A Comparison of Sentiment Analysis Deep Learning Methods and How They Can Be Implemented into a Web Application

UNIVERSITY OF LINCOLN

Isaac Michael Scarrott

SCA16607716

School of Computer Science
Final Year Project

Supervised by

Dr. Elisa Rubegni

Acknowledgements

I would like to thank the following people:

- First of all I would like to thank my family, particularly my parents for their unimaginable support throughout my three years at university. Without you this would not be possible

- Secondly I would like to thank my close friends. Particularly James Bowles, Jamie Turner, Michael Pywell and Joe Duguid. You truly have made my university experience incredible and I would not change it.

- And finally I would like to thank my supervisor Dr. Elisa Rubegni for her guidance and advice in my project.

A Comparison of Sentiment Analysis Deep Learning Methods And How They Can Be Implemented Into A Web Application

Isaac Scarrott

Final Year Bachelors of Science Project
11th April 2019

Abstract

This paper optimizes the word2vec model's hyperparamters when trained on sentiment140 dataset comprising of 1.6 million tweets. Different methods of creating tweet vectors are compared as well as these vectors feeding into different deep learning models, where the results are compared and the best model is then implemented into a web application where users can see the latest trends and get the overall sentiment of the trends. Users can also input their own keyword to get their sentiment from. Overall, the multi layer perceptron proved to gain the best accuracy.

Contents

List of Figures

List of Abbreviations

CBOW Continuous Bag of Words

CNN Convolutional Neural Network

DNN Deep Neural Network

DNN Dense Neural Network

FFNN Feed Forward Neural Network

MLP Multi Layer Perceptron

NLP Natural Language Processing

NN Neural Network

RNN Recurrent Neural Network

SG Skip Gram

SLP Single Layer Perceptron

SVM Support Vector Machine

Chapter 1

Introduction

1.1 Context

Sentiment analysis is associated with other names such as: opinion mining, opinion extraction, subjectivity analysis and emotion extraction. However they all describe the same field of study, which is analyzing peoples opinions, sentiments, evaluation, attitudes and emotions towards entities (Liu, 2012). The idea of sentiment analysis was first put forward in the paper Thumbs up? Sentiment Classification Using Machine Learning Techniques by Bo Pang , Lillian Lee and Shivakumar Vaithyanathan. They set out to improve document classification in the early 2000s by classifying documents by overall sentiment as opposed to topic. They achieved their best accuracy using a combination of support vector machine and uni-grams to achieve a relative accuracy of 82.9% (Pang, Lee and Vaithyanathan, 2002).

More recently you can get consumer versions from companies like Talkwalker who boast an accuracy of 90% using deep learning models and training sets with a size of over ten million (Opitz, 2017). However, Opitz is an employee of Talkwalker so may be bias. Furthermore, she is not in a role related to data science (Content Marketing Lead). Also the description of Talkwalkers model is not provided in any significant detail in the blog post. However, if other academic research is evaluated, it shows some state of the art models are actually achieving an accuracy of around 90%. One example being in the paper: Aspect-based Sentiment Analysis Using Deep Networks and Stochastic Optimization, which boasts 88.52%, 94.30%, 85.63%

and 86.03% in accuracy, precision, recall and F-measure, respectively. This was achieved using Word2vec for conversion of processed corpus and a convolutional neural network for opinion mining (Kumar, Pannuand and Malhi, 2019).

1.2 Project Aims

The aim in the project proposal was to "create a web application that will carry out data analysis, with a focus on sentiment analysis, on at least one social media platform using a current trend or a user inputted keyword and display data that has been curated from the unfiltered data gathered using the trend or keyword". However this has developed into "to create a web application that will carry out data analysis, with a focus on sentiment analysis, on at least one social media platform using a current trend or a user inputted keyword and display semantic data that has been curated to create model that obtains the best accuracy". So the focus of the project has shifted slightly away from the web application and towards finding the best deep learning method to implement into a web application. This means that the quality of the web application may decrease due to having to develop and test many different NN.

1.3 Project Objectives

The objectives in the project proposal were as follows:

- Create an aesthetically pleasing and functional website using HTML5, CSS3 and JavaScript that is able to communicate with the back-end. This will be done using an MVC architecture.
- Store all user names and corresponding hashed passwords in a database. The user must also be able to log in and out at any time.
- Display what is currently trending on Twitter, on the website (the user must be able to change the region of the trends that are being displayed).

- Produce a data analysis tool, where users will be able to analyze trends or a user inputted keyword/hashtag using a search bar.
- Produce a sentiment analysis tool that, when used, not only has a numerical score on the attitude towards the selected data, but also shows graphically the attitude towards the selected data.
- Produce a data analysis tool that tracks the change in attitude towards the topic over a period of time and display this graphically when a trend has been searched or a keyword has been entered in.
- Display multiple statistics such as: the number of tweets, demographics, first tweet and the current top tweet to the user.
- Display insightful information into the topic such as background information on the topic from a website such as Wikipedia.
- Summarize the sentiment in a paragraph in a way similar to the TLDR bot on reddit (http://autotldr.io/), but instead of using one item of text it will take a large amount of different texts and condense it down into one paragraph.

However as the nature of the problem becomes more apparent it is clear that more focused aims are needed, so the new aims are as follows:

- Achieve an accuracy rating of above 80% in sentiment classification using a DNN.
- Create an aesthetically pleasing and functional website using HTML5, CSS3 and JavaScript that is able to communicate with the backend. This will be done using the Flask web framework.
- Produce a data analysis web app, where users will be able to analyze trends or a user inputted keyword/hashtag using a search bar.
- Display what is currently trending on Twitter on the website.

- Produce a data analysis tool that tracks the change in attitude towards the topic over a period of time and display this graphically when a trend has been searched or a keyword has been entered in.
- Display insightful information into the topic such as background information on the topic from a website such as Wikipedia.

Some of the objectives were removed due to them being almost impossible or not adding significantly to the artifact. An example being, the last aim to "Summarize the sentiment in a paragraph in a way similar to the TLDR bot on reddit" which is literally impossible to take a number of tweets based around a keyword with varying topics. So attainable targets have to be set.

Chapter 2

Literature Review

2.1 Sentiment In The Context of Business

Recent research suggests that consumer sentiment of a business affects stock prices, consumption of a product or service and how strong the correlations between sentiment and these variables are. For example, a study done by Souza et al. (2015) concludes that there was a strong correlation between the sentiment of tweets towards a brand and stock returns. Furthermore, the sentiment on Twitter towards a brand presented a relatively stronger relationship with the stock returns compared to traditional news wires. This is also endorsed in a study by Singal (2012), who states a 1% increase in consumer sentiment is correlated with an annualized stock return of 6%. Controlling for overall market movements, a 1% change in consumer sentiment is still related to an annualized stock return of 2%. Thus, information about small changes in consumer sentiment can inform us about stock returns, an important metric of firm performance. Not only can sentiment give us an idea about how our company is currently viewed, it can also make predictions about stock prices in the future. Signal (2012) also adds that consumer sentiment can explain variations in personal consumption expenditure in the hospitality industry in particular.

However, Otoo, of the Board of Governors of the Federal Reserve System, states that an increase in stock prices boosts sentiment (Otoo, 1999), which contrasts the previous studies which state that sentiment boosts stock price. However, Otoo doesnt come to any conclusion about the question Does an increase in stock prices aggregate sentiment because people are wealthier or because they use movements in stock prices as an indicator of future economic activity and potential labor income growth? which suggests that Otoo has just found a correlation between two variables and hasnt come to a definitive conclusion. Even though the studies found contrast, it shows that there is definitely a correlation between stock price/consumer consumption and the sentiment of the consumer/shareholder towards a business. Consequently, it is very important that a business understands its current sentiment towards it in order to try and bring about continuous improvement.

Moreover, sentiment analysis is not only useful in a business context, it is also useful in other fields such as in health and, in particular, mental health. This was proposed in papers like What we really want to find by Sentiment Analysis: The Relationship between Computational Models and Psychological State where Jo et al. (2017), enlightening the relationship between computational models and psychological measurements, states recent sentiment analysis models should be able to explain not only whether the data is negative or positive but also whether the person is negative or positive and then goes on to say If the model is trained well, it can be used to diagnose an emotional disorder. One of example could be in a school. The model could detect potential disorder by analyzing students essays. Furthermore this was found to be successful in a model created by Wang et al. (2013) where they successfully created a model that had an 80% accuracy rating in detecting a depressed author using a set of ten features ranging from the use of first person singular and plural pronouns to the use of emoticons. This suggests that sentiment analysis could also aid doctors in the diagnosis of mental health issues by modifying the model slightly or to use different labels or features.

2.2 Comparison Of Social Media Platforms For Data Mining

According to Statista (2019) the number of monthly active users for Facebook is 2.3 billion and monthly active users on Twitter is 321 million. These statistics were taken from over 22,500 sources, including studies, scientific journals and other relevant statistical publications (Statista, 2019). These statistics reveal that there are over seven times the number of monthly active users on Facebook than on Twitter. Therefore, if a company wants to get an overall more accurate sentiment of their business, and influence as many people as possible, they should invest mostly in Facebook.

