016].
- Normalize: The sklearn version of the normalization algorithm was used due to its optimization [Pedregosa et al., 2011].
- PCA: SKlearn version was used [Pedregosa et al., 2011].
- Tokenization: Part of the embedding pipeline requires the tokenization of the sentences. This was done with the Spacy library, and the "en˙core˙web˙sm" model.[Honnibal and Montani, 2017]

The Datasets

Three datasets were selected to be used for this project. Here it's described how and when they were accessed, stored and embedded into the intermediate representation.

Access

Accessing datasets to train machine learning models is not a standarized process. The developer of every dataset is in charge of the distribution method. Fortunately, two of the three datasets used in this project were distributed by the same organization: the EmotionPush, and Friends datasets were, while CrowdFlower was distributed originally by a company with the same name.

The CrowdFlower dataset was downloaded from the official CrowdFlower website in October 2019. The url to this dataset is `http://www.crowdflower.com/wp-content/uploads/2016/07/text_emotion.csv`. As of May 20, 2020, this link still works, but the website `www.CrowdFlower.com` redirects to `www.appen.com` a company that 'collects [data] to build [...] artificial intelligence systems.' This company offers access to some open source datasets, but the mentioned crowdflower emotion dataset is not listed there. A discussion on this is provided on chapter **??**.

The EmotionPush and Friends datasets were distributed as part of the EmotionX Task, which in turn is part of a set of Social NLP tasks, created by

the Academia Sinica of Taiwan [Chen et al., 2018]. To access this datasets, one must register on the EmotionX 2019 website: `https://sites.google.com/view/emotionx2019`. Access to a google drive is then granted via email, and a zip file with both datasets can be downloaded. This dataset has been used more than once in different analysis on the internet, and it can be therefore accessed without permissions to the official method. Here, the original dataset is used.

Storage

The datasets were downloaded and stored under the project folder 'data'. Since every dataset is provided in different format and under different folder structures, every dataset is simply stored inside a folder with it's name. Under the datasets folder, every selected dataset is accompanied by folders with the embedding model used to embed the dataset. Thus every dataset folder has several subfolders. On these subfolders, a python script called 'embed.py'. This script varies for every model and dataset. In general terms, it extracts the text and label from the dataset, embeds the text into the desired model, and stores it in a 'csv' file under the same folder. The 'csv' file is stored under the name 'embedded.py', except for the FastText model. In this case, there are two embedding approaches, one supervised and one unsupervised. Thus the names of the FastText embedding files are 'embeddings˙supervised.csv', and 'embeddings˙unsupervised.csv'. Every other script creates a single 'csv' file called 'embeddings.csv'.

Thie file structure of the embedded files allows for exploration and experimental scripts to access the embedded data of different datasets, by building a single string with the dataset and model selected. This string must be prepended by the './data/' folder name, and appended with the 'embeddings.csv' string to generate a path that creates accesibility to the different datasets via a python coma-sepparated-value library, such as the built in `csv`, or Pandas [pandas development team, 2020] and it's `read_csv` function. This effectively create a data source to be used in a data pipeline. This approach was selected due to it's simplicity.

3.2 Embedding

The comparisson of the representation of different language models in this project requires a convergence of many different techniques. For this reason it was chosen to embed the datasets into an intermediate format, to later use them in experiments.

The embedding of the datasets is comprised of 5 steps:

1. Loading model, text, and labels.
2. Tokenizing text.
3. Embedding every token into the model latent space.
4. Average the given embedded words.
5. Store the average sentence embedding.

Embedding Methodology

Loading text, and labels was done with either the CSV or the JSON python library, depending on the format of the data.

Tokenizing was done with Spacy's 'en˙core˙web˙sm' model, which allows access to the tokens via an iterator on the model, and the sub-compontent 'text'. An small snippet showing this process is shown in 3.1. This snippet considers a model has been loaded as a dictionary on tokens.

Listing 3.1: Tokenizing with Spacy

```
import spacy
nlp = spacy.load("en_core_web_sm")
for token in nlp("This is a sentence in English"):
  word_embedding = model[token.text]
```

Every token is embedded in this way, but some models might not contain some tokens. In this case, the token is simply skipped. Some tokens with relevant information can be lost with using pretrained models that don't contain the complete vocabulary of the dataset, but it is expected, that the information distribution converge to the real distribution when large number of samples are integrated.

Once every token has been embedded into the model's latent space. A simple average is done across the tokens, keeping the dimensionality of the vector representation, and effectively creating a sentence embedding, represented in the model's latent space. This technique was selected since it's the most common method for sentence representation. With this method, the secuential nature of the tokens in the sentence is lost, in favor of providing a constant sized sentence embedding to compare between methods and datasets.

Lastly, the sentence embeddings are stored along with the label information. For this, the CSV format was selected, due to it's interoperabiliy, and

accessibility. The statistics library Pandas has an excellent csv reader, but the data can also be imported into spreadsheet software, other statistical software, or very quickl loaded on to python with the CSV library. Every CSV file contains a header on the first row. The header is composed by $N + 1$ columns where N is the number of latent dimensions in the model. The last column is the "Emotion" column, where the label is stored. The name of every column starts with the letter 'd', and is followed by consecutive numbers.

Since every pre-trained model is different, there were specific requirements on loading the model and embedding the tokens:

FastText

As previously mentioned, the FastText algorithm is an exception in this project, since it is NOT a pretrained model. The model is trained based on the dataset given. This can be done in a supervised, or an unsupervised manner. Due to the two methods for the usage of the FastText python library, the process of embedding a dataset with it requires two extra text files one with a sentence per line, and a second one, which includes the label as the last word of every line, prepended by two underscores (__).

In both ways of training, the language model is being trained specifically for the dataset vocabulary. For this reason, all tokens will be available in the model's vocabulary, resulting in the most complete language model. This is at the cost of representing only the topics on the dataset. This is therefore also not a general languge model.

Word2Vec

Word2Vec is trained in a very similar way as fasttext. Therefore, the expected results are similar. Word2Vec is treated within the context of this experiments as the pre-trained equivalent of fast text. The same number of latent dimensions, and a similar training approach were used. In this case, if a word in the dataset is not contained in the Word2Vec model, it is droped, and its analysis won't be included in the results of this project. Word2Vec was trained with a very large corpus, it is therefore considered a general language model.

The pre-trained model has been stored under the project folder './models/Word2Vec/GoogleNews-vectors-negative300.bin.gz'. The gensim python library is used to load the model in binary format without having to decompress it. This model is loaded as a dictionary. An example of this is shown in snippet 3.2 that considers an iterator over a tokenized sentence.

Listing 3.2: Loading Word2Vec

```
import gensim
model =
    gensim.models.KeyedVectors.load_word2vec_format(model_path,
    binary=True)
for token in tokenized_sentence:
  word_embedding = model[token]
```

GloVe

Pre-trained GloVe models can be downloaded from the official website `https://nlp.stanford.edu/projects/glove/`. This model was downloaded and stored under the project folder './models/GloVe/glove.6B/glove.6B.300d.txt'. The name of this file contains two numbers: 6B is the number of words that are represented in this model, while 300d is the number of latent dimensions used to represent the vocabulary. This model has been trained for 50, 100, 200, and 300 dimensions. Since a smaller number of dimensions represents a lesser capability for representing complex language concepts [Pennington et al., 2014], the larger version of this model was selected. This also concides with the number of dimensions used in Word2Vec, which makes results easier to compare.

BERT

Although BERT is a pretrained model, it's original distribution is considered to be only partialy trained. On the original paper [Devlin et al., 2019], a fine-tunning task-specific phase is mentioned, and generally required for the model to work best. This finetunning also presents a great infrastructure challenge, since some pre-trained BERT models simply wont fit into a personal computer's RAM.

For this reason, the pre-trained BERT embedding library `https://github.com/imgarylai/bert-embedding` was used. This library allows for a selection of the BERT model, and the embedding of the whole sentence, without tokenization. The result is a json-like dictionary in Python that contains both the original sentence and the embedded sentence.

To be able to run the embedding notebook, provided under the project folder 'exploration/Embedding with bert.ipynb', the following requirements should be met:

- Docker $\geq$ 19.03

- NVIDIA Container Toolkit
- This Docker TF Image:
  `tensorflow/tensorflow:2.1.0-gpu-py3-jupyter`

On Linux, the a correct installation of the nvidia-docker environment would yield a successful run of the following command: `docker run --gpus all --rm nvidia/cuda nvidia-smi`

To build the docker image for this project, one must open a terminal on the 'TF' project folder and run the following docker instruction: `docker build -t bert .` where bert is the name of the image to be created. Once this image has been built, docker can create containers with it. So to run the container necessary for the BERT embedding, the following command is used inside the project folder: `docker run --gpus all -p 8888:8888 -v $(pwd):/tf -it bert`. This last command will run a docker container, based on the "tensorflow:2.1.0-gpu-py3-jupyter" image, connect it to the localhost port 8888, and integrate the project folder to the jupyter server running on the container.

Docker is used to comply with the complex requirements of TensorFlow, CUDA, and the bert-embeddings.Once the Jupyter server is running, the notebook can be opened, and executed. The loading of the model is shown in the following snippet 3.3:

Listing 3.3: Loading BERT

```
from bert_embedding import BertEmbedding
bert_embedding = BertEmbedding(model='bert_24_1024_16',
    dataset_name='book_corpus_wiki_en_cased')
```

Here, the selected model is shown. This is a model with 1024 latent dimensions, trained on the Wikipedia corpus, and with case sensitivity. This means that words lowercase and uppercase letters will be embedded differently.

Within the notebook, a function was created to embed the datasets. This receives three arguments: a list of the sentences, a list of the labels, and the name of the output file, as a string. The embedding function is shown here:

Listing 3.4: Embedding with BERT

```
def embed_and_save(X, Y, outpath):
    E = np.array([np.mean(t[1], axis=0) for t in bert_embedding(X)])
    with open(outpath, 'w', newline='') as f:
        fieldnames = [f"d{i}" for i in range(len(E[0]))] +
            ['emotion']
```

```
        writer = csv.DictWriter(f, fieldnames=fieldnames)
        writer.writeheader()
        for e, l in zip(E, Y):
            writer.writerow(dict({f"d{i}": ei for i, ei in
                enumerate(e)}, **{'emotion': l}))"
```

Running the embeddings for the datasets reported the following data:

Dataset	User	System	Total	Wall
CrowdFlower	user 3h 57min 7s	13min 11s	4h 10min 18s	1h 5min 33s
EmotionPush	user 1h 28min 35s	5min 25s	1h 34min 1s	24min 31s
Friends	user 1h 26min 40s	5min 3s	1h 31min 44s	24min 9s

Table 3.1: Runtimes for embedding datasets with BERT

This is much less than the 'several days' verbaly reported by colegues at the ISMLL. This might be due to the use of pre-trained models, and not running back-propagation to fine-tune the language models.

While running the embeddings, almost no GPU memory was used. This signals that the library is actually not making use of the GPU resources available. This also might mean that the embedding of the datasets might be much faster if the correct hardware resources are used.

At the beginning of the Year 2020, the library seemed a reliable way of getting the embedding done quickly. It allowed for embedding of the complete datasets in matter of minutes. Since I had been warned BERT embeddings could take days, I saw this as a gread advantage, and kept the method. Unfortunately as of May 2020, this library has been depricated. It's unmantained, and has requirements that might only be achievable under very specific conditions. This will not be a problem for reproduction, as long as the library is still available, and the provided docker image is used.

3.3 Analysis

The python scripts to analyze the data are found under the folder './exploration', where they are numbered, and named. The order of the scripts corresponds to the progressive steps in the search for structure in the embedded spaces. The scripts are the following:

1. `01_corr.py`
2. `02_pca.py`

3. `03_tsne.py`

The order of these scripts corresponds to the methodology proposed in this thesis. They generate the visualizations, and test the hypotesis on the data. The last step in the methodology, Clustering, has been performed in all scripts. The scripts each contain a list of strings. Every string in that list is the relative path of one of the pre-processed datasets. These strings can be commented out. In doing so, the analysis will not be run on that specific instance. The full list is declared as follows:

Listing 3.5: Pre-processed datasets

```
dss = ["data/CrowdFlower/FastText/embeddings_unsupervised.csv",
       "data/CrowdFlower/FastText/embeddings_supervised.csv",
       "data/CrowdFlower/GloVe/embeddings.csv",
       "data/CrowdFlower/Word2Vec/embeddings.csv",
       "data/CrowdFlower/BERT/embeddings.csv",
       "data/EmotionPush/FastText/embeddings_unsupervised.csv",
       "data/EmotionPush/FastText/embeddings_supervised.csv",
       "data/EmotionPush/GloVe/embeddings.csv",
       "data/EmotionPush/Word2Vec/embeddings.csv",
       "data/EmotionPush/BERT/embeddings.csv",
       "data/Friends/FastText/embeddings_unsupervised.csv",
       "data/Friends/FastText/embeddings_supervised.csv",
       "data/Friends/GloVe/embeddings.csv",
       "data/Friends/Word2Vec/embeddings.csv",
       "data/Friends/BERT/embeddings.csv"]
```

This was done so to facilitate the integration of new datasets or models to the analysis. As mentioned before, the csv file contains a line for every sentence in the dataset, with the number of columns equal to the dimensionality of the model, plus one, for the label.

