Exploring Emphasis Selection: Is it just a matter of opinion?

Rhiannon Shaw

4th Year Project Report
Artificial Intelligence and Computer Science
School of Informatics
University of Edinburgh

2024

Abstract

Emphasis is incredibly common. Whether changing your vocal pitch for one word when speaking aloud, or making a phrase much larger in a poster to grab a reader's attention, it is all about getting a point across to someone else. When you are writing an academic paper, emphasis is just as important when it comes to helping your readers take away the main point. If what the reader focuses on contrasts with your views, then this could indicate that you have a problem with where you have emphasised words - the wrong words jump out. But, not much work has been done on how to train a language model to predict where this emphasis is in a sentence, and previous work only focuses on one type of text at a time. They either aim to predict emphasis in posters or in presentation slides. In this project, I have found that a Large Language Model can successfully predict emphasis regardless of the style of text. I have conducted a survey in order to test the theory that the LLM might only learn the opinions of annotators and have found that there must be some subconscious rules to the way we want to emphasise sentences that carry across annotators and the type of text. Although annotators may disagree frequently, we can now say that the task of emphasis selection is not just about an opinion.

Research Ethics Approval

This project obtained approval from the Informatics Research Ethics committee.
Ethics application number: 234962
Date when approval was obtained: 26/01/2024
The participants' information sheet and a consent form are included in the appendix.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Rhiannon Shaw*)

Acknowledgements

I would like to thank my supervisor, Adam Lopez, for providing feedback and guidance on my project. I would also like to thank my fellow students that were assigned to Adam Lopez for the suggestions and feedback they gave during our weekly meetings. For the available datasets and for defining the background for this task, I give thanks to Amirreza Shirani, Giai Tran, Hieu Trinh, Franck Dernoncourt, Nedim Lipka, Jose Echevarria, Thamar Solorio, Seokhwan Kim, and Paul Asente. For first defining the model that my model is based on, I thank Vipul Singhal, Sahil Dhull, Rishabh Agarwal, and Ashutosh Mod from the IITK team in SemEval 2020 Task 10.

Lastly, thank you to anyone that participated in my survey, I appreciate that you took time out of your day to answer it.

Table of Contents

Chapter 1

Introduction

1.1 Background

Emphasis is an important aspect of spoken and visual media. For example, if you were telling a story to someone, you may speak louder on certain words or lengthen them more than usual when they are crucial for the story to make sense to your listener. In visual media such as in a poster or advert, emphasis is created by increasing the size of a word or displaying it in a contrasting colour to the rest of the poster in a bid to draw a reader's attention to it. An example of this can be found in Figure 1.1 where the phrase "GET 1 FREE!" aims to catch a shopper's attention to the deal it is advertising by being larger than the other text. Predicting where this emphasis appears within different pieces of visual media has been given the name "Emphasis Selection"Shirani et al. [2019].

To be more exact, emphasis selection is the task of taking a sentence and assigning each word in it a value from 0 to 1 as a representation of how much we think a word should be emphasised if we were to talk aloud or display it in say, a poster. If we were to use our visual example of emphasis from Figure 1.1, then the most simple interpretation of this example would be where "BUY 1" has no probability of being emphasised and "GET 1 FREE!" has a probability of 1 being emphasised as seen in Table 1.1. However, in most texts, there is more to it than what is or is not emphasised which is the reason that

Figure 1.1: A visual example of emphasis

Emphasis Probabilities	0	0	1	1	1	1
Word	Buy	1	Get	1	Free	!

Table 1.1: Example of emphasis selection using 1.1.

we represent this task as predicting probabilities rather than a binary classification task. In datasets for this task, the emphasis probabilities for each piece of text were created from a number of annotators that chose whether they thought a word was emphasised or not. The specifics of these datasets and the task at hand along with the evaluation methods used for it are discussed in chapter 2.

Emphasis selection has not been researched in as much detail compared to other NLP tasks. It is similar to keyword extraction, however, unlike keyword extraction which focuses on finding important phrases in the text, emphasis selection aims to simulate how people would emphasise words when reading or speaking it. Speech is what most papers on emphasis focus on as it is used to create a more natural sounding text to speech model. Yet there is still some research on emphasis within pieces of texts, albeit in order to aid in visual emphasis rather than the style of a piece of writing.

Papers that do focus on emphasis selection take inspiration from ”Learning Emphasis Selection for Written Text in Visual Media from Crowd-Sourced Label Distributions”Shirani et al. [2019], the paper responsible for first defining the task and how to evaluate it (See Section 2.3). With there even being a shared task on it for SemEval 2020 Shirani et al. [2020]. Works based on the original paper and the shared task investigate how well types of language models performed at finding the right words to emphasise on posters. The dataset and what it contains is explained in more detail in Section 2.2, however overall it contained a mix of short quotes, advertisements, social media posts, and posters from multiple sources.

The first approach to this task was to use 2 layers of bi-directional LSTMs (Long short-term memory networks)Shirani et al. [2019], but transformers were instead the most common type of model found in the shared task and most outperformed the original LSTM modelShirani et al. [2020]. And out of the transformers used, BERT Devlin et al. [2019] and its variants such as RoBERTa Liu et al. [2019] were popular choices. Methods trialed within this shared task include creating extra features for the model to use when fine-tuning such as telling the model when a word was capitalised or what Part of Speech tag, what type of word, it is. Although in most cases these models were capable of predicting the correct emphasis probabilities, they struggled with sentences with less common word structures or with sentences that had multiple words that had around the same emphasis. The former is most likely caused by how small the dataset is, whilst the latter is partially due to the evaluation method as the models still predicted those multiple words as being emphasised, just not in the right order from most to least emphasised.

After the SemEval task, another dataset, PSED Shirani et al. [2021b], was developed that used texts completely different to the ones seen in the original dataset. Instead of short texts or wisdom quotes, this dataset primarily featured much longer pieces of texts that came from presentation slides. And just like the original dataset, PSED

was used within a shared task. This time, the task was hosted in CAD 2021 Shirani et al. [2021a] and had far fewer teams. Again, transformer-based models were the most popular and outperformed the example LSTM for the task. Therefore it is safe to say that using transformers are the way to go for predicting emphasis selection as they perform decently on different datasets. Something new that came from this task was that one team used a variant of BERT trained on scientific papers, SciBERT Beltagy et al. [2019], as the presentation slide dataset included many scientific words, and found that this improved their overall model. Indicating that vocabulary could play a part in how well a model performs if choosing a model that had been fine-tuned on a similar subject worked better than a more general model like BERT.

1.2 Project Goals and Motivation

One problem with both datasets is that there are less than 10 annotators for both sets and that there is a lot of disagreement between said annotators. Although the teams in the shared tasks were thorough in experimenting with how to make their models as accurate as possible in their specific shared tasks, they do not explore if the models have learnt rules on how to emphasise sentences or if they have picked up on the opinions of the annotators in their respective datasets. This is what I intend to explore within this project- whether the task of emphasis selection is primarily opinion-based or has some underlying rules to it that persist across different annotators and types of texts. Another question I answer within this project that is connected to my main goal is whether the current resources available are enough to avoid the model from over-fitting, something that would suggest that the current datasets do not show a broad enough range of examples.

When it comes to the type of rules one may expect, I will be relating back to suggestions found in ”Style: Lessons in Clarity and Grace” Williams and Bizup [2016] as the book aims to help people write clearer and the aim of using emphasis is that we want our emphasised words to make points clear to our readers. An example of a rule found in the book is that the author suggests that the last word in a sentence has extra emphasis placed on it.

But how does emphasis help make things clear for readers? Imagine you want to make an important point in your paper on a new algorithm that you have created. You want to show how much better it is than previous algorithms. In most cases, your algorithm is better than previous work, but it performs badly on very specific scenarios. So you write the sentence: “my algorithm is a vast improvement compared to older work, although it can be far more computationally expensive in its worst case”. This leads the reader to ask themselves: “what is this worst case and how bad can it get”? But after this sentence, instead of elaborating on this question, you praise your algorithm for the rest of the paragraph and perhaps even the rest of the paper. To a reader, this is frustrating to read as the tone of your paper contrasts too strongly between the example sentence’s negative ending and the positive words following it. A contrast in tone can create incoherence in a piece of writing if done incorrectly, and incoherence can lead to confusion among readers.

Overall, the aim of this project is to see if the niche task of emphasis selection can be used more generally than previous work to predict where emphasis is in any written piece of text, instead of specifically posters, advertisements, or presentation slides as discussed in Section 1.1 and Section 2.2. This would indicate that there are subconscious rules to where we want to place emphasis even when reading text. And if we know that it is a viable task for any type of text, then it could be used to help writers see where they have put emphasis in a sentence when they are revising a piece of work.

1.3 Contributions

My aims and contributions whilst working on this project is as follows:

1. I created a training, development, and test set by combining existing datasets on Emphasis Selection.
2. I replicated an approximation of the model found in IITK's paper Singhal et al. [2020].
3. I replicated metrics other than the most common evaluation metric, Match_M, by following what is described within Shirani et al. [2019]. More information about these metrics is found in subsection 2.3.2.
4. I created a different classifier that uses classes rather than probabilities to predict emphasis probabilities.
5. I created various custom vocabularies to add to the model's base vocabulary.
6. I fine-tuned models and performed experiments on them by fine-tuning them either on the modified classifier, on a different vocabulary, or by changing 1 parameter at a time in the model.
7. I created a short survey that consisted of 5 sentences that participants were asked to annotate to obtain emphasis probabilities.
8. I explored possible trends that appear within the two available datasets and within the survey to find more generalised rules to predicting emphasis.

Chapter 2

Dataset Design and Evaluation Methods

2.1 The Task

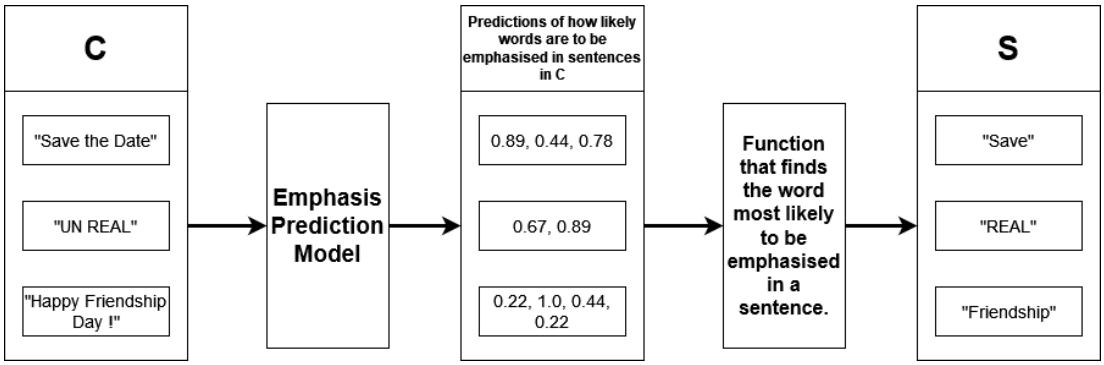


Figure 2.1: A breakdown on the steps of the emphasis selection task if the subset S was created by finding the word with the highest emphasis probability.

The task used in this project is the task of emphasis selection first defined in the paper: ”Learning emphasis selection for written text in visual media from crowd-sourced label distributions” Shirani et al. [2019]. Therefore it uses the definition of emphasis selection that is discussed there. The task is defined as follows:

”Given a sequence of words or tokens $C = x_1, ..., x_n$, we want to determine the subset S of words in C that are good candidates to emphasize, where $1 \leq |S| \leq n$.”

At its most basic elements, the aim of emphasis selection is to use a model that takes in sentences, C, as input and outputs a list for each sentence in C. The output list for a sentence is a prediction on how much the model thinks each word in that sentence should be emphasised. The output lists can then be used to find the subset S for each sentence. However, actually finding S is not what we are focusing on, but, as an example, one method of finding subset S of a sentence would be to take the word with the highest predicted probability of being emphasised in that sentence. For the rest of the project and within previous work, we focus on how to get the predicted probabilities,

the emphasis predictions, of each sentence using a model. The full breakdown of this task has been visualised in figure 2.1.

2.2 Datasets

2.2.1 Available datasets

Currently there are three datasets that are suitable to use for my project that are used in previous work. The SemEval2020 dataset is comprised of the Spark Dataset, a collection of short texts from fliers, adverts etc. created by Adobe Spark, and the Quotes Dataset, a collection of quotes from Wisdom Quotes. Lastly, there is the Presentation Slides Emphasis Dataset (PSED), the dataset used in the CAD21 task, that is comprised of texts that come from presentation slides of various topics, such as presentations from ACL Anthology and .GOV websites.

For all datasets, texts are broken up by word and by punctuation and each sentence is represented as a list of words, part of words, or punctuation. The elements of these lists are called tokens. For example, ”we’re” would be split into two tokens: ”we” and ”’re”. The full sentence: ”We’re going to a party.” would thus appear as: [”We”, ”’re”, ”going”, ”to”, ”a”, ”party”, ”.”] if it was then read into a python program using the function given in the shared task. The reason to split punctuation up into its own tokens is that some punctuation such as an exclaimation mark can give us information about where emphasis should be in the words before it but might not necessarily be as likely to be emphasised itself.

What makes looking at all of the datasets useful is that the styles and average lengths of the texts in the datasets are varied. Whilst the Spark dataset has an incredibly small average text length of around 6, the PSED dataset’s average text length is much higher at around 78 words per text- the Quotes dataset is between the two. The reason for the PSED dataset to have so many longer texts is because each text is an entire presentation slide with bullet points also adding to the total length. In table 2.1, we can see the statistics for each dataset.

Dataset	Max	Avg	SD	Tokens	Texts
Spark	29	6.24	4.64	UNK	1,195
Quotes	38	13.99	5.47	UNK	2,681
Complete SemEval Dataset	38	11.6	6.33	44,976	3,876
Available SemEval Dataset	38	11.74	6.4	36,785	3,134
Available CAD (PSED) Dataset	180	77.23	33.62	109,756	1,421

Table 2.1: Statistics on the number of tokens per sentence per dataset.

But why include two different versions of the SemEval dataset and not include the complete PSED dataset statistics? The available datasets in the table can be found in the SemEval2020 task and the CAD21 task github pages and are what are used for this project. However there is one problem - the test sets are not publicly available, therefore the datasets I have are incomplete and have a smaller number of sentences. The

PSED paper only announces word length, not the total number of tokens per sentence, therefore I cannot calculate the complete dataset's statistics. Whilst looking for the complete datasets, I found that before Codalab switched over to their new server, it was possible submit models to both of the tasks so that they can be evaluated on the test sets. Account creation is now only allowed on the new server, but these competitions were not moved over to the new server and thus if you do not already have an account, you cannot enter your model into the competition. I requested access to the test set but I did not receive a response. After getting no response, I then decided on a plan. I would combine what I had available to create my own dataset.

