Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección Académica

Sentiment Analysis through Conversational Data

por

Alexander Espronceda Gómez

como requisito parcial para obtener el grado de

ingeniería en tecnología de software

Febrero 2022

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección Académica

Sentiment Analysis through Conversational Data

por

Alexander Espronceda Gómez

como requisito parcial para obtener el grado de

ingeniería en tecnología de software

Febrero 2022

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Subdirección Académica

Los miembros del Comité de Tesis recomendamos que la Tesis «Sentiment Analysis through Conversational Data», realizada por el alumno Alexander Espronceda Gómez, con número de matrícula 1742000, sea aceptada para su defensa como requisito parcial para obtener el grado de Ingeniería en Tecnología de Software.

El Comité de Tesis

Dra. Satu Elisa Schaeffer
Asesora

Dra. Sara Elena Garza Villarreal	Dr. Romeo Sánchez Nigenda
Revisora	Revisor

Vo. Bo.

Dr. Fernando Banda Muñoz
Subdirección Académica

San Nicolás de los Garza, Nuevo León, febrero 2022

Contents

List of Figures

List of Tables

Agradecimientos

Me gustaría agradecerle a la Dra. Satu Elisa Schaeffer por haberme apoyado a realizar este proyecto después de tanto tiempo, si no la hubiera conocido en clase y me hubiera vuelto su becario probablemente no me habría animado a trabajar en un proyecto así. También a Cecilia Jael Aguilar Aranda, quien, como participación en un verano científico, me apoyó en búsqueda de conjuntos de datos relevantes para utilizar en este proyecto.

A mi padre José Artemio Espronceda Estrada, quien nunca ha dejado de apoyarme, aún en los momentos difíciles. A mi madre Yadhira Lizet Gómez García, quien me dió la inspiración para trabajar en este proyecto. Y a mi hermana Abigail por mantenerme aterrizado en los momentos donde nada parecía tener sentido.

A mis amigos Alfonso, Luis, Anna, Jorge y Oscar, por ofrecerme el apoyo que no tendría de otra manera y acompañarme en las buenas y en las malas.

Resumen

Alexander Espronceda Gómez.

Candidato para obtener el grado de Ingeniería en Tecnología de Software.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: Sentiment Analysis through Conversational Data.

Número de páginas: 52.

Objetivos y Método de Estudio: En esta tesis se propone generar un software conversacional que interprete el texto introducido por un usuario y determinar su estado de ánimo.

El método de estudio utilizado hará un análisis comprensivo de las redes neuronales, así como también de reconocimiento de patrones y recopilación de datos que nos dé comprensión moderada de algo tan voluble y a veces impredecible como lo es la mente humana.

Contribuciones y Conclusiones: El algoritmo de entrenamiento utiliza un conjunto de datos específico para predecir, dentro de lo posible, qué está sintiendo una persona al momento de escribir alguna oración o frase. El algoritmo es open-source por lo que cualquier persona puede añadir o quitar módulos según se requiera.

La conclusión de esta tesis es que es posible que un algoritmo de Red Neuronal Recurrente reconozca los patrones de una frase, pero se tiene que tener un conjunto de datos limpio, bien distribuido y que incluya palabras que sean difíciles de clasificar erróneamente.

Firma de la asesora: ______________________________
Dra. Satu Elisa Schaeffer

Abstract

Alexander Espronceda Gómez.

As a candidate to obtain the degree of Software Technology Engineering.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Title of the study: Sentiment Analysis through Conversational Data.

Number of pages: 52.

Objectives and Study Methods: In this thesis a conversational software is proposed, which interprets the text entered by a person and determines how they are feeling at the moment.

The study method used will make a comprehensive analysis of neural networks, as well as pattern recognition and data collection that will give us a moderately sufficient understanding of something as sometimes unpredictable as the human mind.

CONTRIBUTIONS AND CONCLUSIONS: The training algorithm uses a specific dataset to predict, as accurately as possible, what a person is feeling when writing a sentence or phrase. The algorithm is open-source so anyone can add or remove modules as needed.

This thesis' conclusion is that it is possible for a Recurrent Neural Network algorithm to recognize patterns in a sentence, but one has to have a filtered, well-distributed dataset that includes words that are difficult to misclassify.

Firma de la asesora: ____________________________
Dra. Satu Elisa Schaeffer

Chapter 1

Introduction

Human beings are social beings, this is widely known. To survive, we must band together and communicate with each other, bonding in the process. This is thanks to a neural process called *empathy*, which is defined as a three-part process that happens in our brains [12]. That happens roughly like this:

- Emotional simulation centered in the limbic system, which makes us mirror the emotional elements we're watching.
- Processing the perspective in the prefrontal and temporal cortex.
- Assessing the course of action to take, either showing compassion or doing something else. This is assumed to be based in the obitofrontal cortex, as well as several other parts of the brain.

This is clearly what is usually considered a human-only behavior, but there are studies that indicate that apes, dogs and rodents have been observed to take action at the presence of distress signals, either from humans or other members of their own species [20]. If this is true, theoretically, a machine could be taught to process signals of distress and react accordingly using a learning algorithm.

Sentiment Analysis through Conversational Data

1.1 Justification

At first, the objective was to create an algorithm that could serve as a makeshift therapy chatbot that people could use when they were confused about their own feelings, but as time has passed, a lot of things have happened in my life regarding people with close-to-none empathy. This project could prove especially useful towards people who have trouble discerning when to console someone or having an idea of how other people or even themselves feel, such as the case of people with Asperger's Syndrome or other forms of high-functioning autism. To this end, the decision was made to work on this project.

1.2 Hypothesis

Empathy consists in a pattern of neurochemical reactions triggered by different situations. Machine learning could learn to identify these patterns. The hypotesis of this thesis is that machine learning could help people with a vague sense of empathy or self-knowledge to discern what they are feeling.

1.3 Objectives

In this section, the objectives proposed for this thesis are established.

1.3.1 General Objectives

The objective of this project is to determine how the person that writes the input text is feeling according to the words in it. This could be achieved thanks to the technology present in machine learning algorithms and an extensive amount of datasets.

1.3.2 Specific Objectives

- Generating an algorithm capable of detecting key words related to mood in text.
- Predicting successfully the mood according to the input given.
- Giving feedback on the input, reinforcing it if positive or giving empathetic words if negative.

1.4 Metodology

The tools that are used in this thesis are mostly Python-based, such as TensorFlow, a neural network framework. This, combined with natural language processing tools and several filtering techniques will be used to achieve – or at least approach as close as possible to – the expected results.

1.5 Structure

The content in this thesis is divided in several chapters, each one of them talking about different information about either the topics that are relevant to the scope of this project or the general process that has happened to reach the goal.

In the second chapter, relevant concepts are discussed and expanded upon for better understanding of what this project's purpose. In the third chapter, existing literature is analyzed and compared to the present work, with comprehensive information and related concepts applied to each one of them.

In the fourth chapter, the project's design, inner workings, and the tools used are described. In the fifth chapter, the inputs and outputs are described, and several experiments are conducted and analyzed for a better understanding of the role of every component of the algorithm.

Chapter 2

Background

Technology in the past decades has been advancing exponentially. So much, in fact, that we can relegate data analysis to them for better accuracy and reliability than what a human can possibly achieve. This is what is called as Machine Learning (sometimes referred only as ML). There is a variety of scenarios where it is useful, such as pattern recognition, which relates extensively to most of this project's work. In this chapter, some key concepts will be explained for easier comprehension of this thesis and the project itself as a whole.

2.1 Machine Learning

Machine learning can be described, broadly and figuratively speaking, as a black box where some data is inserted as an input and numbers come out of it as an output [24]. Some more advanced models of ML allow some internal parameters inside this figurative black box to be able to be tampered with, so that some characteristics of the input data can have effect on the output, these parameters are called *weights* [3]. Most ML algorithms have two stages: training and validation:

- Training processes the inputs and makes educated guesses, and in case of guessing incorrectly, depending on the obtained result, the weights are changed accordingly.

- Validation is as simple as it sounds, some input is fed to the algorithm and information needs to be compared to the real results to test the accuracy percentage.

One of these models that is one of the most used nowadays is the one called *Neural Network*.

2.1.1 Neural Network

A neural network works by using *neurons* that utilize layers that individually weigh the input given to them from the initial text or, if this has been processed already, from another neuron [3]. Likewise, similar to how biological brains work, these algorithms can only predict reliably if given enough data to train and validate their outputs with.

2.2 Sentiment Analysis

Sentiment Analysis (or Opinion Mining, as it is also known) as a tool for data analysis is arguably a recent happening. The term was coined in 2003 and has evolved ever since [15]. This type of data analysis has a lot of potential usages that have yet to be implemented in the daily life.

2.2.1 Concept

The specific execution of the algorithm varies depending on the intended purpose, but the concept and process that is used is generally the same:

- The sentence to analyze is broken down to its component parts, this process is called *tokenization*, and the resulting products are called, fittingly, *tokens*.
- Every token is then tagged, making it part of an internal dictionary or *lexicon*
- A score is assigned to every token depending on the used dataset.

The end score could be left as-is or can be reintroduced to the algorithm for a multi-layered approach depending on its focus [2].

2.2.2 Tokenizing

Tokenizing is the process that happens while making tokens, the way it works is very straightforward: every word in the lexicon that a machine can read is assigned a number for easier reading. Taking the following example:

$$\boxed{\text{This is an example text}}$$

We can tell there are 6 words in the example phrase. So the tokenizing process would make the example look in the following way:

$$\boxed{1, 2, 3, 4, 5, 6}$$

where 1 corresponds to the word "This", 2 corresponds to "is", 3 to "an" and so on.

The interesting part about this process would happen if we used another example phrase, like the following:

$$\boxed{\text{This is another example}}$$

If we did the tokenization process, it would be processed in this way:

$\boxed{1, 2, 7, 4}$

Since the internal lexicon already knows some of the words in this second example, it reuses their token, adding new ones (in this example, "another" is 7) if needed.

This is fairly useful for a machine learning algorithm, since it will not have to compare such massive amount of characters in a string each time, and it would only need to evaluate integers. Whether its focus is either frequency or comparison.

Chapter 3

Related Work

The problem proposed in this thesis is not something new by a long stretch, since sentiment analysis was developed for this very purpose. There are many applications that already apply this kind of Machine Learning for several purposes. In this chapter, some related projects are listed and analyzed.