Every analysis script separates the embedded sentence from the label, into two structures:

- X: contains all the embedded sentences, and is therefore of size NxM, where N is the number of sentences in the dataset, and M is the number of latent dimensions in the model.
- Y: contains all the labels of the dataset, and is of size $Nx1$.

Correlational Analysis

The correlational analysis runs the numpy `corcoef` [Oliphant, 2006] algorithm between every dimension of the model, and the labels vector. A snippet of

the algorithm can be seen in listing 3.6

Listing 3.6: Corrlelation Algorithm

```
1  cors = []
2  for emotion in ind:
3      y = (Y == emotion).astype(int)
4      cor_p_sent = []
5      for j in range(X.shape[1]):
6          x = normalize(X[:, j].reshape(-1, 1)).reshape(-1)
7          c = np.corrcoef(x, y)[1,0]
8          cor_p_sent.append(c)
9      cors.append(cor_p_sent)
10 cors = np.array(cors)
11 x = np.nan_to_num(cors)
```

The correlation is done between every dimension, and every emotion. Therefore, the labels vector is filtered with the selected emotion, as it can be seen on line 3. This results in a vector of size N filled with zeros, except in the places where the selected emotion is the label. This ones-and-zeros vector is the reason why the dimensions vector is normalized. The latent space of every model is different. By normalizing it, we restrict the embedding values between 0 and 1, since the default normalization algorithm uses the L2 norm. The next step, evaluating the numpy corrcoef function, returns the Pearson product-moment correlation matrix. A matrix is formed from the correlations of every dimension, against every emotion. The resulting matrix is then of size MxE, where E is the number of the emotions labeled in the dataset.

Linear Dimentionality Reduction

For a linear dimensionality reduction algorithm, PCA has been selected. The methodology here only differs from the linear correlation analysis in that a PCA transformation is preformed before examining correlations. It looks as follows:

Listing 3.7: PCA correlation Algorithm

```
projection = PCA().fit_transform(X)
cors = []
for emotion in ind:
    y = (Y == emotion).astype(int)
    cor_p_sent = []
    for j in range(projection.shape[1]):
```

```
        x = normalize(projection[:, j].reshape(-1, 1)).reshape(-1)
        c = np.corrcoef(x, y)[1,0]
        cor_p_sent.append(c)
    cors.append(cor_p_sent)
cors = np.array(cors)
x = np.nan_to_num(cors)
```

Non-Linear Dimentionality Reduction

The non-linear dimensionality reduction algorithm selected was the T-distributed Stochastic Neighbor Embedding (TSNE). This uses the distribution information from the embedded sentences to search for a linearly-separable two-dimensional projection. This algorithm was selected for its relationship to visualizations. The multi-core algorithm from the MulticoreT-SNE library was used [Ulyanov, 2016].

Clustering Analysis

The clustering algorithm used was SciPy's linkage algorithm, a part of the cluster.hierarchy library [Virtanen et al., 2020]. This algorithm is an aglomerative, or bottom-up clustering algorithm. It measures euclidian distance between the selected data points, and clusters them one by one. The result of this algorithm can be seen as a dendrogram plot, next to every heatmap in the results section 4.2.

Chapter 4

Experiments

The amount of experiments, data and visualizations created in this project are more than a dedicated reader is confortable reading in one single pass. For this reason, the experiments and results in this chapter first presented with a simple example, the EmoLex. This example will guide the reader through the methodology to analyze the datasets, and introduce the intuitions presented through simpler visualizations with lesser data. With this intuitions in mind, the results of the other datasets, and language models are presented. In this way, the EmoLex works not only as an introduction to the methodology and results, but also as a lax baseline.

4.1 EmoLex

As an introductory experiment, EmoLex has been embedded into the abstract vector representation of the GloVe language model. This model was selected due to its one-to-one relation between word and embedding, and it's context indepedence. This means that a word will get one single embedding no matter what other words apear next to it. This in comparison to BERT, where the embedding of a single word deppends on the tokens, words, and sentences that come with it. The EmoLex has single words related to emotions, so it must be noted that this experiment and it's results relate to word embedding, and not sentence embedding.

Correlational Analysis

A correlational analysis answers the question of how much does every dimension of the language model relates to every emotion. The result of this model is a correlation matrix where every element of that matrix is a num-

ber between -1 and 1. In the case of the EmoLex, abstracted into the GloVe laguage model, it is a $8x300$ matrix, where the rows correspond to the 8 emotions represented in the EmoLex. This matrix is visualized in Figure 4.1. The plot shown in this figure is called a clustermap. It was created with pythons Seaborn library, and consists of two parts:

The main part is the visualization of the matrix. In this visualization, every number of the matrix is given a color from a range of colors shown in the colorbar. The range is automatically set from the maximum and minimum number in the matrix. This allows to see paterns in the releationships between the vector space dimentions and the emotions, if there are any.

The second visualization of this plot is the left dendrogram. A dendrogram is the visualization of a clustering structure. This clustering is done through a nearest neighbor algorithm, using euclidian distance.

A correlational plot is used here to analyze the deviation of the distribution of the representation of variables in the different dimensions of the vector space. In this case, there are 8 variables. A uniformely random distribution of the representation would yield a correlation of 0.125. Thus a correlation of 0.125 or lower is considered random. In our baseline reference by Hollis et al. [Hollis and Westbury, 2016], where they analyzed 9 variables the minimum significant correlation reported was 0.35. This is more than three times the correlation of a uniformely random distribution: 0.111$\underline{1}$

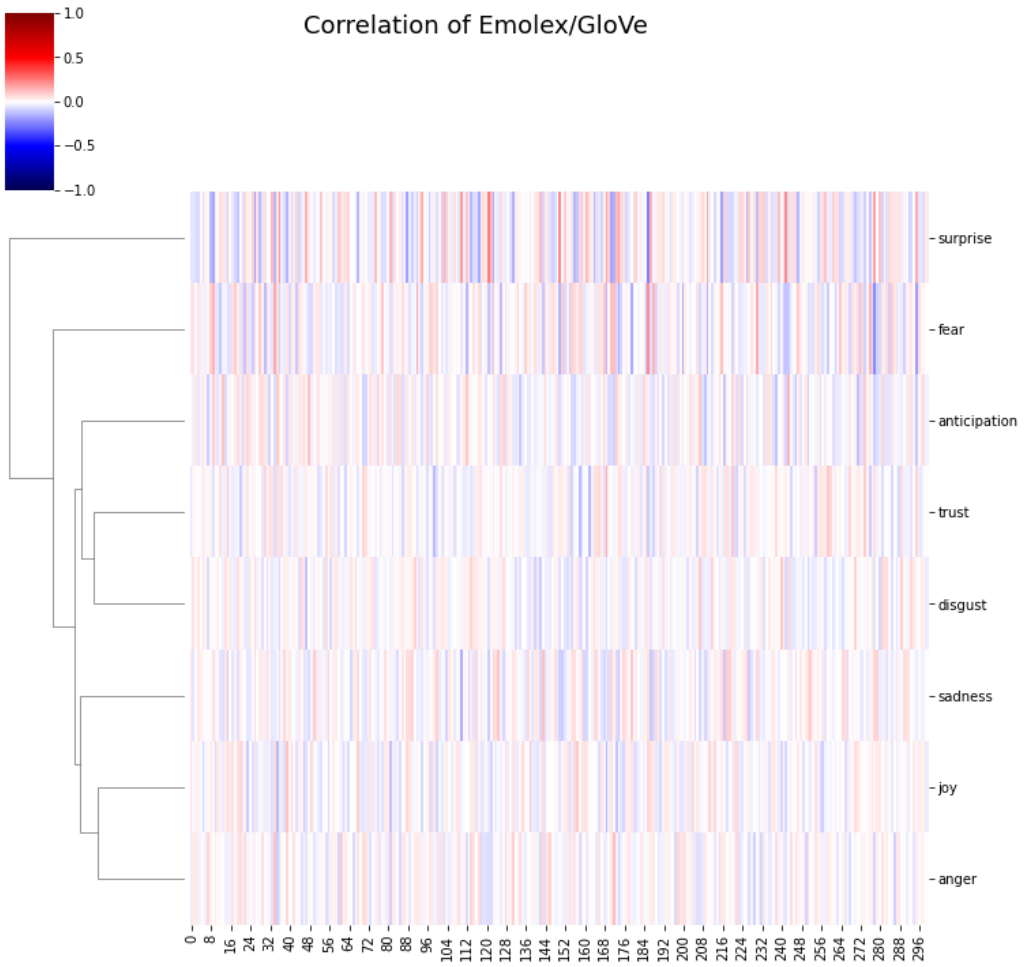

Figure 4.1: EmoLex Correlation Plot

In Figure 4.1 it can be observed that represented by the GloVe model, the words contained in the EmoLex are uniformely distributed through the dimensions. The maximum correlation is 0.251, between the emotion 'Trust' and dimension 121 of the model. Although this is double the correlation that could be found throgh chance, it does not satisfy the baseline condition, and thus, we cannot consider that there is a linear correlation between the emotions of the EmoLex, and the dimensonal representation of the Language Model.

The corresponding clustering algorithm shows that through their vectorial representation, the two most related emotions are joy and anger. This does not comply with the findings of dichotomical emotions, fitting into a valence model, as shown in the 2016 study [Chacón, 2016].

These two results are enough to reject the hypotesis that there is a linear

correlation between the emotions in the EmoLex and the vectorial representation of the GloVe model.

A further correlational study can be done solely to the labels of the EmoLex. This is here called an Emotion-to-Emotion correlation, and it's obtained through the self-correlation matrix of the one-hot-encoding of the EmoLex emotion labels. The clustermap of said matrix can be seen on Figure 4.1. An organic distribution of these would show the expected hierarchical clustering between positive and negative emotions. This plot, although labeled as related to the GloVe, is independant from the language model, since it only looks at the labels, but the name has been kept for consistency.

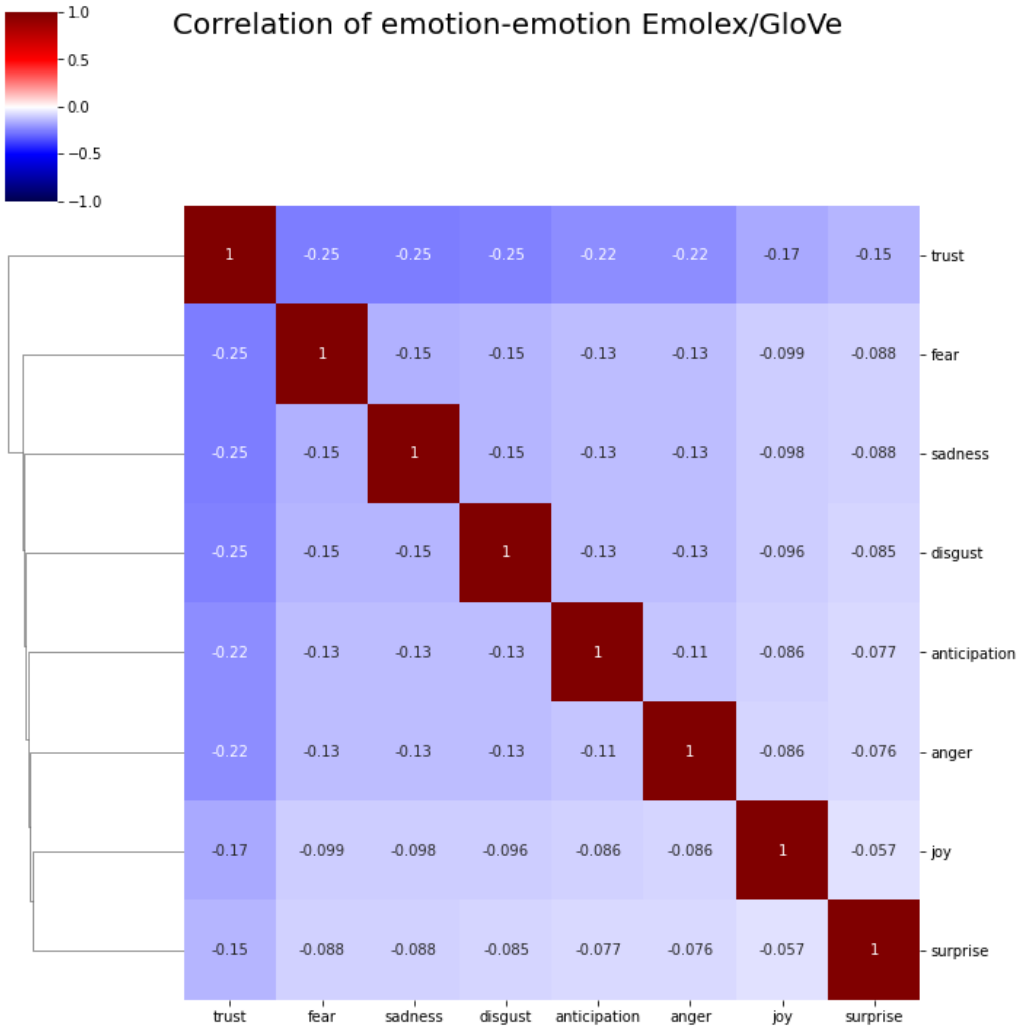

Figure 4.2: EmoLex Correlation Plot

The EmoLex is not an organic corpus, and thus the labels do not represent the hierarchical clustering shown in the 2016 study [Chacón, 2016]. Instead the emotions are uniformely distributed. This is expected for a lexicon.

