BITS PILANI

Meme Sentiment Analysis(Classification)

Dr. Sujith Thomas[1], Urvil Jivani[2], and Anupreet Singh[3]

[1]Professor, Department of Computer Science, BITS Goa

[2]Student, Department of Computer Science, BITS Goa

[3]Student, Department of Computer Science, BITS Goa

Acknowledgment

First of all, we are deeply grateful to the supreme almighty, with whose blessings we were able to accomplish this humble work. As a special mention, we would like to express our gratitude to our mentor Dr. Sujith Thomas under whom we worked on this project. His encouragement and constant support have led us to a successful attempt on this project. We would also like to thank Prof. G Raghurama, director of BITS Goa, for allowing us to study at this college. His constant support and untiring work helped us facilitate this learning experience. We want to thank Computer Science and Information Systems Departments and its faculties for their continuous support in academics, due to which we were able to work on this project. Knowledge learned from them came to be an immense help for this project.

Abstract

In this era of growing social media usage by people from all age groups, Memes are tracked by almost everyone on various platforms for entertainment. But Meme Creation might hurt the sentiments of someone(can include animals as well) or some group of people because a Meme directly or indirectly is making fun of some of their characteristics. So it is essential to detect such content with the help of machines and automatically remove it. We took the help of Deep Neural Networks, which have solved many classification tasks with better results than other methods. Scientifically this area of research is very niche and by far the best problem to judge the performance of a Multimodal model(as we have image and text as input variables). Task and dataset(used in this experiment) are designed to be difficult for unimodal models if we can't miss any input part text or image; hence they struggle to reason. So we will be exploring two required fields of research, Computer Vision and Natural Language Processing, for learning proper encoding of the image and text, respectively, and use them to classify a meme as Hateful vs. Non-Hateful (i.e., binary classification).

Key words: Computer Vision, Natural Language Processing, Deep Learning, Multi-Modelling

CONTENTS

1 INTRODUCTION

With growing research in the field of Computer Vision and Natural Language Processing. There are very few occasions where we use both of them together as it is difficult to tune both problems into the same problem. Multimodal tasks are interesting because many real-world problems are multimodal in nature—from how humans perceive and understand the world around them to handling data that occurs on the Internet. Many researchers also believe solving multimodal problems with human-level performance will be a big step toward AGI(Artificial General Intelligence). Multimodal meme classification measures truly the multimodal understanding and reasoning, as we have seen in the case of VQA(Visual Question Answering) and Machine Translation simple language-based baseline models without sophisticated multimodal understanding performed remarkably well. This is because we can come up with a meme where we can take a harmless image and equally harmless text but when mixed together becomes mean. In upcoming sections, this argument will be discussed further and introduce the objective, describe the data preparation process, and its splits.

1.1 OBJECTIVE

- Measuring progress on multimodal understanding and reasoning.
- Hate speech detection for real-world application.

1.2 DATA DESCRIPTION

In this experiment dataset used is from a Competition(**link**) organized by Facebook AI. All the memes were constructed from scratch using custom-built tools and third-party annotators were trained to employ hate speech definition. Hatefulness Definition employed to label dataset for this challenge - "*A direct or indirect attack on people based on characteristics, including ethnicity, race, nationality, immigration status, religion, caste, sex, gender identity, sexual orientation, and disability or disease. We define attack as violent or dehu-*

manizing (comparing people to non-human things, e.g. animals) speech, statements of inferiority, and calls for exclusion or segregation. Mocking hate crime is also considered hate speech.". These memes were reconstructed from the original memes that are present in wild on the Internet because it avoids noise from OCR since our reconstruction tool records the annotated text. Data were annotated by a third-party annotation company as shown in

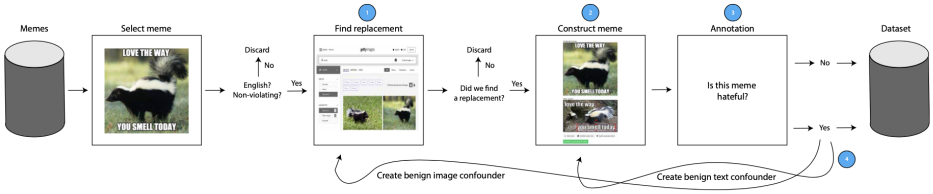

Figure 1.1: Flowchart for Annotation process according to [1]

figure 1.1.