Dataset	Example Sentence
Spark	Best fall foliage trips in U.S.
Quotes	Learning is a lifetime process but there comes a time when we must stop adding and stop updating.
PSED	Sustainable Communities Strategy • The bedrock of SB 375 • Combining housing and transportation strategies • Reaching the GHG emission target

Table 2.2: Example sentences to highlight the styles of the different datasets

2.2.2 Creating the dataset

To create my own test set, I gathered all of the labelled data from the training and development sets released for the shared tasks and combined them. I then split the full set into new training, development, and test sets with a rough split of 80/20/20. It is not exact as some sentences were repeated within the original datasets and were removed after the split to ensure the test set shared no sentences with the other sets. Another reason for combining the sets was to see if a model still learns to select emphasis in vastly different styles of sentences. For example, the presentations dataset contains more scientific words than the quotes dataset, has longer sentences, and any bullet points that were in the slide are also annotated. It is useful to have a flexible model that can work with different styles in writing and previous work does not explore this in any way.

However, combining the datasets is not an advantage over having the original complete datasets. The major downside of creating my own test set is that I will be unable to compare my model's evaluated performance with any other previous work. Because some datasets are more or less difficult for models to work on, I cannot assume that the model I create would perform just as well on the complete dataset's test sets as it would on mine, they share no sentences!

Therefore when I discuss the results of evaluating on my test set in section 2.3, I focus on the differences between the models I have implemented and trained. When it comes to evaluating my models against previous work, the closest I can get to a test set alike the ones seen in the shared tasks is to evaluate my model on only the test texts that came from the SemEval task or only the test texts that came from the CAD task. Doing this

gets me as close to the original difficulty of the unavailable test sets with my dataset, but it will never be an exact match. Any results from my model against previous models cannot be taken as fact that it is better or worse than them, only that it would perform around that value if the test set is of the same difficulty as the training and development data.

2.2.3 How are emphasis probabilities calculated?

Although I have explained the problem and the datasets that are available, I have yet to go into detail on how the datasets themselves calculate how emphasised a word should be. The aim of any model using these datasets is to learn emphasis selection. Therefore, the texts in the datasets have been evaluated by a set of annotators for where they thought words should be emphasised.
The annotators were given sentences and asked to label each word as either being emphasised or not. These labels were then used to calculate the emphasis probability of that position in the sentence. Of course, the probability is dependent on the word, but the same word could appear twice in a sentence and have a different probability. The probabilities are also independent of any other text as they were annotated separately, with texts coming from different sources. Using the annotations, the emphasis probabilities are calculated with the following formula:

$$\frac{\text{Sum(number of annotators that thought a word was emphasised)}}{\text{Number of annotators}}$$

For example, the phrase ”UN REAL” has two words for the annotators to tag. 6 out of 9 annotators believed ”UN” should be emphasised therefore the emphasis probability of the first position in the sentence is $\frac{2}{3}$. For the next word, ”REAL”, 8 annotators thought this word should be emphasised therefore the second position's probability is $\frac{8}{9}$.

To do emphasis selection, the model needs to learn how to assign the right emphasis probability to words by using these annotated sentences as training data. Thankfully, the available datasets had those probabilities included in them. Otherwise, calculating the emphasis probabilities would have been another step in preparing the dataset before I could fine-tune a model.

In tables 2.3 and 2.4 you can see examples from the SemEval dataset. The SemEval dataset had 9 annotators in total, the CAD dataset had 8 annotators after some preprocessing to increase how much annotators agreed with one another in the dataset.

2.3 Evaluation Methods

2.3.1 Match$_m$

Now that we know what the model is trying to learn and replicate, we can now define how it shall be evaluated.
The evaluation method used in all previous work is $Match_m$. The following definition of it also comes from ”Learning emphasis selection for written text in visual media

Annotators that **did not** believe a word to be emphasised	4	8	6	8	2	7
Annotators that **did** believe a word to be emphasised	5	1	3	1	7	2
Emphasis Probabilities	0.56	0.11	0.33	0.11	0.78	0.22
Words	Time	to	pick	your	bracket	!

Table 2.3: SemEval Dataset Emphasis Probabilities example 1. Total number of annotators: 9.

Annotators that **did not** believe a word to be emphasised	7	5	1	7
Annotators that **did** believe a word to be emphasised	2	4	8	2
Emphasis Probabilities	0.22	0.44	0.89	0.22
Words	Keep	moving	forward	.

Table 2.4: SemEval Dataset Emphasis Probabilities example 2. Total number of annotators: 9.

from crowd-sourced label distributions" Shirani et al. [2019] but has been reworded. For every example x in the test set D_t, create a set of m words from the top m emphasis probabilities of sentence x and call it $S_m^{(x)}$. Then choose the top m estimated emphasis probabilities that the model predicts for sentence x and call the set $\hat{S}_m^{(x)}$.Then $Match_m$ is calculated using the formula:

$$\text{Match}_m := \frac{\sum_{x \in D_t} |S_m^{(x)} \cap \hat{S}_m^{(x)}| / (\min(m, x))}{|D_t|}$$

What makes this metric more useful than others suggested in the paper is that when we think about the task's second aim, which is to get a subset of the most likely to be emphasised words, then it is more important for the model to be able to predict the words most likely to be emphasised regardless of the difference between the actual and predicted probability than correctly predicting the probabilities of less likely words.

Many texts in the datasets have lots of words with little emphasis in them so if we evaluated a model on how well it can predict the annotators' emphasis probabilities, then we may end up selecting a model that is amazing at predicting words with a low emphasis probability but not the best at finding the most emphasised words. Therefore, confidence of the model is unimportant, only what words the model predicts to have the highest emphasis. However, this does mean that if two models performed identically to each other according to $Match_m$, one model could still be closer in confidence to the ground truth than the other. Other evaluation methods are proposed in Shirani et al. [2019], but none could be used to compare my model to previous work thus $Match_m$ was the evaluation method focused on the most in this project.

2.3.2 Other Metrics

Although Match$_m$ is the preferred method for this task, more common evaluation metrics are discussed and used in "Learning emphasis selection for written text in visual media from crowd-sourced label distributions" Shirani et al. [2019]. They do, however, need to be adapted to this task and how to do this has been left up to interpretation. Therefore, it is important to explain how these metrics work with emphasis selection and how I have implemented them.

Top K

Top K is useful to calculate as it can be used to find the precision, recall, and F1 score of a model - a much more common metric than Match$_m$! Top K is similar to Match$_m$ as it takes the top k most emphasised words predicted by the model from the sentence and compares it to the actual probabilities. However, unlike Match$_m$, top k does not require the intersecting set between the predicted probabilities and the actual probabilities as precision and recall are calculated using their broader definition in information retrieval tasks rather than the usual definition seen in classification tasks. The reasoning behind this is that as we are working with probabilities, we have no exact right or wrong answers needed for calculating true or false positives or negatives. We also care more about what words the model predicts is the most emphasised rather than how close it gets to the actual probabilities. We will build off of the following definitions that come from chapter 8 of "Introduction to Information Retrieval"Manning et al. [2008] :

$$\text{Precision} = \frac{\text{num of relevant items retrieved}}{\text{num of retrieved items}} \tag{2.1}$$

$$\text{Recall} = \frac{\text{num of relevant items retrieved}}{\text{num of relevant items}} \tag{2.2}$$

Retrieved words will be what the model predicts, relevant words will be chosen using the test set's actual probabilities. In the context of Top K, the number of retrieved items/words will be k. With retrieved items down, what was considered a "relevant" word still needed to be figured out. For this task, we want to see what words are most emphasised therefore words with high actual probabilities are more relevant than words that none of the annotators believed should be emphasised. For example, let us say that a word with an actual probability of over 0.5 is relevant, anything less is irrelevant. Then we can convert the actual probabilities into binary classes where

$$\begin{cases} 0/\text{irrelevant} & \text{probability} < 0.5 \\ 1/\text{relevant} & \text{probability} >= 0.5 \end{cases} \tag{2.3}$$

The value of K affects what metric is most useful to us. At small values of K, precision is most important because it tells us how often the model and annotators agreed that a word was highly emphasised. At larger values of K, precision will drop as there are only so many words with an emphasis probability of over 0.5 in a sentence. At small values of K, recall will be small, especially if the sentence has many highly emphasised words, as it will not be able to take the same number of words as there are highly

emphasised words. At higher values of K, recall instead tells us how often the model completely misses when a word should be highly emphasised. F1, as always, is then the balance between the two - how often the model correctly predicts words to be highly emphasised versus how often it misses them out. If the cut-off is decreased, then we can expect these metrics to increase as more words are considered relevant and there are less chances for errors to happen, the opposite happens if the cut-off is increased.

Due to how much the metrics in Top K rely on both the value of K and the cut-off probability, it is not the best metric for this task as it can be manipulated for or against our favour, especially considering there are no set values for top k that other works use. For example, if we were to set the cut-off to 0 then the precision would be unnaturally high because of how many relevant words there now are. We could then conclude that the model always finds the most highly emphasised words, which would be wrong.

ROC AUC

ROC AUC, also known as the Area under the ROC curve, is also a more well known metric that is useful because it allows us to see how well the model can predict emphasis compared to it randomly being chosen. To create the ROC curve, the same method as how the relevant items in top k were calculated can be used to turn our probability model into a binary classifier. Let us say that the probability used to decide whether a word is in class 0 or class 1, the cut-off, can vary. Then we could have many cut-off values between 0 and 1 to simulate trying different thresholds on the classifier. The only other difference is that both the predicted and actual probabilities are converted to these binary classes - an adaption of calculating the ROC curve to our task as the actual probabilities are often originally discrete classes, not probabilities, when we want to calculate this metric.
Now that the results can be converted to a binary format, we calculate the number of true and false positives and negatives to get the True Positive Rate (Recall) and the False Positive Rate at that current cut-off value.

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.4}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{2.5}$$

Repeating this process of converting the probabilities to the binary classes and calculating the TPR and FPR give us all of the points to create the ROC Curve. The ROC curve can be plotted with the FPRs as the x-axis vs the TPRs in the y-axis. AUC is simply the area under the created curve.

This implementation does have a flaw - if a model does not produce a high TPR and high FPR at any cut-off value then the AUC can be incredibly low. In a normal implementation of a ROC curve, the bounds are from 0 to 1 for both TPR and FPR and the curve is expected to go from [0,0] to [1,1]. If there are no high TPR and high FPR values at the same time, then the curve in my implementation will stop before [1,1], severely limiting the possible AUC. Manually adding these bounds leads to a suspiciously high AUC for all models, therefore I chose not to bound the metric. One

potential way to fix this problem would be to use multi-class ROC AUC and place the probabilities within bins, an idea explored for an alternative classifier implemented in subsection 4.1.1. Multi-class ROC AUC can be calculated in a couple of ways. By either comparing one class to all others or by computing all possible pairs of classes. This leaves us to decide how precise the actual probabilities should be as having too many would result in a much larger combination of comparisons, but too little and we risk losing information about the model as the model now has an easier time predicting the right class.

I eventually decided to stick with my flawed version of ROC AUC as it can still be helpful in its own way, just not the AUC itself. Plotting the TPR and FPR through the different cut-off values can help give an idea of how often the model likes to predict words with a high emphasis, say, around 1, instead of taking the safer approach of predicting many words with a slightly lower probability.

Chapter 3

Implementation

3.1 Language and Platform used

My project was written in Python. I decided to use python because it already had multiple libraries that could load and tune pre-trained transformer models, such as huggingface's transformers library or pytorch.

As LLMs are computationally expensive to train and run, it would have taken too long to use my laptop to train the number of models that I wanted to test. I therefore chose to use Google Colab whenever I needed to implement or run my model because they offered more powerful GPUs with cuda cores and all of the packages I required could be imported or installed with pip in the notebook.
To make it easier to switch between Colab notebooks and to make the files more readable, commonly used functions such as the model and classifier, loading a fine-tuned model, and the metric functions were kept in a python file.

3.2 Reproducing previous work

Before experimenting with my own model, I created a classifier based on the specifications discussed in the IITK SemEval 2020 submission Singhal et al. [2020]. Without access to the original model or test set, I could only guess on some aspects of the implementation. At its most basic level, IITK's implementation is a pre-trained LLM and classifier combined together into one model. The reason I decided to choose this work instead of higher ranking models from the SemEval task such as implementations by the groups Ernie or Hitachi Huang et al. [2020], Morio et al. [2020] was because they either lacked the specifications needed for me to be able to confidently recreate their work or did not implement a classifier on top of the fine-tuned model. I required the classifier to experiment with as it allowed me to more easily implement my probability bucket variant. Hitachi's implementation also requires finding the optimal learning rates of all of the pre-trained models used, I could not use their optimal values as my training dataset will have a different optimal value. This would be too time consuming. CAD21 models were not considered for reproduction as the majority of papers are not available

and the best performing one that is, Cisco Ghosh et al. [2021], is close to the baseline model.

However, I did take inspiration from CAD21. In the shared task paper, it mentions that one of the teams that does not have a paper available, DeepBlueAI, explored how useful it was to fine-tune SciBERT to the task. They found that because the PSED is full of technical terms SciBERT outperforms BERT as it was trained on technical papers. Therefore I chose SciBERT as my base model instead of BERT because my dataset would also have those technical terms that BERT struggled with.

3.2.1 Base Implementation

IITK's model consists of two parts. The pre-trained model and the classifier. Although the paper uses multiple models and averages their scores, I chose not to do this as my interest laid more into experimenting with the vocabulary of models and the representation of probabilities given to the model. Time constraints also encouraged me to focus on one model as trying to optimise many of them before averaging them together would be time consuming. Although I did not combine them, I have fine-tuned a variety of models both uncased and cased which will be an experiment discussed in subsection 4.2.1.1. However for simplicity and consistency, I will refer to the pre-trained model as BERT in this section as changing the loaded model does not require changing the implementation apart from when the model is first loaded in. Input sentences go through the BERT model and come out of it as a 768 dimension vector. The vector is then passed to the classifier to compute the emphasis probabilities. During fine-tuning, the loss used is binary cross entropy loss as all output values are between 0 and 1.

BERT is a well known LLM and thus there are many libraries that can be used to implement it. For this project, models were loaded from HuggingFace using their transformers library for the AutoModel class. HuggingFace is useful because they provide a BERT tokenizer for tokenizing data, a subject discussed in 3.3, and switching models from BERT to SciBERT easily can be done by changing the model and vocabulary location when initialising the AutoModel class.

BERT is initialised with a learning rate of 2e-5 and uses the Adam optimiser during fine-tuning. This implementation uses the default number of layers available, 12, with no frozen layers.

Classifier The classifier was built using the pytorch nn module. The library was chosen instead of tensorflow/keras because pytorch offered various quantization methods to be used later on in the project and it was easier to understand as a novice. Although I did explore the quantization methods, they were not as relevant to this project and any experiments on the subject have been left out.