3.1 Related Projects

In this section, some literature is listed which proposes projects which have similar approaches to the present work, and some others that may not have the same objectives in mind but use algorithms that could be applied as well.

3.1.1 Similar Approaches

Blenn et al. [7] describe three different text classificators with a focus on sentiment analysis from Twitter:

- Twitter Sentiment, which uses a Maximum Entropy algorithm[1].
- Tweet Sentiments, which uses Support Vector Machines[2] for classifications.
- Lingpipe, which uses both previous algorithms and also Naive Bayes[3]

Morris et al. [17] mention Koko, which uses the OpenAI API which is a counseling app for distressed teenagers in need of immediate psychological support, composed of a chatbot and sentiment analysis capabilities while Bird et al. [6] propose a chatbot developed to comprehend instructions, classifying them internally with a predefined bank of words, and reacting accordingly.

3.1.2 Sentiment Analysis in Other Areas

Pang et al. [19] draft out a movie review algorithm that was capable of detecting if the review was either positive or negative depending on the words used, and, Wang et al. [23] propose an algorithm that correlated the air pollution levels with the sentiment expressed in people's tweets. Capuano et al. [9] mention a hierarchical attention network to detect the polarity of a customer's review, with the added bonus of being capable of learning from new data. Chiril et al. [10] propose an algorithm that can detect hate speech in text using natural language text classification across several topics. Ahmad et al. [1] write about a classification system to detect if a tweet was deemed as extremist or non-extremist depending on the vocabulary used and a deep-learning algorithm. Similarly, Röchert et al. [21] report a Recurrent

[1]This algorithm works by having the bias that certain characteristics repeat more in certain categories in text. If no bias is found, the distribution is uniform [18].

[2]Binary algorithm that can sort between two classes, or opt for classification in a "one-versus-everything else" basis [22].

[3]This algorithm utilizes weights expressed in *-1*, *0*, or *+1* depending on the sensitivity of specific characteristics [5]. Works very similarly to a classic perceptron, which only uses *0* or *1*.

Neural Network that detect political statements in YouTube comments while also classifying them in *positive*, *negative*, or *other* depending on the topic.

3.2 Comparative Analysis

Since the projects included in this chapter are all focused in the same branch of algorithm, they have some concepts in common with each other and, in turn, with this project. Some of them are:

Machine Learning The type of algorithm needed for automatic processing, making the machine "learn" (hence the name) over time given enough data.

Neural Network A Machine Learning algorithm that uses weights and filters to output data.

Weights In ML, this is the name given to the internal value that a specific input has after being analyzed by the algorithm. With this, data classification can be achieved.

Text Processing Any type of algorithm that can understand text and output data based on its contents.

Natural Language Processing This is the method used for the algorithm to understand the content of the sentences, this is usually achieved by using tokenization but a preset corpus can also be used.

Sentiment Analysis This involves a ML algorithm, usually a Neural Network, that is able to analyze sentences and classify them according to the words used.

Corpus Preset internal dictionary that the algorithm uses.

Chatbot An algorithm that is able to reply to a prompt using natural language.

3.2.1 Opportunities for Improvement

One of the main positives of working with TensorFlow is the fact that it is a highly reusable code that can very much be ported to any system that can run Python.
It is important to mention GPT-3 as a whole, the framework that Koko – mentioned by Morris et al. [17] – uses is, to date, one of the most impressive AI algorithm to be developed, the downsides being that, being still in beta phase, is very resource-heavy, and its access is reserved to businesses through a fee, very expensive to use for the general public, especially students. That is why in this project, TensorFlow is used, which is free to use, does not need a lot of resources to work and has the advantages of being portable once trained, and also being easily modifiable if needed.

Table 3.1: Comparison between existing literature and the present work: ✓indicates the fulfillment of a criterion, otherwise × is used.

Project	**Neural Network**	**Text Processing**	**Sentiment Analysis**	**Chatbot**	**Open Source**
Blenn et al. [7] Maximum Entropy	✓	✓	✓	×	×
Blenn et al. [7] Support Vector Machines	✓	✓	✓	×	×
Blenn et al. [7] Lingpipe	✓	✓	✓	×	×
Morris et al. [17]	✓	✓	✓	✓	×
Bird et al. [6]	✓	✓	×	✓	✓
Pang et al. [19]	✓	✓	✓	×	✓
Ahmad et al. [1]	✓	✓	×	×	✓
Wang et al. [23]	✓	✓	✓	×	✓
Capuano et al. [9]	✓	✓	✓	×	×
Chiril et al. [10]	✓	✓	×	×	✓
Röchert et al. [21]	✓	✓	✓	×	✓
The present work	✓	✓	✓	✓	✓

Chapter 4

Project Design

The tools that are used in this paper are mostly Python-based, such as TensorFlow, a neural network framework. This, combined with natural language processing tools and several filtering techniques will be used to achieve – or at least approach as close as possible to – the expected results. Having all the concepts in mind, the proposed project has a major component which is the Machine Learning tools surrounded by several small modules such as the GUI and the chatbot components. As for the data used in this project, most of it comes from cleaned, classified tweets partitioned in training and testing datasets. In this chapter, the design of the project is explained.

4.1 Inner Workings Design

In this section, the data and the relations with the algorithm is explained, with a focus on the design itself.

4.1.1 Datasets

The datasets used in this project, as previously mentioned, consist in around 40,000 semi-clean, classified tweets in 13 categories [11], but as the scope for all of those labels exceeds the one proposed in this paper, the ones taken into consideration are as follows:

- Sadness
- Neutral
- Happiness
- Fun
- Worry
- Boredom

Even after eliminating non-critical labels, since the remaining labeled samples are not evenly distributed, leaving them as-is led to very inaccurate results, so a generalistic approach was opted for, classifying the end results in "Good", "Neutral" and "Bad" depending on the overall wellness percieved from the input. This final filter works only with the training data, and works as follows:

- Sadness and Worry are in the "Bad" category.
- Neutral and Boredom are in the "Neutral" category.
- Happiness and Fun are in the "Good" category.

4.1.2 Text Filtering

Since the chosen dataset is imported almost straight from Twitter with poor grammar, misplaced symbols, emojis and similar things, some cleanup has to be done to ensure peak performance.

- First, all text must be converted to lowercase.

- Then, all of the punctuation marks had to be discarded.
- After that, the stopwords[1] have to be omitted as well.
- Finally, for easier analysis, a process called stemming[2] is applied, so that all of the tenses of every verb are evaluated the same way while also avoiding corpus bloating.

These last processes were possible thanks to NLTK[3], which has its own repository of stopwords and stems. An example for this applied to data in the training dataset is as follows.

So sleepy again and it's not even that late. I fail once again.

Following the filtering order, first all the characters are converted to lowercase.

so sleepy again and it's not even that late. i fail once again.

After that, the text is stripped from all non-alphabetic characters.

so sleepy again and it s not even that late i fail once again

Next, all stopwords are culled from the sentence.

sleepy even late fail

The last step is to apply stemming to all the able remaining words, in this case, the adjective "sleepy" stems from sleep.

sleepi even late fail

[1]Words that are not vital for the sentence's meaning.

[2]Reducing a verb to its most basic components.

[3]Natural Language Toolkit, tool used specifically for these case scenarios. `https://www.nltk.org/`

4.1.3 Neural Network

For this project, as mentioned in Chapter 3, TensorFlow was opted for because of its characteristics such as being free to use, not needing a lot of resources to work and the advantages of being portable once trained. All of these traits are what makes this project unique and easily scalable. An LSTM Neural Network was opted for because of the increased accuracy that it offers compared to a regular Recurrent Neural Network.

Basically, LSTM is a subtype of a Recurrent Neural Network which has a certain amount of data be stored for longer periods of time so it can be used for future connections.

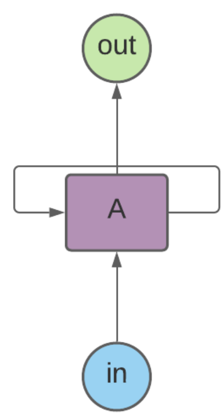

Figure 4.1: Basic Structure of a Recurrent Neural Network, where A represents the algorithm used.

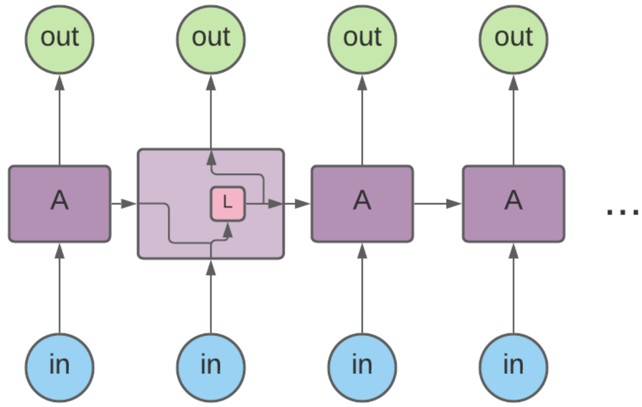

Figure 4.2: Structure inside an algorithm in a basic Recurrent Neural Network, where L represents the layers used.

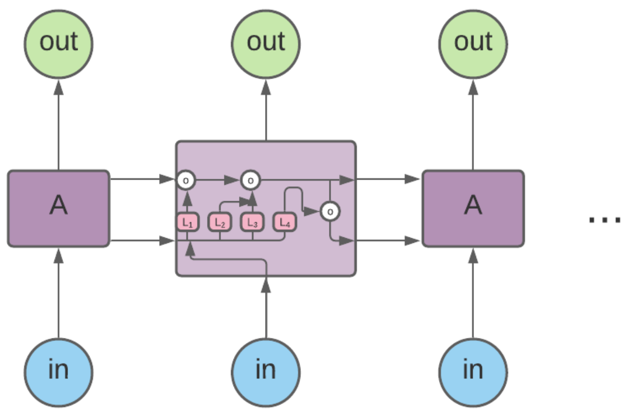

Figure 4.3: Structure inside an LSTM Neural Network, where the o represent the functions that operate the data the data when traveling from one layer to another [4].

4.2 TOOLS

This project is built on Python v3.8.10, The libraries used for this project to come to fruition are TensorFlow[4] v2.6.0 and Keras[5] v2.6.0 for the Neural Network section.

[4]`https://www.tensorflow.org/`

[5]`https://keras.io/`

Natural Language Toolkit[6] v3.5 (also known as NLTK) for the tokenization and stemming process. Chatterbot[7] for the chatbot's output. And, lastly, pygame[8] v1.9.5 for the GUI.