Furthermore, a study by Hughes et al. (2012) states that Twitter users showed significant correlations with the personality traits; conscientiousness, openness and sociability. This suggests that Twitter users are more critical of a sub-par product or service and take obligations to others seriously due to their conscientiousness trait. Moreover, they are open to be influenced about their perception of a business due to their openness trait. Therefore Twitter would be a good platform for a business to use due to potentially being able to get constructive criticism about where their business needs further improvement whilst still being able to influence opinions and, therefore, create a positive sentiment. In the same study Hughes et al. (2012) also suggested that Facebook is correlated with the traits; neuroticism, extraversion, openness and sociability. Neuroticism contrasts conscientiousness as, instead of giving constructive feedback, individuals displaying this trait are more likely just to act off pure emotion. Furthermore, similar to Twitter, Facebook users also show openness, which is positive for improving perceptions and therefore sentiment. Overall, Hughes et al. (2012) findings suggest that a sentiment analysis on Twitter seems to be the best choice in order to get accurate sentiment and to be able to use the data collected to improve sentiment.

2.3 Language Feature Extraction

Feature extraction is a key part of sentiment analysis as picking the correct method of feature extraction, as well as fine tuning this method will improve the accuracy of the model significantly. One set of features that has been very popular since Pang, Lee and Vaithyanathan (2002) first introduced and applied the concept of sentiment analysis is the use of something called an n-gram. An n-gram is described by Ramisch (2008) as any sequence of (contiguous) words in a text. For example a bi-gram of the sentence sentiment analysis is useful would be analysis is or is useful. In a study by Pak and Paroubek (2010), they were able to get an accuracy of 81% by using bi-grams and also by getting rid of common words, called stop words, that dont add sentiment to the sentence. 81% seems to be the best accuracy that can achieved using purely n-grams in something like a support vector machine as Singh and Husain (2014) were also able to achieve an accuracy rating of 81.15%.

However, using purely n-grams appears to be more common in the past decade as there is a clear shift in academic literature about sentiment analysis towards the use of a word embedding model (Lacobacci, Pilehvar and Navigli, 2015) with a neural network architecture. On Google Trends you can see that the interest for a word embedding model like Word2Vec clearly surpasses the interest term n-gram in late 2016 (Google Trends, 2019). In the past years many different approaches to creating word embeddings were developed such as Neural Probabilistic Language Model, Word2Vec , GloVe, Doc2Vec and fastText to name a few (Perone, Silveira and Paula, 2018). Each of these approaches have their advantages and disadvantages in the context of sentiment analysis.

Word2vec is among the most popular and the fastest word embedding toolkits, created by Mikolov et al. (2013) at Google. When comparing Word2vec to another model, it is not only faster than one of its main competitors, GloVe, but it is more accurate also (Mikolov et al., 2017). Moreover, the fastText library created by Bojanowski et al. (2017) outperformed the GloVe model on all of the datasets used (Mikolov et al., 2017). This is also back up in a study by Schnabel et al. (2015), who also found that on average Word2Vec outperformed GloVe. Interestingly, however, Mikolov et al. (2017) doesnt go on to discuss how well the fastText model does compared to the Word2Vec model even though the paper compares the Word2Vec and GloVe models individually to the fastText model. However, when reviewing a paper by Bairong et al. (2017), where a system has to generate a sentence in response to a user input, they found that the fastText model had the best BLEU score in comparison to the Word2Vec or GloVe model. The BLEU score is a judgment on whether or not the generated sentences are natural and informative for the user.

Within the Word2Vec model there are two architectures to produce a word vector

- a skip gram model or continuous bag of words model (Mikolov et al., 2013). Both of which have their own advantages and disadvantages in the context of sentiment analysis. In a study carried out by Ling et al., (2015) they found that the skip gram architecture out-performed the CBOW architecture. However, the difference in accuracy was only 0.11%. Therefore, in theory, it shouldnt really matter which architecture you use. However in research done by Alayba et al. (2018), where they tried to improve their Word2Vec model for use in sentiment analysis, they conclude that they were going to use a CBOW architecture with 200 dimensions, based off the findings in the word similarity. With this they also managed to achieve an accuracy of around 90%, which could be considered to be state of the art. Overall, based on the findings of these two studies, the difference in output between the two architectures is minimal and choosing which architecture varies from dataset to dataset.

On the basis of the academic literature reviewed, it is clear that using either the fastText or Word2Vec models are the best models to use to create word embeddings. However, the question must be asked, which is best in the context of sentiment analysis? Cliche (2017), carried out a study to find out which variation of pre-trained word embedding model and neural network performed the best. The results show that on each type of neural network (LSTM or another RNN) the word2vec model had the best accuracy every time, on multiple different training sets. This is also backed up by Yu, Lee and Wong (2016) who carried out an evaluation of the result from the IALP 2016 shared task on Dimensional Sentiment Analysis for Chinese Words (DSAW) which seeks to identify a real-value sentiment score of Chinese words. In this study a model that uses word2vec out-performs the fastText model that was submitted in both of the performance metrics. However there was only one submission that did use the fastText model, where nine of the submissions used Word2Vec. Therefore the question must be asked, if more submissions used fastText, would it out-perform Word2Vec? Nevertheless, Word2Vec did out-perform fastText in both of the studies. Therefore the evidence suggests that Word2Vec is the most effective model for sentiment analysis.

2.4 Deep Learning For Sentiment Analysis

The best method of classifying items of text has been debated even since Pang ,Lee and Vaithyanathan (2002) first published their paper about sentiment analysis where they compared the accuracy of Naive Bayes classifier, maximum entropy classification and support vector machines. In their finds they found that support vector machines, on average, performed best and Naive Bayes performed, on average, the worst. However, they do note that the difference between the accuracy of the three arent large. They also go on to say, when talking about a movie review in the style that contradicts itself over and over, bag-of-features classifiers would presumably find these instances difficult, since there are many words indicative of the opposite sentiment to that of the entire review. This shows a clear flaw in these types of machine learning classifiers. However in a study by Go, Bhayani and Huang (2009) on the sentiment of tweets, with the addition of emojis, they found that the maximum entropy classifier performed the best out of the three classifiers. Once again though, the margin of difference in accuracy between the three is minimal.

However, in recent years, there has been a shift towards using deep learning and neural networks to perform sentiment analysis, particularly convolution neural networks and recursive neural networks. One reason for this is because the bag of words models like support vector machines or naive bayes ignores factors like word order and suffers from data sparsity (Zhang, Wang and Liu, 2018). Deep learning called Recursive Neural Tensor Network was proposed in 2013 by Socher et al. as a method of sentiment analysis. In this paper they managed to improve the (at the time) state of the art sentiment detection by 5.4% using deep learning. This was a huge jump for a state of the art artificial intelligence.

Furthermore, even more recently, a RNN model called long short-term memory has been widely used in research into more effective ways of sentiment analysis. In a research paper by Tai, Socher and Manning (2015) for a fine grained classification of sentiment (where the rating were very negative, negative, neutral, positive, and very positive) the LSTM model came out on top with an accuracy of 51%. However, in the binary classification (positive or negative), the CNN came out on top. Furthermore in a study by Yin et al. (2017) they compared the performance of LSTMs, GRUs and a CNN carrying out various natural language processing tasks including sentiment analysis. On sentiment analysis of movie data the GRU out performed the LSTM and the CNN with an accuracy rating of 86.32%, with the LSTM coming in second with an accuracy rating of 84.51% and finally the CNN achieving an accuracy rating of 82.38. So this study contradicts Socher and Manning (2015) study where the CNN beat the LSTM.

However, when carrying out this project due to the hardware available has to take into account factors like training time and resources required. In general, the convolutional layer in a CNN is about 5-10x faster than an optimized LSTM layer. However, a different neural network called a Simple Recurrent Unit which is designed to provide expressive recurrence, enable highly parallelized implementation of a deep neural network (Lei, Zhang and Artzi, 2017). In a study by Lei, Zhang and Artzi (2017) they were not only able to match speed and accuracy on a sentiment analysis task, but were also able to marginally improve on the speed and accuracy of the other CNNs and RNNs they created.

So with all the factors considered what deep learning architecture should be used in this project? Factors such as resources available, time available and complexity of the model should all be considered. Taking all factors into account for this project, a CNN would be sufficient to achieve suitable accuracy and isnt too resource heavy, however LSTMs and other deep learning methods could be explored during the project.

Chapter 3

Methodology

3.1 Project Management

Project management is a well developed and accepted field in the exercise of professional expertise. Effective project management is a key aspect in the success of the whole project. In a study comparing the relationship between project management methodology and project success it states having a comprehensive set of project management methodology elements including tools, techniques, process capability profiles and knowledge areas is linked to project success (Joslin and Mller, 2015). This is why not only executing a project management methodology is key in a successful project, but also researching and selecting the right methodology for the project is so important in the success of a project.

Successful project management is defined as achieving project objectives, within time and within cost, whilst achieving a desired quality (Kerzner and Kerzner, 2017). A Guide to the Project Management Book of Knowledge (Project Management Institute, 2017), identifies industry accepted activities in regular to best practices and can be use to create a project management methodology for a variety of projects. Project management is so important as it leads to projects; meeting objectives, being more predictable, increases the chances of success and being able to meet deadlines. However, poorly managed projects may result in missed deadlines, poor quality of work and a worst case scenario of a failure to achieve the objectives for which the project was undertaken (Project Management Institute, 2017). So for a project to be successful it must have a clear project management methodology.

In the book: A Guide to the Project Management Body of Knowledge (Project Management Institute, 2017), they split project management into nine distinct sections. However, due to the nature of my project (no shareholders, completed as an individual with the help of a supervisor) the project will only be split into three sections by combining some of the nine sections into one section. The first being risk management.