PCA

By maximizing the ammount of information represented by the first components through a linear transformation of the data, a PCA analysis provides a perspective on the representation of variables in the model that cannot be seen through a simple correlation.

By visualizing a scatter plot of the first components or PCA dimensions, if there is a linear separation between the labels it can be visualized. Even in the case of data that is not linearly separable, a gradient of the distribution of the data could indicate the existance of a structure.

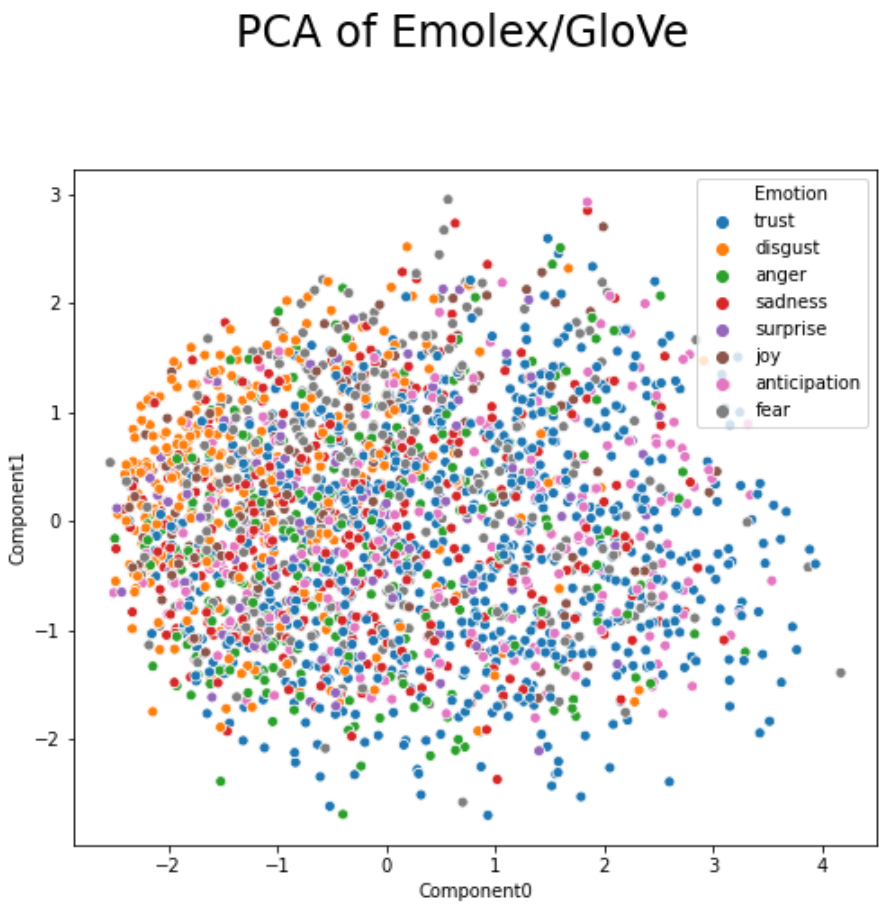

Figure 4.3: EmoLex Scatter plot of PCA

In Figure 4.1 we can see the scatter plot of the words of the EmoLex, projected on to the first two components of the PCA transformation. Here we can observe, that the words labeled with the emotion disgust are mostly centered to the top-left of the plot. Although this might seem like the indication of a structure, we cannot certantly point at it. This plot is useful for visualizing linear separation, but that linear separation does not show in this case. To be able to visualize what the PCA transformation does to the vector space, the correlation matrix is shown next.

The correlation matrix shown next is the same type of visualization as the one in Figure 4.1, but this is done to the transformed dataset. This means that the x-axis shows the number of the component instead of the dimension

of the language model.

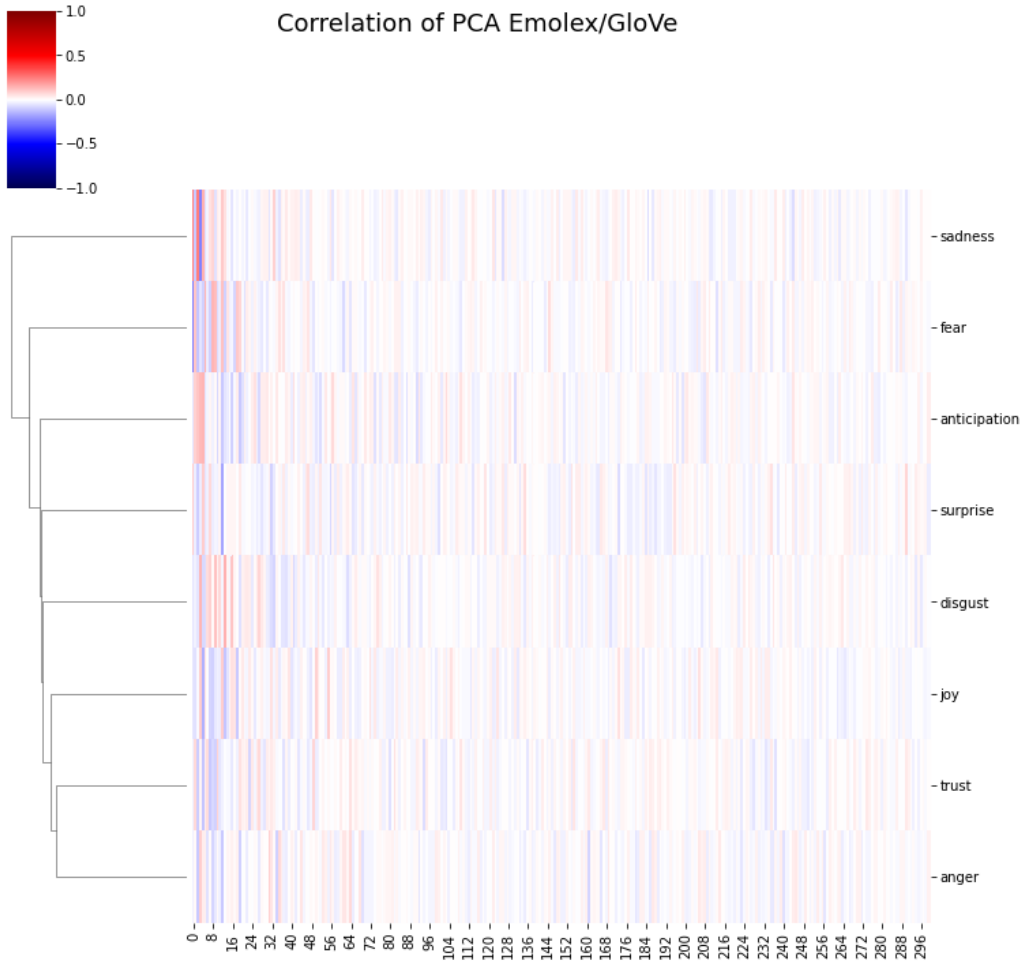

Figure 4.4: EmoLex Correlation of all PCA components

By plotting the correlation matrix of the PCA transformation of the EmoLex, we can observe that most of the variability is concentrated on the first components. Still in this case, the maximum correlation is 0.2567, barely higher than on the linear correlation. The hierarchical clustering shows no precence of a valence-correspondant clustering.

Considering that the PCA is a linear transformation we do not expect to see better correlations between the model dimentions, and the concepts, but a dense representation that does not require looking at all dimensions. For this reason, when looking at PCA correlations, only the first eight dimensions will be shown. Eight is an arbitrary number that allows a square correlation matrix, since it's the same number of emotions for this dataset.

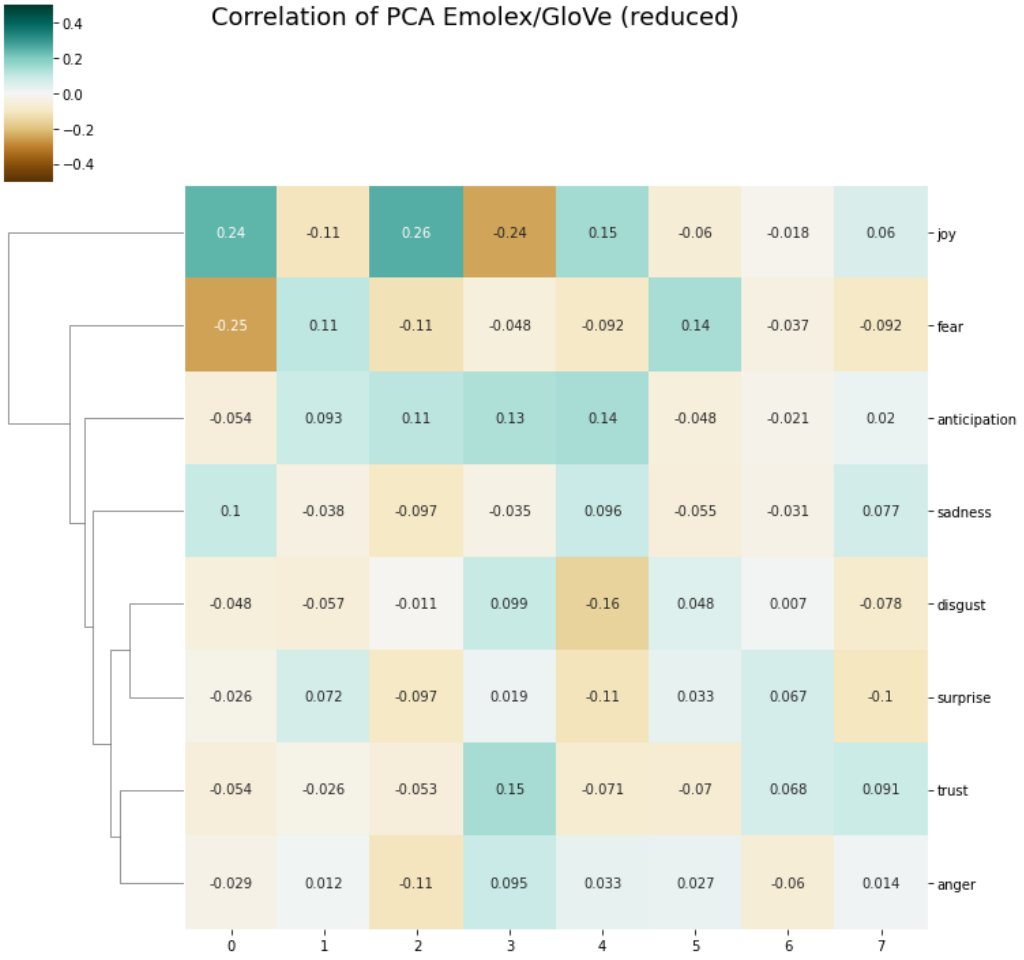

Figure 4.5: EmoLex Correlation of first PCA components

Figure 4.1 shows the first 8 dimentions of the PCA transformation for this model and dataset. A different color scheme has been selected to easily identify between the linear correlation visualization, and the PCA-transformed correlations. This is the exact same data as the one used to create Figure 4.1, but since only the most information-dense dimensions are being shown, the hierarchical clustering is much different. Here we can observe that what had no apparent structure now seems to cluster disgust with surprise, and trust with anger. By observing only the first component, we can see that if it activates, we can expect that the emotion Joy is not expressed in the embedding, with a 24% probability, and that fear is not expressed with 25% probability. This results are not very concise, but this is expected due to the artificial nature of the dataset, and the subjectivity of the concepts. In the expreiments section of this chapter we aim at describing how much of that

variability is due to the subjectivity of the concept.

TSNE

Through this, non-linear visualization algorithm we can plot a scatter that will concentrate the maximum variability possible on two dimensions. The result is the plot on Figure 4.1.