Dataset consists of 10k memes after filtering out low-quality examples. it comprises 5 classes of memes:

- multimodal hate
- unimodal hate (one or both the modalities were hateful)
- benign image confounders(corresponding to multimodal hate memes)
- benign text confounders(corresponding to multimodal hate memes)
- random non-hateful

From this data set for 10k memes, we used around 8.5k images in total for our experiment due to computational limitations. We used 80% data for training and 20% for validation out of 8.5k memes.

1.3 Constraints

- Computational Limitation(Only 8GB RAM Laptops for running models as competition rules restricted everyone to use Google Colab and Kaggle).
- Worked virtually in such unprecedented times of Covid-19 interactions were not com-

parable to those which we might have done if we were at college.

2 METHODOLOGY

2.1 APPROACH 1: UNIMODAL APPROACH

The Unimodal approach takes into account only one aspect of the data. Using a text-based model would likely be the best approach to building a Unimodal classifier, as we have to classify the images as offensive or non-offensive based on the text. Thus we use BERT (Bidirectional Encoder Representations from Transformers) [2], a SOTA model to classify texts.

2.1.1 PRE-PROCESSING DATA

Preprocessing plays an important part in text-based classification. The following preprocessing pipeline was used:-

- As the text is short (Figure 2.1), we first treat the text in the same way as tweets and remove any URLs or '@' mentions.

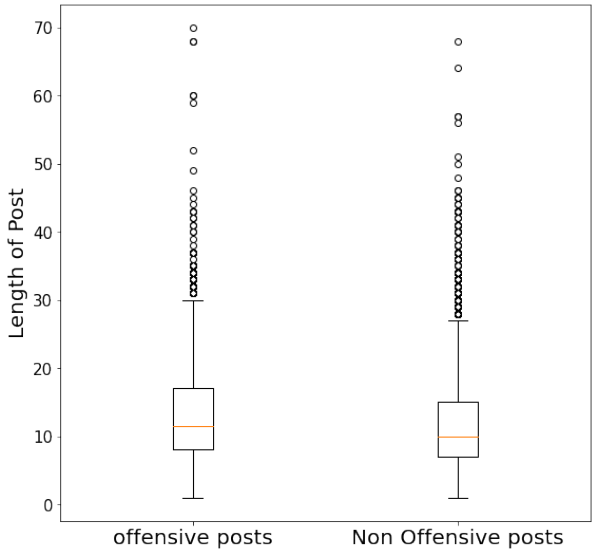

Figure 2.1: Boxplot showing the distribution of lengths of the texts

- After this, we noticed that some texts contain emojis in them. So we use the python **Emoji library** to convert them into their word formats. For Example, the xD emoji (Can't type in Emoji format here) is converted into ":Laughing Face Emoji" in the

text. Care is taken to convert an emoji only once in the text, as there can be multiple consecutive appearances of the same Emoji.

- We noticed that there are multiple occurrences of contractions like "I've", "'tis", "don't", "Can't" etc. As the text classification we aim to do is sentimental, It'd be wise to expand such contractions accordingly with their context. So we used the **pycontractions library** that is built upon Java8 to expand such contractions.
- After all this, we refer to the box plot (Figure 2.1) and decided that all texts with less than 50 words should be used and the rest should be considered outliers.

The text preprocessing timeline is shown (Figure 2.2).

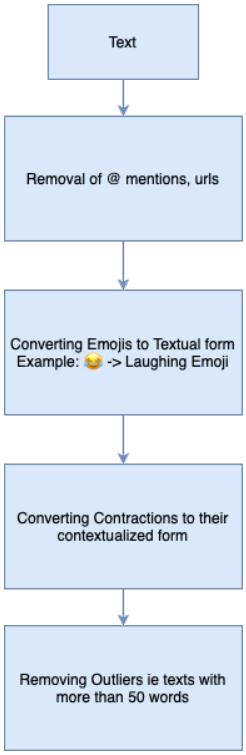

Figure 2.2: Flowchart showing the text preprocessing pipeline

Please refer to the word clouds to better visualize the textual data. Figure 2.3 shows the word cloud for the whole dataset, while Figure 2.4 shows the word cloud for the offensive samples.