The classifier consists of 3 linear layers, 2 ReLU layers, 1 dropout layer, and 1 sigmoid layer. The sigmoid layer is what outputs the probabilities. Probabilities are for individual tokens in the sentence rather than the full sentence so we do not run into the problem of

the entire sentence's emphasis probabilities summing up to 1, you could have a sentence full of highly emphasised words with no problems.

As the BERT models used are not their large variants, the first linear layer takes 768 dimension vector from the BERT output and converts it to a 300 dimension vector. The second linear layer goes from 300 dimensions to 20, leaving the last layer to go from 20 to 1. If the large variant of BERT was used, the hidden dimensions would be 900 and 40 instead. I am unsure of their reasoning behind these numbers but perhaps their intuition was to go from the BERT output to their maximum input length, 300 tokens.

The dropout for the classifier layer is set to 0.3. The paper does not indicate the number of dropout layers used, only that there were more than one. In preliminary testing, I found that one layer was enough for decent results and chose to only include one to keep the classifier as simple as needed until further testing was done. Therefore I performed an experiment to see if there was a significant improvement in using more or less dropout layers. The full experiment and its results are explained in subsection B.0.3 but is touched upon within the next chapter.

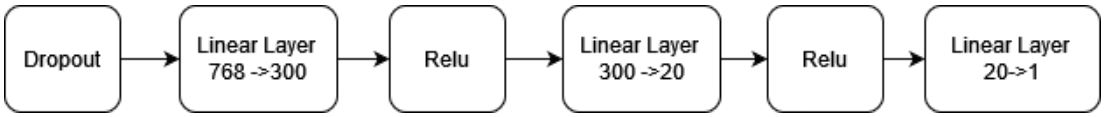


Figure 3.1: Implemented Classifier

3.2.2 Alternatives

Alternatives to using the defined model were tested by IITK. This ranged from freezing layers of BERT to using a BiLSTM or GRU layers instead of the classifier. However, they found the best results when the model was initialised to the specifications listed above. The classifier is also consistent over different pre-trained models. Thus I have replicated the most optimal model that they could find.

Previous works that make use of BiLSTMs are rarer than works that use transformer models. Although there is the implementation specified in Shirani et al. [2019], the best performing models in both the SemEval 2020 task and CAD21 task are transformer based models with only a few teams using BiLSTMs at all for either the main model or the classifier. Because there was more work done in previous papers using transformer based models and because BERT and its variants are popular enough that there are sufficient resources on them that I felt confident in using this implementation.

3.3 Tokenization and formatting predictions

3.3.1 Tokenization

However, before I could think of fine-tuning my model or experimenting with my classifier, I needed to process the training and development data into a format that my model could take as input. As a concept, it sounded easy. But in practice, this was one of the biggest challenges in this project.

BERT only accepts sentences that have been encoded. Encoded sentences are lists of integers that correspond to tokens, parts of words. Tokenization is therefore the process of breaking a string into those smaller pieces of words which can then be assigned an id. This is more efficient than just storing every single word as some words can be made up of parts that are shared between multiple words, thus saving space and allowing the model to create more words that it might come across. For example, imagine we have the words: ”playing” and ”partying”. These could be split up into ”play” + ”ing” and ”party” + ”ing”. The model stores 3 words: ”play”,”party”, and ”ing” as ids. Then the model can recognise the 4 words: ”play”,”party”,”playing” and ”partying” with only 3 ids. Considering the number of words that use ”ing”, a lot of extra words can be created that way.

The problem here is how the dataset is formatted. Sentences are lists of words, punctuation is split up as well. Each entry has its own emphasis probability. Reformatting a list of strings into a string is easy. What to do after tokenizing the words back into that list of strings is harder, the tokenized list is often longer than the original sentence! The emphasis probabilities no longer match with the words of the sentence and because we’re wanting the model to predict each token, the original emphasis probabilities cannot work with the tokenized list.

To fix this, some values need to be duplicated or changed. One possible idea would be to average out the probability over all of a word’s tokens. However, this is detrimental to long, but highly emphasised, words. The method that I have went with is to take whatever probability corresponded to the original token before it has been split up and repeat it for the number of tokens the word was split up into. To calculate how many copies of a probability are needed for the tokens and labels to line up, each word is tokenized separately, and the number of tokens is then used to append the right number of copies to a new list.

Table 3.1: Tokenization example on a text with more tokens that emphasis probabilities

Before Tokenization		
Emphasis Probabilities	0.78	0.67
Words	Midnight	Hike

Hike is split up into ”Hi” + ”#ke” but there are only two emphasis probabilities therefore ”#ke” has no corresponding probability.

After Tokenization			
Emphasis Probabilities	0.78	0.67	?
Words	Midnight	Hi	#ke

”#ke” is assigned the same probability as ”Hike” would have to fix this problem.

After adjusting labels			
Emphasis Probabilities	0.78	0.67	0.67
Words	Midnight	Hi	#ke

Formatting Predictions

Just like how the tokenized training data was often longer than the original training texts, the model's predictions were often longer than the original texts because it was predicting on said tokenized and then encoded texts. Therefore the predictions outputted from the model cannot be evaluated as is and must be processed to be the correct lengths before the predictions can be evaluated.

Like when the texts are tokenized, the number of segments a word is split into is needed. But once we have those lengths, there are various ways that we could use to decide how we compress multiple tokens and their multiple probabilities back into one emphasis probability. Let us again take ”Hike” as an example. The model will predict the probabilities of both ”Hi” and ”#ke”, not ”Hike” as one word. We know the word has been split in two, but we could use a variety of methods:

1. We could take the average of all of the tokens that create the word.
2. We could take the first token's probability. In this example, we would take ”Hi”'s probability and discard the prediction for ”#ke”
3. We could discard the probabilities that are a certain threshold less than the maximum probability predicted for the tokens in the word and calculate the average from what is left. Imagine if the threshold was set to 0.5. In this example, if ”Hi” was predicted to have a probability of 1 and ”#ke” has a predicted probabilty of 0.3, then the probability of ”#ke” would be discarded and the average of what is left, in this case only ”Hi”, would be calculated and used.

In the end, I chose to go with the first idea: compute the average of all tokens that create a word and use that as the word's emphasis probability. BERT is capable of seeing the entire text's context and tokens that are a part of a word, but not the start of it, are distinguished by starting with ”#” therefore tokens that create words should already have a similar emphasis to each other as BERT learns that a # means that a token is connected to the one before it. This rules out idea number 3. Choosing idea 1 over 2 is so that no prediction is completely lost as there may still be some discrepancies between tokens.

Chapter 4

Experiments and Discussion

Up until now, the model implementation has followed previous work as closely as it possibly can. This section will explain and discuss the results of the experiments I have performed using the model discussed in Chapter 3 as a baseline.

To ensure that there is no confusion, I will summarise the model here. The baseline model outputs predictions about the emphasis of a word/token as a probability between 0 and 1. The pre-trained transformer used in this model is SciBERT Cased, it also uses the default SciBERT Cased vocabulary. My intuition for choosing this model over the uncased version is that the model may benefit from seeing case in scenarios where a word is likely to be emphasised because it is CAPITALISED when others are not. The classifier that takes the output from SciBERT consists of 3 linear layers with output dimensions of 300, 20, then 1 dimension, 2 ReLU layers, a dropout layer, and a sigmoid layer. SciBERT's dropout layers and the classifier's dropout layer have all been set to 0.3 in the base case. The loss function used is Binary Cross Entropy Loss. Lastly, the model starts with its learning rate set to 2e-5 and uses the AdamW optimiser.

As a way to distinguish the differences between many probabilities, I have colour-coded the majority of tables within this chapter and any other table after this chapter to display the colour red in varying levels of saturation depending on their values. The closer a probability is to 1, the brighter hue of red the table cell will be. I aimed to make the tables as readable as possible to make analysis easier for both me and you.

4.1 Modifications

4.1.1 Probability Bucket Classifier

Along with the classifier created from previous work that is used in the baseline model, I have created a classifier that has set classes for predicting probability for ranking words between the numbers 0 to n where 0 is not at all emphasised and n is the most emphasised.
The idea here is that probability can be sorted into a number of buckets. At the most extreme case, we could say that any word in the training set with a probability equal to or over 0.5 is emphasised and any probability under is not emphasised. This unusual

method has not been looked into but is useful to consider and experiment with as it could make the model more strict on what it predicts as emphasised.

To do this, I removed the sigmoid layer and modified the dimensions of the second and third linear layers of the base classifier. The second layer still has an input dimension size of 300 but now outputs a 200 dimension vector. The third layer then takes the larger vector and outputs the same number of dimensions as the number of probability bins. Because the model was no longer working with only binary values, binary cross entropy loss wasn't used for this classifier. Instead, categorical cross entropy was used.

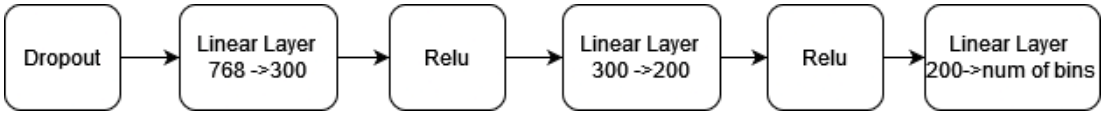


Figure 4.1: Modified Classifier

Apart from modifying the classifier, the pre-processing of the training and development data needed to be changed as well to put the emphasis probabilities into bins.The equation for creating the bins from the probabilities is shown below with "p" signifying the current emphasis probability of a word:

$$\text{n bins} = \text{round}((n-1)\text{p}, 0) \tag{4.1}$$

The experiment and the reasoning behind the number of bins used for the models trained is discussed in subsection 4.2.3.

4.1.2 Custom Vocabulary

When the pre-trained model's tokenizer finds an unknown word that it can't create out of words or combinations of letters, it assigns the word the tag of: "[UNK]" which removes context within the sentence. As previous work found that SciBERT performed better on the technical texts in the PSED dataset compared to BERT, it led me to believe that SciBERT's advantage in predicting the PSED dataset may partially be caused by a larger vocabulary that includes more scientific words. SciBERT does indeed have a larger vocabulary than BERT - 2120 tokens larger in the cased version - and if those extra tokens helped in any way, then it is useful to run an experiment on it. Another reason to research this is that we can see if we would need to add domain-specific language in cases where emphasis selection is used on technical papers as although the rules of where emphasis may not change, the missing words within the text may impact the model.

For the first custom vocabulary, I used Tensorflow's bert_vocab_from_dataset function for producing the custom vocabulary from the training set. This resulted in a list of around 2500 of the most common words or word pieces within the dataset. However, depending on the model, only 100 to 200, of these words were added to a model at a time as only tokens that did not appear in the model's base vocabulary are used. Lastly,

I created both a cased and uncased version of the custom vocabulary in anticipation of testing the model using different parameters.

However this was not the only custom vocabulary developed. Where Tensorflow's function added mostly word-pieces that were not in the original vocabulary such as "##cord" from say, "record", I also created a vocabulary of only the 1000 most common words using the spacy library's tokenizer as it did not split the words into word-pieces. To ensure that only words that were the most used within the training dataset, and therefore the most specific to the domain, I checked that all words appeared in at least 5% of the training set's texts before taking the top 1000 from that list.

As "Style: Lessons In Clarity and Grace"Williams and Bizup [2016] suggests that nouns are more effective in the last position of a sentence, the position that gets "special" emphasis placed on it, I generalised this to mean that nouns may be more likely to be heavily emphasised or dictate what will be emphasised. Therefore the last vocabulary created used spacy's POS tagger to extract all of the proper nouns used within the training set. I expected most nouns to be able to be recreated by SciBERT but names, companies, or places, especially those without an English Language origin, could have combinations of letters that are less commonly used such as "CDC" being a combination of "C", "#D","#C".

4.2 Evaluation

Not only were there models fine-tuned using the modified classifier defined in section 4.1.1, but there were also models fine-tuned on the custom vocabularies or by modifying certain starting parameters. Once a model was fine-tuned, it needed to be tested against the test set. Each model would predict the entire tokenized test set and have its results reformatted back into the right number of tokens. If the probability bucket classifier was used, then the prediction was first divided by n before being processed.

After the predictions were collected and processed, the model's results were put through the evaluation methods explained in 2.3 and written to 3 tables that could be analysed further. Why 3 tables? Well, the test set was a combination of the datasets found on the SemEval 2020 task and the CAD 21 task and because we predicted the entire test set, the texts predicted could be sorted back out into texts that came from either dataset. Therefore the models were evaluated on how well they did on the entire dataset, on texts that came from the SemEval 2020 dataset, and on texts that came from the CAD 21 dataset. This is to account for if one dataset would be considered harder for the model to predict over the other and so that we can compare the model to previous work as closely as possible.

In the rest of this section, we will first describe what was changed within the models and compare the impact of these changes on the models performance when predicting the full dataset, the sentences that came from SemEval 2020 in the full test set, and the sentences that came from CAD 21 in the full test set. Then the best models from each of those sections will be compared with each other along with a model that has been trained using the best parameters from each experiment. Lastly, this new model will be the model used within Chapter 5 to predict the survey sentences.

The metrics that the models will be evaluated against are the metrics specified in Section 2.3. These will include RANK, which is the average of the calculated Match 1 to 5, the average F1 for top 1 - 5, the average precision that was used to calculate the F1, and the AUC for the model's predictions. The reason to include both the precision and the F1 is that precision in top k is defined by how often highly emphasised words from the actual emphasis probabilities appear within the top k words that the model predicts. This makes precision the closest metric to Match M whilst also being a metric that models outside of this task have used, such as within recommendation systems.

4.2.1 Pre-trained Model Parameter Experiments

The experiments in this section pertain to how the pre-trained model can be modified and will focus on how different pre-trained models perform, and if adding to the vocabulary to the model helps it in any way. Other experiments that were not included in this section due to the constraints of this dissertation include exploring different dropout probabilities and starting learning rates.

4.2.1.1 Evaluating Pre-trained Models and their vocabulary

For this experiment, different pre-trained models were tested instead of the pre-trained model used in the baseline model. This was to see if one model is better than the other when it comes to emphasis selection on a larger dataset compared to the shared tasks. These models were those used frequently in previous work and include: SciBERT uncased, BERT cased and uncased, RoBERTa, distilBERT cased and uncased Sanh et al. [2020], and XLNet Yang et al. [2020].