3.1.1 Risk Management

Risk management is the identification and management of uncertainty in a project (Chapman and Ward, 2003). A Guide to the Project Management Book of Knowledge (Project Management Institute, 2017) states that there are seven risk management processes; planning risk management, identifying risk, performing qualitative risk analysis, performing quantitative risk analysis, planning risk responses, implementing risk responses and monitoring risks. In the project proposal a risk management matrix is created where five different risks are identified and given them an overall risk rating between one and twenty-five based on how likely they are to happen and, if they do happen, how much will they impact the project.

The risk matrix will serve as my main tool in completing risk management.

3.1.2 Time Management

There is no clear set definition of time management (Claessens et al., 2007). However, Lakein and Leake (1973) define it as prioritising and planning tasks required to achieve these goals. Furthermore, in the book A Guide to the Project Management Book of Knowledge (Project Management Institute, 2017), they define it as the process required to manage the timely completion of the project. Furthermore, they also split time management up in to the processes; processes schedule management, define activities, sequence activities, estimate activity duration's, develop schedule and control schedule. These processes were created in the proposal of this project in the form of a gantt chart.

Risk	Explanation	Likelihood*	Impact*	Score**	Mitigation
Hashed passwords might be cracked	Due to the passwords being hashed, it is hard for a user to get the hashed password as it is a one-way system. However, if it does happen, the user's passwords could be leaked. Furthermore, the user could use the password for many different applications.	1	4	4	Ensure sufficient up-to-date hashing is being used and get user to use complex enough passwords as the most common ways passwords are cracked is guessing passwords, dictionary attacks, and brute-force attacks.
Data might not be displayed correctly whilst testing	There is a chance that the front end and back end may not communicate correctly and the data may not display correct. This would result in providing the user with false information due to the data not being tested properly.	2	3	6	Test the MVC properly with many different variables and test it regularly to ensure the data being provided is correct. Also allow users to report data they believe to be incorrect in order to be able to respond quickly and correct this. Also use a controlled set of test data where the output is known and compare the expected output to the actual output.
Website may go down	The website may go down temporarily so no one will be able to use this	2	2	4	Although you cannot predict when a website will go down, if it is in your control, work to get the website up and running again. However, if it is out of your control, try and get regular updates on when it will be up and inform users, if required.
Test data that is used for machine learning may not be accurate	Because I will probably be using a pre-made set of test data for my machine learning, the data may not be accurate and the sentiment analysis may not be accurate	1	4	4	Make sure that the test data I use is firstly from a reliable source and then check it myself to make sure it is accurate.
I won't be able to obtain a Twitter developer account	I need a Twitter developer account in order to be able to access Twitter API's	1	5	5	I will have to use another platform instead of Twitter, such as Facebook.

Figure 3.1: Risk matrix created in the project proposal

A Trello board was then created in order to keep track of what task currently are currently in progress and what tasks needs to be done. This was very much in the style of a Kanban board. Kanban is based on three principles; to visualize, limit work in process and manage flow. Creating a Trello board fits almost all of these principles as seen below.

The time frame for each of the five main tasks was set out in the gantt chart. However, for the smaller tasks it was more difficult to estimate the exact amount

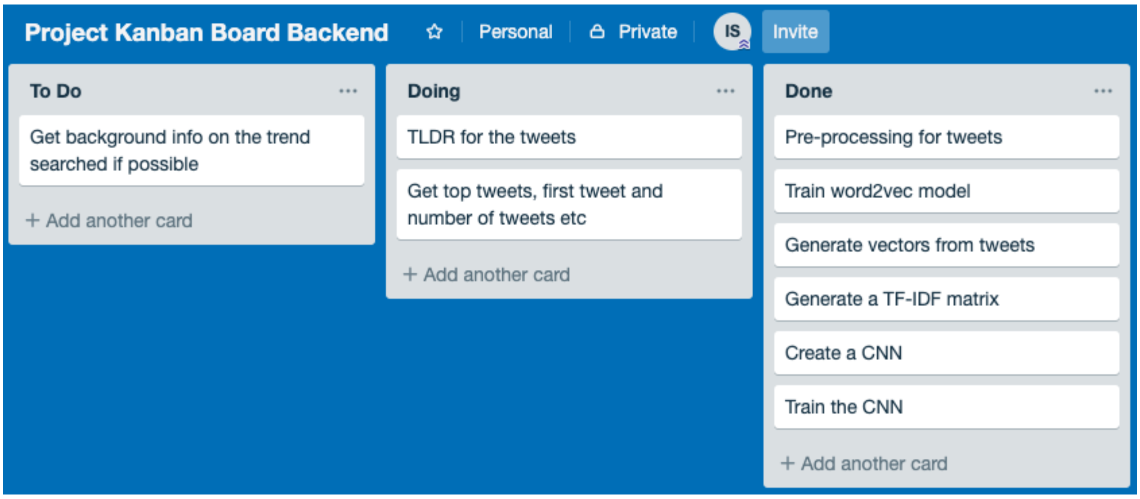


Figure 3.2: The backend Kanban board

of time that it would take.This was especially true for something like deep learning using my CNN as the developer had never done deep learning before and didnt know what the training times would be. This meant that the number of hours it would take to complete a task had to be dynamically added. This was done using a daily to do list like the one seen below in my iCloud notes app.

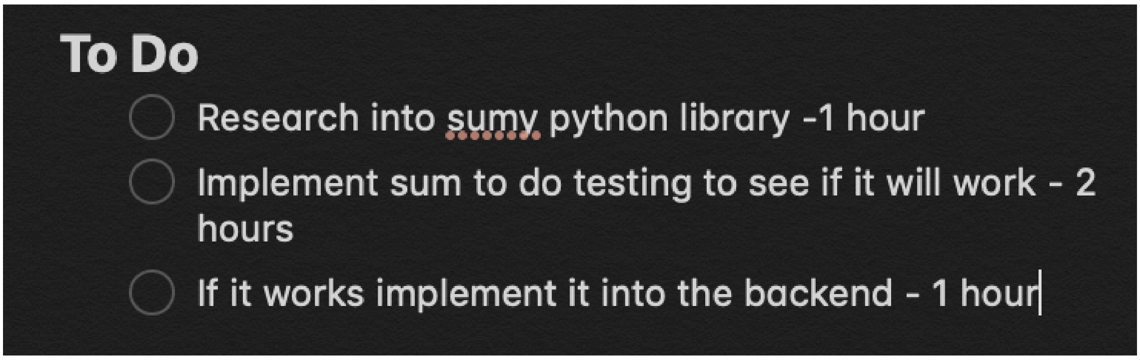


Figure 3.3: An example of my daily to do list that are attempted that day

3.2 Software Development

The software development life cycle is a methodology for designing, building and maintaining information and industrial systems (Bassil, 2012) . There are many different models for software development, all with their advantages and disadvantages. However, due to the nature of this project such as there being no client and it being a code heavy project, a non-iterative methodology called Kanban was chosen. Kanban is a lean approach to projects, which aims to reduce waste and increase productivity (Ahmad, Markkula and Oivo, 2013). In recent years it has grown to be one of the biggest methodologies for software development. Kanban is mainly based around a Kanban board as shown in *figure 3.1* and mentioned in the previous section. However, Kanban is often fused with an agile framework called SCRUM. This is often called scrumban or leagile. Liz Keogh (2011) is an independent Lean and Agile consultant and claims that both methodologies put people and their interactions at the heart. Both have a clear focus on value, fast delivery and the continuing growth of the team and its ability to achieve those valuable deliveries. However, due to the nature of the project (no clients), this project uses kanban methodology. Furthermore, eXtreme programming practices have also been incorporated in the project such as; simple design, refactoring, 40 hour work week (adjusted) and coding standard.

Kanban is particularly useful in this particular project as the kanban board (see *figure 3.1*) allows me to visualise easily everything that is to be done in the day and maps clear progress with the done section on my trello board. This is a key source of motivation as it gives a sense of achievement while consistently pushing you to work on new tasks. New task are only pushed when needed and the kanban board has a maximum of three tasks currently active. This allows the developer to not burn out and ties in well with the 40 hour work week in XP. This provided enough structure to the project so that stories would get done in sufficient time. However, it didnt provide too much structure so that it would be hard to introduce new features to development or tasks into development. EXtreme programming was used as it would increase the quality of output of the code as the project is very code heavy, and XP nominates code as the key activity throughout the software project (Beck and Gamma, 2000). A simple design ensured that the project doesn't become over complicated the interface of the website as, in theory, it is just a tool to look at sentiment analysis and therefore the sentiment analysis should be at the forefront and focus of the website. Re factoring and a coding standard ensured that not only the code works, but the way it looks and runs is not too complicated and will be consistent throughout. Re factoring the code every so often is important as when new features are added, code could become spaghetti, and this ensures well structured code. And finally, a 40 hour work week has been adapted for this project to more like a 15 hour work week to ensure progress is made on the project without the developer burning out.

3.3 Tools & Machine Environment

Deciding on tool sets and machine environment methodology is crucial for any project to succeed, especially in such a quickly developing field such as deep learning. This section will describe the methodology that will be used to develop the code, such as languages, libraries used, version control and hardware used.

3.3.1 Python

For this project, the two languages that were considered, based on their machine learning and data analytic libraries available to them, were Python and R. When comparing the two, python is a not only a good language to do research and prototyping, but it is also a good language to build production systems (McKinney, 2012) and it is used by companies such as Instagram, Spotify and Amazon (Cleveroad Inc., 2018). However, Python is also a high level interpreted programming language, therefore runs slower than languages like Java or C++ (McKinney, 2012). On the other hand R is also a very powerful programming language that is used by companies such as Google, Bank of America and Shell. However, R is mainly used in academic research and not in production unlike python. However due to the vast array of libraries available and it working with web frameworks like Django and Flask, Python is the most suitable choice to carry out this project.