Figure 2.3: Overall Word-cloud

Figure 2.4: Offensive Samples Word-cloud

To better understand why the task focuses significantly on the context of the words spoken, we made a Radar plot (Figure 2.5) for some of the common words.

As we can see in the Radar Plot, racist words like "muslim", "white", "black" appear more frequently in offensive posts. But some "Offensive looking" words like "fuck" appear more

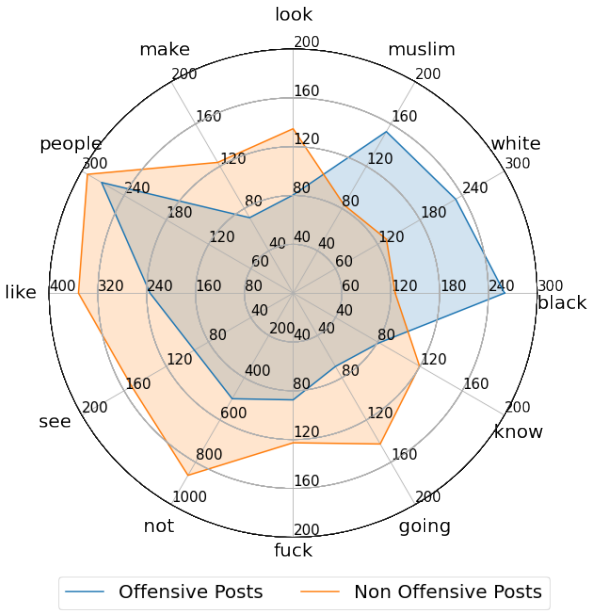

Figure 2.5: Radar Plot

commonly in non-offensive posts as they may have a positive meaning when given context. Thus the Radar Plot proves that identifying context is an essential metric for this problem.

Finally, after all this preprocessing, we finally fit BeRT on the dataset.

2.2 Approach 2: Multimodal Approach

As seen from the results of the Unimodal approach, we need to extract features from both image and the text to classify the images correctly. This is because memes are precisely contextualized by seeing both their visual and text parts. A simple linear approach wouldn't be able to classify it correctly. Many examples are shown (Figure 2.6).

Figure 2.6: Example showing both image and text are important

2.2.1 PRE-PROCESSING DATA

The Multimodal approach uses the Unimodal approach, as discussed above, for the textual part. The preprocessing pipeline remains the same, and the model is also BeRT. But instead of classifying based on features provided by BeRT, we concatenate its output with the image features.

The image features are extracted using multiple SOTA pre-trained models, namely Resnet50 [3], VGG16 [4], MobileNet-V2 [5]. As with the textual features, image preprocessing also plays an important role. The image preprocessing pipeline is as follows:-

- All images are resized to 255 x 255 x3. This ensures that the minimum image dimensions are as required by the pre-trained image models.
- Next, the images are center cropped to sizes 224 x 224 x 3. As memes generally have top and bottom texts, this cropping ensures that we focus more on the central part of the image.
- After center cropping, we make the images flip randomly on the horizontal axis. This ensures that the model learns to recognize features and helps in augmentation.
- , In the end, the images are normalized using the standard mean and std variables used for RGB images.

The image preprocessing pipeline is shown (Figure 2.7).

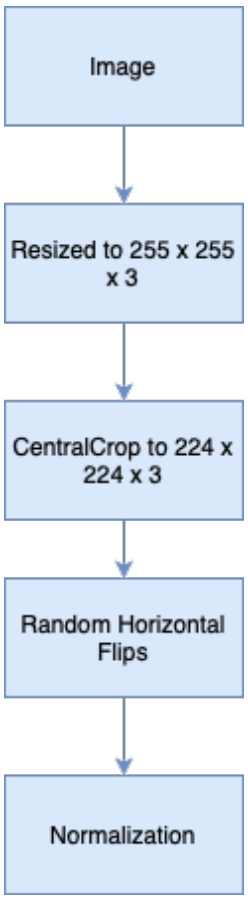

Figure 2.7: Flowchart showing the image preprocessing pipeline

After all this preprocessing, the images are ready to be passed on to the model.