SciBERT is a more specific version of BERT trained on scientific papers therefore the initial intuition was that BERT may perform worse on test sentences that come from the CAD 21 dataset compared to SciBERT as these sentences come from more technical sources. Therefore BERT was included to compare how much help the extra training was to SciBERT in this task. distilBERT was also expected to not perform as well as other models because of its size - distilBERT is much smaller than any other model (being 40% smaller than BERT) but that makes it far more suited to tasks that require faster predictions such as, say, predicting an entire document, and was chosen to see if the number of parameters a model has affects how well it performs on this task. Lastly, RoBERTa and XLNet are interesting as they both only have cased models and aim to fix problems in the original BERT model. RoBERTa is a version of BERT that has modified hyperparameters and has been further trained on a much larger corpus than the original model. XLNet is even further removed from BERT as it learns tokens in a different way. Where BERT and models like BERT learn to predict by hiding, or other words masking, 15% of the tokens in a training set and then predicting those hidden tokens, XLNet predicts every token but not always in the right order.

Base Vocabulary

The experiment seen in Table 4.1 puts all of these different models against each other to see which one performs the best using the mean of their Match 1-5 values called

”RANK”. In this table, the models are fine-tuned using their base vocabulary. It gives a sense of how well the models have learnt the task without any of the custom vocabulary.

Table 4.1: Test set predictions of different models using their base vocabulary ordered by RANK, the average of Match 1-5.The baseline model is in bold.

Complete Test Set				
Model	RANK	AVG F1	AVG Precision	AUC
bert base uncased	0.783	0.581	0.607	0.46
bert base cased	0.782	0.579	0.606	0.536
allenai scibert scivocab cased	**0.746**	**0.554**	**0.581**	**0.688**
roberta base	0.709	0.531	0.556	0.594
distilbert distilbert base cased	0.697	0.523	0.548	0.644
allenai scibert scivocab uncased	0.676	0.507	0.531	0.767
xlnet xlnet base cased	0.673	0.509	0.535	0.749
distilbert distilbert base uncased	0.664	0.496	0.521	0.76
SemEval Test Sentences				
Model	RANK	AVG F1	AVG Precision	AUC
bert base cased	0.822	0.692	0.733	0.841
bert base uncased	0.812	0.693	0.733	0.73
allenai scibert scivocab cased	**0.811**	**0.688**	**0.729**	**0.921**
xlnet xlnet base cased	0.777	0.663	0.703	0.903
roberta base	0.773	0.658	0.697	0.926
distilbert distilbert base cased	0.765	0.652	0.691	0.908
allenai scibert scivocab uncased	0.757	0.648	0.687	0.904
distilbert distilbert base uncased	0.749	0.639	0.678	0.89
CAD Test Sentences				
Model	RANK	AVG F1	AVG Precision	AUC
bert base uncased	0.753	0.463	0.473	0.421
bert base cased	0.74	0.46	0.47	0.487
allenai scibert scivocab cased	**0.678**	**0.414**	**0.425**	**0.645**
roberta base	0.642	0.398	0.406	0.535
distilbert distilbert base cased	0.625	0.387	0.397	0.592
allenai scibert scivocab uncased	0.592	0.359	0.367	0.728
distilbert distilbert base uncased	0.574	0.346	0.355	0.723
xlnet xlnet base cased	0.564	0.348	0.356	0.704

Surprisingly, BERT was the model that was the best at predicting the most emphasised words within a text in both the texts that came from the SemEval dataset and the texts that came from the CAD datasets. However, like the shared task for CAD 21, all models struggled more on the CAD task texts than they did for SemEval’s dataset. This indicates that the CAD 21 dataset is indeed ”harder” for the models to learn and predict. That dataset having longer texts on average could be the reason here, as all models perform better on shorter texts, but it could also be because there are many more words that have an emphasis probability of 0 within the set which would encourage a model to not emphasise words as strongly as it should. Let us say that in a sentence with

10 words, one is highly emphasised with a probability of 1 and 9 have no emphasis whatsoever. The model correctly predicts than 1 word out of 9 is emphasised. If the model predicts the wrong word to be emphasised with a probability of 1, then it is off by 2. 1 for emphasising the wrong word, and 1 for not emphasising the correct word But if it predicts an emphasis of 0.5 for a incorrect word, then the model is only off by 1.5, 0.5 for the incorrect word and 1 for the correct word.

What backs up this theory is how low the AUC is for some of these models. The AUC is calculated from the curve made by plotting the True Positive rate against the False Positive Rate for different cut-off probabilities that map the predicted and actual probabilities to a binary 1 or 0 depending on if they are higher or lower than the cut-off. Let us take BERT uncased as our example as it displays a significantly worse AUC than SciBERT cased although it does better in the RANK metric. If we plot the False Positive Rate and True Positive rate at different cut-off probabilities for the CAD Test Texts in Figure 4.2, we can see that model does not predict any probabilities over around 0.7 as there are neither any false positives or true positives beyond this point. Therefore the sparsity in the datasets is something that must be overcome when it comes to the task of emphasis selection.

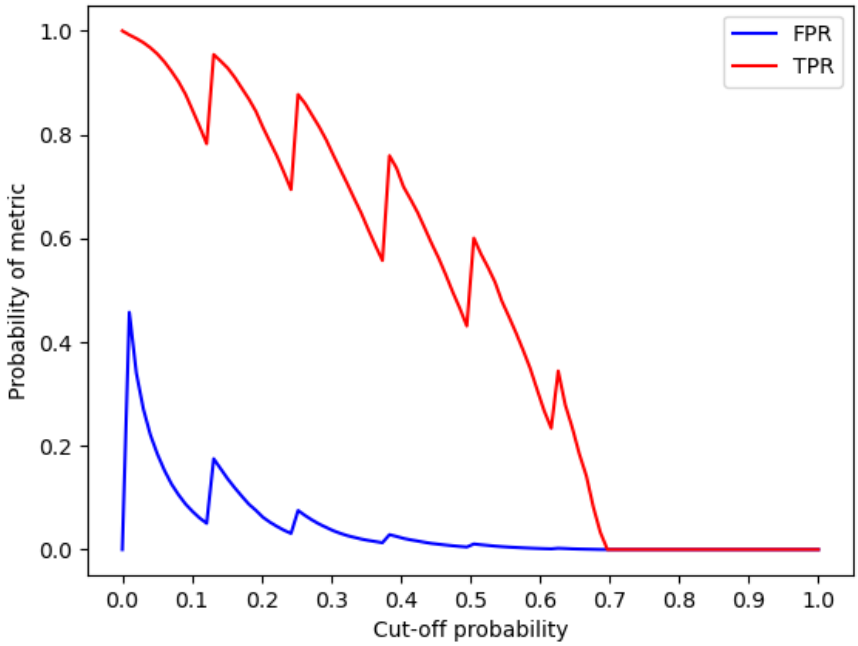


Figure 4.2: False Positive Rate vs True Positive Rate for bert uncased on the CAD Test Texts at different cut-off probabilities

Custom Vocabularies

The experiment seen in Table 4.2 shows the calculated RANK of models fine-tuned using the custom vocabularies specified in subsection 4.1.2 against the model fine-

tuned using the base vocabulary. The Base column is the RANK of the base vocabulary, the Tensorflow column is the custom vocabulary that was created using the bert_vocab_from_dataset function, the spacy column is the custom vocabulary created by using spacy to find the most common words, and the Proper Nouns column is the custom vocabulary that adds all proper nouns to the spacy vocabulary.

Table 4.2: RANK Comparison between how well models perform using their base vocabulary VS the custom vocabularies explained in subsection 4.1.2.The baseline model is in bold.

Complete Test Set				
Base Vocab	Original	Custom	Spacy	Proper Nouns
bert base uncased	0.783	0.72	0.78	0.768
bert base cased	0.782	0.704	0.792	0.752
allenai scibert scivocab cased	**0.746**	**0.74**	**0.817**	**0.778**
roberta base	0.709	0.628	0.676	0.712
distilbert distilbert base cased	0.697	0.71	0.679	0.734
allenai scibert scivocab uncased	0.676	0.706	0.795	0.731
xlnet xlnet base cased	0.673	0.562	0.669	0.622
distilbert distilbert base uncased	0.664	0.633	0.683	0.77
SemEval Test Sentences				
Base Vocab	Original	Custom	Spacy	Proper Nouns
bert base cased	0.822	0.753	0.813	0.783
bert base uncased	0.812	0.78	0.813	0.786
allenai scibert scivocab cased	**0.811**	**0.785**	**0.856**	**0.827**
xlnet xlnet base cased	0.777	0.659	0.765	0.739
roberta base	0.773	0.688	0.735	0.768
distilbert distilbert base cased	0.765	0.768	0.74	0.774
allenai scibert scivocab uncased	0.757	0.769	0.841	0.774
distilbert distilbert base uncased	0.749	0.712	0.705	0.788
CAD Test Sentences				
Base Vocab	Original	Custom	Spacy	Proper Nouns
bert base uncased	0.753	0.658	0.746	0.75
bert base cased	0.74	0.653	0.771	0.722
allenai scibert scivocab cased	**0.678**	**0.694**	**0.777**	**0.727**
roberta base	0.642	0.566	0.616	0.654
distilbert distilbert base cased	0.625	0.648	0.617	0.694
allenai scibert scivocab uncased	0.592	0.64	0.747	0.688
distilbert distilbert base uncased	0.574	0.55	0.661	0.753
xlnet xlnet base cased	0.564	0.462	0.568	0.5

Overall, no custom vocabulary significantly benefited the model. The tensorflow vocabulary was detrimental to most models, the spacy vocabulary that did not add all of the proper nouns performed around as well as the base vocabulary, and lastly we have the spacy vocabulary that did add the proper nouns was the vocabulary that shows the most promise in improving the models' performances. We must take these results

with a grain of salt however as performance varies between models fine-tuned using the exact same parameters. For example, although the spacy dataset does not add many words to the base vocabulary, SciBERT cased performed a lot better with this extra vocabulary. Therefore as no model benefits from all vocabularies if we count cased and uncased as the same model, we cannot say that simply adding any common vocabulary will help the model in any way. The problem here is that the training and test sets are rather small and the majority of common words for this task are already in this model.

4.2.1.2 Dropout

Lastly in the line of experiments that directly affect the pre-trained model, we have the dropout layer experiment. When initialising the pre-trained model, we can specify the probability that weights randomly get set to 0 when passed through the pre-trained model's dropout layers. For example, if we set the dropout probability to 0 then the pre-trained model would never drop weights and therefore could be more prone to over-fitting. Therefore if we see a dip in how well the model performs when we lower the dropout probability, then we can confirm that the model has a higher tendency to over-fit on the training set. For this experiment, the dropout of the classifier was fixed at 0.3, only the pre-trained model's dropout probabilities were changed. The values chosen range from both extremes - from a probability of 0 to a probability of 1 - to see when the dropout probability becomes a problem. The table for this experiment is table A.2.

4.2.2 Classifier Parameter Experiments

Along with experiments that affect the pre-trained model, I fine-tuned multiple models to examine aspects of the classifier in order to see how much the classifier impacts the model as a whole. This includes the probability bucket classifier introduced in Subsection 4.1. However, in order to discuss the results of the probability bucket classifier and other experiments in more detail, tests that produced results that required smaller discussions have been moved to Chapter B in the appendix. Therefore I will briefly summarise the moved experiments before moving onto the main experiment of this section, the modified classifier.

Varying the number of linear layers

The baseline model used 3 linear layers to convert the pre-trained model's output from a 768 dimension vector down to a 1 dimension vector that could be normalised in a sigmoid layer to get the emphasis probabilities. But previous work does not say why they needed 3 layers. Therefore this experiment aimed to test if the model is just as effective if it only uses one or two linear layers. The results suggested that the model is indeed as accurate at predicting emphasis probabilities when it has less linear layers. This also means that when future work tries to improve how well models performs on an emphasis selection task, that they should focus on the pre-trained model's parameters and the quality of the training dataset as the classifier does not significantly affect the model.

Varying the number of dropout layers

The baseline model used has one dropout layer, but previous work suggests that they may have used more than one for the model specified in IITK's paper Singhal et al. [2020]. Thus I aimed to see if the dropout layer within the classifier is necessary or if there should be more - needing more dropout layers of the same probability would suggest that the pre-trained model is incapable of counteracting all of the effects of over-fitting. In this experiment, extra models were fine-tuned with either 0, 2, or 3 dropout layers. The results suggested that one dropout layer has little impact on the model as a whole. 2 layers appear to be better than 1 or 3 but the difference, especially in the average precision, is small enough to not be able to confirm if this is the case or if this was simply luck as a model fine-tuned with the exact same specifications as another model may be slightly more or less accurate than the original.

Varying the dropout probability

For the last small experiment, I fine-tuned models with different dropout probabilities for the classifier's dropout layer. Just like the pre-trained model version, only one type of dropout changes in this experiment. This time the pre-trained model's vocabulary stays put and the classifier's dropout is what changes. The same values were tested with the classifier as with the pre-trained model for consistency. In this experiment, we see how much the classifier affects how much the model over-fits the training data. Interestingly, the same values that perform well with the pre-trained model also perform well on the classifier. It looks like using the values 0.9 or 0.1 for both the pre-trained model's dropout value and the classifier's dropout value is the most beneficial combination with this dataset, unlike the baseline that is set at a probability of 0.3.

If we were to fine-tune the model for longer or on a larger dataset then it would be better to use a probability of 0.1 for both the pre-trained model and classifier. This is because using such a high dropout probability could affect how much information the model has as the majority of weights are set to 0. I am unsure why it has not happened here for either experiment but either the model requires less weights than the model has to successfully predict emphasis prediction or the model has not trained for long enough for it to be a problem.

4.2.3 Probability Bucket Classifier

For this experiment, models were fine-tuned using the classifier defined in 4.1 and then compared to the baseline which is the model with 1 label in the table. The number of labels corresponds to the number of bins that the model used. The values selected explore the extremes of both too many and too few classes. If 1 label was selected using the modified classifier then the model would predict every word to be 0, therefore the minimum labels that the classifier can use is 2. The training data was rounded to 2 decimal places therefore increasing the number of bins to over 101 was expected to not achieve much but help show if there was a trend to increasing the bins. The results for this experiment can be seen in table

What stands out the most, apart from the obvious difference in RANK between the

baseline and all other models in this experiment, is that the area under the curve for the ROC curve is extremely low for any probability bucket classifier model. This metric was calculated in the exact same way to every other experiment and the function that calculates the area came from scikit-learn so any problems would be from creating the ROC curve itself. Because the ROC curve plots the true positive rate against the false positive rate, it is useful to examine these rates at different cut-offs like in figure 4.2. Let us examine the model that was second-best to the baseline, the model that used 11 bins.