4.3 Inner Workings

In this section, I will highlight the most important parts of this project's code and their function. In case of needing further insight on the code used, the repository is online at `https://github.com/Alex-Ego/Affective-Computing-VN`.

4.3.1 Text Filtering

After the dataset has been properly located and ready to be used, the cleanup discussed earlier in this chapter happens in the following code snippet:

```
def tokenizing_process(message):
    # Pre-tokenizing
    tokens = word_tokenize(message)

    # Making them lowercase
    tokens = [w.lower() for w in tokens]

    # Filtering the punctuations
    table = str.maketrans('', '', string.punctuation)
    stripped = [w.translate(table) for w in tokens]

    # Filtering non-alphabetic characters
    words = [word for word in stripped if word.isalpha()]
```

[6]`https://www.nltk.org/`

[7]`https://chatterbot.readthedocs.io/en/stable/`

[8]`https://www.pygame.org/news`

```

    # Removing stopwords
    stop_words = set(stopwords.words('english'))
    words = [w for w in words if not w in stop_words]

    # Stemming words
    porter = PorterStemmer()
    stemmed = [porter.stem(word) for word in words]

    # Joining the resulting string
    message = " ".join(stemmed)
    return message
```

This works in the same way and order as specified earlier in this chapter.

4.3.2 Neural Network

After the text has been properly classified and ready-to-test, this is the structure of the neural network.

```
model = tf.keras.Sequential([
    layers.Embedding(input_dim=vocab_size,
                            output_dim=embedding_dim,
                            input_length=max_length),
    layers.SpatialDropout1D(0.15),
    layers.Bidirectional(layers.LSTM(32, dropout=0.15,
recurrent_dropout=0.15)),
    layers.Dense(8, activation="tanh"),
    layers.Dense(4, activation="softmax")
])
```

As the code shown indicates, this neural network has 3 layers: LSTM for classification, and two Dense, the former is to slim down the input data from the previous layer, and the latter is for classification which can be one in 4 categories, which include the three previously discussed categories, and one reserved for error/unknown purposes.

4.3.3 Portability

Training the Neural Network every time is not needed, since there is a way to save the model in a *.hdf5* file and the corpus in a plain *.txt* file with the following code snippet:

```
model_location = os.path.join(abs_location, "nndata/model")
keras.models.save_model(model, model_location +
"/sentimental_analysis.hdf5")
with open(model_location + "/tokens.txt", "w") as f:
    f.write(tokenizer.to_json())
    f.close()
```

This is fairly useful, because training it every time is very time and resource consuming. Having this as an option opens the path for more applications in less powerful systems.

Chapter 5

Data Experiments

In this chapter, the parts that compose this project as well as their context are shown. Also, some experiments are demonstrated for comparison with different parameters that could affect this thesis' project's overall accuracy.

5.1 Inputs and Outputs

In previous chapters, it has been specified that this algorithm takes an text input and, according to its contents, a message is shown as an output. The breakdown is as follows.

5.1.1 Inputs

The input that is given is cleaned up and tokenized – as shown in Project Design –, this is then added to an internal corpus that has weights set for every word in it, effectively working as scores. Every word has a different score in every label, whether it is positive or negative. This score is added up and the highest final score will be the one that the algorithm will detect as the most probable for the text input. However, this has its caveats, small sentences are more likely to be miscategorized because one word can have different applications in the scope of this project, for a more accurate analysis a longer sentence must be written.

5.1.2 Outputs

Depending on the final score, the algorithm will choose a random sentence related to the detected sentiment, this is, as of the time of writing, very rudimentary, but the fact that it is built in Python this can be a building block for a more robust, context-conscious, reply system. In the training module, however, four extra values are part of the output as well: `Loss`, `Val_loss`, `Accuracy` and `Val_accuracy`. These values are standard in every Neural Network algorithm to observe how poorly the evaluation does within the training dataset, and what the accuracy percentage is, respectively. The `Val_` counterparts of these values are the same, but applied to the validation dataset.

5.2 Experiments

In this section, various experiments of this project are shown with varying training data and parameters with the respective accuracy and loss graphs. The purpose of these experiments is to determine if the parameters chosen for this project are optimal and, if not, correct them and know the reason behind the improvement. The parameters that could potentially have a great impact on the output of the classification – and therefore are the best to experiment with – are the following:

- Used datasets: This could influentiate the amount of words in the corpus and have a big impact on how some words are percieved
- Training epochs: How many loops does the algorithm go through before being considered fully trained, if this number is too high it could result in *overfitting*,

which is, in casual terms, the Neural Network equivalent of overthinking.

- Units in the LSTM layer: This unit system, albeit small in the overall scale of things, could make-or-break the algorithm if not tuned correctly.

- Categorized sentiments: Reducing the scope of the project could potentially benefit the overall accuracy of the remaining sentiments.

The amount of improvement with each experiment is shown with loss and accuracy graphs, which are evaluated every epoch the algorithm is trained. Lower loss and higher accuracy are preferred.

5.2.1 Experiment 1: Base Experiment

In this experiment, we look at the base version of this thesis' project.

Table 5.1: Experiment 1's defining characteristics.

Datasets Used	2: Gupta [14] and Mohammad [16]
Epochs	10
LSTM Layer	32 units
Categorized Sentiments as "Bad"	"Sadness", "Worry", and "Fear".
Categorized Sentiments as "Neutral"	"Neutral" and "Boredom".
Categorized Sentiments as "Good"	"Happiness", "Fun", "Joy", and "Love".

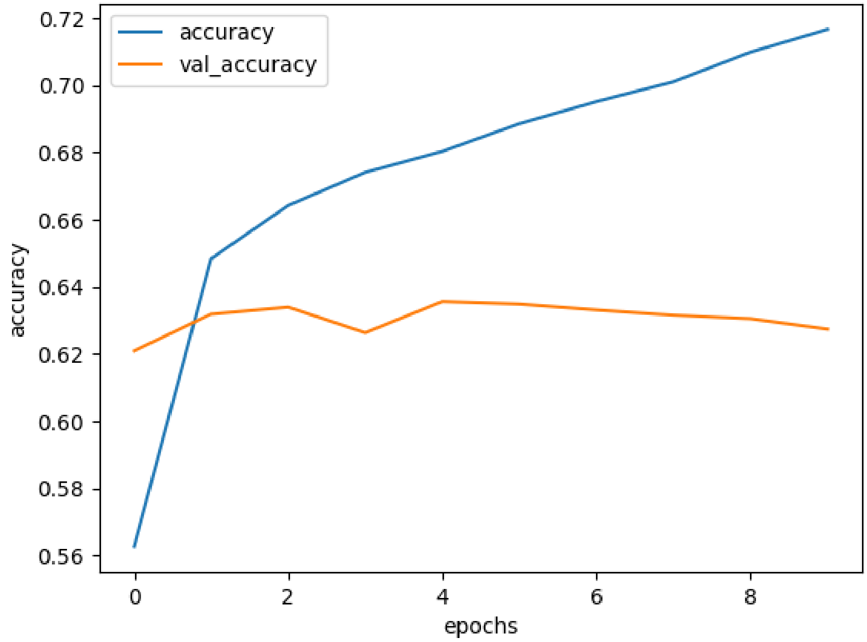

Figure 5.1: Accuracy Graph of Experiment 1

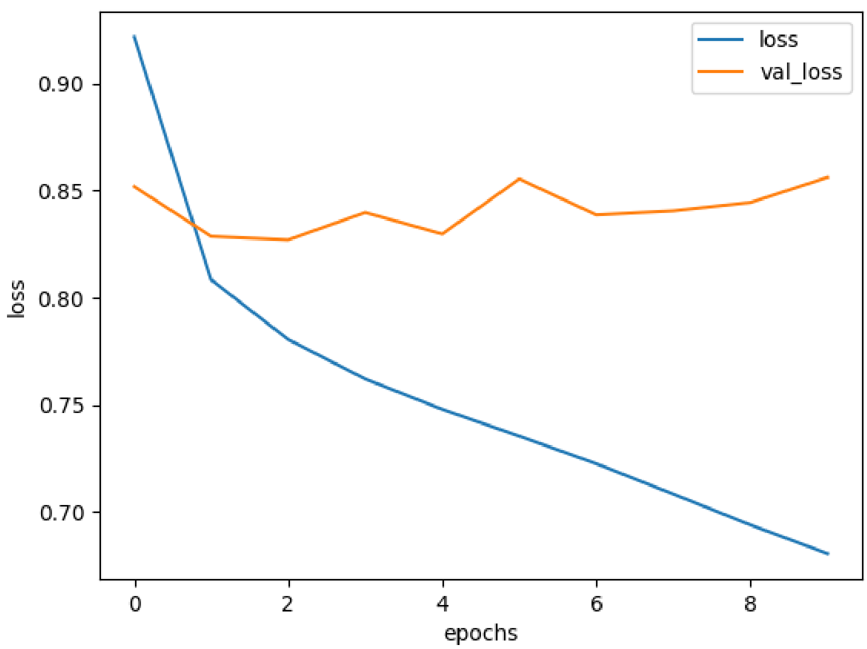

Figure 5.2: Loss Graph of Experiment 1

5.2.2 Experiment 2: More Datasets With Reduced Data Scope

This experiment takes sentences from one more dataset and 3 less categorized sentiments: “Fear”, “Joy”, and “Love”.

Table 5.2: Experiment 2’s defining characteristics.

Datasets Used	3: Gupta [14], Mohammad [16] and Govi [13]
Epochs	10
LSTM Layer	32 units
Categorized Sentiments as “Bad”	“Sadness”, and “Worry”.
Categorized Sentiments as “Neutral”	“Neutral” and “Boredom”.
Categorized Sentiments as “Good”	“Happiness”, and “Fun”.