The model is an ensemble of the three models above, namely, VGG16, Resnet50, MobileNet-v2, and BeRT.

The model has the following pipeline (Figure 2.8). Each image is passed through the three models, and the output is flattened. The three models' flattened output is concatenated and sent through dense layers. Alongside this, the BeRT model extracts the textual features later concatenated with the ones above. This final feature vector is passed through dense layers to get the predicted labels.

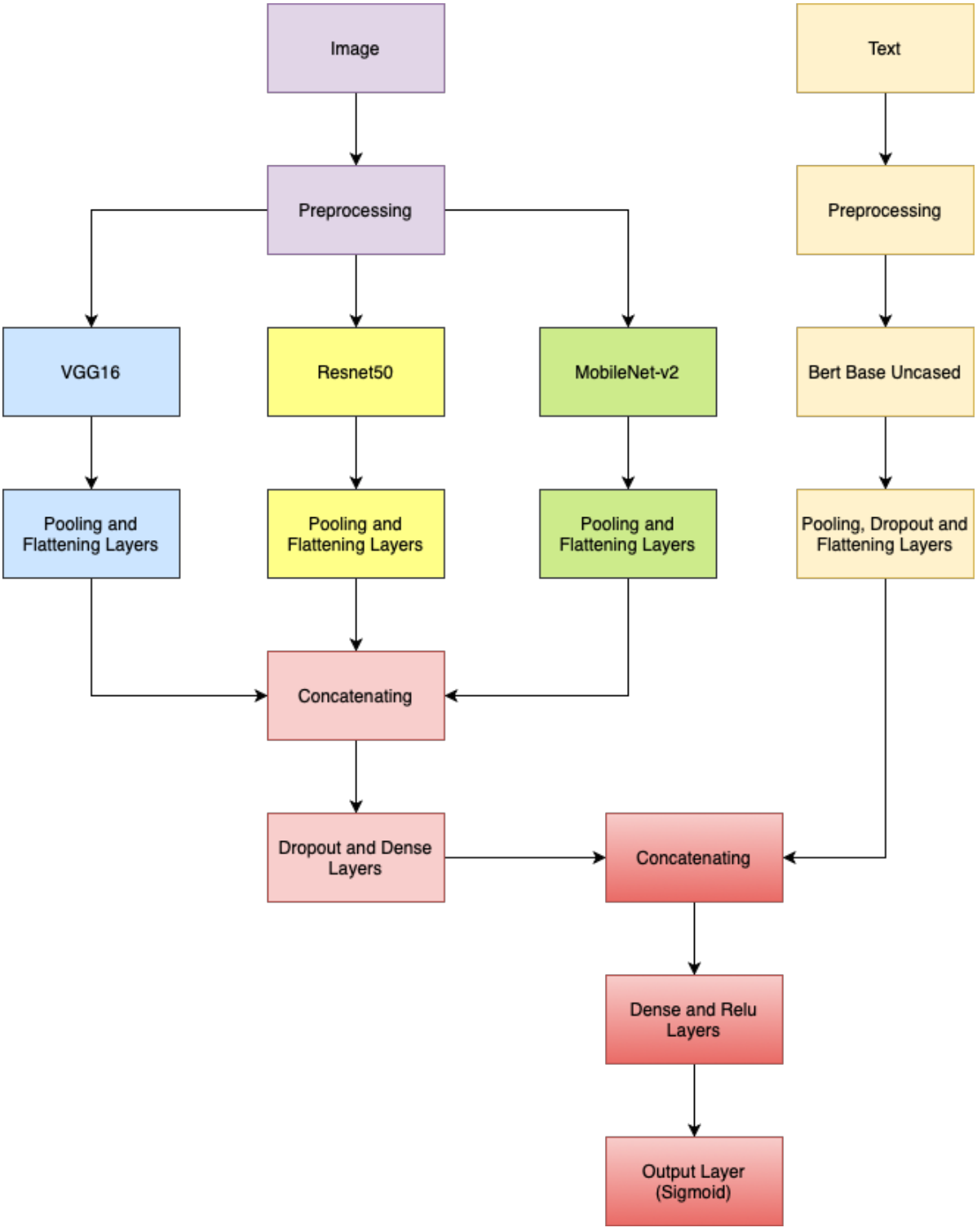

Figure 2.8: Flowchart showing the complete Multimodal pipeline

3 Results

3.1 Approach 1: Results