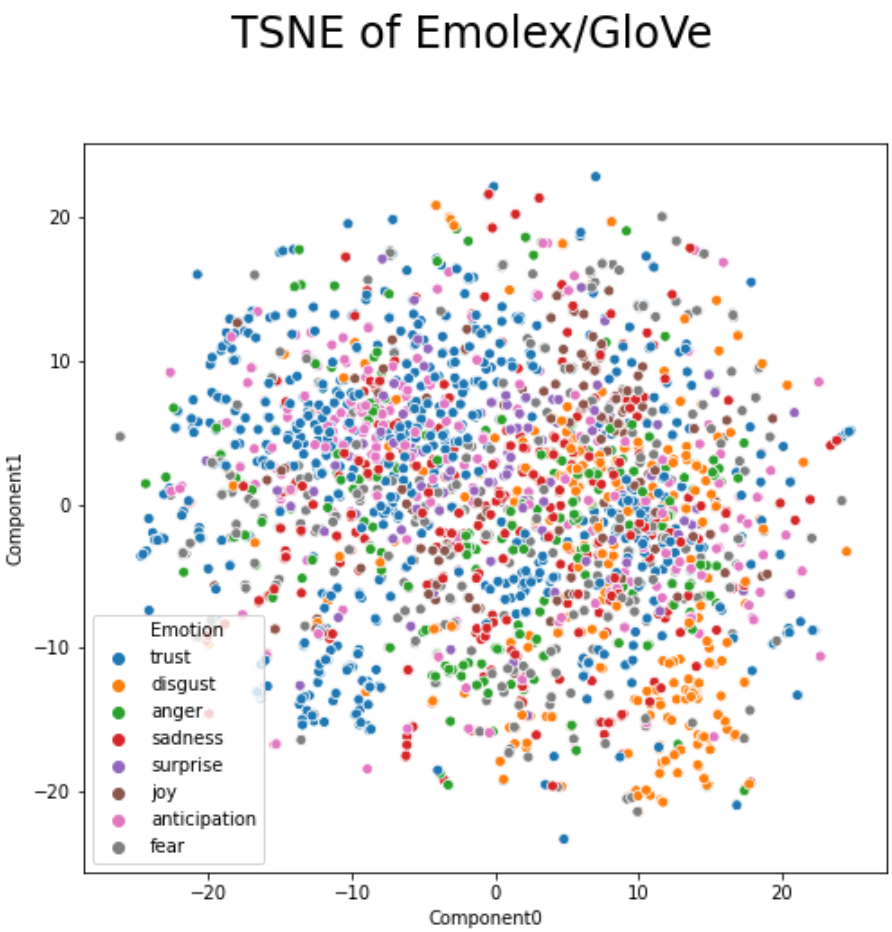

Figure 4.6: EmoLex Scatter plot of TSNE

In this plot we can observe that two major distribution centers are created: one on the top of the plot and one on the bottom. The two groups are

represented by two emotions: trust and disgust. These two are indeed polar opposites on the Plutchik model.

Within the these two groups, we can also observe sub-clusters. A further analysis of the plot has been done with the help of a bokeh interactive plot, present in html format in the plots folder of the repository. Here we can read the words in the subcusters.

One of the subclusters with the trust label is the one formed by the words: aposite, apostolic, chapland, vicar, episcopal, parish, deacon, ordained, priest, congregation. The subcluster also contains the word reverend, labeled with the emotion joy, and is very close to the subcluster with the words convent, nun, monk, abbot, and cannons, labeled with trust, and mystic, labeled with surprise.

With this example we can comment on what we expect to see in further experiments. Subclusters are formed by words that have semantic relationship, as it is expected from a functional language model, and the emotions labeled on to that word are an expression of the context of the person labeling the word, more than the intrinsic emotion of the word. This is conformant with the theory of constructed emotions.

These subclusters form an important part of the analysis, and have therefore been given a name: 'semantic islands'. A semantic island is a cluster of datapoints that have a semantic relationship.

As it is expected from a non-organic dataset, the structure to be found seems artificial. The distribution of the labels and words is uniform, and no conclusion must be drawn form it to apply to general language. Nontheless, the visualization of the EmoLex as a dataset lets us develop the beggining of an intuition, and a technique as of how to approach this problem. In the following section, the same methodology will be followed to analyze the selected datasets and models.

4.2 Results

The following results are separated into three: The correlation analisis, the PCA, or linear transformation, and the TSNE, or non-linear transformation.

Correlation Analysis

A linear correlation is now compared between the four forementioned language models. The results that best abstract the specifics of this dataset are those of the FastText model, since it has been trained specifically for this dataset. This is our baseline, and thus will be presented first:

FastText

With the FastText model, the maximum correlation was shown in the supervised approach, with 33.45% correlation between the hapiness emotion, and it's third dimension. The visualization of said model is shown in Figure 4.2.

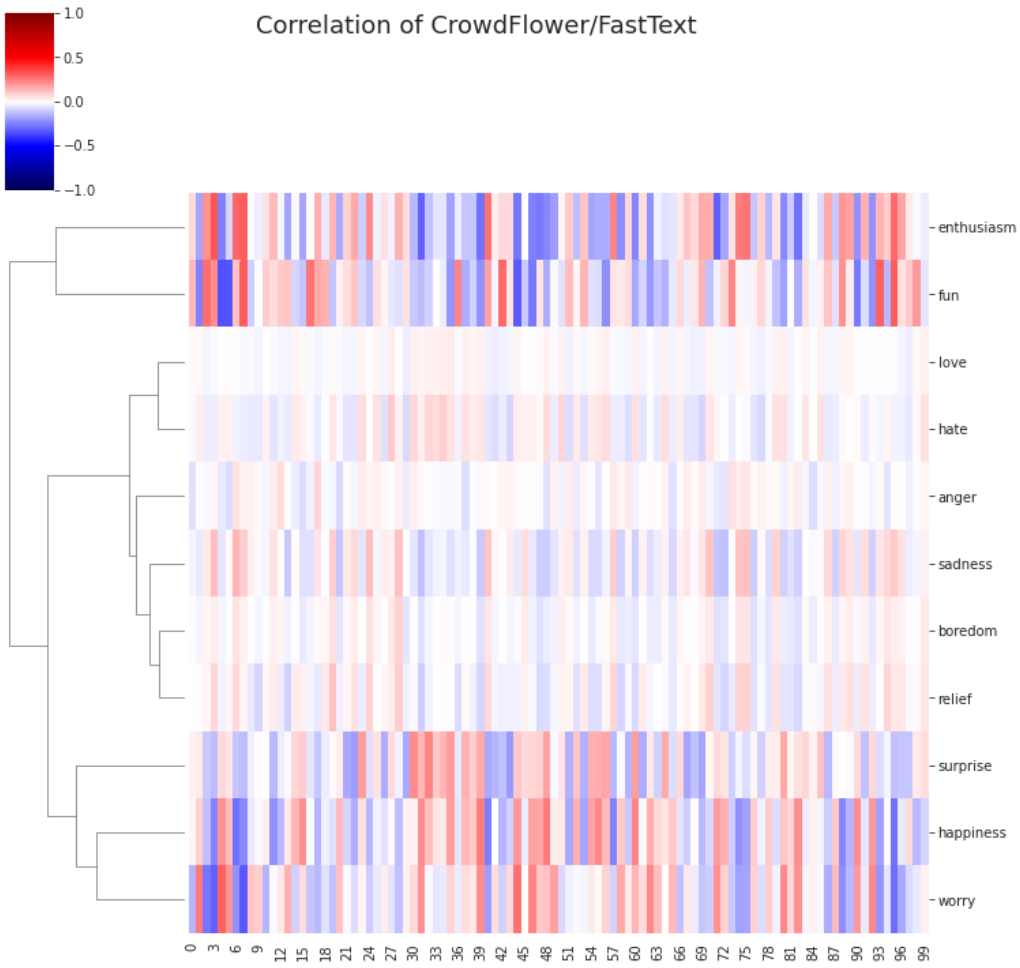

Figure 4.7: Correlation plot for FastText

A hierarchical structure of the concepts can also be observed, with two main groups: fun and enthusiasm in one, with a high activation of all dimensions (be it possitive or negative), and the rest on a second group. The latter can be subdivided into two. The first group contains emotions of surprise, happiness, and worry, while the second one contains the rest of the emotions with very little activation.

An interesting observation is that love and hate have been clustered to-

gether, even if the activation of the vector space is very reduced.

Word2Vec

Word2Vec is the first pre-trained model to be reviewed. This model has been observed to contain an abstraction of valence, and it is thus expected to show it. Hollis et al. report that 208 dimensions out of the 300 dimensions of te model correlate with the concept of valence. They unfortunately do not mention how much the dimensions correlote [Hollis and Westbury, 2016]. In this case, the model's maximum correlation was between the happiness concept, and it's 43th dimension, with 11.94%. This is means that most of the dimensions did not correlate significatntly with the concepts we are exploring, since the random threshold is 9.09%, and the maximum correlation found barely surpases that, with most of them under the treshold.

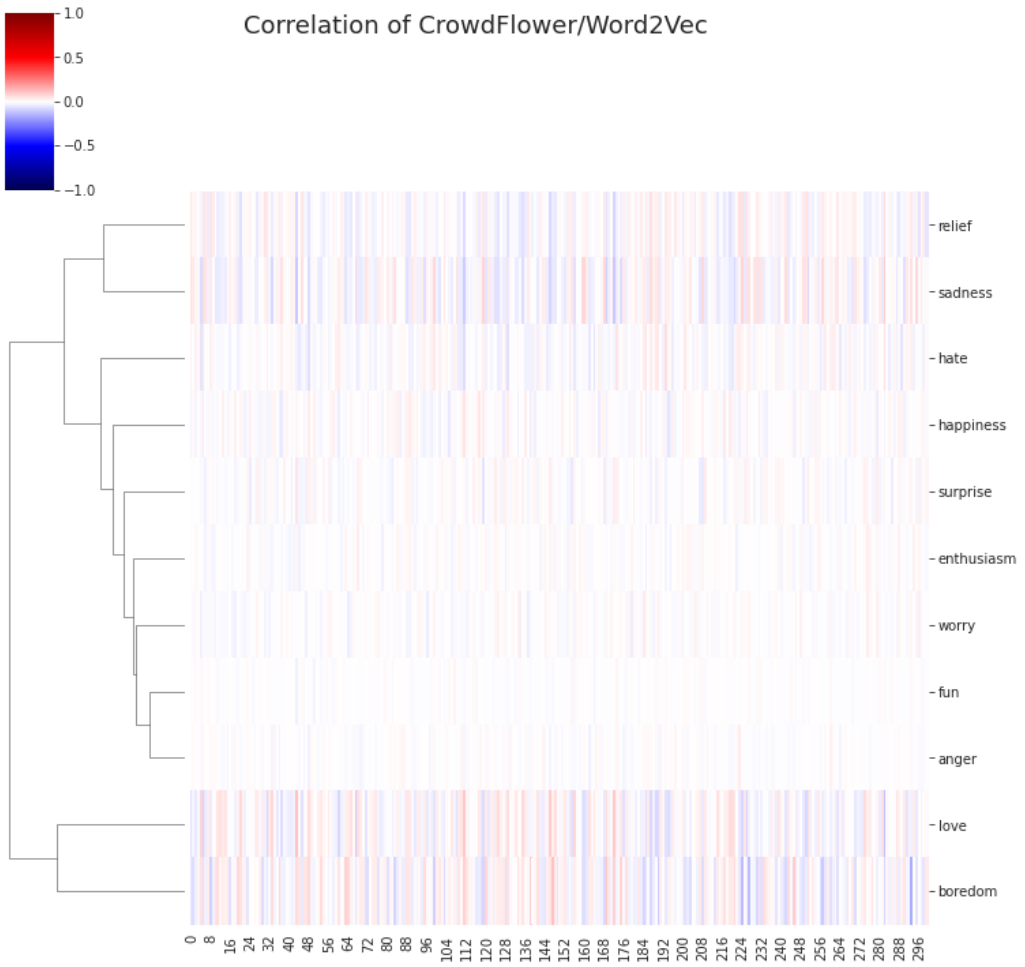

Figure 4.8: Correlation plot for Word2Vec

Figure 4.2 shows the mentioned correlations. At first sight, it might seem that the clustering is similar to the one shown with FastText, with two main groups, one of which is again separated into two, but the emotions clustered are not the same. Clustered together are love and boredom, releif and sadnes, and fun and anger. These are neither opposites, nor subsets of a valence side.

GloVe

The GloVe model presents a higher complexity than the Word2Vec. Thus, better results are expected. Here, the best correlation was 11.97 %, between the worry concept, and the 164 dimension.This is again, barely above the random treshold.