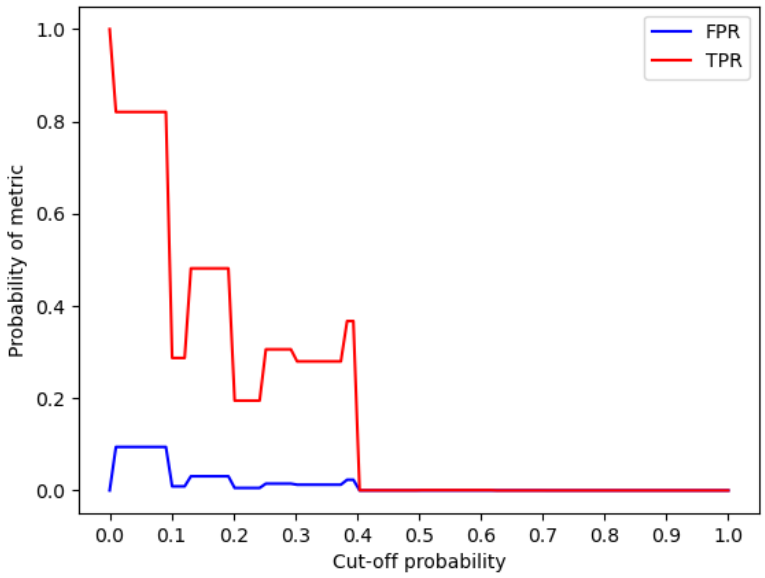


Figure 4.3: True Positive Rates and False Positive Rates at different cut-off probabilities for the model with 11 bins/labels.

What has happened in Figure 4.3 is that the model frequently predicts words at a much lower emphasis than the actual probabilities. Its maximum predicted probability is even lower than 4.2 at around 0.4. Sadly, this all means that the probability bucket classifier is not a suitable substitution for the baseline classifier as it negatively impacts how well the model learns how to predict emphasis by encouraging it to predict many words with a low emphasis.

4.2.4 Comparison to previous work

Now that we have tested various models to see what does or does not work, we can fine-tune a model on only the very best parameters and see how it compares to the two shared tasks previously done on emphasis selection. This new model was fine-tuned with the current parameters:

Before we continue, it must be noted that as the dataset is not the actual test set, the results of my model will be an approximation to how the model could have performed on the shared tasks, not how well it would have performed. Just like with other experiments,

Table 4.3: Probability Bucket classifier models ordered by RANK. The baseline model is in bold.

Complete Test Set				
Labels	RANK	AVG F1	AVG Precision	AUC
1	**0.746**	**0.554**	**0.581**	**0.688**
51	0.623	0.456	0.477	0.121
11	0.605	0.446	0.468	0.07
21	0.586	0.439	0.46	0.084
6	0.57	0.433	0.451	0.101
16	0.556	0.409	0.426	0.274
1001	0.555	0.405	0.424	0.029
501	0.555	0.402	0.419	0.05
201	0.527	0.392	0.41	0.029
101	0.523	0.387	0.404	0.043
2	0.493	0.433	0.466	0.057
SemEval Test Sentences				
Labels	RANK	AVG F1	AVG Precision	AUC
1	**0.811**	**0.688**	**0.729**	**0.921**
51	0.708	0.61	0.646	0.462
11	0.697	0.601	0.636	0.146
2	0.678	0.65	0.689	0.209
21	0.675	0.59	0.624	0.257
101	0.672	0.583	0.615	0.103
16	0.671	0.575	0.608	0.557
6	0.665	0.592	0.627	0.278
201	0.647	0.563	0.597	0.026
1001	0.634	0.546	0.58	0.087
501	0.625	0.535	0.565	0.083
CAD Test Sentences				
Labels	RANK	AVG F1	AVG Precision	AUC
1	**0.678**	**0.414**	**0.425**	**0.645**
51	0.535	0.295	0.298	0.041
11	0.508	0.284	0.29	0.05
21	0.493	0.282	0.287	0.051
501	0.483	0.264	0.265	0.038
1001	0.473	0.257	0.258	0.019
6	0.471	0.267	0.264	0.05
16	0.436	0.235	0.233	0.21
201	0.401	0.212	0.213	0.026
101	0.367	0.183	0.182	0.029
2	0.301	0.205	0.231	0.007

we can find this approximation by splitting the test set back into texts that come from the SemEval task or the CAD task. However, comparing my model to previous work allows

Table 4.4: The model parameters of the current model being compared to previous work.

Model	Vocabulary	Starting Learning Rate
BERT uncased	Bert Base Uncased	4.6e-05
Labels	**Pre-trained Model Dropout**	**Classifier Dropout**
1	0.9	0.1
Number of Linear Layers	**Number of Dropout Layers**	
2	2	

us to evaluate if fine-tuning on both datasets impacts how well the model performs on the separate datasets. If the model performs poorly compared to either the SemEval or CAD tasks, then the model might not have learnt a way to predict emphasis probabilities on different styles of writing. The following tables depict how well my model performs against previous models. Table 4.5 compares against models in the SemEval 2020 shared task and table 4.6 compares against models in the CAD 21 task.

Overall, it appears that CAD 21 dataset is definitely the harder dataset of the two given that previous models have such a low score and my model struggles on this dataset more than the SemEval one. From the results, it can be assumed that my model would be able to predict both test sets if the actual test sets were of the same difficulty as my test set. It might do a bit worse, or it might do a bit better, but we can say that fine-tuning on the combined dataset does not severely impact the model's predictions on a domain that it should be able to perform well at if fine-tuned on the separate dataset. In fact, it just solidifies the fact that there is some inherent rules that the model can learn that transfers across datasets, a topic talked about and explored in more detail in the next section.

Models	ERNIE	Hitachi	IITK	Randomseed19	**My Model**	SemEval Baseline
RANK	0.82	0.81	0.81	0.81	**0.79**	0.75

Table 4.5: Comparison of my model VS the SemEval 2020 models

Models	**My Model**	UBRI-604	DeepBlueAI	Cisco	Baseline	Zouwuhe
RANK	**0.65**	0.53	0.52	0.52	0.47	0.47

Table 4.6: Comparison of my model VS the CAD 21 models

Chapter 5

Survey

5.1 Motivation

Because there are a few datasets available, and each dataset only had a maximum of 9 annotators, there may still be more patterns in the way that people annotate emphasis that are not present in the current data. Shirani et al. [2019] and Shirani et al. [2021b, 2019] found a significant enough Kappa score for them to assume that people do have different opinions on what words should be emphasised. But what happens if we add more people into the mix? This was one of the goals of my survey - to see if there is a common pattern to how people emphasise sentences regardless of the source of the sentence and the backgrounds of the annotator. Of course, I could rely on the current datasets but increasing the number of possible opinions is what I believe should be looked into further. This is especially important as we do not want the model to simply be learning the opinions of the current annotators when the emphasis probabilities created by a much larger audience could disagree with them.

Along the lines of searching for patterns, I would like to focus on how emphasis selection relates to the suggestions in the emphasis and stress section of ”Style:Lessons in Clarity and Grace”Williams and Bizup [2016]. There, it is said that words at the end of a sentence are more stressed than those at the start. There is also a hierarchy when it comes to what types of words create emphasis. Nouns or verbs are more commonly emphasised or stressed than prepositions. Therefore my exploration of the results will look into if all datasets and my survey have a common pattern to where emphasis is.

Lastly, I wanted extra, different, sentences for my model to predict using text that was yet another style that is not seen in any of the other datasets. Specifically, text that could be read as a small paragraph in a book. Something that doesn't need to be emphasised graphically unlike the posters, adverts, and presentation slides that my model was fine-tuned on. Another difference is that the datasets are randomly chosen from various sources, texts do not follow on from one another. This would not be the case in an academic paper. However, I chose to create a short story instead of an excerpt from an academic paper as I wanted the survey to be simple and accessible to any participant and it was much easier to implement the rules from ”Style:Lessons in Clarity and Grace”Williams and Bizup [2016] in this format. Choosing to create a short

story gives me a better idea on how well the model could work on longer documents. It also brings in more examples from a task with very little work done on it.

5.2 Structure

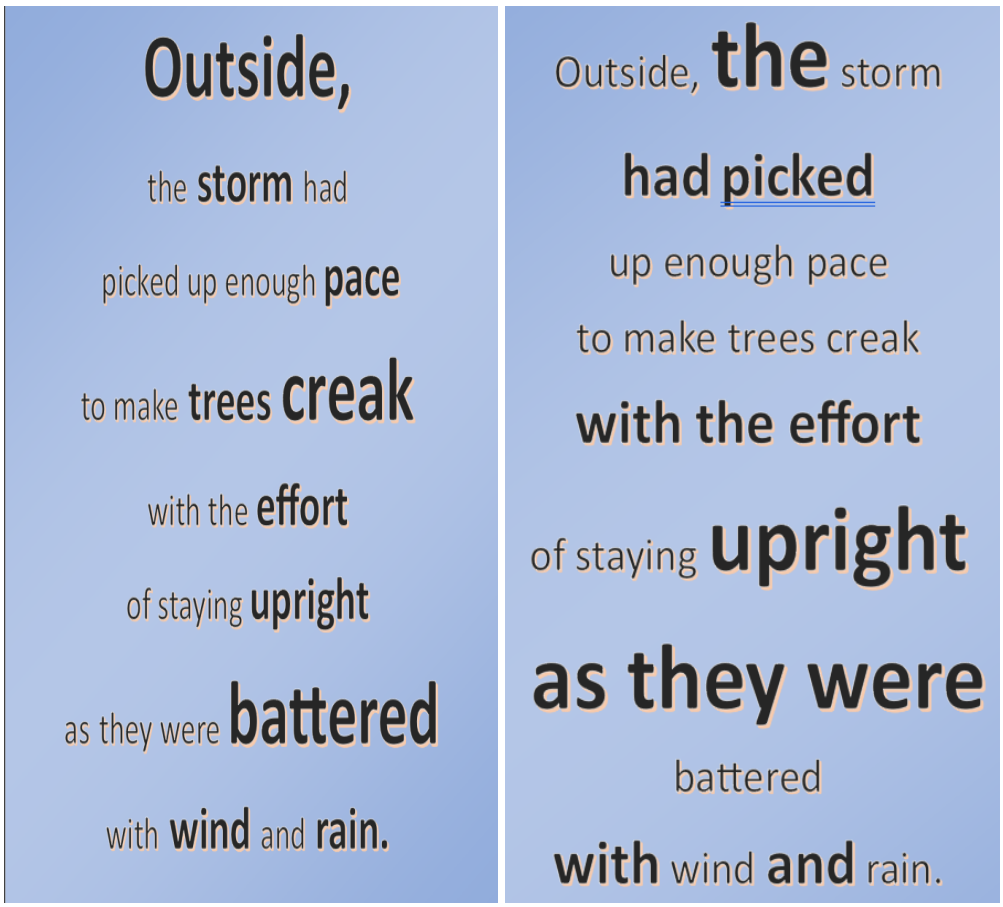
The survey's aim was to get as many participants as possible. The task of annotating can also be seen as tedious. Because of this and the fact there was no reward for participants to keep on annotating sentences, I kept the survey short. Overall, there are 5 sentences that participants were asked to annotate, and one example sentence to help them understand what emphasis selection is. In the 5 main sentences, punctuation is split up from words to mimic the format of the datasets and because certain punctuation such as exclamation marks might be seen as having their own emphasis. All the sentences together create the short passage as follows:

1. Outside, the storm had picked up enough pace to make trees creak with the effort of staying upright as they were battered with rain and wind.
2. He felt like he had forgotten something.
3. He tried to focus on his book regardless of that feeling, but an annoyingly persistent dripping noise put a stop to that.
4. That's when he remembered- there was a hole in his roof!
5. And now not only did he have an increasingly large hole in his roof, but also a sizeable puddle soaking into his carpet.
6. Today was not his day.

The first section of the survey provides two example annotations using sentence 1 to help participants understand how the rest of the survey will work and that there are no right or wrong answers when they are annotating the sentences. I decided that it was important to provide a clear walk-through on how emphasis selection works before going into the actual survey because the task is so open-ended, I did not want people to worry about giving the wrong answers. Because previous work relies on visual media, I chose to show the two versions of sentence 1 as two posters, seen in figure 5.1, as well as rewrite them with emphasised words in bold. The aim for the two posters was for the poster seen in 5.1a to be easier to understand at a first glance compared to the poster seen in 5.1b and to walk participants through why that might be the case from my perspective.

After the introductory section of the survey, participants were then asked two questions for each sentence from 2 to 6:

1. What words do you think **are emphasised** in this sentence?
2. What word do you think is **most emphasised** in this sentence?



(a) Variant 1 (b) Variant 2

Figure 5.1: Two possible ways to emphasise sentence 1

5.3 Results

5.3.1 Participant results

By the end of two weeks, there had been 8 responses to the survey. Each participant answered every question. The responses from the participants will be shown in the next two sections depending on the type of question. If the question asked for participants to answer what words they thought were emphasised in a sentence, then the collected responses will be shown in Section5.3.1.1. If the question asked for participants to answer what word they thought was most emphasised, then the collected responses will be shown in 5.3.1.2.

To calculate inter-annotator agreement, Fleiss' Kappa will be used. This metric specialises in calculating the inter-annotator agreement between 2 or more annotators and is a variant of Cohen's Kappa, a metric that only compares 2 annotators at a time. For calculating the method, the statsmodel library sta was used therefore the explanation of the metric will rely on how their method is implemented.

Fleiss' Kappa is calculated by the following equation:

$$\begin{aligned} \kappa &= \frac{p_o - p_e}{1 - p_e} \\ p_o &= \text{The observed agreement of the annotators} \\ p_e &= \text{The expected agreement of the annotators} \end{aligned} \tag{5.1}$$

The observed agreement is the mean of the percent agreements of each word in each sentence. The percent agreements in this scenario is how far off the survey results are from a word being rated as emphasised or not emphasised by all participants. For example, a word that no-one believes is emphasised has a percent agreement of 1 - the same is true if if everyone thought a word was emphasised. Thus the furthest away the agreement can be is when half of the participants believe a word to be emphasised.

The expected agreement is calculated using the marginal frequencies of how many words were said to be emphasised vs not emphasised. These frequencies are squared and then summed for the expected agreement.

The range of Fleiss' Kappa is from -1 to 1. If the score is negative, then the participants disagreed more than the expected agreement. If the score is positive, then the participants agreed more than the expected agreement. However, there is no threshold where we can say that the participants agree or disagree significantly. But the value is useful as it can be turned into a z score and from that, we can get a p-value.

5.3.1.1 Emphasis probabilities

The emphasis probabilities were created similarly to how the emphasis probabilities in the original datasets were calculated [2.2.3], however the survey lacked the difference between the B and the I tags and therefore did not have a separate tag for annotators to say whether they thought a span of emphasised words followed on from each other. Although this lowers the amount of information we can gather from analysing only the results of the survey, this does not make a difference to the emphasis probabilities themselves.