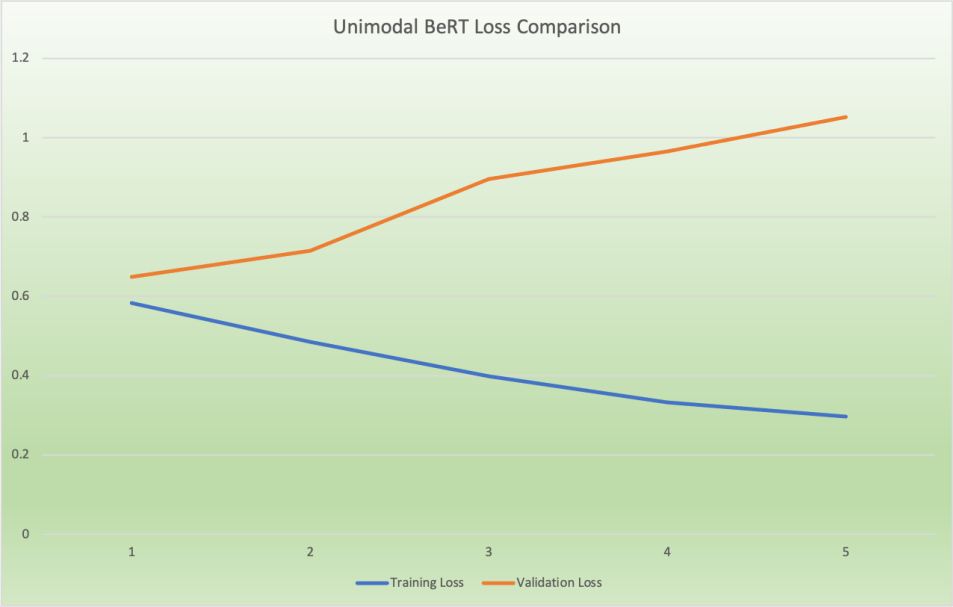

Figure 3.1: Graph showing the model loss for Unimodal approach

As expected, the Unimodal approach cannot correctly classify the dataset as it cannot identify the context. As seen in the model loss graph (Figure 3.1), we can see that the model overfits. The best model accuracy from the five epochs for the validation data is only *66.7%*, and for the training, data is only *83.6%*. Table 3.1 shows the epoch loss values.

Thus we can safely say that the unimodal approach is terrible and wouldn't give excellent results.

Epoch	Training Loss	Validation Loss
1	0.5833	0.649
2	0.4854	0.7142
3	0.3988	0.895
4	0.3316	0.9653
5	0.2962	1.051