Python was first developed in the 1991 by Guido van Rossu as an extension to a language called ABC (Python Software Foundation, 2019), with the most recent release being 3.7.2. Python 3.7 was used throughout the project. Pythons most popular and most powerful libraries include; Numpy, Pandas, Matplotlib and iPython (McKinney, 2012), Gensim and Tensorflow. Pythons extensive number of libraries are one of the key reasons why it is used by so many for data analysis like sentiment analysis. The library Numpy has the goal to create the corner-stone for useful environment for scientific computing (Oliphant, T.E., 2006). It adds to Python a fast and efficient multi-dimensional array object, functions for performing element-wise computations with array or mathematical operations between arrays, linear algebra operations and fourier transform. Numpy was a key element in handling data in Python, alongside another library called Pandas, which provides rich data structures and functions to make working with structured data such as the CSV files (that the training data is in) fast, easy and expressive. (McKinney, 2012). These two combinations makes Python an overall very powerful language for data processing and is used in the project a great amount.

For the deep learning aspect, the libraries Gensim and Keras were used. Gensim offers a parallelized Word2Vec implementation in Python (Vuurens, Eickhoff and de Vries, 2016). This was used to train, save and load models that were created using word2vec. Keras is also a key library used as it is a deep learning framework for Python that you can define almost any time of deep learning model (Chollet, 2017). In this project, it will be run off the TensorFlow backend. TensorFlow is a machine learning system that operates at large scale and in heterogeneous environments (Abadi et al., 2016.).

As far as IDEs go, for the start of the project, sublime text was used. However, within the first few hours of programming it became apparent that sublime text was not suitable for data analysis, mainly due to the lack of variable explorer inbuilt. After some research and browsing websites such as reddit, Spyder was the chosen IDE. Spyder is part of the anaconda package. This not only had a variable explorer, but had an iPython console built into it. This enabled me to view what was in the Numpy arrays and Pandas with ease. Furthermore, in the latter stages, research done into other IDE's and PyCharm was used for some aspects. However, the large bulk of the project was done in Spyder.

3.3.2 HTML, CSS, Javascript and Bootstrap

When developing the frontend of an application, the objective of the project was to create a website. To do this the obvious choices to use are; HTML, CSS and Javascript. Bootstrap was also employed as a front-end framework, which is described as a sleek, intuitive, and powerful front-end framework for faster and easier web development (Jain, 2015). For this reason, due to the project not being too heavily focused on a novel front end, this will be used to aid me in developing the website. However, due to time constraints, a template was used for the website. As for the IDE, Brackets was chosen due to it being built for markup language, the vary array of plugins and the live preview.

3.3.3 Flask

Flask will be used as the web framework so that the backend and the frontend of the website will be used.

3.3.4 Version Control

Version control is a key tool when developing the artifact as it enables you to follow the iterative changes you make to your code. Hence, you can try different things with new thoughts yet dependably have the alternative to return to a particular past adaptation of the code you used to produce specific outcomes (Blischak, Davenport and Wilson, 2016). It also allows you to be able to collaborate with other developers easily. However, due to this being coded and developed individually, this was not a concern to the project.

The version control system used in this project is Git, with the use of the tool Github Desktop and the Github website. Git is describe on their website as a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. (Git, 2019).

3.3.5 Hardware

In terms of hardware for this project, it was developed on a Macbook Pro (2017, 2.3 GHz Intel Core i5, 8 GB 2133 MHz LPDDR3, Intel Iris Plus Graphics 640 1536 MB) running Mac OS Mojave. However, executing code like training the LSTM had to SSH into, a more powerful desktop PC (Quad Core Intel i5, GTX 970 4GB, 16GB RAM) running Ubuntu 18.10 was needed . For this a remote kernel in Spyder was used, which was then connected to a Macbook Pro. This was due to the Macbook Pro heating up too much when executing certain code, such as training the CNN and particularly the LSTM.

3.4 Testing

Software Testing is the process of executing a program or system with the intent of finding errors (Myers et al., 2004). The testing methodology is split up into two distinct sections; white box testing and black box testing. The black-box approach is a testing method in which test data are derived from the specified functional requirements without regard to the final program structure (Perry and William, 1989). This is usually done by a system tester who doesn't know much about the system.

However white box testing is very dissimilar to black box testing hence the name. White box testing is testing the system with knowledge of the system, usually done by a developer.

Both of these are equally important as white box will test the obvious potential flaws in the program, while black box will test unexpected inputs that a developer will not think of. This will all be done in a table in section 4.8.

Chapter 4

Implementation

4.1 Dataset

A dataset called Sentiment140 was utilized for this project. This dataset was created by an algorithm developed by Go, Bhayani and Huang (2009) using an algorithm specified in their paper. Their algorithm was able to achieve an accuracy of around 83% using Maximum Entropy algorithm. However, this has since been further developed upon and improved to create the dataset that exists today.

The dataset has a total of six column detailing; the sentiment, the tweet ID, date of the tweet, a boolean value always set to false called query_string, the user who tweeted it and the tweet itself. However for sentiment analysis, all that needs to be taken is the text and the sentiment. Also, the sentiment is a value from 0-4 with 0 being negative and 4 being positive. All the training data in the project is either 0 or 4 and the testing data having values 0, 2 or 4.

Furthermore, in late 2017, Twitter introduced a new character limit of 280 characters, doubling it from the previous limit of 160 characters. This could also be an disadvantage of the dataset. However, there is no dataset available of this size; therefore it will be the best dataset to train my model.

4.2 Pre-processing

In Kotsiantis, Kanellopoulos, and Pintelas (2006) paper titled Data pre-processing for supervised learning, they express how import data pre-processing is on the performance of a supervised machine learning algorithm. They also go on to say that the success of the model is usually dependant on the quality of the data that they operate on. If the data is inadequate, or contains extraneous and irrelevant information, machine learning algorithms may produce less accurate and less understandable results, or may fail to discover anything of use at all. So pre process is a key part of the implementation process

Due to the dataset used in the project being explicitly tweets, the information from the text that needs to be filtered out is; any usernames , any URLs, any punctuation, any numbers, and make it all lowercase. This ensures that the tweet contains only text. However, this does remove emojis and therefore some information about the sentiment is lost.

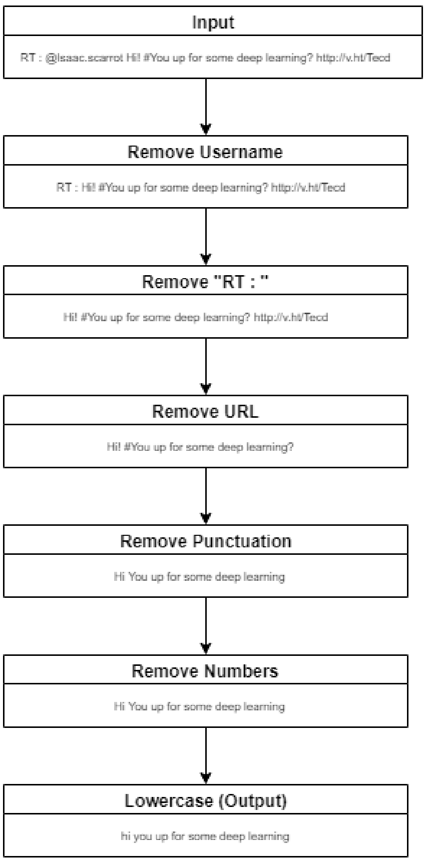


Figure 4.1: Steps for text pre-processing visualized

The tools within Python used for pre-processing is exclusively modules that come with Python. Regular expressions are used for all pre-processing, apart from making it all lowercase and removing punctuation. To remove the username from a tweet, the regular expression below is used

$$@[A-Za-z0-9]+$$

. This simply looks for the @ symbol, [] looks for a single character present in the set (with the set having the ranges A-Z, a-z and 0-9) and finally + matches the set between one and infinite times. To remove the RT : from tweets the regular expression RT : is used which removes anything that matches it exactly.

Removing the URL was a bit more difficult due to it being in many forms such as; www.google.com, www.google.co.uk, http://www.google.com or even https://www.google.com. However, it just needed to recognize the start of the URL, which will usually come in the form of http://, https:// or www.. Adding to this, if someone posts a link like old.reddit.com it will not recognize this. However, this is only a very small minority and therefore shouldnt affect that data at all due to the word2vec model not having it in its vocabulary. To remove the URL the regular expression below is used.

$$((http(s)?:)|(\dot{www}))(S)*$$

The regular expression will first of all look for the http and then will look if there is an s proceeding it. It will then look for :// and then look for a www.. If any, or both, of these are found then it will remove them. It will then look for between zero and infinite non white space characters to remove the rest of the URL using the expression below

$$(S)*$$

.

To remove punctuation the translate function built into Python was used. This returns a copy of the string where all characters have been translated using a table and deleting characters in the second variable passed. An almost brute force method is used to delete characters using every single character on the keyboard. Finally, removing numbers was an easier task by using the regular expression below.

$$\dot{+}$$

. matches any digit.

This was all the text pre-processing that was required. The only other types of pre-processing that were considered and trialled were stemming and the removal of stop words. Stop words were not included in pre-processing due to some of the words that may be removed could provide vital sentiment to the tweet. For example in NLTKs list of English stop words, they include not and this is vital at providing context to the tweet; therefore they were not removed. Due to the size of my dataset, there was no need for stemming as word frequencies are very high and every word from the dataset is in the vocabulary.