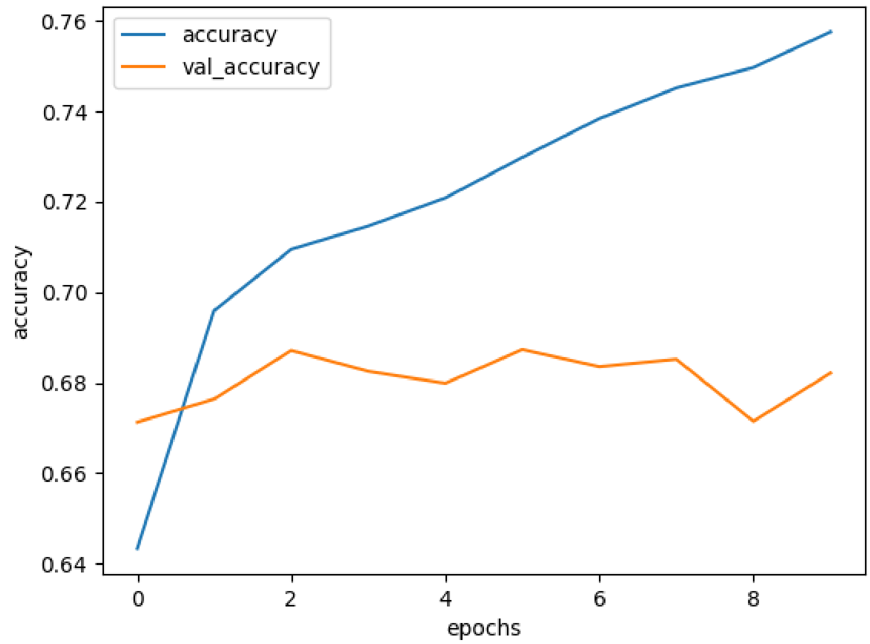

Figure 5.3: Accuracy Graph of Experiment 2

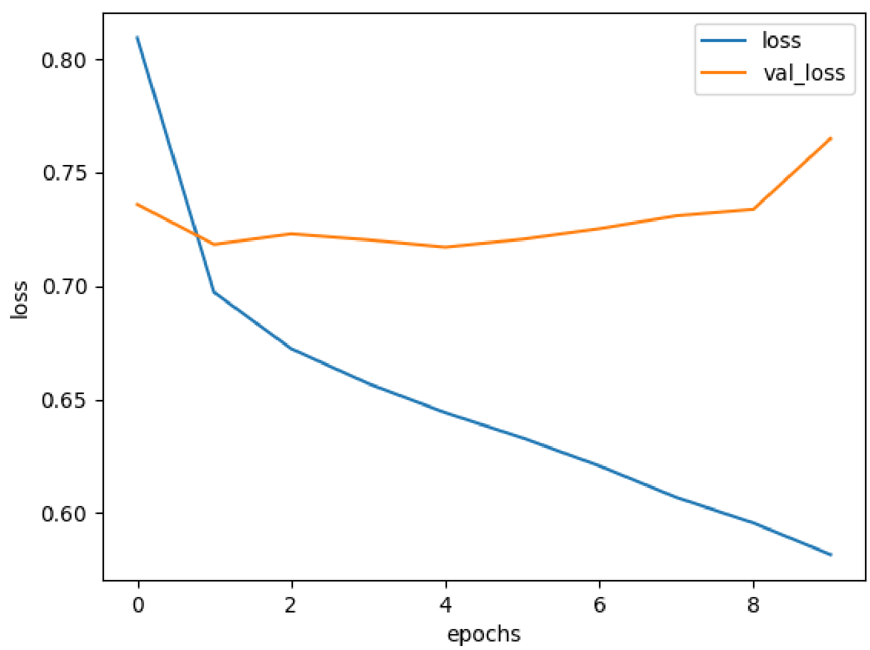

Figure 5.4: Loss Graph of Experiment 2

5.2.3 Experiment 3: Augmented LSTM Units

Largely the same as Experiment 2, but the LSTM has more units to work with.

Table 5.3: Experiment 3's defining characteristics.

Datasets Used	3: Gupta [14], Mohammad [16] and Govi [13]
Epochs	10
LSTM Layer	64 units
Categorized Sentiments as "Bad"	"Sadness", and "Worry".
Categorized Sentiments as "Neutral"	"Neutral" and "Boredom".
Categorized Sentiments as "Good"	"Happiness", and "Fun".

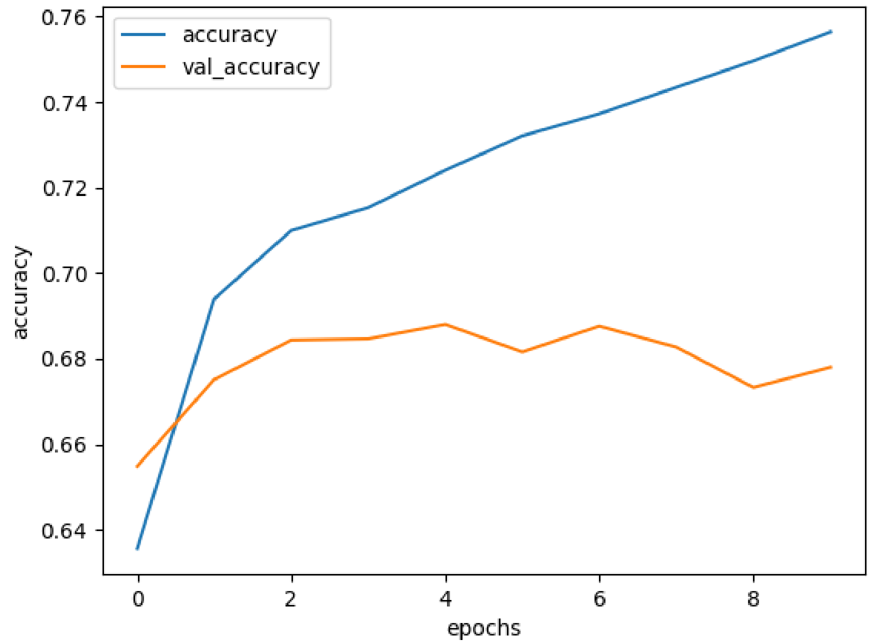

Figure 5.5: Accuracy Graph of Experiment 3

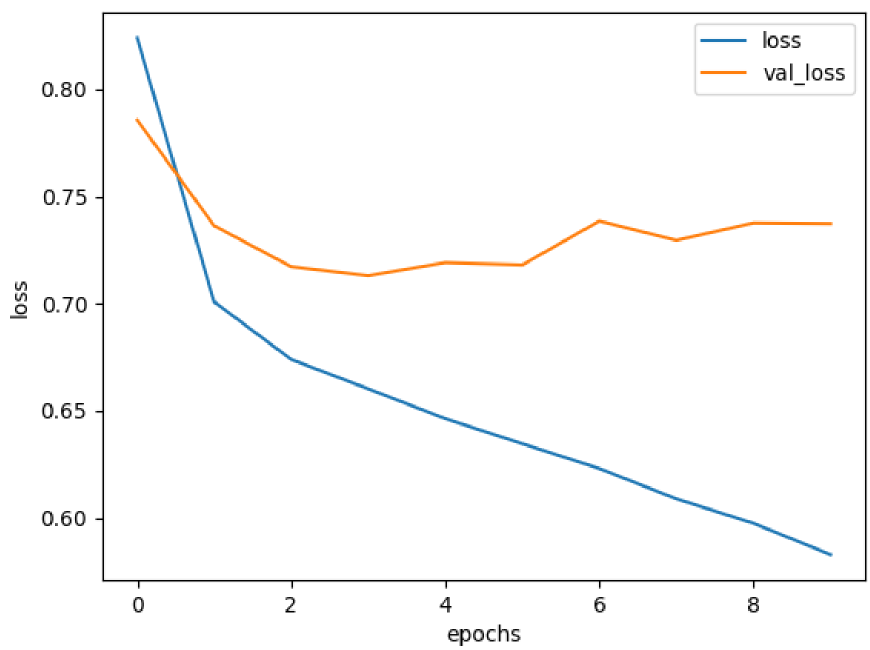

Figure 5.6: Loss Graph of Experiment 3

5.2.4 Experiment 4: Augmented Datasets Without Reduced Data Scope

A mix between Experiment 1 and 2. Three datasets with the full sentiment categorization and LSTM with 32 units.

Table 5.4: Experiment 4's defining characteristics.

Datasets Used	3: Gupta [14], Mohammad [16] and Govi [13]
Epochs	10
LSTM Layer	32 units
Categorized Sentiments as "Bad"	"Sadness", "Worry", and "Fear".
Categorized Sentiments as "Neutral"	"Neutral" and "Boredom".
Categorized Sentiments as "Good"	"Happiness", "Fun", "Joy", and "Love".