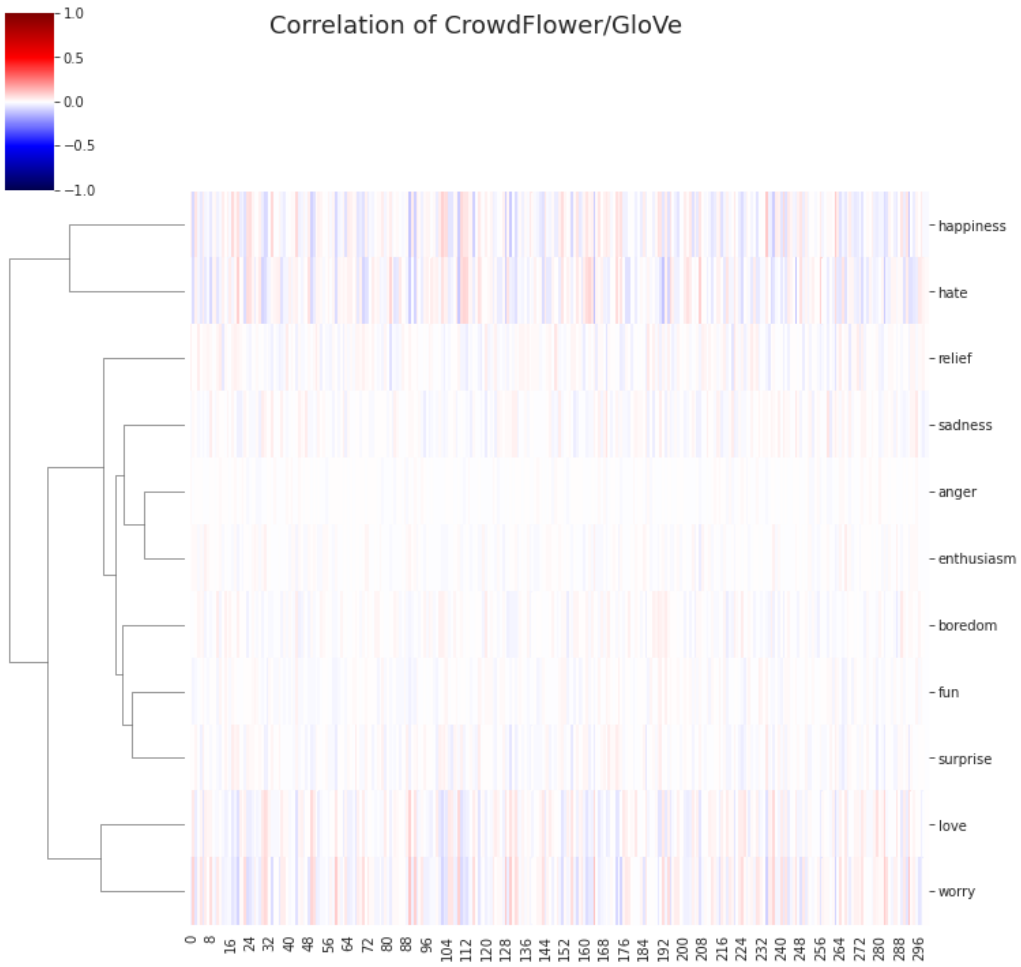

Figure 4.9: Correlation plot for GloVe

The clustering in Figure 4.2 presents a similar structure, with different concepts as the last two models. Hate and happines form a group, while the rest separate into two subgroups: one with love and worry, and the rest of the concepts, with almost no correlation, in a single big cluster.

BERT

BERT is the most powerful language model used within this project. Still, it presented a maximum correlation of only 13.87% between the love concept, and dimension 305. Although slightly better than the previous results, it is still within the treshold of random sampling.

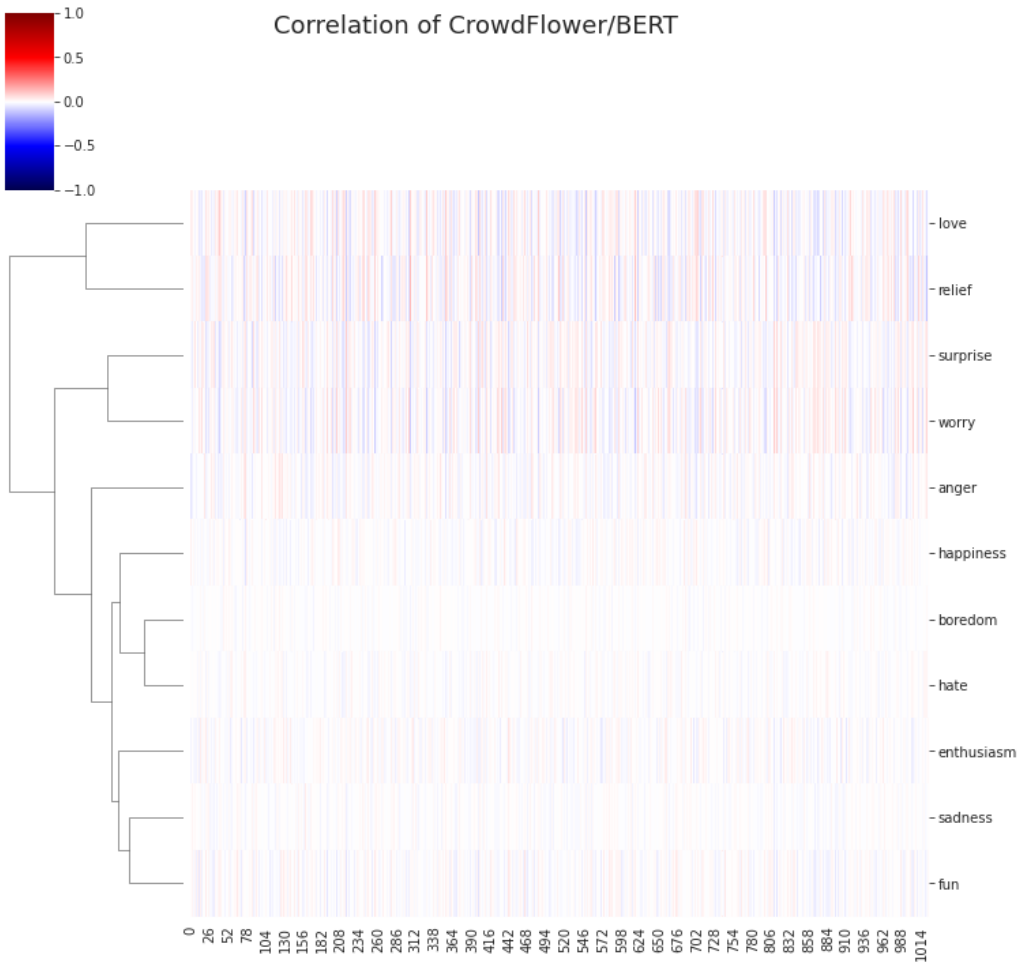

Figure 4.10: Correlation plot for BERT

On Figure 4.2, we can observe that the concepts that most activate the latent dimensions in this model are love, relief, surprise, and worry. Love and Relief cluster into one group, Surprise and Worry in another. Not much structure can be observed, but sadness and fun did cluster into the less activated cluster.

Analysis Discussion

As we can see with these analysis, the linear interpretation of the dimensions of an embedded dataset, through a pre-trained language model does not provide consistent information about the representation of those concepts by the language model.

Linear Transformation Analysis

A linear transformation of the vector space generated by the laguage model can concentrate the information of said model on very little dimensions. This allows for a different analysis of the embedding of the concepts. For this analysis, we have selected the 11 top components of the PCA trasformation. This allows us to see the numeric values in the visualization.

FastText

The baseline FastText analysis shows that the mostly correlated model was the supervised model, shown in Figure 4.2. This shoed a maximum correlation of 36.85% with the concept of hate. This is a slight improvement over the non-transformed analysis.

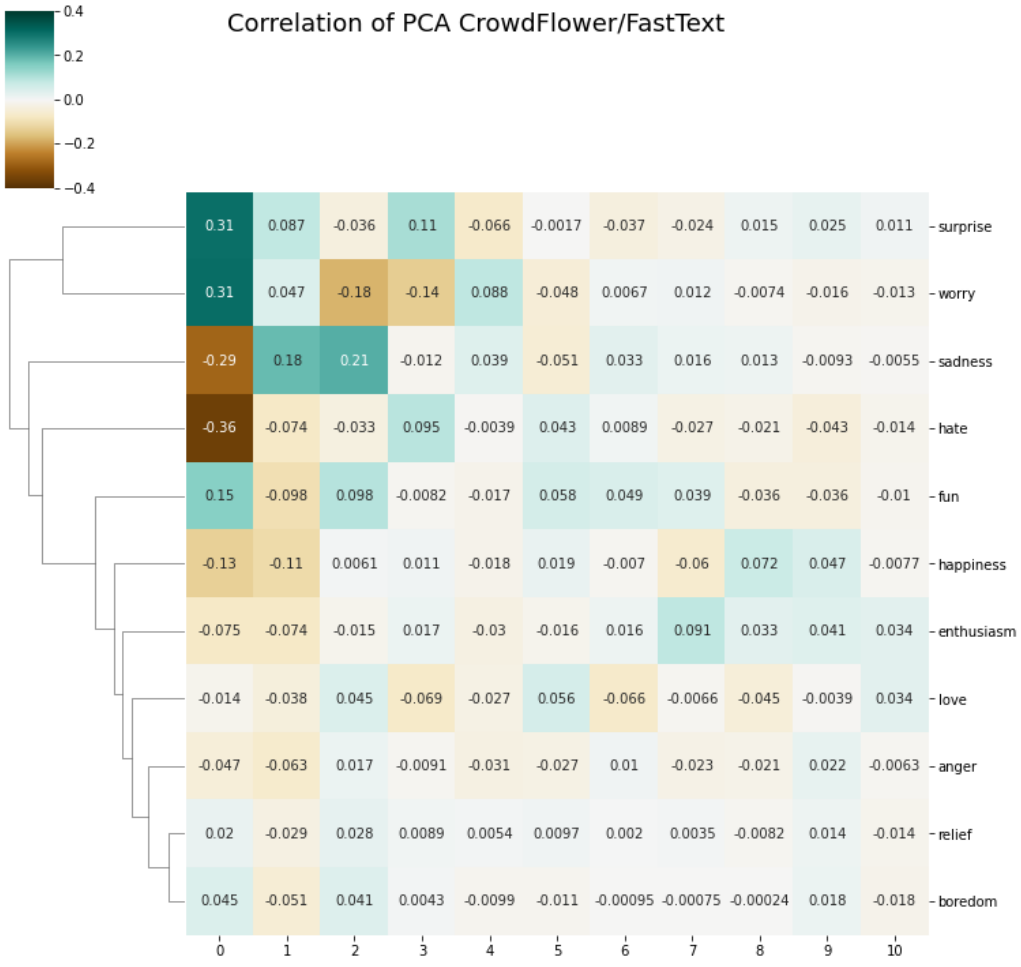

Figure 4.11: PCA Correlation plot for FastText

Figure 4.2 shows that the first component of the transformation has a high positive correlation with the concepts of Surprise and Worry, while keeping a high negative correlation with Sadness and Hate. One grouping between Surprise and Worry is present, with the rest of the concepts in a second cluster.

Word2Vec

The transformation of the Wort2Vec representation presents the worst representation of concepts seen. The maximum correlation present is between the Love concept, and the second component.

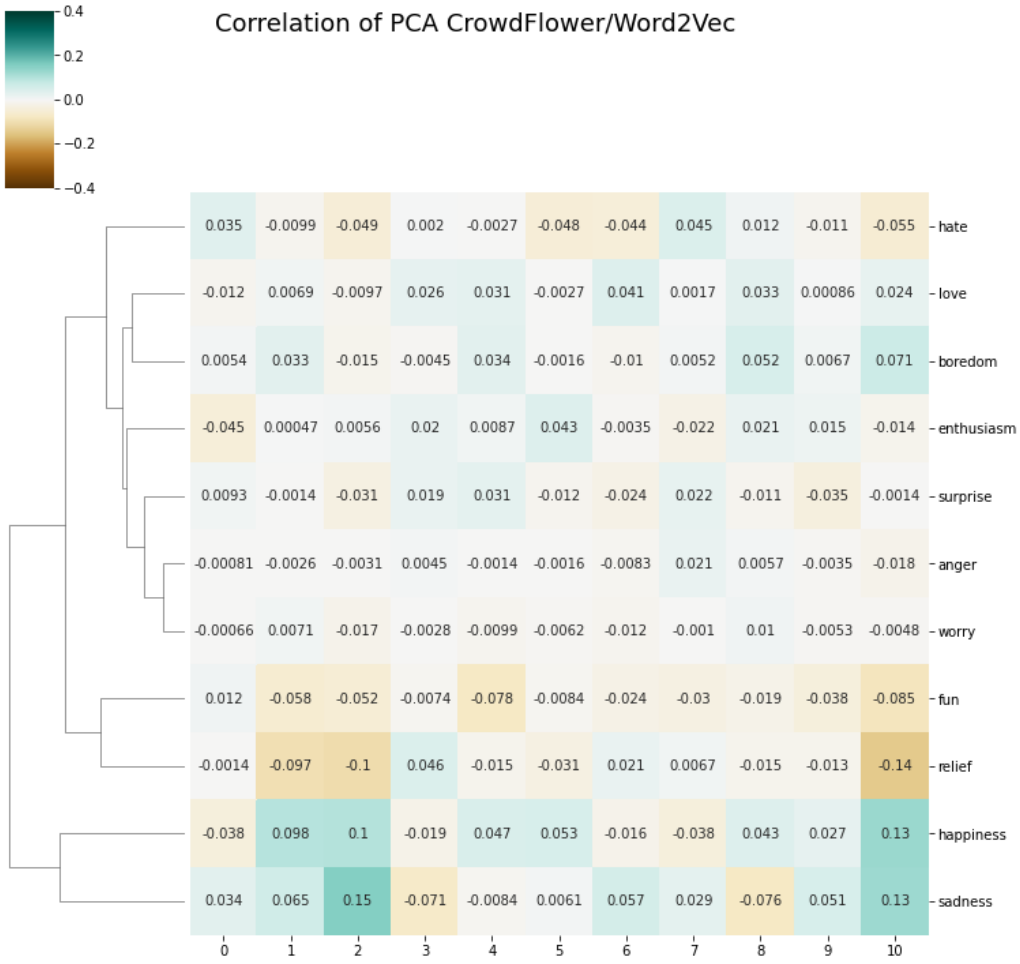

Figure 4.12: PCA Correlation plot for Word2Vec

Figure 4.2 shows the results of this analyzis. Here, with a very low correlation, sadness and hapiness have been clustered together, while relief and fun create the second most distinct group. The dichotomy of sadness and happiness is as expected from the baseline papers, but the correlation is much lower than if only valence is taken into account. Another relevant grouping to mention is that of Anger and Worry. Although the direct correlations between the given concepts and the components of the transformed vector space are not statistically relevant, the clustering that can be done by analyzing these corresponds with that of parts of the Plutchik model, and accounts for dichotomy of emotions in a valence model of affect.