Furthermore, the sentiment number is a value between zero and four and therefore the data need to be between zero and one. To do this, you divided a numpy array by four so it could be fed into a NN for training.

4.3 Word2Vec Model

Word2Vec is a two layer unsupervised NN (Wohlgenannt and Minic, 2016) created by Mikolov et al. (2013), which creates vector representations of words learned by the model that have been shown to carry semantic meaning (Rong, 2014). The vector has dimensions that are specified by the user, usually between 100-300. Any more than this will create random noise that will potentially add no context to the word. Any less than this will mean that there is a strong possibility that there would not be enough information about the word and therefore not enough information about the context of the word.

These dimensions are called the word vector and are learned by learning about the context of the word using a window of words around the given word. For example, the words Facebook and Twitter would be mapped as similarly as the words McDonalds and Burger King. Taking words and creating a vocabulary of word vectors is very useful, as with the unique numerical context of the word rather than just a string of characters. Consequently mathematical operations can be performed on the vector. For example, a common operation to perform is King-Man+Woman and a well trained model should output queen, or Candle-Fire+Electricity should output light bulb. Furthermore, the model can now be fed it into a NN for it to learn about the word vectors using a labeled dataset.

Word2Vec has two models within it; continuous bag of words and skip gram. The basic concept of skip gram is it takes a given word and tries to predict the words around it, however CBOW does the opposite of this and it take an array of words and tries to predict the word in the middle.

Averaging the vectors of each word for each model is often the best method to get accurate word vectors. However Mikolov et al. (2013), the creators the the Word2Vec model, states the skip-gram model works well for rarer words. Therefore, due to the training data being tweets that will contain rare words like slang words, this will be the better model to use to create our word2vec model.

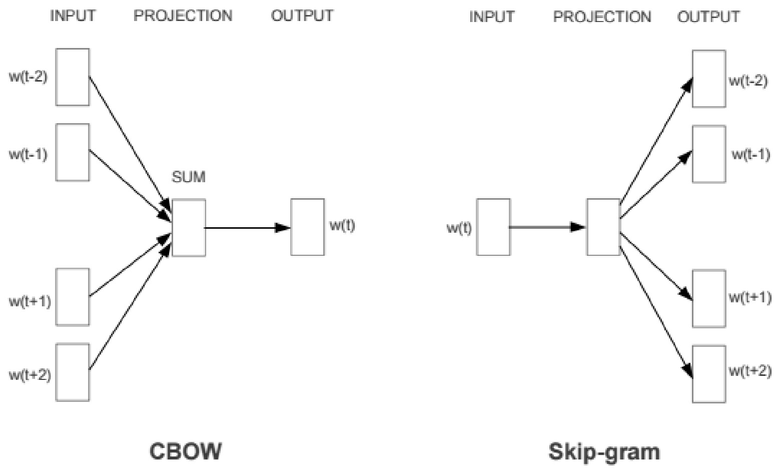


Figure 4.2: A comparison of the input and output of CBOW and SG

To implement the Word2Vec model, the library Gensim was used. Gensim is an open-source powerful library used for machine learning and natural language processing that includes the fastText, Doc2Vec and Word2Vec algorithms. However, only the Word2Vec model will be used based on research done in previously. You simply import the word2vec model from the Gensim moduele and call the word2Vec function. The input is all of the words in your training and test data tokenized.

There are many hyper-parameters that you can modify such as using SG or CBOW, size, window, workers and many more. However, the only modified the hyper-parameters are the ones listed. As discussed before, the skip-gram model is used. As for the size of the vector, the most optimal number was 100 features based on the output of the similarity to other words

Window is the maximum distance between the current and predicted word in a sentence. This was set to 6 due to tweets having a maximum of 280 character or approximately 56 words. A small window size ensures that enough context is captured from the surrounding words whilst not getting context from words that are not related. These were found to be the optimal parameters for creating the model for word vectors.

However, using the Word2Vec model pre-trained by Google was also experimented with. It is approximately a 3.5gb file of approximately 3 million word vectors with each vector having 300 dimensions. This, however, did not improve accuracy as shown in previously and is significantly more resource intensive.

4.4 Deep Learning

Even though in previously it was decided that CNN was the deep learning NN to be used to learn about the sentiment of tweets, it would be interesting to see how other models perform compared to the CNN. To do this a multilayer perceptron, a convolutional neural network and a long short term memory, all implemented using Keras module, which was discussed in section x. The tweet vectors as discussed in a previous section that are fed into the model. The hyper-parameter used for the word2vec model are optimized for usage on the CNN model; however, they should be optimal for all the NNs.

4.4.1 Keras & Tensorflow

Keras is essentially a wrapper for other lower level frameworks such as tensorflow, Theano and CNTK. However, the most widely used and the default backend to use with Keras is TensorFlow. Each model is defined as a sequence of layers using the Sequential Class. Furthermore, you simply add the chosen layer block by block to the network, defining its hyper parameters when calling the function of the layer.

When completing the model the optimizer, loss and metrics parameters were utilized. Optimal optimizer parameters are chosen from model to model; however, the metric of accuracy and binary cross entropy for the loss function are always used. Binary cross entropy is used because this is a binary classification problem. This is also called binary log loss due to the graph being a log function as shown in figure 4.3.

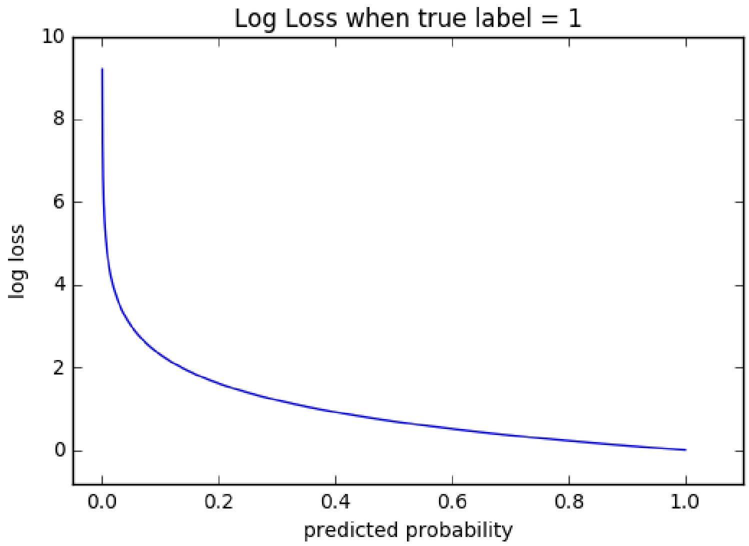


Figure 4.3: Binary Log Loss Graphed

The loss function is the value that wants to be minimized down to as close to zero as possible so that the value is closer to the truer values, or in binary cross entropy the predicted probability being 1 or 100%. This is a key measure of overfitting or underfitting.

Tensorflow is the most popular machine learning library in the world at the moment. Tensorflow was developed by Google to take advantage of the vast amount of data that they have available; consequently it is a very powerful machine learning library. Tensorflow is lower level than Keras and therefore Keras hides lots of the complexities of tensorflow.

Tensorflow also has TensorBoard inbuilt into it, which is a visualization tool you can analyse loss and accuracy and create a graph of all the layer of your model, which is a huge advantage of tensorflow as it can be used to analyse your model within this project.

4.4.2 Convolutional Neural Network

CNNs became popular in 2012 when Krizhevsky, Sutskever and Hinton created a CNN for the ILSVRC-2012 competition ImageNet classification, winning by a margin of a 10% more accurate algorithm. Therefore it is mostly used for analyzing visual imagery due to it being able to detect patterns in data. However, performing other forms of analysis such as sentiment analysis can produce very optimal results due to its pattern recognition ability(Shen et al., 2014).

The CNN implemented in the project is based off the CNN implemented in the paper by Ouyang et al. (2015). You can see the template used in figure 4.4.