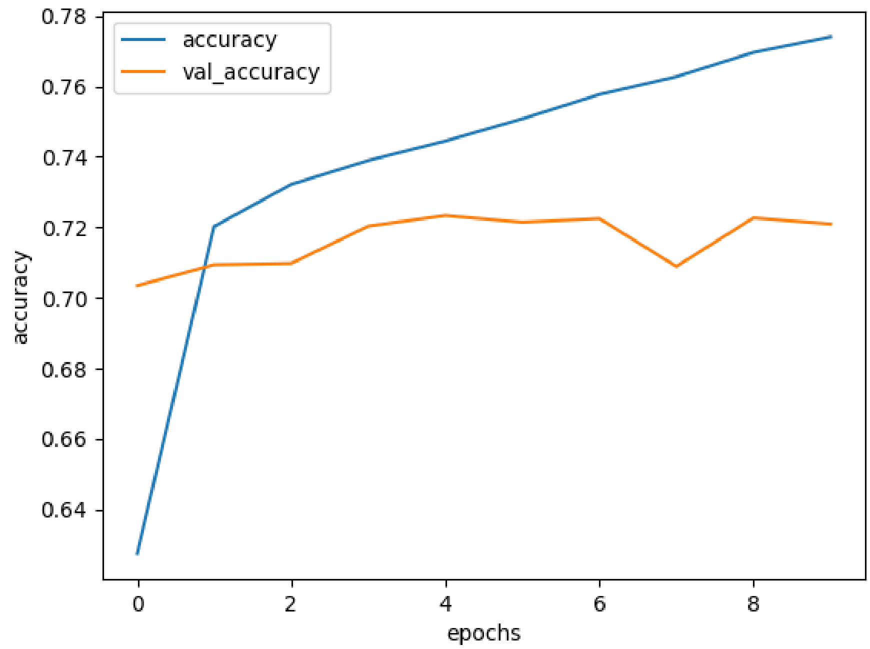

Figure 5.7: Accuracy Graph of Experiment 4

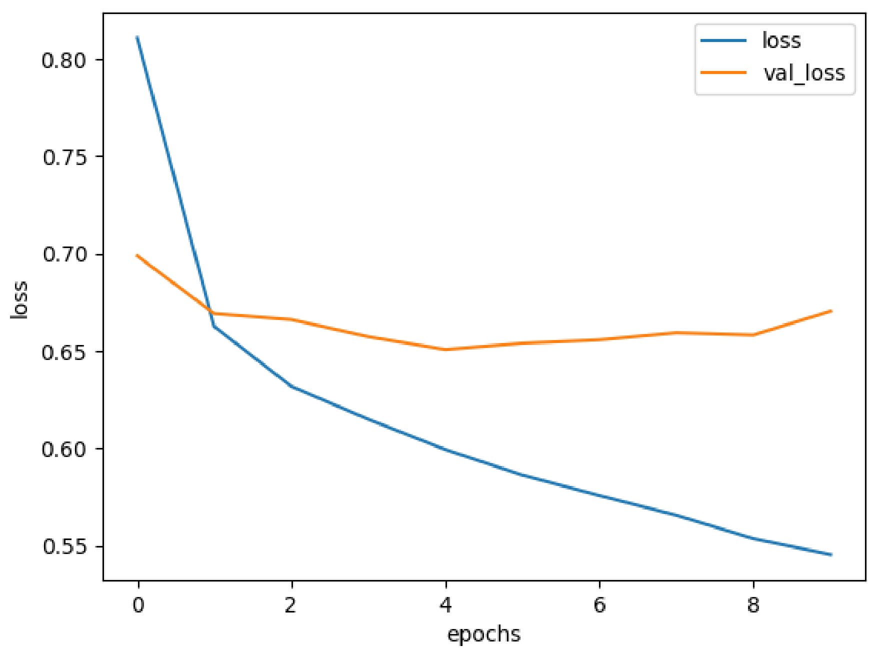

Figure 5.8: Loss Graph of Experiment 4

5.2.5 Experiment 5: Extra Sadness Dataset

Same as Experiment 4, but with an added dataset with only “Sadness” sentences.

Table 5.5: Experiment 5’s defining characteristics.

Datasets Used	4: Gupta [14], Mohammad [16], Govi [13], and Calefato et al. [8]
Epochs	10
LSTM Layer	32 units
Categorized Sentiments as “Bad”	“Sadness”, “Worry”, and “Fear”.
Categorized Sentiments as “Neutral”	“Neutral” and “Boredom”.
Categorized Sentiments as “Good”	“Happiness”, “Fun”, “Joy”, and “Love”.

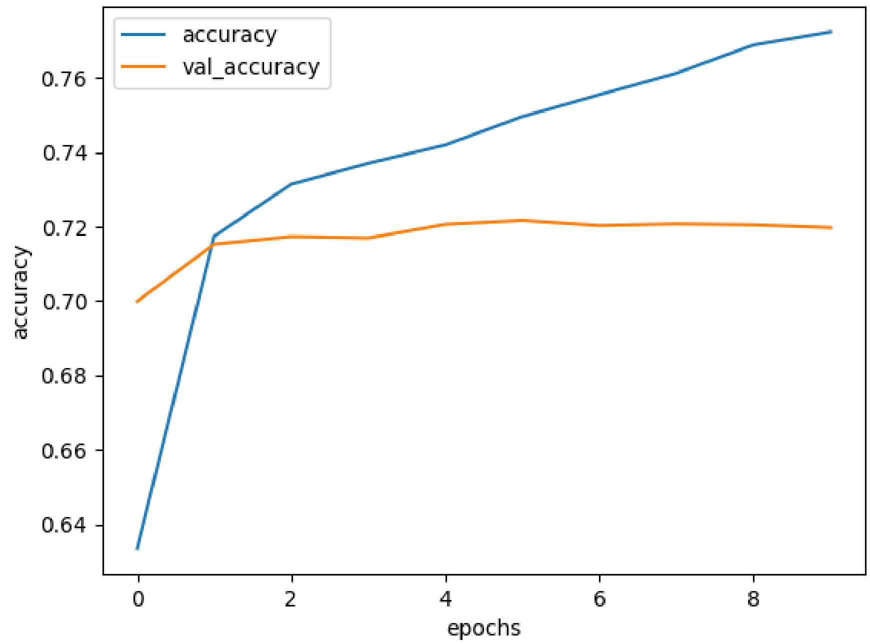

Figure 5.9: Accuracy Graph of Experiment 5

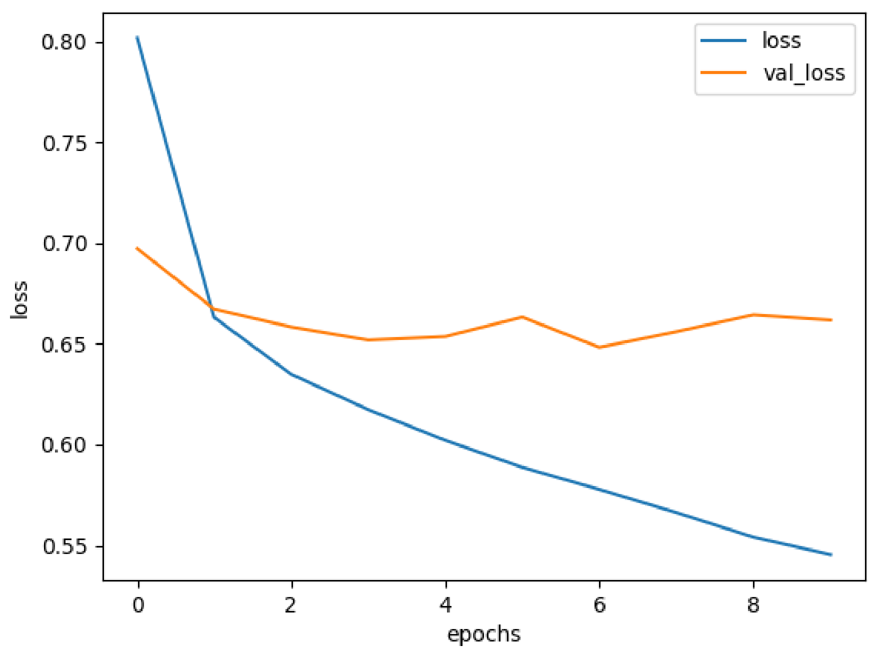

Figure 5.10: Loss Graph of Experiment 5

5.2.6 Experiment 6: Reduced Classification Scope

This is the largest change on an experiment, the "Neutral" category has been completely disabled with the purpose of seeing how the rest of the data would be classified as.

Table 5.6: Experiment 6's defining characteristics.

Datasets Used	4: Gupta [14], Mohammad [16], Govi [13], and Calefato et al. [8]
Epochs	10
LSTM Layer	32 units
Categorized Sentiments as "Bad"	"Sadness", "Worry", and "Fear".
Categorized Sentiments as "Neutral"	N/A
Categorized Sentiments as "Good"	"Happiness", "Fun", "Joy", and "Love".

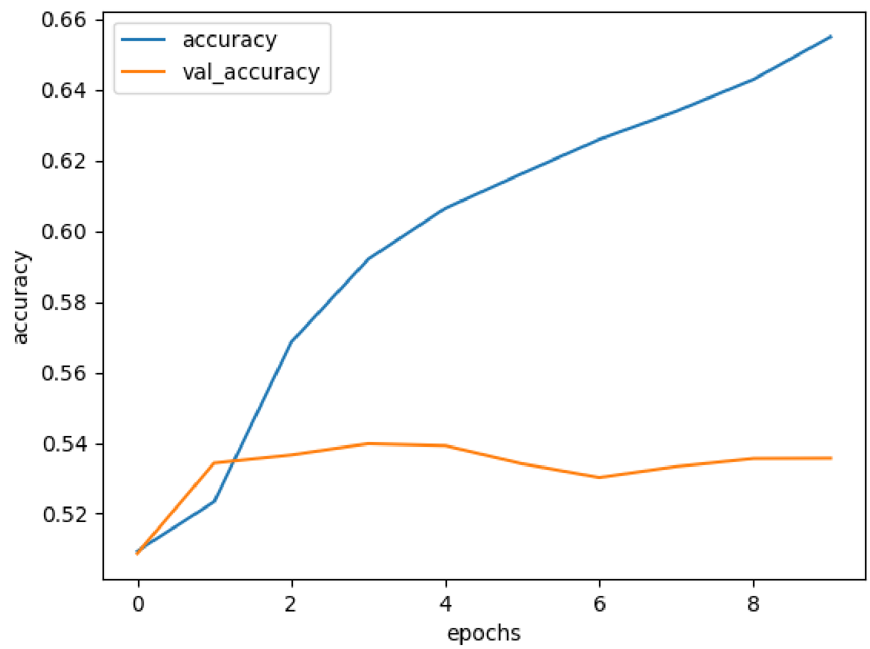

Figure 5.11: Accuracy Graph of Experiment 6

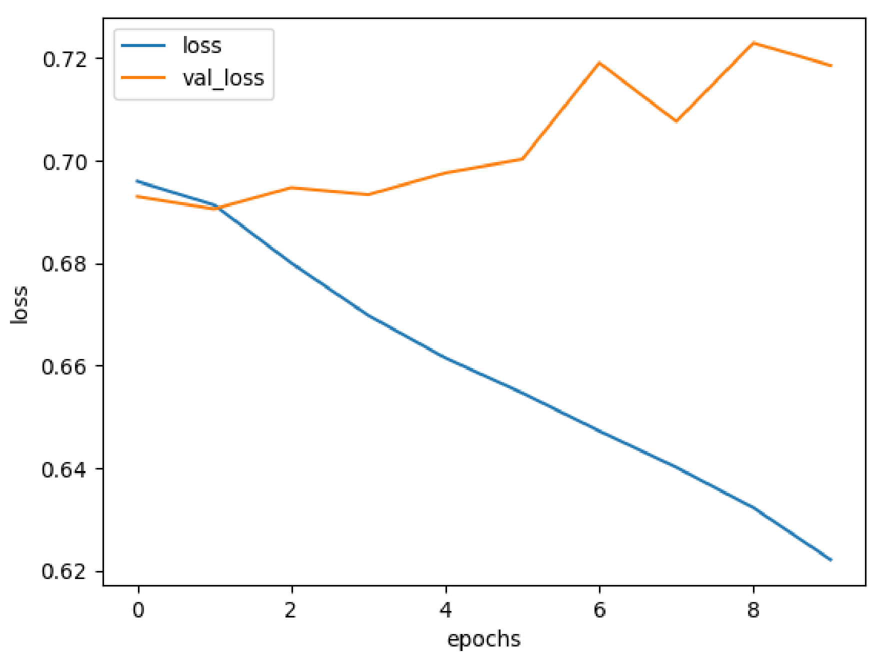

Figure 5.12: Loss Graph of Experiment 6

5.2.7 Experiment 7: Reduced Epochs

Largely the same as Experiment 5 with half the epochs. This with the purpose of seeing if the data has been overfit.