GloVe

The GloVe model shows a very low correlation, even worst that the Word2Vec model, which seems contradictory. The maximum correlation is 9.58% lower than the threshold of random choice.

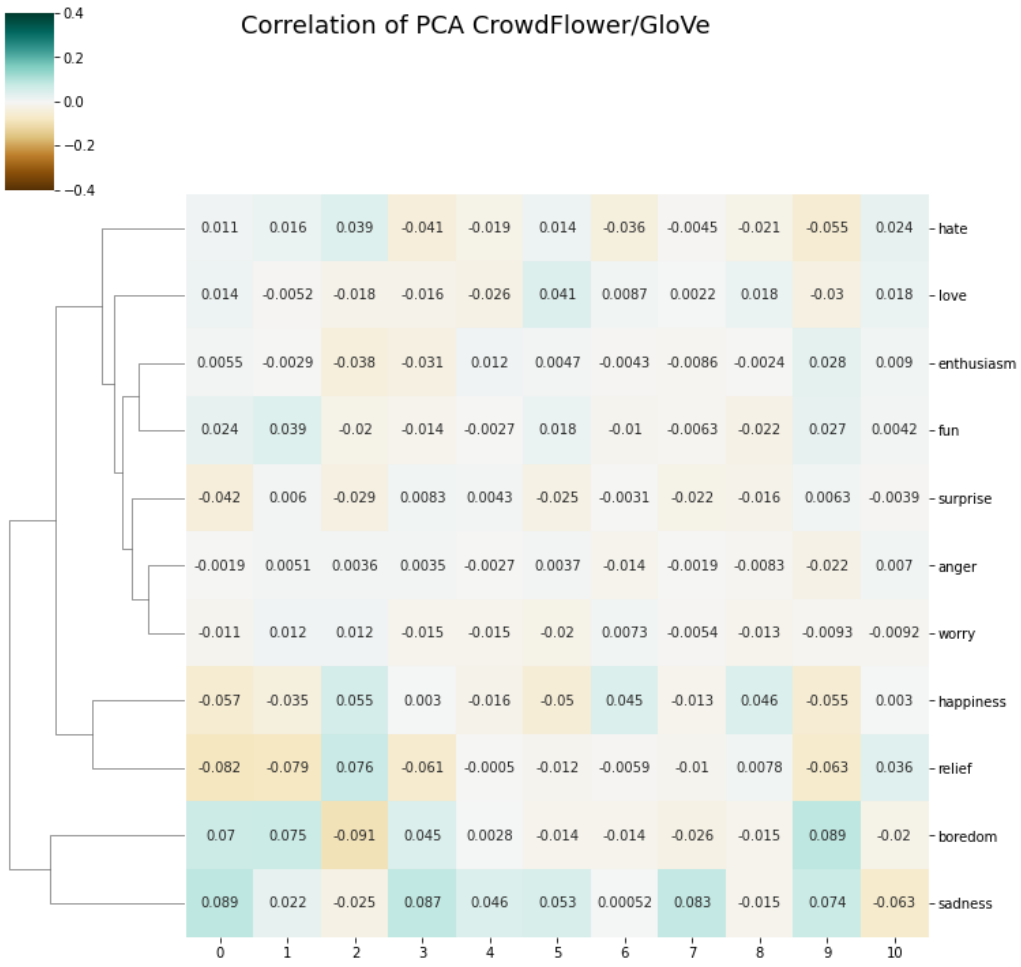

Figure 4.13: PCA Correlation plot for GloVe

Figure 4.2 shows how the low correlation of concepts and components of the PCA-transformed vector space yields no results that relate to emotion models. Even so, a three-group clustering is seen. This might indicate that this clustering is more related to the dataset, than to the language model.

BERT

The BERT model shows a maximum correlation of 10.28% with between the component number 5 and the concept of Love. The correlations are not as high as expected, for the most powerful model in this project.

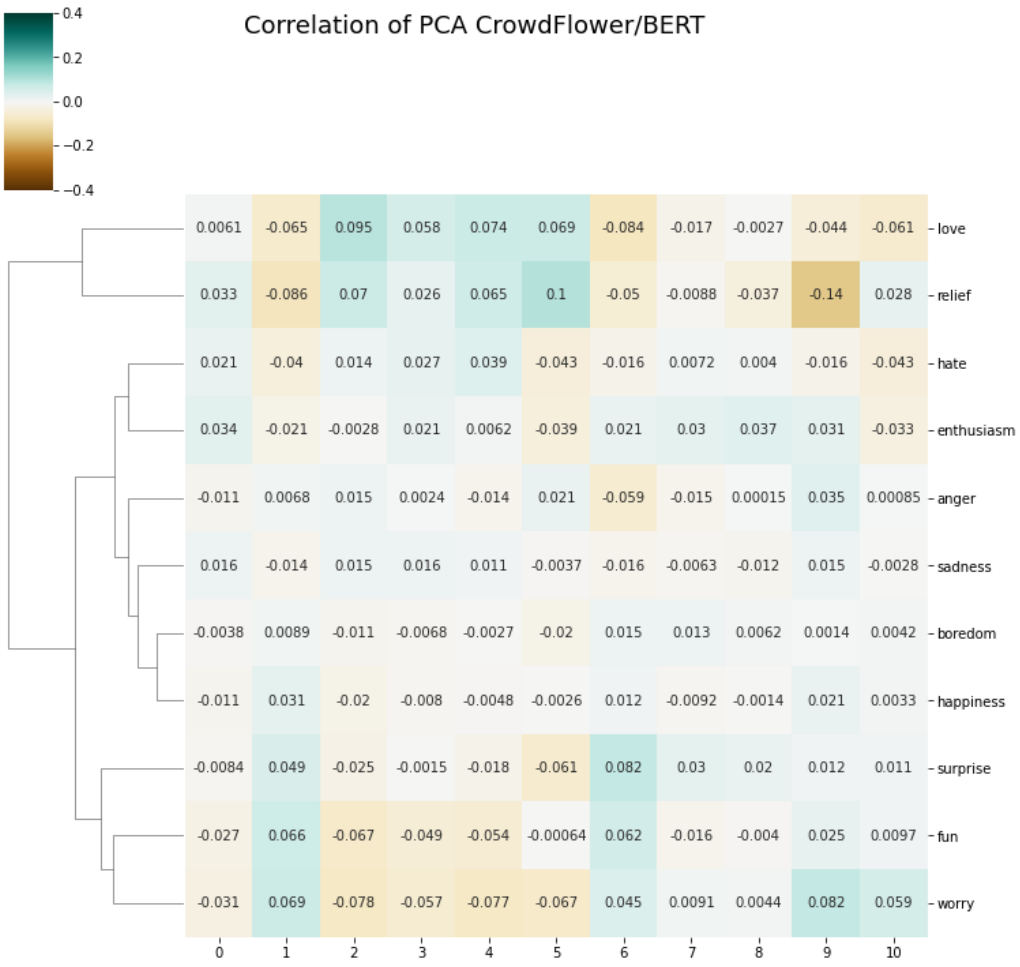

Figure 4.14: PCA Correlation plot for BERT

Figure 4.2 shows how the results don't seem to be as densely populated by high correlations. This could be due to the high dimensionality of the model. With this PCA transformation 1024 dimensions are being reduced down to 10, which in comparison with the other models, is a much bigger reduction. No clustering seems relevant, when compared with emotion models.

Analysis Discussion

As mentioned before, a dimensionality reduction implies that some information will be lost by the model. If the information captured was already low, and the number of dimensions is significantly reduced, the results can end up being worst than random guess.

Non-linear Transformation Analysis

The TSNE non-linear dimensionality reduction allows for two-dimensional scatter plots that maximize the distance between groups in a dataset. A point in every scatter plot represents a sentence in the dataset. The color is the emotion label of that sentence.

FastText

As with the last two studies, the FastText approach is a suppervised language model trained on this specific dataset. For this reason, it's also expected to have the TSNE scatter plot with the most clearly separate groups.

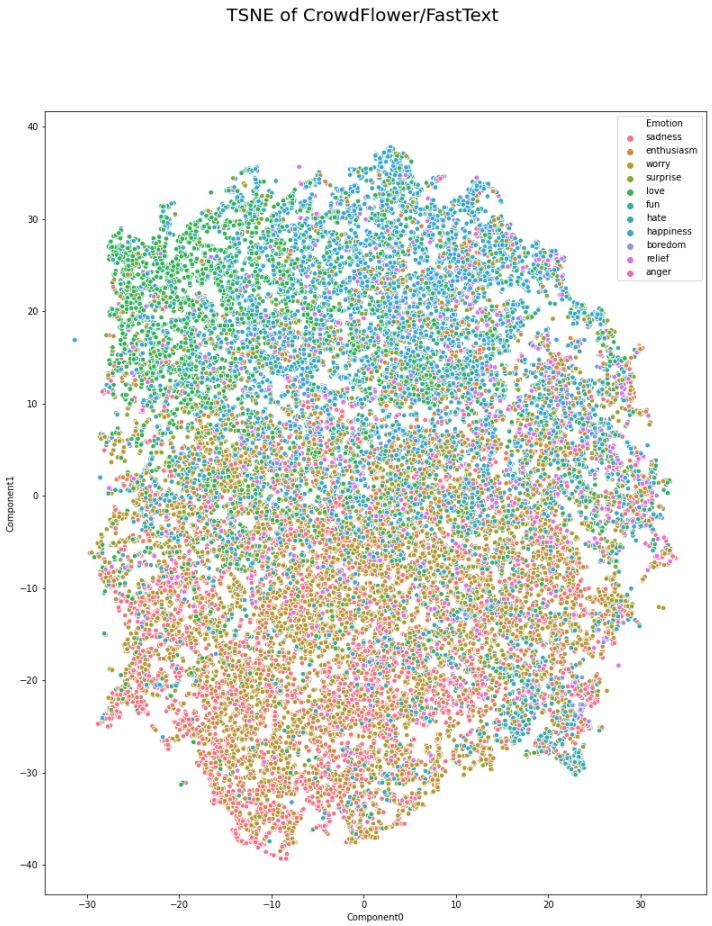

Figure 4.15: Scatter plot for TSNE of FastText

It can be observed that Figure 4.2 shows clear gradients between groups. These are not linearly separable, but do comply with the emotion's valence value. Positive valenced emotions like Love, Fun and Happiness are present in the top part of the visualization (the positive side of component 1), while the negative emotions like Anger, Hate, and Sadness are presented in the lower part of the visualization. There is no clear separation of groups around the origin.

Word2Vec

As expected, the Word2Vec scatterinng does not present such clear groups as the ones shown by the FastText model. There is a main group of datapoints that fall around the origin, and a very slight gradient with positive valence at the bottom of component1, and negative valence at the top.

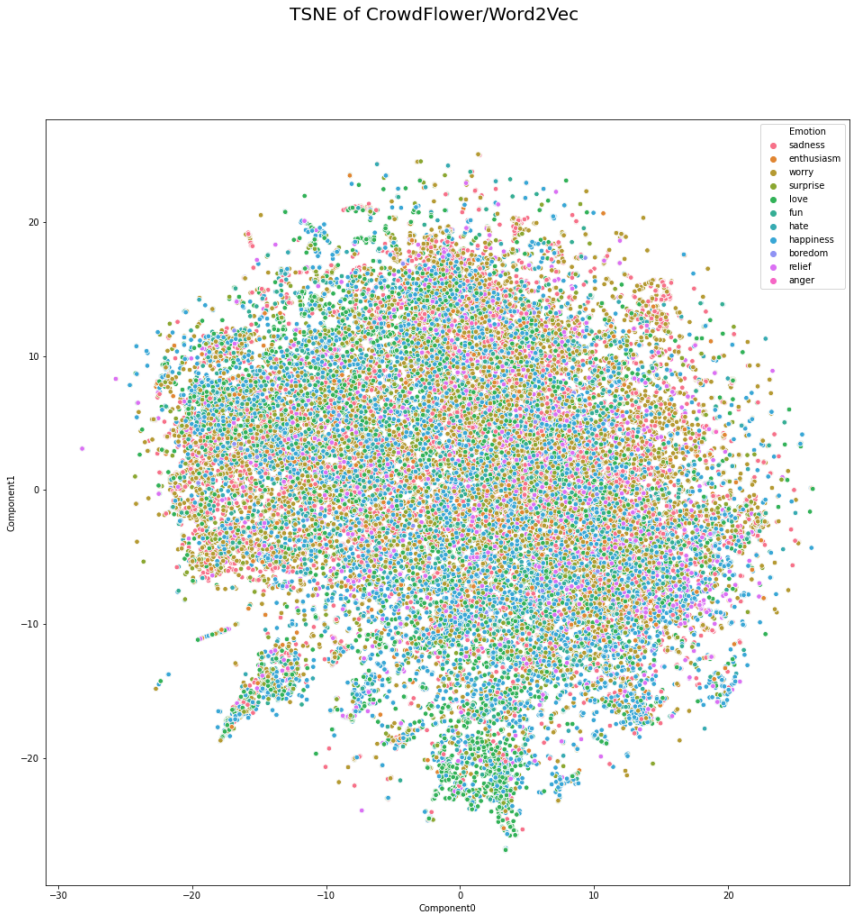

Figure 4.16: Scatter plot for TSNE of Word2Vec

Figure 4.2 shows a phenomenon present in pre-trained models. Some data points are separated from the main cluster, but are maintained relatively close to each other. These are sentences that are similiar in meaning, sometimes even identical sentences, but contain a distict emotional meaning.