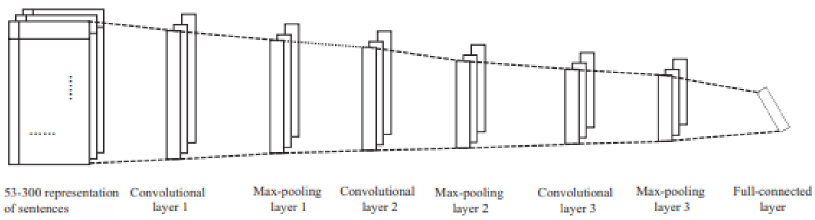


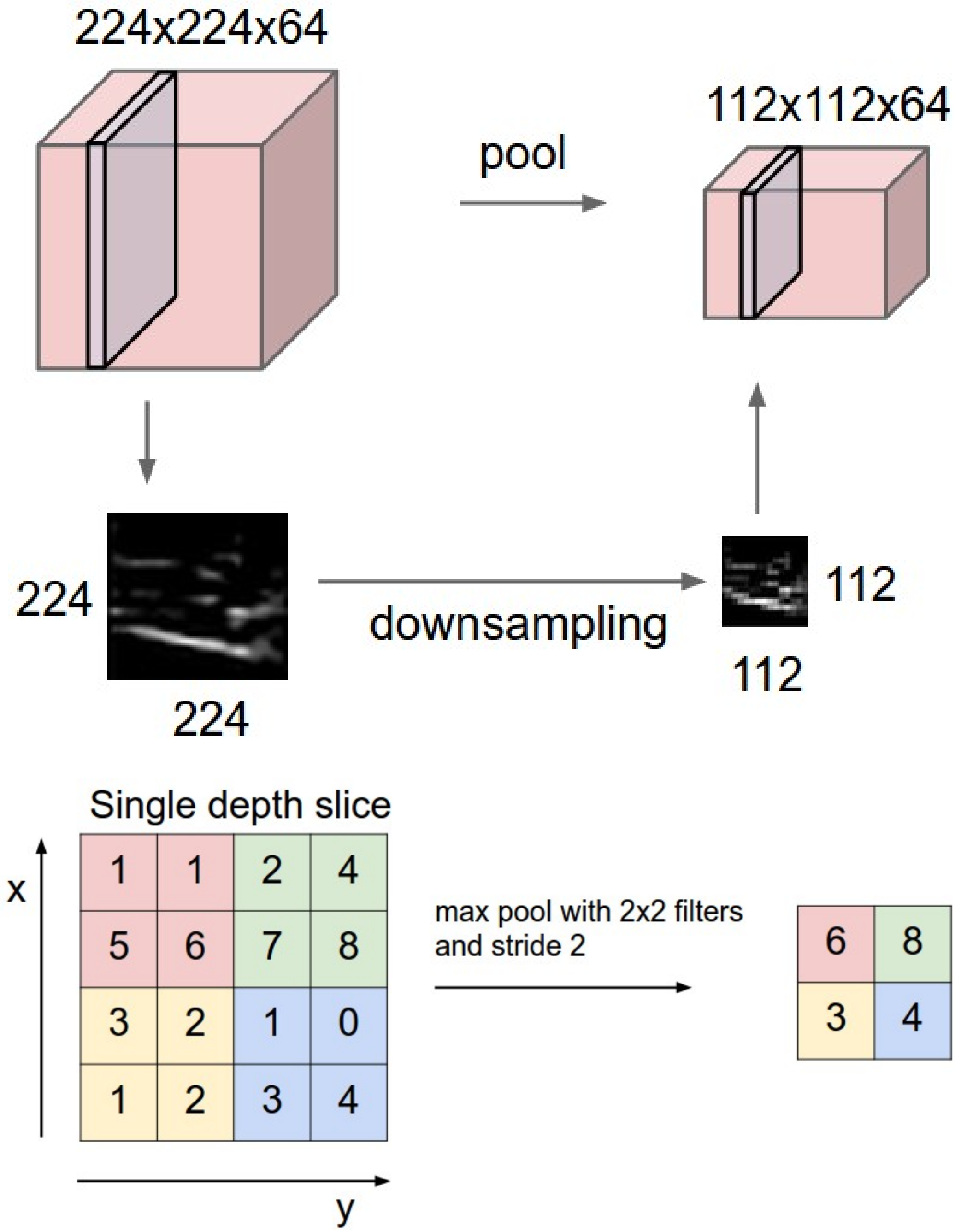
Figure 4.4: CNN designed by Ouyang et al.

However, the input is slightly changed to suit our input and the last maximum pooling layer is a global maximum pooling layer into a densely connected layer.

The input layer takes the padded sequence of tweets. The number of features taken in is 250, which was set when we padded the tweets. This goes into the embedding layer. The embedding layer will take the sequence of tweets and embed each index with the corresponding vector given in the embedding matrix, which was created earlier using the word2vec model that was trained. So essentially it will take a positive integer and turn it into a dense vector of size 280 in this models case. This will treat the sentence like an image.

Next it is fed into a maximum pooling layer which downsamples the volume spatially, independently in each depth slice of the input volume (Stanford, 2019).

This will help stop a very common problem of overfitting, which could have been a huge problem in our model due to the lack of dropout layers which also prevents overfitting. However, in this project, dropouts werent needed due to the corpus being so large.



These two layers are then repeated again (convolutional and maximum pooling layers), with a third convolutional layer being added on. However with each convolutional layer added, the kernel size is increased by one. All the convolutional layers were activated by an argument called relu which stands for Rectified Linear Unit. These are used as they preserve information about relative intensities as information travels through multiple layers of feature detectors, which is crucial for keeping the context in sentiment analysis. This is then fed into a densely connected layer with

256 nodes and relu activation again. Finally the output layer is a dense layer with the sigmoid activation due to the output wanting to be either 0 or 1 depending on the probability of the input. This only has one node due to us having one output.

4.4.3 Feedforward Neural Network (MLP)

A feedforward NN is potentially the simplest type of NN that you can create with the simplest being a single layer perceptrons.

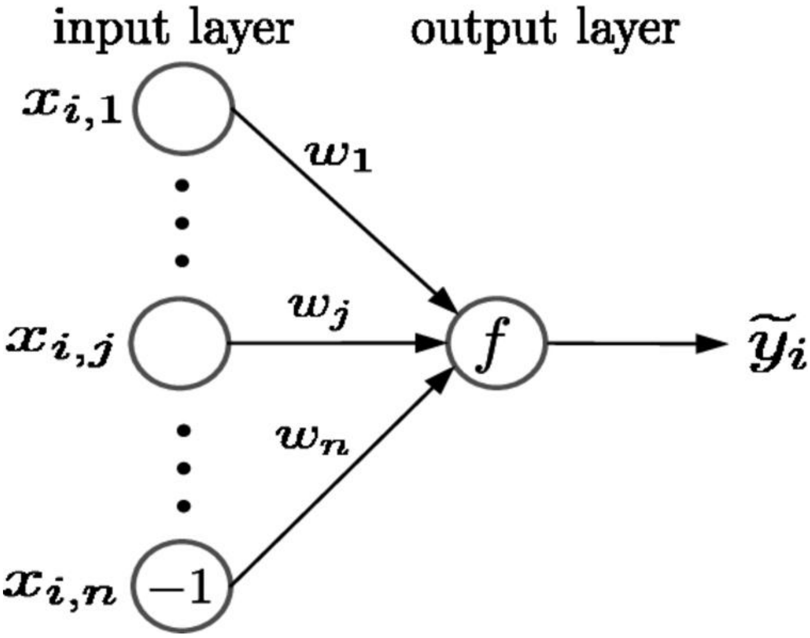


Figure 4.5: Basic design of a SLP

However, a huge limitation of a single layer perceptron is its inability to solve XOR due to it not being a linear function. To solve this Multilayer perceptrons were introduced (where there are at least 2 layers with usually a maximum of three layers) where, due to the addition of the hidden layers, it can deal with non linear inputs.

For the feedforward network simplifying the word vectors down into 100 features can take place by taking the average word vector with each word being multiplied by a TF-IDF matrix. TF-IDF is used in determining the importance of a word in a set of documents or, in this case, a set of tweets. It does this by giving it a weighting by determining the relative frequency of words in a specific document compared to

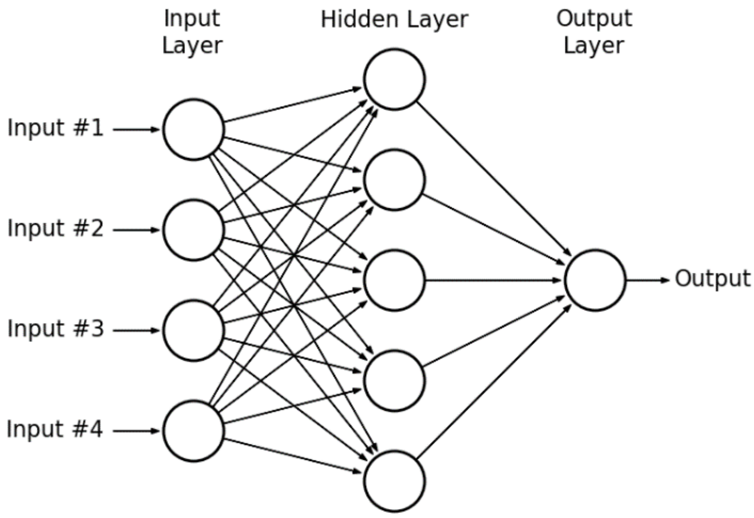


Figure 4.6: Basic design of a MLP

the inverse proportion of that word over the entire corpus (Ramos, 2003). This is useful as, for example, if you compare the top TF-IDF of an article about cities in the United Kingdom youd expect the word with the highest TF-IDF to potentially be London. However, in an article about cities in Netherlands youd expect it to be Amsterdam.The equation for TF-IDF of a selection of tweets can be written as follows, where T a set of tweets and t T, w is a word:

$$w_t = f_{w,t} \cdot \log\left(\frac{T}{f_{w,T}}\right)$$

To split this down even more, the first part is term frequency and the second part is inverse document frequency and you simply multiply these two values together to get the TF-IDF number.

This is a relatively simple algorithm to implement. However, the machine learning library scikit-learn has an inbuilt function that will calculate TF-IDF matrix when given the sentences or, in this case, tweets. You simply import scikit-learn and then call the function TfidfVectorizer and then call the function fit_transform with the raw tweets. This will generate the matrix, which can be saved for later use.

However, this alone does have its limitations as it just uses the current word with no context around it, which is crucial for sentiment analysis. Adding to this, if you combine this with word2Vec, this could add more information about each word to the tweet therefore increasing the accuracy of the MLP.

Creating tweet vectors using word2vec and the TD-IDF matrix generated is a relatively simple task. As discussed before, an average of the word vectors in the tweets would be used. First the word2vec model that has been trained is loaded in alongside the TD-IDF matrix. Each tweet is fed into a function where they are tokenized. Then each word in the tweet is summed after multiplying the word vector by the weighting value generated by the TF-IDF. Each dimensions is then divided by the number of words in a tweet to create a 100 dimensional tweet vector.