Table 5.7: Experiment 7's defining characteristics.

Datasets Used	4: Gupta [14], Mohammad [16], Govi [13], and Calefato et al. [8]
Epochs	5
LSTM Layer	32 units
Categorized Sentiments as "Bad"	"Sadness", "Worry", and "Fear".
Categorized Sentiments as "Neutral"	"Neutral" and "Boredom".
Categorized Sentiments as "Good"	"Happiness", "Fun", "Joy", and "Love".

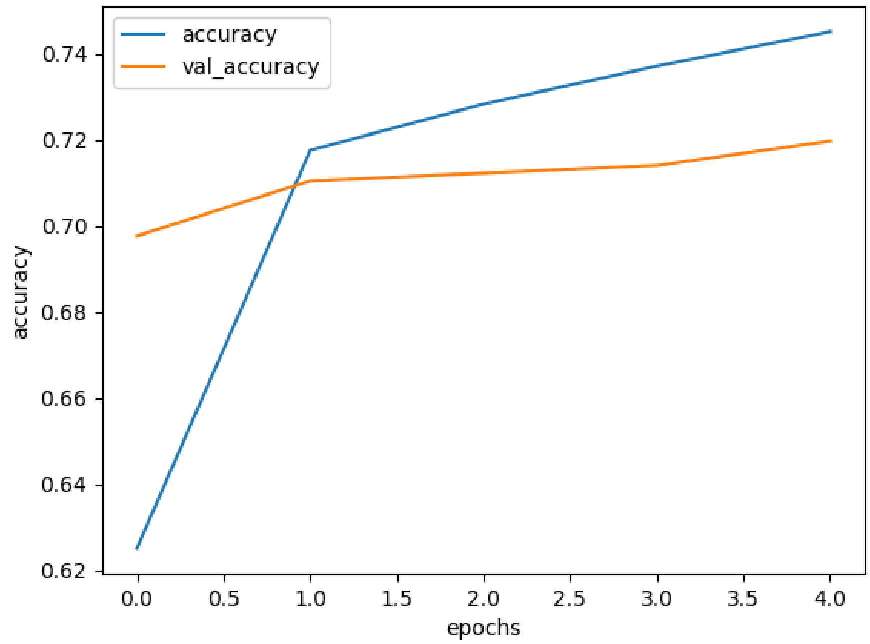

Figure 5.13: Accuracy Graph of Experiment 7

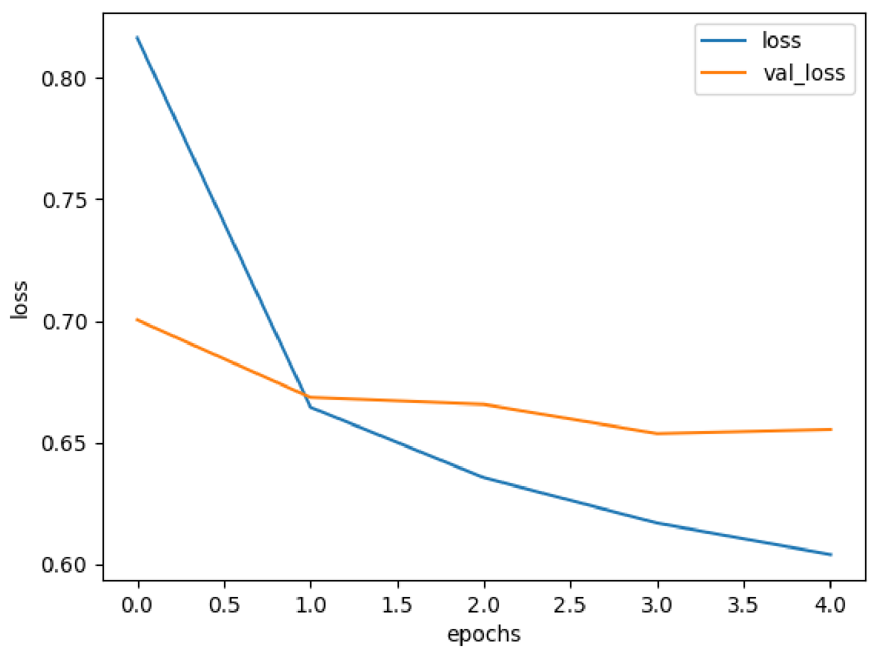

Figure 5.14: Loss Graph of Experiment 7

5.2.8 Experiment 8: Added Stop Words

Same as the experiment 7, the difference being that the top 3 most recurrent words in the datasets are flagged as stop words in an attempt to mitigate the bleed between categories.

Table 5.8: Experiment 8's defining characteristics.

Datasets Used	4: Gupta [14], Mohammad [16], Govi [13], and Calefato et al. [8]
Epochs	5
LSTM Layer	32 units
Categorized Sentiments as "Bad"	"Sadness", "Worry", and "Fear".
Categorized Sentiments as "Neutral"	"Neutral" and "Boredom".
Categorized Sentiments as "Good"	"Happiness", "Fun", "Joy", and "Love".

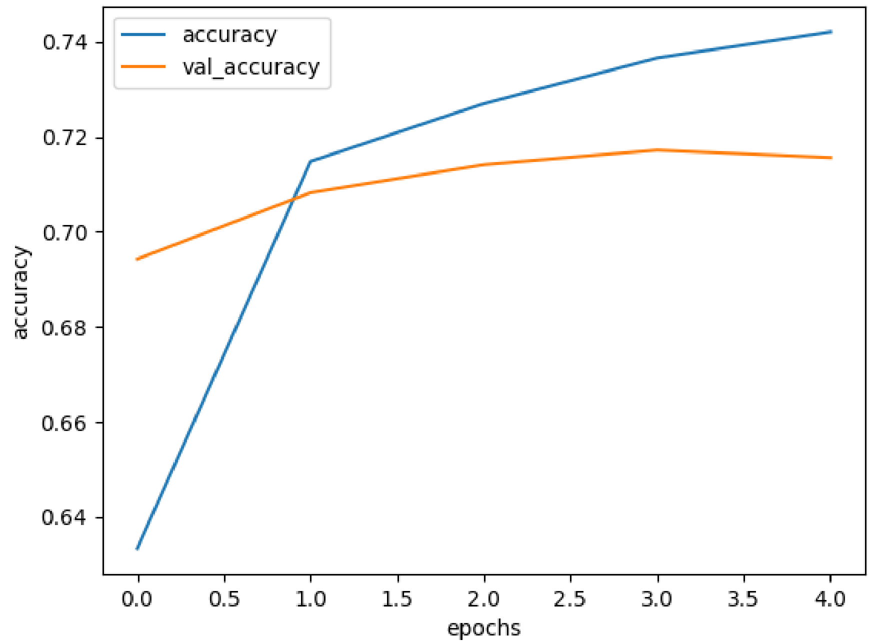

Figure 5.15: Accuracy Graph of Experiment 8

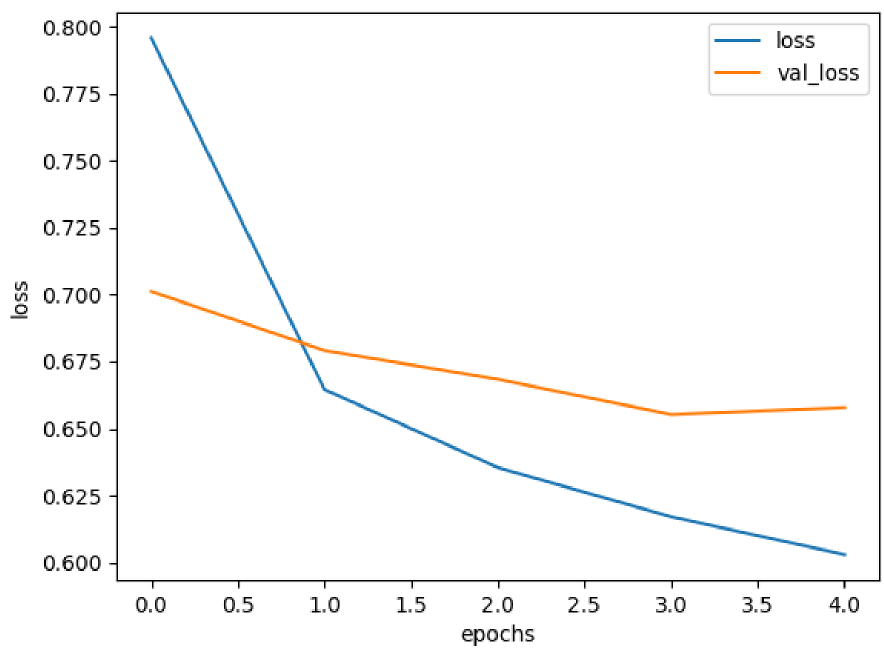

Figure 5.16: Loss Graph of Experiment 8

5.2.9 Experiment 9: Extra Stop Words and Reduced Classification Scope

Keeping in track with experiment 8, with the added extra of also reducing the scope of the classification, reducing the categories to “Good” and “Bad”.

Table 5.9: Experiment 9’s defining characteristics.

Datasets Used	4: Gupta [14], Mohammad [16], Govi [13], and Calefato et al. [8]
Epochs	5
LSTM Layer	32 units
Categorized Sentiments as “Bad”	“Sadness”, “Worry”, and “Fear”.
Categorized Sentiments as “Good”	“Happiness”, “Fun”, “Joy”, and “Love”.