GloVe

For the GloVe model, the gradient of positive and negative valenced emotions is not as clear as with the Word2Vec model.

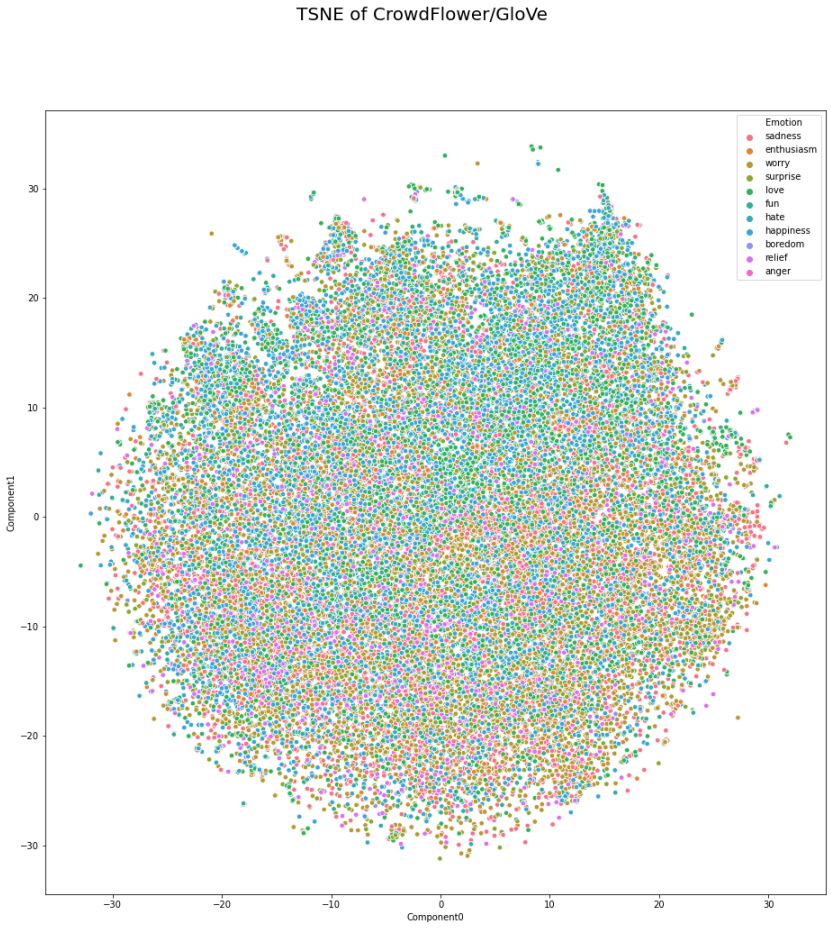

Figure 4.17: Scatter plot for TSNE of GloVe

On Figure 4.2 the dataset is represented mostly as a gaussian distribution of scattered datapoints. If there are relevant features of this representation, they are on the top of the visualization, where most of the positively valenced emotions are. There, the semantic clusters, seem to be more common than anywhere else in the plot.

BERT

With this analysis, BERT comes out as the model that creates representations that are linearly separable, even after using average representation of sentences.

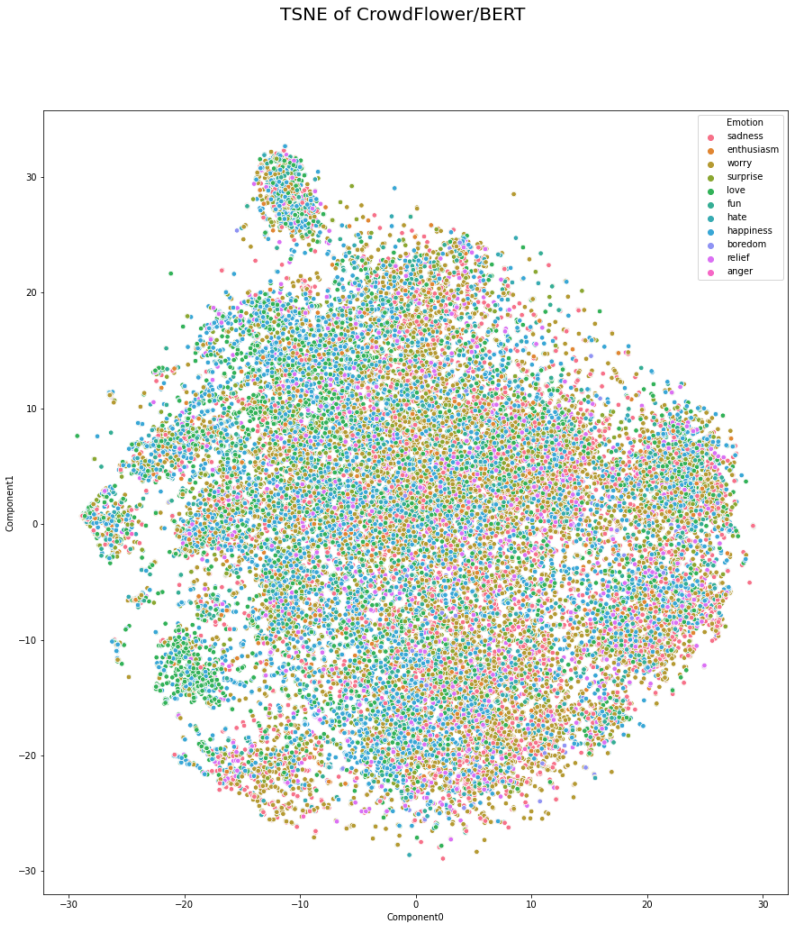

Figure 4.18: Scatter plot for TSNE of BERT

In Figure 4.2 several clusters can be observed, with a very characteristiic one composed of mostly Love, Fun, and Happiness labels around (-17, -12). This cluster can be considered as a positive valence cluster. Although there are several groupings of sentences, there seems to be no clear gradient of valence in the model, but valence might be a variable to be measured inside clusters of datapoints.

Analysis Discussion

All models presented in this project are created through non-linear methods. The pre-trained models are also optimized to be used with neural networks.

neutral	16894
worry	16840
happiness	10336
sadness	10284
love	7610
surprise	4360
fun	3532
relief	3042
hate	2640
empty	1606
enthusiasm	1510
boredom	358
anger	212

Table 4.1: Class distribution for CrowdFlower dataset.

For this reason, a non linear approach is expected to yield the best result for a classifier.

Result Analysis

The models presented here seem to have very little representation of emotions, but a strong relationship with valence. Although this could indicate that the models for human language do not conceptualize emotions, some cases have been found, that might indicate that it is in fact, the dataset that makes it difficult to cluster the conceptualization of certain emotions.

Class unbalance

The class distribution of the CrowdFlower dataset is severely unbalanced. Table 4.1 shows the classes, including neutral, which has been removed from this analysis.

For this reason, the majority of the datapoints visualized in this project are Worry, Happiness, and Sadness.

Semantic Clusters

In the TSNE visualizations it's hard to pin down which datapoints represent what. For this reason, interactive visualizations have been made to explore freely the 2D space generated by this trasformation. This visualizations can

be found in the project repository. By examining these visualizations several observations can be made.

The first one is that the models abstract semantics pretty well, as expected. Creating semantic clusters, or 'Meaning Islands' for sentences that express the same idea, even if it's with different words. That's the case of the Star Wars island. This is a cluster in the Word2Vec TSNE transformation that clusters tweets sent on May 4th. (**Note**: the model did not have access to the date of the tweet, May the 4th is considered Star Wars day by the franchise's Fanbase) This cluster has a center around (26, -5). Tweets from this cluster include text like 'Happy Star Wars Day! May the 4th be with you', 'Happy Star Wars day!', or simply 'May the 4th be with you!'. Other tweets nearby include mentions of Bank's day, National days, or Fight Club's 10th aniversary.

Another interesting feature of this model is the 'ALL-CAPS' peninsula on Figure 4.2, on the other side of the plot, at around (-25, 11). This feature's most compact cluster are tweets of people whishing a happy mother's day, but also features some tweets cursing (with and without mentioning of mothers in the cursing), whishing happy birth day, and one about Star Wars day. All these tweets have in common that at least some part of it is written in all capital letters.

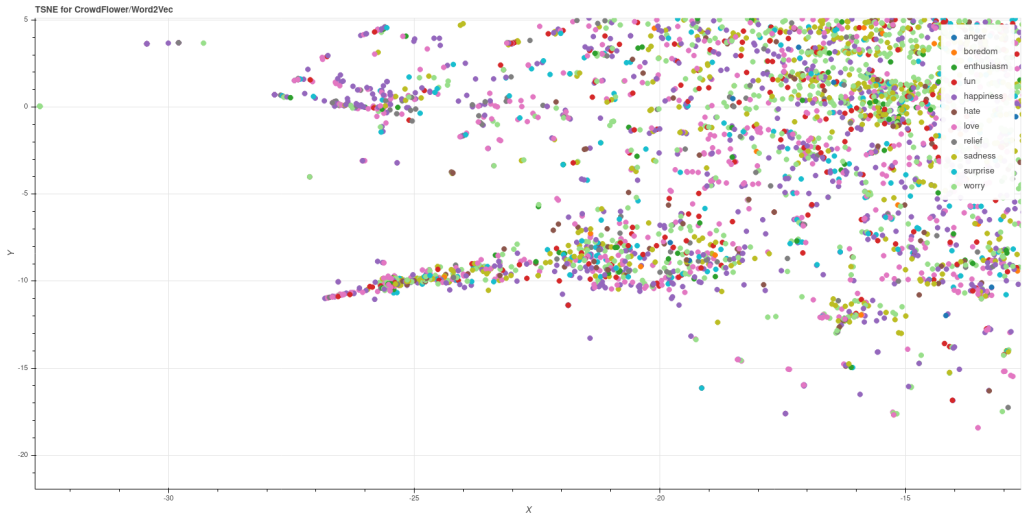

Figure 4.19: A zoom into the All-Caps peninsula for the TSNE transformation of the CrowdFlower DS under the Word2Vec LM

Other interesting cluster is around (24, -10) in the Word2Vec projection, where most of the tags contain the word Happy, but are tagged with the emotion love.

In the GloVe projection, most tweets whishing for a happy mother's day seem to be disperese on the top region of the plot, but do not generally cluster.

A consecuent observation of these clusters is that very similar tweets have different emotional labels. This does not necesarily mean that the labels are wrong. But it does speak about the meaning that that tweet had for the person that asigned the label. Emotions are acording to theory, context dependant. Tweets for Mother's day are labeled with the emotions Love, Happiness, Fun, and even Wory.

A third observation comes from examining the dataset's text, where some of them are not in english. Some others are just links. In general there is an overload of people whishinig a happy mother's day and a happy Star Wars day. Both of these dates happen in May.

FastText Approach

The FastText Language Model, trained with this dataset, seems to have taken the three most frequent labels, and separated them to two different extremes of the vector space. This is exactly what a spervised model should do, but by doing so it sacrifices the clustering between sentences that might be related, but writen in a different manner.

Chapter 5

Conclusion

The experiments presented in this project help understand some of the reasons as of why the field of emotion research is so hard to manage in precise and conicse manners. By asuming there is a model of universal emotions, to be able to generalize to all languages and populations, one can neglect the specificity of very relevant context-dependant emotions. On the other side, by creating models that adapt to populations and languages, the generalization of those specific concepts is lost.

5.1 Discussion

Language models seem not to have a strong representation of the concepts of emotions as labeled in datasets, but datasets with labeled emotions have serious problem when it comes to emotions that present more than one of the emotions from the selected emotion model. In many cases, dataset labeling selects arbitrary emotion models.

Valence, on the other hand, is present on every dataset and languge model. Since every emotion of universal models of emotions fits into a dichotomical model of valence, a top to bottom hierarchical clustering can be done with the concepts that are represented in a language model, to aproach the labeling of a dataset, and even to create a model of emotion in text.

Even the most simple of the examined language models abstracts very complex concepts, and is able to discern between very fine differences. In the case of BERT, those differences can even be on the context of the text.

On the CrowdFlower dataset

The CrowdFlower dataset has been used on many baseline papers cited on this project, and as a main dataset for this project. Unfortunately, I noticed too late, that there is no actual paper describing its creation. The company that created it doesn't exist anymore, and the labeling seems to have partially been done automatically. Many labels seem incorrect, or were doubtably labeled out of context. This is a major problem to consider in the present project, and must be resolved in future iterations.