The model is very simple. There is the input layer that is then fed into two hidden dense layer that is then fed to an output layer. The input layers have 200 dimensions, which correlates to the size specified when training the word2Vec model. Next is the first hidden layer in the form of a densely connected NN layer, with the activation of relu, which was discussed before. The number of nodes in the first hidden layer is 128 as, after experimentation with different number of nodes, this seemed like the optimal number. The second hidden layer is almost exactly the same but takes an input from the dense layer and has a total of 32 nodes. Again, this is the optimal number found to use for generating the best accuracy. Finally, the output layer is a dense layer with the sigmoid activation due to the output wanting to be either 0 or 1 depending on the probability of the input. This only has one node due to us having one output.

4.4.4 Recurrent Neural Network (LSTM)

LSTM was only partially investigated and tested due to it taking so long to train and the success of the previous networks.

However to create a LSTM we simply input into a LSTM layer in keras that has 128 nodes and then output to a dense layer with 1 node and an activation function of sigmoid as discussed before.

4.5 Web App Development

For web development a mixture of HTML, Bootstrap for CSS, minimal javascript and flask was used. Flask is a micro web framework used to link the frontend and the backend (Python) of a web app. It was used instead of Django due it it being more pythonic than Django and will fit the needs of this web app perfectly.

Bootstrap makes CSS a lot easier as you simply define in the class the layout of the chosen element and it will apply the correct CSS automatically.

4.5.1 Design

For the design of the website, designs were first drafted using pen and paper, unfortunately these designs were lost. However, after creating them with pen and paper, Adobe Photoshop was then used to create some examples of what each web page would actually look. A blue colour scheme was followed, which is also similar to the colour scheme of Twitter.

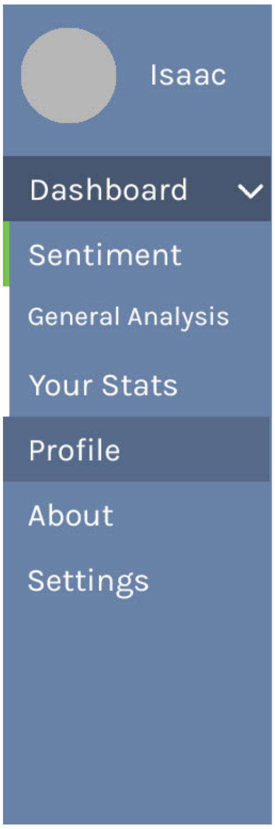


Figure 4.7: Inital idea of the template for the website

4.5.2 Website Development

For web development a template for the dashboard was used (Bootstrap, 2018). This was then modified by removing the graphs, changing the navigation bar items and adding relevant items in relation to sentiment analysis onto the dashboard.

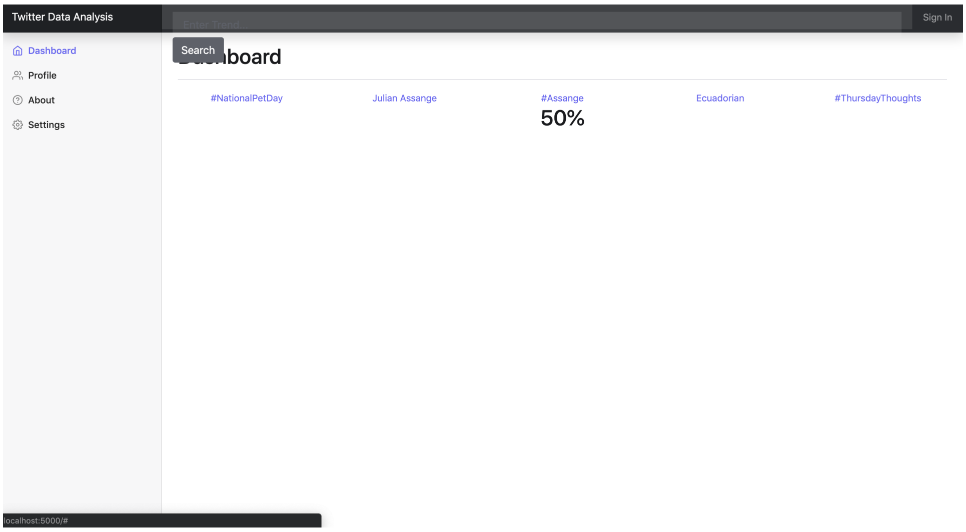


Figure 4.8: Modification of the template to show current trends and search for a keyword to get the sentiment of

4.5.3 Flask

As mentioned before, flask is used for the web framework. The app.py file contains a function to get the current trends, a function to get the sentiment of the current trends, a function to take an input from the search bar to get the sentiment of a current trend and will return the most frequent unigram or bigram when you search for a trend. Below is a figure of what output you get when you search happy.

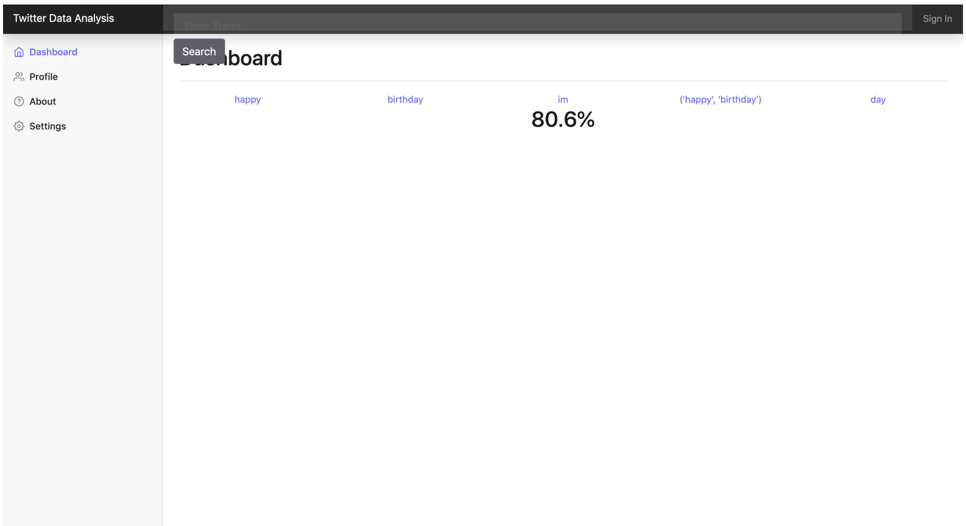


Figure 4.9: Using the word happy to get tweets from twitter and calculating the sentiment

Chapter 5

Results & Conclusion

First we tested the best method of pre-processing, which was described in figure 4.1, where tweets are reduced down to text only in order for it to be fed into the word2vec model. Then a Word2Vec model was fine tuned with the Sentiment140 dataset. The optimal hyperparamters for best word similarity are; using the skip gram model, using a vector size of 200 features, a window size of 6 and keeping every single word that the model sees. This enabled us to create the best word vectors in order to give the word context which is crucial for sentiment analysis.

This word2vec model was then used to create tweet vectors that would be fed into a MLP, CNN and a LSTM.The results are displayed in the table below.

Model	Epochs	Accuracy	Loss
CNN	1	0.74	0.62
MLP	3	0.83	0.49
LSTM	1	0.73	0.59

The results presented are surprising as MLP is more accurate by a margin of 10%. So therefore the model on the website is the MLP.

Overall the Twitter sentiment analysis web app seems to work well. For example Virgin Trains has made headlines as the owner stated virgin train could vanish from the UK "after its partner Stagecoach was barred from three rail franchise bids."

(BBC, 2019). So if we search V̈irgin Trainsö̈n our sentiment analysis tool the result can be seen in the figure below

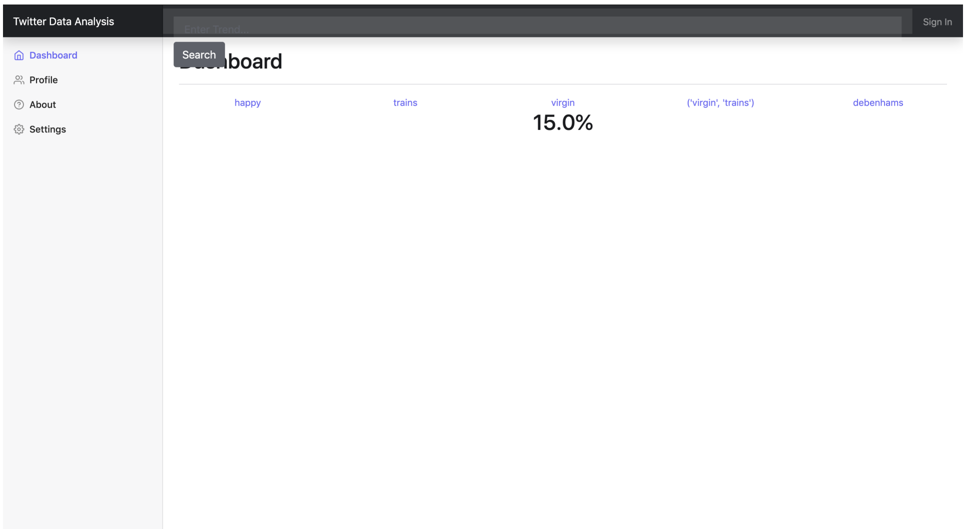


Figure 5.1: Result when we search for ”Virgin Trains”

I think this is a clear indicator that the sentiment classifier has a good understanding on what trends are positive and what trends are negative especially when we look at figure 4.9 as well. So overall the webapp at it's base level is a success.

However if I had more time I would tweak the hyperparameters more on the CNN and LSTM to try and improve accuracy as well as build the web app up to display more information about trends and to collect data about trends.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al. (2016), Tensorflow: A system for large-scale machine learning, *in* '12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)', pp. 265–283.