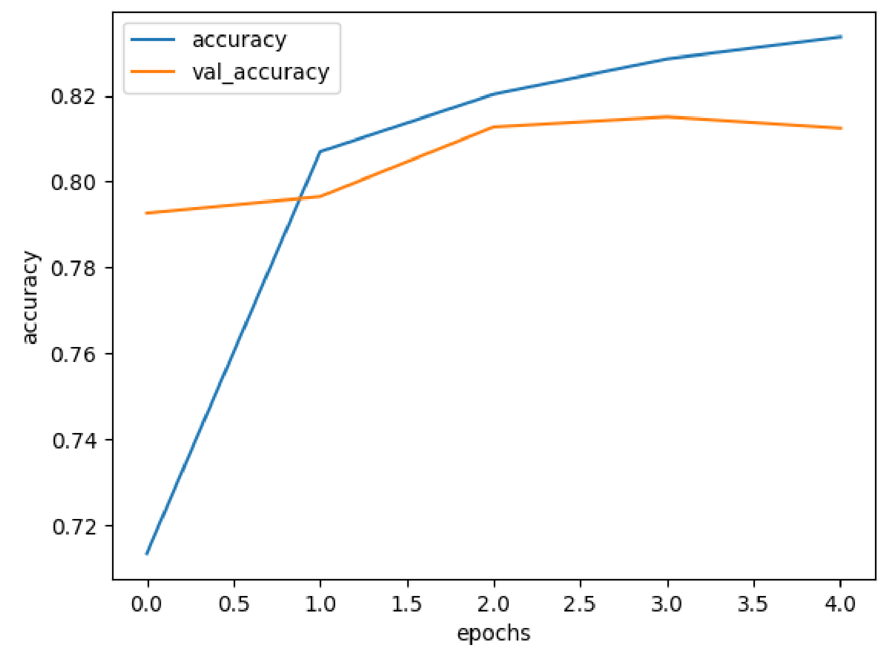

Figure 5.17: Accuracy Graph of Experiment 9

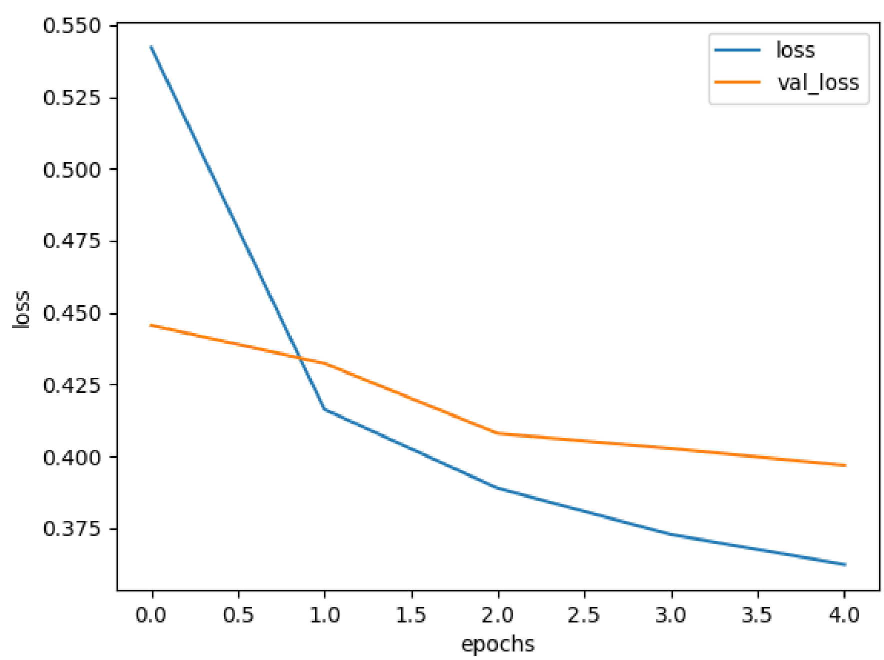

Figure 5.18: Loss Graph of Experiment 9

5.3 Results

In this section, the results of the experiments from the previous section will be discussed. Later on, a hypothesis will be formulated based from them. For reference, all these experiments were subjected to the same basic test inputs post-training:

- "Good"-labeled sentences: "I'm happy" and "happy happy happy happy happy happy"
- "Neutral"-labeled sentences: "I don't feel anything" and an empty input
- "Bad"-labeled sentences: "I am very sad right now" and "sad sad sad sad sad sad sad"

Table 5.10: Experiment results

	Training		Cross-Validation	
	Loss	Accuracy	Loss	Accuracy
Experiment 1	0.6857	0.7151	**0.8499**	0.6308
Experiment 2	0.5956	0.7576	0.7649	0.6821
Experiment 3	0.5829	0.7564	0.7373	0.6780
Experiment 4	0.5455	0.7741	0.6704	0.7110
Experiment 5	0.5442	0.6620	0.6620	0.7162
Experiment 6	0.6222	0.6550	0.7186	0.5357
Experiment 7	0.6041	0.7451	0.6555	0.7097
Experiment 8	0.6030	0.7421	0.6579	0.7156
Experiment 9	0.3624	0.8337	0.3871	**0.8124**

5.3.1 Interpretation

On Experiment 1: Base Experiment, the post-training results were promising, sentences with obvious “Good” and “Neutral” related words were correctly analyzed. But, as Figures 5.1 and 5.2 show, the validation results were considerably worse than the control data scores, this is due to the “Bad” score behaving erratically even when using obvious “Bad”-related words, this could be explained by the disparity between words used on the datasets – not many words were repeated on these –. Even so, overall this had one of the best accuracies across the experiments.

On Experiment 2: More Datasets With Reduced Data Scope, using one more dataset and less classes grouped with each category was opted to mitigate the “Bad” score while trying to keep data across the categories balanced. This, however, caused the algorithm to categorize every sentence from the input as happiness regardless of the words used, and sometimes even going over the “0” category used exclusively to categorize unclassifiable sentences and catching overflow errors.

On Experiment 3: Augmented LSTM Units, seeing that Experiment 2 had failed to predict correctly, more units were given to the LSTM to see if this lowered the chance of error, but as Figures 5.5 and 5.6 demonstrate, this got generally the same results. This determined that the defining factor was elsewhere.

On Experiment 4: Augmented Datasets Without Reduced Data Scope, the base algorithm from Experiment 1 was brought back to work with the three datasets used on Experiments 2 and 3 to verify if the extra dataset was too lopsided to work with. It was found that having the full roster of emotions taken into consideration (“Sadness”, “Worry”, and “Fear”, “Neutral” and “Boredom”, “Happiness”, “Fun”, “Joy”, and “Love”) actually helped the prediction scores. Even so, “Bad” category sentences were still not being recognized correctly, so more experimentation was needed.

On Experiment 5: Extra Sadness Dataset, knowing that it is very likely that there simply is not enough “Bad”-related data to train with, another dataset solely compromised sentences categorized as “Sadness” was added. Overall, this experiment had very good accuracy and loss as shown in Figures 5.9 and 5.10. This, however, had no noticeable effect on the prediction output post-training since the “Bad” scores were still very low with paired inputs.

On Experiment 6: Reduced Classification Scope, some drastic changes were made to experiment with the datasets used, this consisted in completely eliminating the “Neutral” category, only taking into consideration “Bad”, “Good” and “0” with the intention of seeing the algorithm’s behavior. This, as one would expect, broke everything and did not categorize anything correctly and had the worst scores overall of all experiments.

On Experiment 7: Reduced Epochs, taking again into consideration the results given from Experiment 5, the problem of overfitting was also a concern, and since the epoch count was a parameter yet-to-be manipulated. Instead of using the usual 10 epochs, 5 were used. This did not have any noticeable effect in the prediction output, however.

On Experiment 8: Added Stop Words, manually marking the words that represent the most bleed between categories (“work”, “go”, “nt”) still wasn’t enough to fix the categorization problem, even though there was a slight benefit in the accuracy and loss values.

On Experiment 9: Extra Stop Words and Reduced Classification Scope, reducing the data category scope while also maintaining the stopwords mentioned on experiment 8 filtered, brought overall the best results as of yet. This can be attributed to the fact that the most significant category bleed happened between “Neutral” and “Bad” since they shared a lot of the word pool.

5.3.2 Discussion

The experiment with the best performance overall was Experiment 9: Extra Stop Words and Reduced Classification Scope, with the scope adjustment the goal of predicting, at least partially, the feeling of a person's input was achieved. Thanks to an exhaustive search conducted inside all of the datasets used it was found out that the most-used words in the datasets pertaining to the "Bad" and "Neutral" category were mostly ambiguous words that, out of context, could be interpreted as disinterest or overall any other emotion.

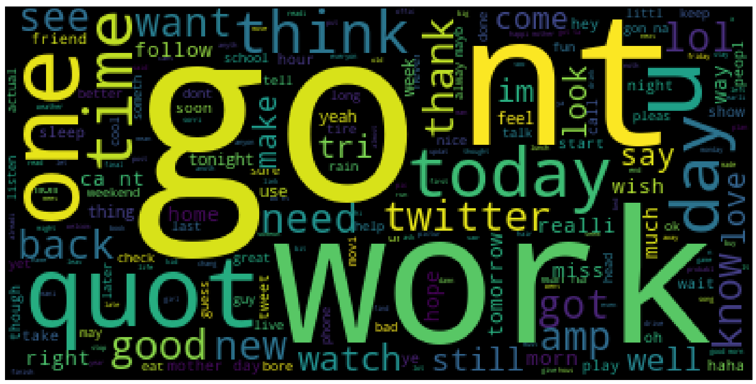

Figure 5.19: Word cloud of the "Neutral" category's corpus previous to the filtering in Experiment 8: Added Stop Words

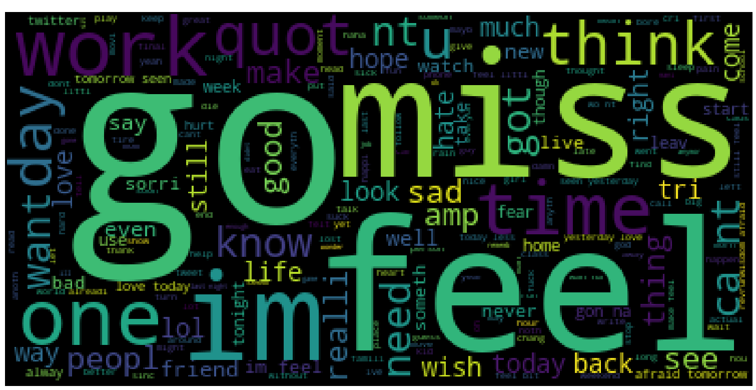

Figure 5.20: Word cloud of the "Bad" category's corpus previous to the filtering in Experiment 8: Added Stop Words

This, combined with the fact that some of the datasets used were plagued with orthographical errors, greatly affected the performance of this algorithm's original scope. Thus, Experiment 9 had to reduce the scope.