For this section, the Fleiss' Kappa was 0.572. This is considerably higher than that seen in Shirani et al. [2020] with 0.246 and Shirani et al. [2021a] with 0.2092, however there are multiple reasons that can explain this large difference. The number of participants, 8, equals the number of annotators used in the PSED dataset Shirani et al. [2021a] therefore the number of annotators does not influence the score. However, the survey is much, much, shorter than the original datasets which will definitely be a factor in increasing the Kappa score. In a larger dataset, there are many more chances for ambiguous sentences or clauses where annotators disagree on what should or should not be annotated. As all of the sentences in the survey were written by one person, the style of the survey sentences may also play a part. They were written to be easy to understand, easy to understand sentences on principle should be less ambiguous than more complex sentences. The style of a sentence playing a part on how often the annotators agree is not a new idea, in fact, sentences that annotators found less understandable in the PSED dataset were removed altogether as they lowered the Kappa score significantly Shirani et al. [2021b]. Overall, we cannot assume that this survey

performed well simply because it has a high kappa score. Instead, we will focus on comparing the average position of emphasis and the most commonly emphasised POS tags in the survey responses with the other datasets in section 5.3.2.

Table 5.1: Sentence 2

Responses	2	7	0	0	8	5	0
Emph Probs	0.25	0.875	0	0	1	0.625	0
Words	He	felt	he	had	forgotten	something	.

Table 5.2: Sentence 3

Part 1							
Responses	2	3	0	8	4	1	7
Emph Probs	0.25	0.375	0	1	0.5	0.125	0.875
Words	He	tried	to	focus	on	his	book
Part 2							
Responses	4	0	1	3	0	1	0
Emph Probs	0.5	0	0.125	0.375	0	0.125	0
Words	regardless	of	that	feeling	,	but	an
Part 3							
Responses	3	4	4	7	0	0	7
Emph Probs	0.375	0.5	0.5	0.875	0	0	0.875
Words	annoyingly	persistent	dripping	noise	put	a	stop
Part 4							
Responses	0	0	0				
Emph Probs	0	0	0				
Words	to	that	.				

Table 5.3: Sentence 4

Part 1							
Responses	2	0	0	1	8	0	1
Emph Probs	0.25	0	0	0.125	1	0	0.125
Words	That	's	when	he	remembered	-	there
Part 2							
Responses	0	0	8	0	1	8	2
Emph Probs	0	0	1	0	0.125	1	0.25
Words	was	a	hole	in	his	roof	!

5.3.1.2 Max Emphasis

The full survey results for this section can be seen in C.0.1.

Table 5.4: Sentence 5

Part 1							
Responses	0	2	1	1	0	0	1
Emph Probs	0	0.25	0.125	0.125	0	0	0.125
Words	And	now	not	only	did	he	have
Part 2							
Responses	0	5	7	8	0	0	6
Emph Probs	0	0.625	0.875	1	0	0	0.75
Words	an	increasingly	large	hole	in	his	roof
Part 3							
Responses	0	1	0	0	5	7	2
Emph Probs	0	0.125	0	0	0.625	0.875	0.25
Words	,	but	also	a	sizeable	puddle	soaking
Part 4							
Responses	0	0	6	0			
Emph Probs	0	0	0.75	0			
Words	into	his	carpet	.			

Table 5.5: Sentence 6

Responses	7	0	8	2	3	0
Emph Probs	0.875	0	1	0.25	0.375	0
Words	Today	was	not	his	day	.

In this section, the Fleiss' Kappa was 0.373. This is understandable as for the majority of words with any sort of vote towards them, only 1 or 2 annotators believed they were the most emphasised. This leads to more annotators not choosing the most popular choice, and with the way that Fleiss' Kappa is calculated, lowering the annotator agreement. As no other dataset looked into this type of question, there is no score to compare to. Instead, it is more useful to see where participants agree or disagree in order to understand how people's thinking changes when they are confined to choosing 1 word to emphasise.

The probabilities calculated here represent how many people thought a word was most emphasised in a sentence. They were only allowed to choose one word therefore it is much more sparse than the usual emphasis probabilities calculated. For the below table, any tag with no word that was picked at least once as being the most emphasised will not be shown. Because different categories of verbs appear in the results, they will be shown as separate rows and combined within the row **VERB**.

By analysing this table, we can see that although nouns are more commonly thought to be most emphasised, annotators are more likely to agree that verbs are the most emphasised words in a sentence. This as interesting as it contradicts the ordering of most to least emphasised words specified in "Style:Lessons in Clarity and Grace"Williams and Bizup [2016].

Table 5.6: POS Tag Results

Current Tag	Words with max emphasis	Avg Prob	Highest Prob
VB	1	0.625	0.625
VBD	3	0.167	0.25
VBN	1	0.875	0.875
VERB	5	0.4	0.875
NN (Noun)	7	0.232	0.5
DT (Determiner)	1	0.125	0.125
RB (Adverb)	3	0.375	0.75
JJ (Adjective)	1	0.125	0.125

Another surprise was the number of adverbs that participants voted as the most emphasised words of a sentence, especially how many of them voted for ”not” in sentence 5. One would initially think that like adjectives, adverbs would be a less common choice. But if we analyse the adverbs that have been chosen, we can see that each word provides a lot of information within the sentence or within the English language in general. ”Now” in sentence 4 is an example of where it is a general case. Time is an important part of communication. As participants were prompted to answer in a way that would convey a story most clearly, words that convey time becomes important if you need to quickly tell a reader an order of events. This idea can be backed up by 2 participants choosing ”Today” to be the most emphasised word in sentence 5.

However, as is the case with the previous section, we cannot be certain that the patterns seen here will transfer to a larger dataset. Further work that explores this specific part of emphasis selection on a larger and more varied dataset would be the best way to prove or disprove what types of words annotators believe are most emphasised within a sentence.

5.3.2 Dataset Comparison

To explore the survey effectively, we must compare it to previous work. One way in which to do this is to try and find any common patterns within the datasets. Finding common patterns within all of the datasets and the survey also will indicate if emphasis selection can be generalised across different types of texts. Focusing on the inspiration for this project, ”Style:Lessons in Clarity and Grace”Williams and Bizup [2016] suggests that words at the end of a sentence are more stressed/emphasised than words closer to the start of the sentence. More importantly, it tells us what types of words the author believes should be most emphasised. The aim of this section is to explore if the survey and datasets’ most emphasised words are close to the end of the sentence and what words have the largest average emphasis probabilities.

5.3.2.1 Position of max emphasis

It is important to note that in this section, maximum emphasis relates to the word with the highest emphasis probability in a sentence using the normal emphasis probabilities within the datasets/survey. The survey’s version of maximum emphasis cannot be

compared to emphasis probabilities as the latter is a less limited task which could cause annotators to act differently to if they knew they must only choose one word to be emphasised.

When thinking about finding the position of maximum emphasis, the only special case is when two or more words have the exact same maximum emphasis probability. In this case, we use the mean of all of the positions in the sentence where this maximum value is found. We could take the first instance or last instance of the word but this would bias the results towards favouring the start or end of a sentence.

After finding the maximum position, the position must then be divided by the sentence length to convert the distance into the range: [0,1], with 0 being the start of the sentence and 1 being the end of the sentence. This ensures that how far away the maximally emphasised word is to the start or end of a sentence is not dependent on how long the sentence is. Thus if the most emphasised word is in the middle of a 7 word sentence, and another sentence's most emphasised word is also in the middle, then the two maximum emphasis positions would both be 0.5.

SemEval	CAD	Survey	All
0.431	0.467	0.405	0.442

Table 5.7: Average position of maximum emphasis

However, if we look at the average positions for maximum emphasis in table 5.7, we find that the most emphasised word is usually nearer the middle, but often closer to the start of the sentence, rather than the end. This is true even with the 5 examples from the survey. From this, we can say that although the author may find that emphasis should be placed on the last word of a sentence, readers prefer to emphasise words in the first half of a sentence more strongly instead. This does not mean that the last word in a sentence does not matter or does not benefit from its position, but that key information in most texts are more effective if pushed to the start of a sentence and that readers notice this. To truly test the idea of extra emphasis being placed on the last word in a sentence, analysis of the average emphasis per position would need to be calculated.

5.3.2.2 POS tagging

The full set of tables can be seen at C.0.2.

For this analysis, all datasets and the survey needed to be annotated with POS tags. Thankfully, the SemEval dataset came annotated with POS tags and tells us which POS tagger was used for their results, the Stanford POS tagger. In order to keep consistency between the POS tags found in the SemEval dataset and the POS tags needing to be annotated onto the CAD dataset and my survey, I opted to use the same tagger specified in the task.

By simply taking the average emphasis probability of all words with the same tags, we can see that the majority of words have little to no emphasis, especially for the CAD

Match 1	Match 2	Match 3	Match 4	RANK
0.4	0.5	0.53	0.65	0.52

Table 5.8: Model Match$_m$ results on survey sentences

dataset. The sparsity in the number of words with a high emphasis indicates that there is a pattern to how people annotate emphasis regardless of the type of text or annotator background. Or, more simply, there is a pattern to how people know what words not to emphasise. Take the Determiner tag ”DT” as an example. Although there are 8098 instances throughout all of the datasets and the survey, the average for this tag does not surpass 0.15. That is, on average, nearly 1 annotator per word that believes it should be emphasised!

Comparing the results to how the author for ”Style:Lessons in Clarity and Grace”Williams and Bizup [2016] ranks types of words within chapter 11, from most to least emphasised, we see a couple of common patterns between them. Nouns often rank highly regardless of how common they are, adjectives are close behind, adverbs after that, and then prepositions far, far, below them. Exactly as described. As the rest of the author's predictions were correct when it came to emphasis and ”semantic weight” in a sentence, we can assume that verbs fit in between adjectives and adverbs.

5.3.3 Model Prediction

Lastly, and most importantly, can the model that I have fine-tuned manage to predict emphasis similarly to participants that are different to the annotators from the datasets. Another challenge for the model to overcome is that the style of the survey sentences are dissimilar to those that it fine-tuned on. If it can predict emphasis on the survey sentences to a degree similar to the current datasets, then the model has learned a general solution/set of weights that still works over domains and opinions it might not have seen yet. The reason this is so important is because if it cannot do this, then we can't be sure if how we choose to emphasise words is determined by a set of rules and our opinions or strictly based only on opinion.

To give the model the best chance of performing decently on this task, I have chosen to use the same model specified in Table 4.4. The final results are the mean of all of their predictions. The full set of predictions can be found at C.0.3.

Surprisingly, the results are slightly lower for the survey. The model appears to be able to find the correct words just over half of the time on average, but missing one word in a small survey such as this creates a much larger drop in the evaluation score than if it were to miss one word in the test set. The differences between the survey results' emphasis probabilities and the model's predictions show us where the model still struggles to emulate human annotators. The model's predictions in general are often lower than the survey's probabilities, especially where the majority of participants agreed that a word was emphasised. In fact, the model rarely goes above 0.6 in its predictions. If we were to measure the model by how close it gets to the original probabilities, then it would perform poorly. A possible reason for this is that it has learned to produce a low probability as the majority of words within its training datasets

have 0 or little emphasis as discussed in 5.3.2.2. Therefore instead of learning that some words have incredibly high emphasis, it has learnt that many words in a sentence have a little or moderate amount of emphasis. One way that this could be minimised would be to remove sentences from the combined dataset where the average emphasis for the sentence is below a threshold or the number of words with some emphasis are below a certain percentage of the entire sentence. This could encourage the model to increase its prediction probabilities as words with at least some emphasis would be more likely in the modified dataset.

However, as we currently do not care for exact probability predictions, it is amazing to see that the model can indeed predict emphasis probabilities - it hasn't simply learned an opinion! This indicates that the task of emphasis selection consists of some latent rules combined with an annotator's opinion. These rules are the same across different domains and different sets of annotators. From what has been analysed, the latent rules of what words we deem to be emphasised is dependent on position and type. With words that convey the most meaning closer to the beginning of a sentence, such as a paragraph topic in an introductory sentence, having the most emphasis. Now that we know that emphasis selection does have common factors that the model can learn, it would be interesting to delve deeper into said common factors to see if a more concrete description of the task could be created for future use.

Chapter 6

Conclusion

Throughout this project we have seen how emphasis probabilities can be calculated, how they are used, and how we can predict them. Most importantly, we have found that emphasis selection is a task that must have a general solution or rules to it that a model can learn to predict and our exploration of what is similar between the datasets and the survey helps to back up this theory. However, investigating these rules in more detail is left as a potential angle for future papers.

I succeeded in my biggest goals of replicating previous work as best I could with the available datasets and seeing how changing parameters changes how well the model does on the test set. These experiments showed me that the dataset is balanced enough to avoid the model overfitting, but that the number of words that have no emphasis can cause models to be more conservative with their predictions.

When it came to modifications on the baseline model, I found that my classifier was woefully inaccurate compared to the original implementation and avoided predicting words with a high emphasis probability. I also found that adding pronouns to a tokenizer's vocabulary can help give the model more context, but that it would be more useful to add vocabulary in a setting where there could be many words that the vocabulary does not have such as in a highly technical scientific paper.

Performing a survey on emphasis selection was important to see if the model could predict emphasis when given text that was annotated by different people from its training data. The results showed that the model was capable of this, and that it had learnt to predict on something other than the annotators' opinions, and therefore I looked into similarities between the survey and the combined dataset. Overall, I found that the similarities in what words are or are not emphasised often line up with the suggested order of emphasis in "Style: Lessons in Clarity and Grace"Williams and Bizup [2016], but that the last word of a sentence is not the most common spot for the highest emphasis.

In conclusion, future work in emphasis selection can rest easy knowing that there are some fundamental rules to the task, although we do not know them yet.

Bibliography

URL https://www.statsmodels.org/dev/generated/statsmodels.stats.inter_rater.fleiss_kappa.html.

Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1371. URL https://aclanthology.org/D19-1371.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Sreyan Ghosh, Sonal Kumar, Harsh Jalan, Hemant Yadav, and Rajiv Ratn Shah. Cisco at aaai-cad21 shared task: Predicting emphasis in presentation slides using contextualized embeddings, 2021.

Zhengjie Huang, Shikun Feng, Weiyue Su, Xuyi Chen, Shuohuan Wang, Jiaxiang Liu, Xuan Ouyang, and Yu Sun. ERNIE at SemEval-2020 task 10: Learning word emphasis selection by pre-trained language model. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1456–1461, Barcelona (online), December 2020. International Committee for Computational Linguistics. doi: 10.18653/v1/2020.semeval-1.190. URL https://aclanthology.org/2020.semeval-1.190.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to Information Retrieval, July 2008. URL https://www.cambridge.org/highereducation/books/introduction-to-information-retrieval/669D108D20F556C5C30957D63B5AB65C/evaluation-in-information-retrieval/1DE7AC4EC9996792D8590212D735AF15. ISBN: 9780511809071 Publisher: Cambridge University Press.