Correctly identifying and representing emotions in text

Considering the observations of the experiments in this project, suggestions as of how to create and use a model of language, or a dataset that correctly captures emotional concepts can be made.

The first comments are on the dataset. A dataset with self-contained texts is suggested. Since the message of the text must be as context independant as possible. This does not mean that a discussion, like the case of Emotion-Push or Friends cannot be analyzed, but if analyzed, it must be done as a single data point, maybe with variable emotion along the discussion. Twitter is not a bad example of a corpus for emotion detection, but retweets and responses must be excluded, and a sample across time is suggested, to avoid the influence of events.

The text of the dataset should be a single language. Different languages will require different models of emotions. Labeling, if done by humans, should be voted on, or, if multiple labels are posible, they should be counted and voted on by intensity. In this case the labels are only to be considered the "Perceived" emotion. Which labels the emotion of the reader, and no the emotion of the original author of the text.

For a dataset trying to capture the emotion of the person writing the text, the best approach is by self-reported emotions. This means that only texts that explicitly express the experience of an emotion by the writer can be taken into account. This can be combined with a text-similarity expansion of the dataset. An example of this is selecting tweets that contain the frase: 'I feel...' followed by an emotion word, or descriptor. This specific example is similar to the way Language Models learn from context. Anything that follows that sentence, can be interpreted as a self report of an internal condition. A further analysis on this example could yield interesting results, and is considered as future work.

If a model of emotions is used to select the labels for a dataset, the consideration of what an emotion, along with the interpretation and clear

differences of those emotions in the specific language of the dataset must be made. A dataset that does not explicitly express the reasons to use an emotion model should not be used. Eckman's model, for example, although universal, is based on faces. If the dataset is not of faces, Eckman's model is discouraged in favor of a more complete model of emotion in laguage.

On average representation of sentece embedding

For this project, a sentence embedding was simply the average of it's token's embeddings. Although simplistic, this way of representing sentences is not optimal, since the sequential nature of the tokens is lost. The problem with this is that by including the secuential nature of the sentence, like when using a RNN-based model to classify, modifies in a non-linear way the vector space, and requires much more data.

Non-separable emotions

As seen in this project, and presumably, also known by the reader, many emotions can be experienced at the same time. The lines between what can clearly be different emotions can get blurry as their intensity increases or the context changes. A better way of representing such abstract and complex concepts could be through statistical models. Gaussian models for example, could easily describe the distribution of emotions in the vector space. This is considered a future approach to this problem.

Emotion Words

The word with which one describes an emotion, the emotion name, is one of the most powerful cognitive tool that a person has to understand, manage, and express their emotions. By concentrating on emotion names, machine learning models could improve their results. Contrary to this, the results of this project show that many datasets label texts including the word 'Happy' as other emotion. When using a dataset with labeled emotions, a cross analysis of the labels and their text is suggested, to avoid or at least consider these contradictions.

5.2 Future Work

This project's scope is necesarily narrow. Human language is a complex phenomena, and so are human emotions. There is much to be learned. Here,

some suggestions as for the next steps to consider are made.

Multi-label datasets

Several datasets with labeled emotions were considered and adquired for this project. Their analysis is one of the next steps to be taken. For this, datasets with multi-labeled texts are to be considered. These are specifically difficult to visualize, but could be handeled with models of emotions like the Plutchik model, which describe emotions as superpositional.

Learning an Emotion Model

Language models have proven to be very powerful. They abstract concepts very well, and represent those concepts with efficiency. Given the results of this thesis, I would like to create a model of emotions in text based on Language models. This requires much more text, and probably an approach closer to the self-reported datasets, but this might even be the next step in the study of human emotions. Machine Learning is a tool that lets us examine and observe human phenomena in a way that we have not ever been able to, by examining human generated data.

Thanks to the work done in this project a new method for the creation of an emotional model can be created. A prospect model of emotions in text can be created by using explicit textual exrpessions of emotion. An example of an explicit expression of emotion is: 'I feel sad' By removing the emotional word from this sentence, we can obtain an emotion-neutral emotion context. By parsing a corpus for these specific sentences, maping them on to a pre-trained-model vector space, and reproducing a separation method like PCA, the transformation that better captures the different emotional contexts can be obtained. This transformation can latter be used for non-explicit expressions of emotion. This model of emotions can be context and languge specific, but the methodology is not restricted to any language, or even expressions of emotion. It is in general a method for analyzing conceptualizations of pre-trained language models.

5.3 Conclusion

In hope that this project serves as a pedagogic experience for students working on NLP concept learning, and the visualization of concepts in machine learning models of language, the code, documentation, development environment, development notes, and git repository are available under the given

link in the section on Organizational 3.1. This has been done prurposely. Visualizations are available as well as interactive explorations of the datasets, in the abstract spaces for the representation of the datasets through the language models, as well as the code to reproduce them with other datasets. We hope this work is of help to anyone trying to understand concept learning in ML models of language.

Chapter 6

Appendix

6.1 CUDA

The version of CUDA library used is release: 10.2, V10.2.89

Ablative study: Why there isn't a one to one correlation between embedded dimensions and labels? It's exactly what happens with feature maps. Why not with SLP? Use fasttext to explain the difference

6.2 Emotion Datasets

This table is part of the 2018 study from Klinger et Al. [Klinger et al., 2018]. It presents the datasets they used, and have been considered for this study.

6.3 Python Virtual Environment

The python virtual enviroment was created with VirtualEnv, and VirtualEnvWrapper. The requirements.txt file contains the used libraries to recreate this study. These are:

- torch
- torchvision
- jupyter
- numpy
- matplotlib

Dataset	Author	Year	License
affectivetext	Strapparava & Mihalcea	2007	
crowdflower_data	CrowdFlower	2016	available to
dailydialog	Li Yanrand et al	2017	available to
emotion-cause	Diman Ghazi&Diana Inkpen&Stan Szpakowicz	2015	research onl
EmoBank	Sven Buechel	2017	redistributa
emotiondata-aman	Saima Aman&Stan Szpakowicz	2007	obtainable u
fb-valence-arousal-annon	Preotiuc Pietro	2016	available to
grounded_emotions	Liu, V.&Banea, C.&Mihalcea	2017	available to
isear	Klaus R. Scherer and Harald Wallbott	1990	available to
tales-emotions	Cecilia Ovesdotter Alm	2005	gplv3
emoint			
electoraltweets			

- scikit-learn
- fasttext
- seaborn
- gensim
- spacy
- MulticoreTSNE
- bokeh
- transformers
- fastcluster

Bibliography

[Abadi et al., 2015] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

[Alm et al., 2005] Alm, C. O., Roth, D., and Sproat, R. (2005). Emotions from text: machine learning for text-based emotion prediction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 579–586. Association for Computational Linguistics.

[Aman, 2007] Aman, S. (2007). Recognizing emotions in text. In *Masters Abstracts International*, volume 46. Citeseer.

[Buechel and Hahn, 2017] Buechel, S. and Hahn, U. (2017). Emobank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 578–585.

[Chacón, 2016] Chacón, A. B. (2016). Sonification of emotional phenomena in a social network. Bachelor Thesis. This is my Bachelor thesis.

[Chen et al., 2018] Chen, S.-Y., Hsu, C.-C., Kuo, C.-C., Ku, L.-W., et al. (2018). Emotionlines: An emotion corpus of multi-party conversations. *arXiv preprint arXiv:1802.08379*.

[Corson, 1996] Corson, D. (1996). *Using English words*. Springer Science & Business Media.

[Darwin and Prodger, 1872] Darwin, C. and Prodger, P. (1872). *The expression of the emotions in man and animals*. Oxford University Press, USA.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. arXiv: 1810.04805.

[Ekman, 1992] Ekman, P. (1992). Are there basic emotions?

[Ekman, 1999] Ekman, P. (1999). Facial expressions. *Handbook of cognition and emotion*, 16(301):e320.

[Ekman and Keltner, 1997] Ekman, P. and Keltner, D. (1997). Universal facial expressions of emotion. *Segerstrale U, P. Molnar P, eds. Nonverbal communication: Where nature meets culture*, pages 27–46.

[Feldman Barrett and Russell, 2014] Feldman Barrett, L. and Russell, J. A. (2014). *The psychological construction of emotion*. Guilford Publications.

[Ghazi et al., 2015] Ghazi, D., Inkpen, D., and Szpakowicz, S. (2015). Detecting emotion stimuli in emotion-bearing sentences. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 152–165. Springer.

[Hollis and Westbury, 2016] Hollis, G. and Westbury, C. (2016). The principals of meaning: Extracting semantic dimensions from co-occurrence models of semantics. *Psychonomic Bulletin & Review*, 23(6):1744–1756.

[Honnibal and Montani, 2017] Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

[Huang and Ku, 2018] Huang, C.-Y. and Ku, L.-W. (2018). Emotionpush: Emotion and response time prediction towards human-like chatbots. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 206–212. IEEE.

[Irwin, 1947] Irwin, J. R. (1947). Galen on the temperaments. *The Journal of general psychology*, 36(1):45–64.

[Joulin et al., 2017] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In *Proceedings of the*

15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers, pages 427–431. Association for Computational Linguistics.

[Klinger et al., 2018] Klinger, R. et al. (2018). An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2104–2119.

[Kluyver et al., 2016] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., and Willing, C. (2016). Jupyter notebooks – a publishing format for reproducible computational workflows. In Loizides, F. and Schmidt, B., editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press.

[Lai, 2019] Lai, G. (2019). bert-embedding: Token level embeddings from bert model on mxnet and gluonnlp.

[Li et al., 2017] Li, Y., Su, H., Shen, X., Li, W., Cao, Z., and Niu, S. (2017). Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

[Liu et al., 2017] Liu, V., Banea, C., and Mihalcea, R. (2017). Grounded emotions. In *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 477–483. IEEE.

[Maas et al., 2011] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. page 9.

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. page 9.

[Mohammad et al., 2014] Mohammad, S., Zhu, X., and Martin, J. (2014). Semantic role labeling of emotions in tweets. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 32–41.

[Mohammad, 2012] Mohammad, S. M. (2012). # emotional tweets. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on*

Semantic Evaluation, pages 246–255. Association for Computational Linguistics.

[Mohammad and Bravo-Marquez, 2017] Mohammad, S. M. and Bravo-Marquez, F. (2017). Emotion intensities in tweets. In *Proceedings of the sixth joint conference on lexical and computational semantics (*Sem)*, Vancouver, Canada.

[Mohammad et al., 2018] Mohammad, S. M., Bravo-Marquez, F., Salameh, M., and Kiritchenko, S. (2018). Semeval-2018 Task 1: Affect in tweets. In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

[Mohammad and Turney, 2013] Mohammad, S. M. and Turney, P. D. (2013). Crowdsourcing a Word-Emotion Association Lexicon. *arXiv:1308.6297 [cs]*. arXiv: 1308.6297.

[Oliphant, 2006] Oliphant, T. E. (2006). *NumPy: A guide to NumPy*, volume 1. Trelgol Publishing USA.

[pandas development team, 2020] pandas development team, T. (2020). pandas-dev/pandas: Pandas.

[Paszke et al., 2019] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., dAlché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

[Picard, 2000] Picard, R. W. (2000). *Affective computing*. MIT press.

[Plutchik and Kellerman, 2013] Plutchik, R. and Kellerman, H. (2013). *The measurement of emotions*, volume 4. Academic Press.

[Preoţiuc-Pietro et al., 2016] Preoţiuc-Pietro, D., Schwartz, H. A., Park, G., Eichstaedt, J., Kern, M., Ungar, L., and Shulman, E. (2016). Modelling valence and arousal in facebook posts. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*, pages 9–15.

[Rothe et al., 2016] Rothe, S., Ebert, S., and Schütze, H. (2016). Ultradense Word Embeddings by Orthogonal Transformation. *arXiv:1602.07572 [cs]*. arXiv: 1602.07572.

[Scherer and Wallbott, 1990] Scherer, K. and Wallbott, H. (1990). International survey on emotion antecedents and reactions (isear).

[Schuff et al., 2017] Schuff, H., Barnes, J., Mohme, J., Padó, S., and Klinger, R. (2017). Annotation, modelling and analysis of fine-grained emotions on a stance and sentiment detection corpus. In *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 13–23.

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

[Strapparava et al., 2004] Strapparava, C., Valitutti, A., et al. (2004). Wordnet affect: an affective extension of wordnet. In *Lrec*, volume 4, page 40. Citeseer.

[Ulyanov, 2016] Ulyanov, D. (2016). Multicore-tsne. `https://github.com/DmitryUlyanov/Multicore-TSNE`.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

[Virtanen et al., 2020] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Jarrod Millman, K., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C., Polat, İ., Feng, Y., Moore, E. W., Vand erPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and Contributors, S. . . (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

[Vo and Zhang, 2016] Vo, D. T. and Zhang, Y. (2016). Don't Count, Predict! An Automatic Approach to Learning Sentiment Lexicons for Short Text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 219–224, Berlin, Germany. Association for Computational Linguistics.

[Wolf et al., 2019] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv, abs/1910.03771*.