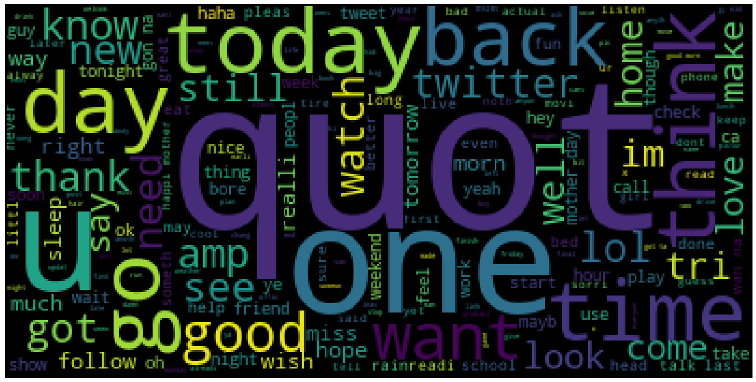

Figure 5.21: Word cloud of the "Neutral" category's corpus post-filtering

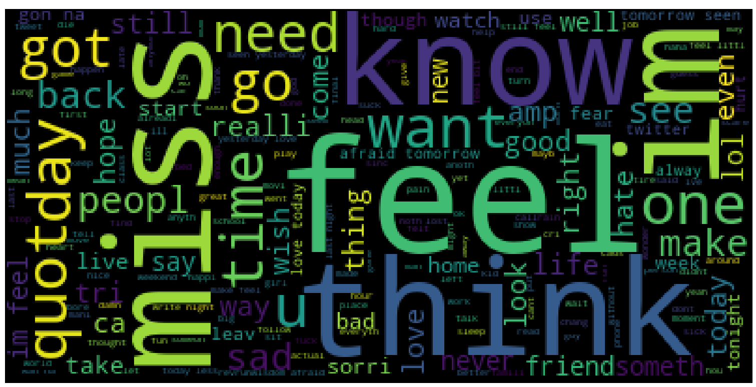

Figure 5.22: Word cloud of the "Bad" category's corpus post-filtering

These results determine that the proposed hypothesis is partially true: given enough data, a Machine Learning algorithm can learn to classify feelings and react accordingly, effectively learning how to identify patterns to an extent. However, high quality and volume data is needed for this to be reliable. Something that was only partially obtained for this project.

Overall, we can determine that, with the use of more consistent data, a favorable result can be achieved with the model used in this project.

Chapter 6

Conclusion

The formulated hypothesis of this thesis was, in short, that if empathy is based in pattern recognition, a Machine Learning algorithm could learn to predict and react according to a presented input.

This project had originally been planned as a therapy-assistant chatbot which had the purpose of soothing people in states of distress. But seeing that there are alternatives with much more powerful technology behind them already, the decision of making this as an open-source alternative to that kind of project was made.

This project works thanks to a Recurrent Neural Network that, being trained with certain datasets with categorized sentences, classifies an input from a person in one of three categories: "Good", "Neutral" and "Bad" depending on the sentiment detected. It was built using Python 3.8.10, making use of the following libraries:

TensorFlow v2.6.0 Used for Neural Network model building.

Keras v2.6.0 Used for Neural Network processes.

NLTK v3.5 Used for filtering words and "stemming".

Chatterbot Used for automated responses from the algorithm.

Pygame v1.9.5 Used for the GUI.

Since the algorithm is built using these tools, it can be considered open-source, so anyone can add or remove modules as needed.

As for the performance of this project, it is hard to call this successful, "Good" and "Neutral" sentences were correctly detected but, even in the best experiment, the algorithm could not detect "Bad"-related sentences with any kind of accuracy, but on the other hand, if we dig deeper down the datasets, we will find that there's not much that a ML algorithm could have done, since –after filtering the stop words– there were mostly just words that could possibly be used in other contexts and not be "Bad"-coded.

In the end, the conclusion reached is that the hypothesis was correct; it is possible for a Machine Learning algorithm to predict how a person is feeling based on an input, albeit some faults can be caused by the datasets used in training it. This can be changed using some quality control on them, or using personalized data specifically catered to this project.

6.1 Future Work

This project would greatly benefit from a dataset that takes into consideration sentences that can be said in any context and still be correctly classified. And, of course, the less ortographical errors there are, the better.

Another thing that could still be improved upon is the GUI, maybe using more modules or a new library altogether that can easily accept text inputs into it can make the project look not only sleek, but professional.

The fact that this algorithm is open-source means that it can easily be expanded upon: layers can be added to the Neural Network, a more robust chatbot-esque module can be added, advanced pre-filtering processes, and the list goes on. But a very critical thing to add would be a fine-tuner can be included for less lopsided classifications in the vocabulary, something that can pre-assign weights to words that appear in the datasets. This will, most likely, get rid of the classification issues that plague the actual version of this project.

Bibliography

[1] Shakeel Ahmad, Muhammad Zubair Asghar, Fahad M Alotaibi, and Irfanullah Awan. Detection and classification of social media-based extremist affiliations using sentiment analysis techniques. *Human-centric Computing and Information Sciences*, 9:1–23, 2019.

[2] Orestes Appel, Francisco Chiclana, and Jenny Carter. Main concepts, state of the art and future research questions in sentiment analysis. *Acta Polytechnica Hungarica*, 12:89–91, 2015.

[3] Taiwo Oladipupo Ayodele. Introduction to machine learning. *New Advances in Machine Learning*, Feb 2010. doi: 10.5772/9394.

[4] José David Bermúdez, Pedro Achanccaray, and Ieda Del'Arco Sanches. Evaluation of recurrent neural networks for crop recognition from multitemporal remote sensing images. In *Anais do XXVII Congresso Brasileiro de Cartografia*, pages 800–804, 2017.

[5] Daniel Berrar. Bayes' theorem and naive bayes classifier. *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics; Elsevier Science Publisher: Amsterdam, The Netherlands*, pages 403–412, 2018.

[6] Jordan J Bird, Anikó Ekárt, and Diego R Faria. Chatbot interaction with artificial intelligence: human data augmentation with t5 and language transformer ensemble for text classification. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–16, 2021.

[7] Norbert Blenn, Kassandra Charalampidou, and Christian Doerr. Context-sensitive sentiment classification of short colloquial text. In Robert Bestak, Lukas Kencl, Li Erran Li, Joerg Widmer, and Hao Yin, editors, *NETWORKING 2012*, pages 97–108, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-30045-5.

[8] Fabio Calefato, Filippo Lanubile, Nicole Novielli, and Luigi Quaranta. Emtk - the emotion mining toolkit, 2019.

[9] Nicola Capuano, Luca Greco, Pierluigi Ritrovato, and Mario Vento. Sentiment analysis for customer relationship management: An incremental learning approach. *Applied Intelligence*, 51:3339–3352, 2021.

[10] Patricia Chiril, Endang Wahyu Pamungkas, Farah Benamara, Véronique Moriceau, and Viviana Patti. Emotionally informed hate speech detection: a multi-target perspective. *Cognitive Computation*, pages 1–31, 2021.

[11] CrowdFlower. Sentiment analysis: Emotion in text, Jul 2016. URL `https://data.world/crowdflower/sentiment-analysis-in-text`.

[12] Robert Elliott, Arthur C Bohart, Jeanne C Watson, and Leslie S Greenberg. Empathy. *Psychotherapy*, 48:1–2, 2011.

[13] Praveen Govi. Emotions dataset for nlp, Apr 2020. URL `https://www.kaggle.com/praveengovi/emotions-dataset-for-nlp`.

[14] Pashupati Gupta. Emotion detection from text, 2021. URL `https://www.kaggle.com/pashupatigupta/emotion-detection-from-text`.

[15] Akshi Kumar and Mary Sebastian Teeja. Sentiment analysis: A perspective on its past, present and future. *International Journal of Intelligent Systems and Applications*, 4:2–4, 2012.

[16] Saif M. Mohammad. Word affect intensities. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018)*, Miyazaki, Japan, 2018.

[17] Robert R Morris, Kareem Kouddous, Rohan Kshirsagar, and Stephen M Schueller. Towards an artificially empathic conversational agent for mental health applications: System design and user perceptions. *J Med Internet Res*, 20, Jun 2018. ISSN 1438-8871. doi: 10.2196/10148.

[18] Kamal Nigam, John Lafferty, and Andrew McCallum. Using maximum entropy for text classification. In *IJCAI-99 workshop on machine learning for information filtering*, volume 1, pages 61–67. Stockholom, Sweden, 1999.

[19] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*, 2002.

[20] Stephanie D Preston and Frans de Waal. The communication of emotions and the possibility of empathy in animals. In *Altruism and Altruistic Love: Science, Philosophy, and Religion in Dialogue*. Oxford University Press, 2002. doi: 10.1093/acprof:oso/9780195143584.003.0025.

[21] Daniel Röchert, German Neubaum, and Stefan Stieglitz. Identifying political sentiments on youtube: A systematic comparison regarding the accuracy of recurrent neural network and machine learning models. In *Multidisciplinary International Symposium on Disinformation in Open Online Media*, pages 107–121. Springer, 2020.

[22] KP Soman, R Loganathan, and V Ajay. *Machine learning with SVM and other kernel methods*. PHI Learning Pvt. Ltd., 2009.

[23] Bingkun Wang, Ning Wang, and Zhongsheng Chen. Research on air quality forecast based on web text sentiment analysis. *Ecological Informatics*, 64, 2021. ISSN 1574-9541. doi: https://doi.org/10.1016/j.ecoinf.2021.101354.

[24] Xian-Da Zhang. Machine learning. In *A Matrix Algebra Approach to Artificial Intelligence*, pages 223–440. Springer, 2020.

Resumen autobiográfico

Alexander Espronceda Gómez

Candidato para obtener el grado de
Ingeniería en Tecnología de Software

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

Tesis:
Sentiment Analysis through Conversational Data

Nací el 17 de Noviembre de 1998 en Monterrey, Nuevo León, el mayor de los hijos de José Artemio Espronceda Estrada y Yadhira Lizet Gómez García.
Soy el primer hijo de la generación en mi familia, por lo que nunca sentí pertenecer, ya que mis tíos eran mucho más grandes que yo y mis primos mucho más pequeños. Por ello, siempre me encontraba pensando maneras de comunicarme con todos ellos "en su idioma" y lo lograba con relativo éxito. Pero a la persona que nunca pude entender fue a mi madre. Así que la mayoría de la inspiración de este proyecto se lo atribuyo a ella.
Me apasiona mucho el área de Análisis de Datos y Aprendizaje Máquina (Machine Learning), así como áreas como el Diseño de Videojuegos y la Psicología, por lo que este proyecto es la culminación entre mis pasiones más grandes para concluir la carrera de Ingeniería de Tecnología de Software.