Gaku Morio, Terufumi Morishita, Hiroaki Ozaki, and Toshinori Miyoshi. Hitachi at SemEval-2020 task 10: Emphasis distribution fusion on fine-tuned language models. In Aurelie Herbelot, Xiaodan Zhu, Alexis Palmer, Nathan Schneider, Jonathan May, and Ekaterina Shutova, editors, *Proceedings of the Fourteenth Workshop on Semantic*

Evaluation, pages 1658–1664, Barcelona (online), December 2020. International Committee for Computational Linguistics. doi: 10.18653/v1/2020.semeval-1.216. URL https://aclanthology.org/2020.semeval-1.216.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

Amirreza Shirani, Franck Dernoncourt, Paul Asente, Nedim Lipka, Seokhwan Kim, Jose Echevarria, and Thamar Solorio. Learning emphasis selection for written text in visual media from crowd-sourced label distributions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1167–1172, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1112. URL https://aclanthology.org/P19-1112.

Amirreza Shirani, Franck Dernoncourt, Nedim Lipka, Paul Asente, Jose Echevarria, and Thamar Solorio. SemEval-2020 task 10: Emphasis selection for written text in visual media. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1360–1370, Barcelona (online), December 2020. International Committee for Computational Linguistics. doi: 10.18653/v1/2020.semeval-1.184. URL https://aclanthology.org/2020.semeval-1.184.

Amirreza Shirani, Giai Tran, Hieu Trinh, Franck Dernoncourt, Nedim Lipka, Paul Asente, Jose Echevarria, and Thamar Solorio. Learning to emphasize: Dataset and shared task models for selecting emphasis in presentation slides, 2021a.

Amirreza Shirani, Giai Tran, Hieu Trinh, Franck Dernoncourt, Nedim Lipka, Jose Echevarria, Thamar Solorio, and Paul Asente. PSED: A Dataset for Selecting Emphasis in Presentation Slides. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4314–4320, Online, August 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.377. URL https://aclanthology.org/2021.findings-acl.377.

Vipul Singhal, Sahil Dhull, Rishabh Agarwal, and Ashutosh Modi. Iitk at semeval-2020 task 10: Transformers for emphasis selection, 2020.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, volume 1, pages 173–180. Association for Computational Linguistics, 2003. doi: 10.3115/1073445.1073478.

Joseph M. Williams and Joseph Bizup. *Style: Lessons in Clarity and Grace, 12th Edition*. Pearson, 2016.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.

Appendix A

Model Experiment

A.0.0.1 Learning Rate

For this experiment, 10 other starting learning rates other than the baseline were used to fine-tune models. The learning rates range from 5e-05 to 1e-05. This range was selected from the learning rates tested in BERT's paper Devlin et al. [2019] all the way to the default learning rate for BERT. I decided to experiment with learning rates as what worked for previous work might not work for this dataset as it uses different examples. The results of this experiment can been seen in table A.1.

It can be seen that the starting learning rate does not greatly affect how well the model performs on the test set as the smallest starting learning rate is right behind the second largest starting learning rate and the worst is even the third largest. Sure, more experimentation may indicate what is the most optimal learning rate for this test set and can be performed given that previous work does this to get the most out of their model, but we are looking for a more general approach on how the model would respond to data other than the test set, not how optimised it is for this one specifically.

Table A.1: Scibert Cased Models fine-tuned using different starting learning rates. Models are ordered by RANK. The baseline model is in bold.

Complete Test Set				
Learning Rate	RANK	AVG F1	AVG Precision	AUC
4.6e-05	0.798	0.585	0.612	0.663
1e-05	0.766	0.565	0.592	0.512
2.3e-05	0.764	0.566	0.593	0.536
1.4e-05	0.763	0.565	0.591	0.659
2.8e-05	0.757	0.56	0.588	0.737
3.2e-05	0.754	0.562	0.588	0.667
2e-05	**0.746**	**0.554**	**0.581**	**0.688**
5e-05	0.741	0.552	0.578	0.685
1.9e-05	0.723	0.541	0.567	0.813
3.7e-05	0.714	0.535	0.559	0.746
4.1e-05	0.701	0.528	0.553	0.742
SemEval Test Sentences				
Learning Rate	RANK	AVG F1	AVG Precision	AUC
4.6e-05	0.839	0.704	0.746	0.886
1e-05	0.831	0.698	0.739	0.844
2.3e-05	0.825	0.696	0.737	0.896
2.8e-05	0.825	0.694	0.736	0.94
3.2e-05	0.818	0.693	0.733	0.92
2e-05	**0.811**	**0.688**	**0.729**	**0.921**
1.4e-05	0.81	0.686	0.727	0.929
5e-05	0.799	0.678	0.718	0.919
1.9e-05	0.785	0.671	0.71	0.932
3.7e-05	0.782	0.668	0.706	0.921
4.1e-05	0.779	0.668	0.706	0.913
CAD Test Sentences				
Learning Rate	RANK	AVG F1	AVG Precision	AUC
4.6e-05	0.755	0.46	0.471	0.628
1.4e-05	0.713	0.438	0.448	0.612
2.3e-05	0.699	0.43	0.44	0.478
1e-05	0.698	0.426	0.436	0.459
3.2e-05	0.688	0.425	0.434	0.622
2.8e-05	0.686	0.419	0.43	0.696
5e-05	0.68	0.419	0.43	0.638
2e-05	**0.678**	**0.414**	**0.425**	**0.645**
1.9e-05	0.658	0.405	0.414	0.782
3.7e-05	0.642	0.395	0.404	0.705
4.1e-05	0.62	0.382	0.39	0.704

A.0.1 Model Dropout

Table A.2: Models ordered by RANK. The baseline model is in bold.

Complete Test Set				
Model Dropout	RANK	AVG F1	AVG Precision	AUC
0.9	0.822	0.6	0.627	0.416
0.1	0.796	0.586	0.613	0.58
0.5	0.792	0.586	0.613	0.554
0.7	0.779	0.575	0.603	0.676
1.0	0.758	0.564	0.59	0.677
0.0	0.752	0.56	0.586	0.756
0.3	**0.746**	**0.554**	**0.581**	**0.688**
SemEval Test Sentences				
Model Dropout	RANK	AVG F1	AVG Precision	AUC
0.9	0.857	0.716	0.758	0.833
0.1	0.846	0.708	0.75	0.92
0.5	0.843	0.707	0.748	0.909
0.7	0.836	0.704	0.746	0.921
0.0	0.818	0.693	0.734	0.936
1.0	0.814	0.691	0.731	0.934
0.3	**0.811**	**0.688**	**0.729**	**0.921**
CAD Test Sentences				
Model Dropout	RANK	AVG F1	AVG Precision	AUC
0.9	0.786	0.478	0.488	0.357
0.1	0.744	0.457	0.467	0.528
0.5	0.74	0.459	0.47	0.499
0.7	0.718	0.44	0.451	0.637
1.0	0.7	0.432	0.44	0.631
0.0	0.682	0.419	0.429	0.718
0.3	**0.678**	**0.414**	**0.425**	**0.645**

Appendix B

Classifier Experiments

B.0.1 Number of linear layers

In this experiment, the number of fully connected layers used within the classifier were tested as the previous work that the baseline model is based on does not go into detail on the reasons behind their decision on choosing 2 linear layers for the classifier. Thus I chose to evaluate if adding the classifier on top of the pre-trained model helps or hinders the model. For each model, the dropout and sigmoid layers have been kept in the same positions as in Figure 3.1. For the 1 layer model, it simply takes the 768 dimension vector that outputs from the pre-trained model into 1 dimension so that it can still be viewed as a probability once it goes through the sigmoid layer. For the 2 layer model, one linear layer takes the 768 dimension vector to a 300 dimension vector, the second takes the vector from 300 dimensions back down to one. It effectively eliminates the last linear layer used in the model. From what we can see in Figure B.1, the extra

Table B.1: SciBERT Cased Models fine-tuned with different numbers of linear layers used in the classifier ordered by RANK. The baseline model is in bold.

Complete Test Set				
Classifier Layers	RANK	AVG F1	AVG Precision	AUC
2	0.766	0.567	0.594	0.578
1	0.747	0.556	0.582	0.627
3	**0.746**	**0.554**	**0.581**	**0.688**
SemEval Test Sentences				
Classifier Layers	RANK	AVG F1	AVG Precision	AUC
2	0.827	0.696	0.737	0.915
1	0.811	0.687	0.729	0.888
3	**0.811**	**0.688**	**0.729**	**0.921**
CAD Test Sentences				
Classifier Layers	RANK	AVG F1	AVG Precision	AUC
2	0.702	0.432	0.443	0.522
1	0.68	0.418	0.426	0.58
3	**0.678**	**0.414**	**0.425**	**0.645**

classifier layers do not affect the model's performance. The model does not need to be as complicated as previous work suggests to get a decently accurate model and indicates that if improvements can be made on the model in the future, it should focus on getting the most out of the pre-trained models rather than on the classifier itself.

B.0.2 Number of dropout layers

Along with the theme of seeing how simple the classifier needed to be, I experimented with the number of dropout layers the classifier needed. In the baseline model, there is only 1 dropout layer before the linear and ReLU layers. For the classifier with 3 dropout layers, there is one before every linear layer. Evaluating how the model performs with more or less dropout layers also allows us to see how much the model needs to be stopped from over-fitting as that is the primary reason to include dropout layers. If it performs well with no dropout layers in the classifier, then the dropout in the pre-trained model is enough to curb any potential over-fitting from happening before it even arrives at the classifier. If the classifier requires more, and significantly improves with more, then the problem may lie in the dataset. Thankfully, the benefits to adding more dropout

Table B.2: Models ordered by RANK. The baseline model is in bold.

Complete Test Set				
Dropout Layers	RANK	AVG F1	AVG Precision	AUC
2	0.789	0.579	0.607	0.62
0	0.754	0.561	0.587	0.686
3	0.752	0.558	0.585	0.616
1	**0.746**	**0.554**	**0.581**	**0.688**
SemEval Test Sentences				
Dropout Layers	RANK	AVG F1	AVG Precision	AUC
2	0.852	0.71	0.754	0.915
0	0.824	0.695	0.736	0.928
3	0.82	0.692	0.733	0.919
1	**0.811**	**0.688**	**0.729**	**0.921**
CAD Test Sentences				
Dropout Layers	RANK	AVG F1	AVG Precision	AUC
2	0.723	0.441	0.452	0.572
3	0.681	0.418	0.428	0.565
0	0.681	0.42	0.43	0.642
1	**0.678**	**0.414**	**0.425**	**0.645**

layers do not appear to be significant enough to indicate that there is something wrong with the training set. If that were the case, then either the training set is too small for this task or the examples were not diverse enough which caused the model to learn a pattern that worked well within the training set which is not in the test set. In fact, SciBERT's dropout layers are enough to successfully handle over-fitting without extra dropout layers.

B.0.3 Classifier Dropout

In experimenting with the dropout probabilities of the pre-trained model, it was necessary to also test how the dropout probabilities affect the classifier. Therefore this is the classifier counterpart of the experiment outlined in subsection 4.2.1.2. This means that for this experiment, the classifier's dropout is what has been changed with the pre-trained model's dropout probability fixed to 0.3.

Table B.3: Models ordered by RANK. The baseline model is in bold.

Complete Test Set				
Classifier Dropout	RANK	AVG F1	AVG Precision	AUC
0.1	0.836	0.609	0.635	0.545
0.9	0.835	0.606	0.634	0.56
0.0	0.8	0.588	0.616	0.615
0.3	**0.746**	**0.554**	**0.581**	**0.688**
0.5	0.745	0.554	0.58	0.721
0.7	0.743	0.553	0.579	0.817
1.0	0.699	0.525	0.55	0.669
SemEval Test Sentences				
Classifier Dropout	RANK	AVG F1	AVG Precision	AUC
0.9	0.877	0.723	0.766	0.892
0.1	0.869	0.72	0.763	0.922
0.0	0.85	0.709	0.751	0.927
0.7	0.814	0.689	0.73	0.932
0.5	0.812	0.688	0.727	0.938
0.3	**0.811**	**0.688**	**0.729**	**0.921**
1.0	0.768	0.658	0.696	0.915
CAD Test Sentences				
Classifier Dropout	RANK	AVG F1	AVG Precision	AUC
0.1	0.802	0.491	0.5	0.492
0.9	0.79	0.484	0.494	0.511
0.0	0.749	0.462	0.472	0.567
0.3	**0.678**	**0.414**	**0.425**	**0.645**
0.5	0.675	0.414	0.423	0.679
0.7	0.668	0.41	0.419	0.786
1.0	0.628	0.385	0.395	0.621

Appendix C

Survey

C.0.1 Most Emphasised Words

Table C.1: Sentence 2

Responses	0	1	0	0	7	0	0
Emph Probs	0	0.125	0	0	0.875	0	0
Words	He	felt	he	had	forgotten	something	.

Table C.2: Sentence 3

Part 1							
Responses	0	1	0	5	0	0	0
Emph Probs	0	0.125	0	0.625	0	0	0
Words	He	tried	to	focus	on	his	book
Part 2							
Responses	0	0	0	0	0	0	0
Emph Probs	0	0	0	0	0	0	0
Words	regardless	of	that	feeling	,	but	an
Part 3							
Responses	0	0	0	1	0	0	1
Emph Probs	0	0	0	0.125	0	0	0.125
Words	annoyingly	persistent	dripping	noise	put	a	stop
Part 4							
Responses	0	0	0				
Emph Probs	0	0	0				
Words	to	that	.				

Table C.3: Sentence 4

Part 1							
Responses	1	0	0	0	2	0	0
Emph Probs	0.125	0	0	0	0.25	0	0
Words	That	's	when	he	remembered	-	there
Part 2							
Responses	0	0	4	0	0	1	0
Emph Probs	0	0	0.5	0	0	0.125	0
Words	was	a	hole	in	his	roof	!

Table C.4: Sentence 5

Part 1							
Responses	0	1	0	0	0	0	0
Emph Probs	0	0.125	0	0	0	0	0
Words	And	now	not	only	did	he	have
Part 2							
Responses	0	2	0	1	0	0	0
Emph Probs	0	0.25	0	0.125	0	0	0
Words	an	increasingly	large	hole	in	his	roof
Part 3							
Responses	0	0	0	0	1	3	0
Emph Probs	0	0	0	0	0.125	0.375	0
Words	,	but	also	a	sizeable	puddle	soaking
Part 4							
Responses	0	0	0	0			
Emph Probs	0	0	0	0			
Words	into	his	carpet	.			

Table C.5: Sentence 6

Responses	2	0	6	0	0	0
Emph Probs	0.25	0	0.75	0	0	0
Words	Today	was	not	his	day	.

C.0.2 POS tagging results

The following tables use the POS tags defined for the Stanford POS tagger Toutanova et al. [2003]. Any tag that had less than 100 entries in the SemEval dataset, the CAD dataset, or the combined total have been ignored. The survey results are an exception to this due to how few words it has in total.