Ahmad, M. O., Markkula, J. and Oivo, M. (2013), Kanban in software development: A systematic literature review, *in* '2013 39th Euromicro conference on software engineering and advanced applications', IEEE, pp. 9–16.

Alayba, A. M., Palade, V., England, M. and Iqbal, R. (2018), Improving sentiment analysis in arabic using word representation, *in* '2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)', IEEE, pp. 13–18.

Alhabash, S. and Ma, M. (2017), 'A tale of four platforms: Motivations and uses of facebook, twitter, instagram, and snapchat among college students?', *Social Media+ Society* **3**(1), 2056305117691544.

Bairong, Z., Wenbo, W., Zhiyu, L., Chonghui, Z. and Shinozaki, T. (2017), Comparative analysis of word embedding methods for dstc6 end-to-end conversation modeling track, *in* 'Proceedings of the 6th Dialog System Technology Challenges (DSTC6) Workshop'.

Bassil, Y. (2012), 'A simulation model for the waterfall software development life cycle', *arXiv preprint arXiv:1205.6904* .

Beck, K. and Gamma, E. (2000), *Extreme programming explained: embrace change*, addison-wesley professional.

Blischak, J. D., Davenport, E. R. and Wilson, G. (2016), 'A quick introduction to version control with git and github', *PLoS computational biology* **12**(1), e1004668.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2017), 'Enriching word vectors with subword information', *Transactions of the Association for Computational Linguistics* **5**, 135–146.

Chapman, C. and Ward, S. (2003), *Project risk management: processes, techniques, and insights*, Wiley.

Claessens, B. J., Van Eerde, W., Rutte, C. G. and Roe, R. A. (2007), 'A review of the time management literature', *Personnel review* **36**(2), 255–276.

Cliche, M. (2017), 'Bb_twtr at semeval-2017 task 4: twitter sentiment analysis with cnns and lstms', *arXiv preprint arXiv:1704.06125* .

Discover 5 Leading Companies That Use Python and Learn Does It Fit Your Project (n.d.).
URL: *https://www.cleveroad.com/blog/discover-5-leading-companies-that-use-python-and-learn-does-it-fit-your-project*

Facebook users worldwide 2018 (n.d.).
URL: *https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide*

General Python FAQ (n.d.).
URL: *https://docs.python.org/2/faq/general.html*

Git (2018).
URL: *https://git-scm.com/*

Go, A., Bhayani, R. and Huang, L. (2009), 'Twitter sentiment classification using distant supervision', *CS224N Project Report, Stanford* **1**(12), 2009.

Google (2019).
URL: *https://trends.google.com/trends/explore?date=today 5-yq=word2vec,ngram,n gram*

Hughes, D. J., Rowe, M., Batey, M. and Lee, A. (2012), 'A tale of two sites: Twitter vs. facebook and the personality predictors of social media usage', *Computers in Human Behavior* **28**(2), 561–569.

Iacobacci, I., Pilehvar, M. T. and Navigli, R. (2015), Sensembed: Learning sense embeddings for word and relational similarity, *in* 'Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)', Vol. 1, pp. 95–105.

Institute, P. M. (2004), *A Guide to the Project Management Body of Knowledge, Third Edition (PMBOK Guides)*, Project Management Inst.
URL: *https://www.amazon.com/Guide-Project-Management-Knowledge-Guides/dp/193069945X?SubscriptionId=AKIAIOBINVZYXZQZ2U3Atag=chimbori05-20linkCode=xm2camp=2025creative=165953creativeASIN=193069945X*

Jain, N. (2015), 'Review of different responsive css front-end frameworks', *Journal of Global Research in Computer Science* **5**(11), 5–10.

Jo, H., Kim, S.-M. and Ryu, J. (2017), 'What we really want to find by sentiment analysis: The relationship between computational models and psychological state', *arXiv preprint arXiv:1704.03407* .

Joslin, R. and Müller, R. (2015), 'Relationships between a project management methodology and project success in different project governance contexts', *International Journal of Project Management* **33**(6), 1377–1392.

Kerzner, H. and Kerzner, H. R. (2017), *Project management: a systems approach to planning, scheduling, and controlling*, John Wiley & Sons.

Kumar, R., Pannu, H. S. and Malhi, A. K. (2019), 'Aspect-based sentiment analysis using deep networks and stochastic optimization', *Neural Computing and Applications* pp. 1–15.

Lakein, A. and Leake, P. (1973), *How to get control of your time and your life*, New American Library New York.

Lei, T., Zhang, Y. and Artzi, Y. (2017), 'Training rnns as fast as cnns', *arXiv preprint arXiv:1709.02755* .

Ling, W., Tsvetkov, Y., Amir, S., Fermandez, R., Dyer, C., Black, A. W., Trancoso, I. and Lin, C.-C. (2015), Not all contexts are created equal: Better word representations with variable attention, *in* 'Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing', pp. 1367–1372.

Liu, B. (2012), 'Sentiment analysis and opinion mining', *Synthesis lectures on human language technologies* **5**(1), 1–167.

McKinney, W. (2012), *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*, " O'Reilly Media, Inc.".

Meeks, E. (2015), *D3. js in Action*, Manning Shelter Island, NY.

Merabti, T., Abdoune, H., Letord, C., Sakji, S., Joubert, M. and Darmoni, S. J. (2011), 'Mapping the atc classification to the umls metathesaurus: some pragmatic applications.', *Studies in health technology and informatics* **166**, 206–213.

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013), 'Efficient estimation of word representations in vector space', *arXiv preprint arXiv:1301.3781* .

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C. and Joulin, A. (2017), 'Advances in pre-training distributed word representations', *arXiv preprint arXiv:1712.09405* .

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. (2013), Distributed representations of words and phrases and their compositionality, *in* 'Advances in neural information processing systems', pp. 3111–3119.

Myers, G. J., Badgett, T., Thomas, T. M. and Sandler, C. (2004), *The art of software testing*, Vol. 2, Wiley Online Library.

Oliphant, T. E. (2006), *A guide to NumPy*, Vol. 1, Trelgol Publishing USA.

Otoo, M. W. (1999), 'Consumer sentiment and the stock market'.

Pak, A. and Paroubek, P. (2010), Twitter as a corpus for sentiment analysis and opinion mining., *in* 'LREc', Vol. 10, pp. 1320–1326.

Pang, B., Lee, L. and Vaithyanathan, S. (2002), Thumbs up?: sentiment classification using machine learning techniques, *in* 'Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10', Association for Computational Linguistics, pp. 79–86.

Perone, C. S., Silveira, R. and Paula, T. S. (2018), 'Evaluation of sentence embeddings in downstream and linguistic probing tasks', *arXiv preprint arXiv:1806.06259* .

Perry, W. E. (1989), *A Standard for Testing Application Software, 1990*, Auerbach Publishers.

Python (n.d.), 'python/cpython'.
URL: *https://github.com/python/cpython/blob/master/LICENSE*

Ramisch, C. (2008), 'N-gram models for language detection', *M2R Informatique-Double diplome ENSIMAG–UJF/UFRIMA* .

Schnabel, T., Labutov, I., Mimno, D. and Joachims, T. (2015), Evaluation methods for unsupervised word embeddings, *in* 'Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing', pp. 298–307.

Singal, M. (2012), 'Effect of consumer sentiment on hospitality expenditures and stock returns', *International Journal of Hospitality Management* **31**(2), 511–521.

Singh, P. K. and Husain, M. S. (2014), 'Methodological study of opinion mining and sentiment analysis techniques', *International Journal on Soft Computing* **5**(1), 11.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. and Potts, C. (2013), Recursive deep models for semantic compositionality over a sentiment treebank, *in* 'Proceedings of the 2013 conference on empirical methods in natural language processing', pp. 1631–1642.

Souza, T. T. P., Kolchyna, O., Treleaven, P. C. and Aste, T. (2015), 'Twitter sentiment analysis applied to finance: A case study in the retail industry', *arXiv preprint arXiv:1507.00784* .

Stanford (2019).
URL: *http://cs231n.github.io/convolutional-networks/*

Tai, K. S., Socher, R. and Manning, C. D. (2015), 'Improved semantic representations from tree-structured long short-term memory networks', *arXiv preprint arXiv:1503.00075* .

Talkwalker.com (n.d.), 'Sentiment analysis just got smarter'.
URL: *https://www.talkwalker.com/blog/sentiment-analysis-just-got-smarter*

Vuurens, J. B., Eickhoff, C. and de Vries, A. P. (2016), 'Efficient parallel learning of word2vec', *arXiv preprint arXiv:1606.07822* .

Wang, X., Zhang, C., Ji, Y., Sun, L., Wu, L. and Bao, Z. (2013), A depression detection model based on sentiment analysis in micro-blog social network, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer, pp. 201–213.

What happens to Debenhams now? (2019).
URL: *https://www.bbc.co.uk/news/business-47868020*

White, D. and Fortune, J. (2002), 'Current practice in project managementan empirical study', *International journal of project management* **20**(1), 1–11.

Yin, W., Kann, K., Yu, M. and Schütze, H. (2017), 'Comparative study of cnn and rnn for natural language processing', *arXiv preprint arXiv:1702.01923* .

Yu, L.-C., Lee, L.-H. and Wong, K.-F. (2016), Overview of the ialp 2016 shared task on dimensional sentiment analysis for chinese words, *in* '2016 International Conference on Asian Language Processing (IALP)', IEEE, pp. 156–160.

Zhang, L., Wang, S. and Liu, B. (2018), ‘Deep learning for sentiment analysis: A survey’, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **8**(4), e1253.