Tag	Avg	%0-0.25	%0.25-0.5	%0.5-0.75	%0.75-1	Total Num
VBN	1.0	0.0	0.0	0.0	1.0	1
NN	0.768	0.0	0.143	0.143	0.714	14
JJ	0.667	0.0	0.0	0.667	0.333	3
VB	0.562	0.5	0.0	0.0	0.5	2
VBG	0.375	0.0	0.5	0.5	0.0	2
RB	0.357	0.429	0.286	0.143	0.143	7
VBD	0.281	0.625	0.125	0.0	0.25	8
PRP	0.125	0.6	0.4	0.0	0.0	5
EX	0.125	1.0	0.0	0.0	0.0	1
PRP$	0.1	0.8	0.2	0.0	0.0	5
CC	0.083	1.0	0.0	0.0	0.0	3
.	0.05	0.8	0.2	0.0	0.0	5
DT	0.047	0.875	0.125	0.0	0.0	8
TO	0.0	1.0	0.0	0.0	0.0	2
IN	0.0	1.0	0.0	0.0	0.0	5
,	0.0	1.0	0.0	0.0	0.0	2
VBZ	0.0	1.0	0.0	0.0	0.0	1
WRB	0.0	1.0	0.0	0.0	0.0	1
:	0.0	1.0	0.0	0.0	0.0	1

Table C.6: Survey Emphasis per POS Tag

Tag	Avg	%0-0.25	%0.25-0.5	%0.5-0.75	%0.75-1	Total Num
NNP	0.538	0.169	0.258	0.317	0.255	1459
JJS	0.523	0.182	0.288	0.271	0.259	170
NN	0.498	0.22	0.277	0.296	0.207	5091
JJ	0.489	0.216	0.295	0.293	0.196	1947
NNS	0.448	0.263	0.32	0.268	0.149	1226
JJR	0.435	0.291	0.338	0.205	0.166	151
VBG	0.417	0.324	0.291	0.271	0.114	598
VBN	0.388	0.353	0.333	0.205	0.109	439
VB	0.379	0.411	0.276	0.191	0.121	2421
RP	0.368	0.403	0.295	0.209	0.093	129
CD	0.339	0.463	0.306	0.13	0.101	415
VBD	0.291	0.569	0.215	0.138	0.077	246
RB	0.286	0.561	0.252	0.135	0.051	2036
VBP	0.272	0.584	0.227	0.128	0.061	1500
PRP$	0.192	0.739	0.204	0.049	0.008	613
.	0.176	0.777	0.194	0.027	0.002	2944
VBZ	0.173	0.772	0.153	0.055	0.02	1838
PRP	0.162	0.79	0.136	0.051	0.022	2735
MD	0.161	0.784	0.156	0.045	0.015	736
WP	0.154	0.82	0.126	0.039	0.015	388
IN	0.151	0.817	0.132	0.039	0.012	2895
WRB	0.149	0.835	0.082	0.059	0.024	255
TO	0.143	0.829	0.132	0.037	0.003	1009
DT	0.136	0.848	0.112	0.033	0.007	2693
CC	0.121	0.855	0.117	0.026	0.002	820
WDT	0.092	0.932	0.041	0.027	0.0	147
,	0.088	0.926	0.066	0.009	0.0	1145
:	0.079	0.918	0.063	0.012	0.008	255

Table C.7: SemEval 2020 Dataset Emphasis per POS Tag

Tag	Avg	%0-0.25	%0.25-0.5	%0.5-0.75	%0.75-1	Total Num
NNS	0.176	0.62	0.275	0.084	0.021	8146
NN	0.168	0.642	0.252	0.085	0.022	17346
VBG	0.164	0.631	0.282	0.074	0.013	1956
NNPS	0.163	0.661	0.225	0.107	0.007	271
JJ	0.151	0.676	0.239	0.072	0.014	8863
VBN	0.134	0.711	0.227	0.053	0.009	2067
NNP	0.129	0.728	0.206	0.054	0.012	23184
VBD	0.111	0.761	0.187	0.043	0.009	1150
RB	0.104	0.801	0.136	0.045	0.019	1876
JJR	0.093	0.826	0.145	0.023	0.006	311
VB	0.09	0.814	0.152	0.03	0.004	3448
JJS	0.078	0.841	0.138	0.022	0.0	138
VBZ	0.077	0.839	0.132	0.025	0.003	1821
RBR	0.065	0.879	0.105	0.016	0.0	124
CD	0.046	0.918	0.072	0.01	0.0	2869
CC	0.046	0.912	0.083	0.004	0.0	3847
VBP	0.044	0.925	0.069	0.006	0.0	2414
FW	0.037	0.949	0.051	0.0	0.0	178
:	0.037	0.947	0.053	0.0	0.0	1556
IN	0.035	0.945	0.053	0.002	0.0	9565
,	0.034	0.948	0.05	0.002	0.0	3410
MD	0.032	0.957	0.041	0.002	0.0	999
DT	0.03	0.956	0.042	0.002	0.0	5397
TO	0.029	0.959	0.04	0.001	0.0	2265
PRP$	0.029	0.954	0.042	0.005	0.0	409
.	0.028	0.962	0.037	0.001	0.0	2585
$	0.026	0.963	0.037	0.0	0.0	109
)	0.026	0.963	0.035	0.002	0.0	839
(	0.025	0.967	0.033	0.0	0.0	765
WDT	0.023	0.973	0.027	0.0	0.0	331
WP	0.022	0.968	0.027	0.005	0.0	219
WRB	0.02	0.976	0.024	0.0	0.0	291
PRP	0.02	0.968	0.031	0.001	0.0	717

Table C.8: CAD 2021 Dataset Emphasis per POS Tag

Tag	Avg	%0-0.25	%0.25-0.5	%0.5-0.75	%0.75-1	Total Num
JJS	0.324	0.477	0.221	0.159	0.143	308
POS	0.323	0.511	0.221	0.107	0.16	131
NN	0.243	0.546	0.257	0.133	0.064	22451
RP	0.23	0.614	0.209	0.123	0.055	220
VBG	0.224	0.559	0.284	0.121	0.036	2556
NNPS	0.222	0.575	0.244	0.134	0.047	320
NNS	0.212	0.574	0.281	0.108	0.038	9372
JJ	0.212	0.593	0.249	0.112	0.047	10813
VB	0.209	0.648	0.203	0.097	0.052	5871
JJR	0.205	0.652	0.208	0.082	0.058	462
RB	0.199	0.676	0.197	0.092	0.036	3919
VBN	0.179	0.648	0.246	0.08	0.027	2507
RBR	0.167	0.751	0.122	0.09	0.037	189
NNP	0.153	0.695	0.209	0.07	0.026	24643
VBD	0.143	0.726	0.192	0.06	0.022	1404
PRP	0.132	0.827	0.115	0.04	0.018	3457
VBP	0.131	0.794	0.13	0.053	0.024	3914
PRP$	0.126	0.825	0.139	0.031	0.005	1027
VBZ	0.125	0.805	0.143	0.04	0.011	3660
.	0.107	0.863	0.121	0.015	0.001	5534
WP	0.107	0.873	0.091	0.026	0.01	607
MD	0.087	0.884	0.09	0.02	0.006	1735
CD	0.083	0.86	0.102	0.025	0.013	3284
WRB	0.08	0.91	0.051	0.027	0.011	547
DT	0.065	0.92	0.065	0.012	0.002	8098
TO	0.064	0.919	0.068	0.012	0.001	3276
IN	0.062	0.915	0.071	0.011	0.003	12465
CC	0.059	0.902	0.089	0.008	0.0	4670
EX	0.051	0.972	0.021	0.007	0.0	145
FW	0.05	0.929	0.055	0.016	0.0	183
,	0.047	0.943	0.054	0.004	0.0	4557
WDT	0.044	0.96	0.031	0.008	0.0	478
:	0.043	0.943	0.054	0.002	0.001	1812
$	0.037	0.939	0.061	0.0	0.0	115
)	0.026	0.963	0.035	0.002	0.0	839
(	0.025	0.967	0.033	0.0	0.0	765

Table C.9: Datasets and Survey Emphasis per POS Tag

C.0.3 Model Prediction Results

Table C.10: Model Prediction of survey sentence 2

Actual Prob:	0.25	0.875	0.0	0.0	1.0	0.625	0.0
Model Pred:	0.066	0.14	0.083	0.12	0.61	0.546	0.179
Words:	He	felt	he	had	forgotten	something	.

Table C.11: Model Prediction of survey sentence 3

Actual Prob:	0.25	0.375	0.0	1.0	0.0	0.125	0.875
Model Pred:	0.034	0.092	0.046	0.272	0.046	0.051	0.173
Words:	He	tried	to	focus	on	his	book
Actual Prob:	0.5	0.0	0.125	0.375	0.0	0.125	0.0
Model Pred:	0.185	0.032	0.04	0.126	0.023	0.039	0.048
Words:	regardless	of	that	feeling	,	but	an
Actual Prob:	0.375	0.5	0.5	0.875	0.0	0.0	0.875
Model Pred:	0.372	0.445	0.415	0.367	0.11	0.086	0.251
Words:	annoyingly	persistent	dripping	noise	put	a	stop
Actual Prob:	0.0	0.0	0.0				
Model Pred:	0.077	0.059	0.04				
Words:	to	that	.				

Table C.12: Model Prediction of survey sentence 4

Actual Prob:	0.25	0.0	0.0	0.125	1.0	0.0	0.125
Model Pred:	0.045	0.05	0.056	0.111	0.339	0.054	0.057
Words:	That	’s	when	he	remembered	-	there
Actual Prob:	0.0	0.0	1.0	0.0	0.125	1.0	0.25
Model Pred:	0.094	0.126	0.445	0.214	0.201	0.465	0.172
Words:	was	a	hole	in	his	roof	!

Table C.13: Model Prediction of survey sentence 5

Actual Prob:	0.0	0.25	0.125	0.125	0.0	0.0	0.125
Model Pred:	0.021	0.034	0.024	0.024	0.02	0.02	0.023
Words:	And	now	not	only	did	he	have
Actual Prob:	0.0	0.625	0.875	1.0	0.0	0.0	0.75
Model Pred:	0.026	0.147	0.154	0.146	0.064	0.035	0.155
Words:	an	increasingly	large	hole	in	his	roof
Actual Prob:	0.0	0.125	0.0	0.0	0.625	0.875	0.25
Model Pred:	0.025	0.024	0.04	0.028	0.316	0.242	0.213
Words:	,	but	also	a	sizeable	puddle	soaking
Actual Prob:	0.0	0.0	0.75	0.0			
Model Pred:	0.101	0.096	0.238	0.099			
Words:	into	his	carpet	.			

Table C.14: Model Prediction of survey sentence 6

Actual Prob:	0.875	0.0	1.0	0.25	0.375	0.0
Model Pred:	0.616	0.146	0.341	0.484	0.579	0.226
Words:	Today	was	not	his	day	.

Appendix D

Participants' information sheet and consent form

As my survey was online, the participant's information sheet and the consent form were merged into one document.

Form:

Project title: Develop AI-powered tools to help scientific authors write with Style
Principal investigator: Adam Lopez
Researcher collecting data: Rhiannon Shaw

This study was certified according to the Informatics Research Ethics Process, reference number 234962. Please take time to read the following information carefully. You should keep this page for your records.

Who are the researchers?
Rhiannon Shaw, Adam Lopez

What is the purpose of the study?
The purpose of this study is to see where people subconsciously want to stress words when reading a sentence in their heads or aloud. The information will then be used to see if the majority of participants agree on the same words and if there is a pattern to what is or isn't emphasised. The purpose of this study is to also see how similarly a language model performs to participants. This language model is being developed as a tool to help with revising academic writing by highlighting emphasis.

Why have I been asked to take part?
It is important that the tool that is being developed isn't simply learning the opinions of the few annotators that helped create the datasets it uses. What words stand out to you in a sentence could be vastly different to someone who has done the task before. Therefore, the reason that you are being asked to take part is that you have most likely never done a task in emphasis selection making your answers valuable in proving if we do have subconscious rules in where we place emphasis.

Do I have to take part?
No – participation in this study is entirely up to you. You can withdraw from the study at any time, without giving a reason. Your rights will not be affected. If you wish to withdraw, contact the PI. We will stop using your data in any publications or presentations submitted after you have withdrawn consent. However, we will keep copies of your original consent, and of your withdrawal request.

What will happen if I decide to take part?
The survey will take place in a Qualtrics online form where every submission is anonymous, no personal data will be gathered in this survey. There are 6 short sections to answer. The first section only includes instructions for the study and an example sentence and answers to help you get an idea of how the study works and what emphasis means in the context of the project. For the other sections of the survey, you will be shown a sentence and asked to select what words you feel are emphasised, or stressed, in the sentence and what word you feel looks most emphasised to you. Please feel free to answer them at your own pace. The last page before you submit your answers will ask if you are completely sure about your participation in the study.

Are there any risks associated with taking part?
There are no significant risks associated with participation

Are there any benefits associated with taking part?
No.

What will happen to the results of this study?
The results of this study may be summarised in published articles, reports and presentations. Quotes or key findings will be anonymized: We will remove any information that could, in our assessment, allow anyone to identify you. With your consent, information can also be used for future research. Your data may be archived for a minimum of 2 years.

Data protection and confidentiality.
Your data will be processed in accordance with Data Protection Law. All information collected about you will be kept strictly confidential. Your data will be referred to by a unique participant number rather than by name. Your data will only be viewed by the researcher/research team Rhiannon Shaw, Adam Lopez. All electronic data will be stored on a password-protected encrypted computer, on the School of Informatics' secure file servers, or on the University's secure encrypted cloud storage services (DataShare, ownCloud, or Sharepoint) and all paper records will be stored in a locked filing cabinet in the PI's office. Your consent information will be kept separately from your responses in order to minimise risk.

What are my data protection rights?
Rhiannon Shaw is the Data Controller for the information you provide. You have the right to access information held about you. Your right of access can be exercised in accordance Data Protection Law. You also have other rights including rights of correction, erasure and objection. For more details, including the right to lodge a complaint with the Information Commissioner's Office, please visit www.ico.org.uk. Questions, comments and requests about your personal data can also be sent to the

University Data Protection Officer at dpo@ed.ac.uk. For general information about how we use your data, go to: edin.ac/privacy-research

Who can I contact?
If you have any further questions about the study, please contact the lead researcher, Rhiannon Shaw, s2077451@ed.ac.uk. If you wish to make a complaint about the study, please contact inf-ethics@inf.ed.ac.uk. When you contact us, please provide the study title and detail the nature of your complaint.

Updated information.
If the research project changes in any way, an updated Participant Information Sheet will be made available on http://web.inf.ed.ac.uk/infweb/research/study-updates.

Consent
By proceeding with the study, I agree to all of the following statements:
- I have read and understood the above information.
- I understand that my participation is voluntary, and I can withdraw at any time.
- I consent to my anonymised data being used in academic publications and presentations.
- I allow my data to be used in future ethically approved research.