Table 3.1: Table showing the loss values over the epochs for Unimodal approach

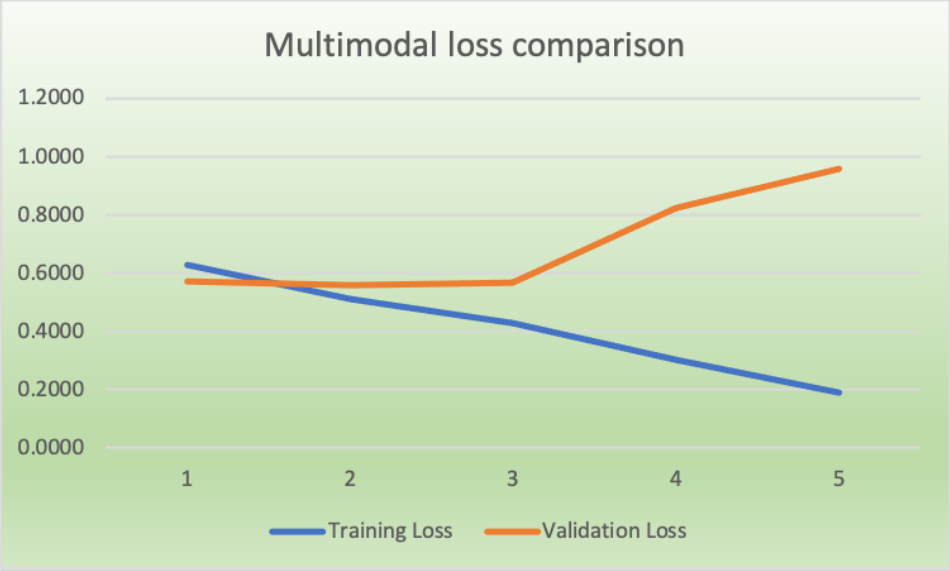

Figure 3.2: Graph showing the model loss for Multimodal approach

3.2 Approach 2: Results

As for the Multimodal approach, we have better results than those for the Unimodal ones. Figure 3.2 shows the loss comparison for the training and validation datasets. From the Graph, we can see that the model begins to overfit during the 3rd Epoch but doesn't have a very significant decrease in validation loss during the first three epochs. So, we can safely say that this approach works better than the Unimodal ones, but it still has problems contextualizing the images. As for Accuracy comparison, the multimodal approach achieves the highest validation accuracy of *72.9%* and a training accuracy of *93%*. Figure 3.3 shows the accuracy comparisons during the epochs. Table 3.2 shows the metrics altogether.

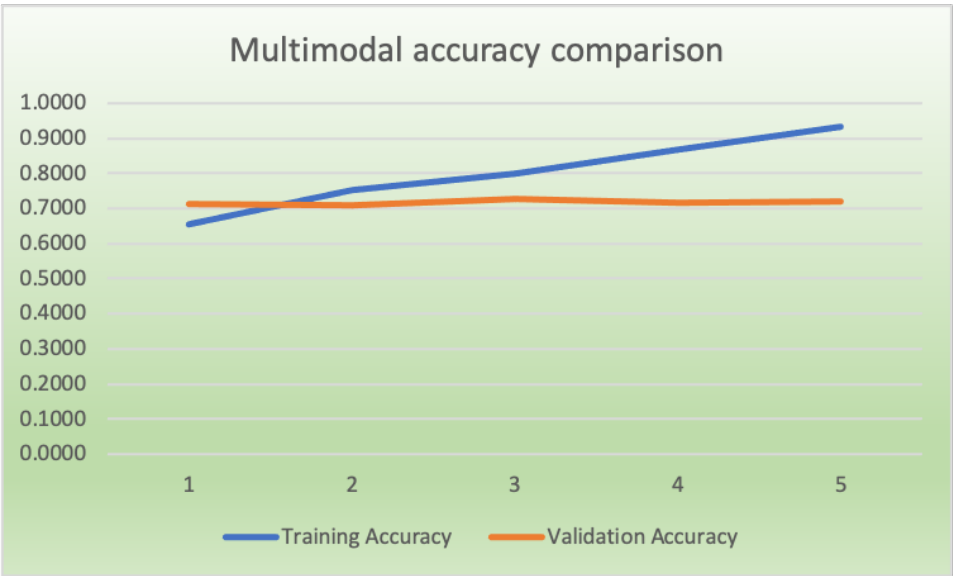

Figure 3.3: Graph showing the model accuracy for Multimodal approach

Epoch	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
1	0.6278	0.6561	0.5726	0.7118
2	0.5104	0.7544	0.5582	0.7101
3	0.4307	0.8006	0.5676	0.7292
4	0.3044	0.8692	0.8231	0.7182
5	0.1898	0.9330	0.9584	0.7211

Table 3.2: Table showing the metric values over the epochs for Multimodal approach

4 Conclusion

Compared to the baselines shown in Table 4.1, our approach works better than the baseline models but is still far away from the human accuracy levels. Thus we need to improve our detection methods further. We can still try to improve our accuracy further using the future approaches mentioned in the next section.

Baseline	Accuracy
Unimodal ImageGrid[1]	0.5200
Unimodal TextBeRT[2]	0.5920
Multimodal ViBeRT[6]	0.6110
Approach 1	0.6670
Approach 2	0.7292
Humans	0.8470

Table 4.1: Table showing the Baseline accuracy

5 Future Scope

To understand the context of the images, we need to identify the objects/subjects that are present in the images. To do this, we propose the following ways:-

- We can use Image Captioning to generate a textual description of the images. Combining this with the already given text might provide a better model. We propose the approach given in Figure 5.1.

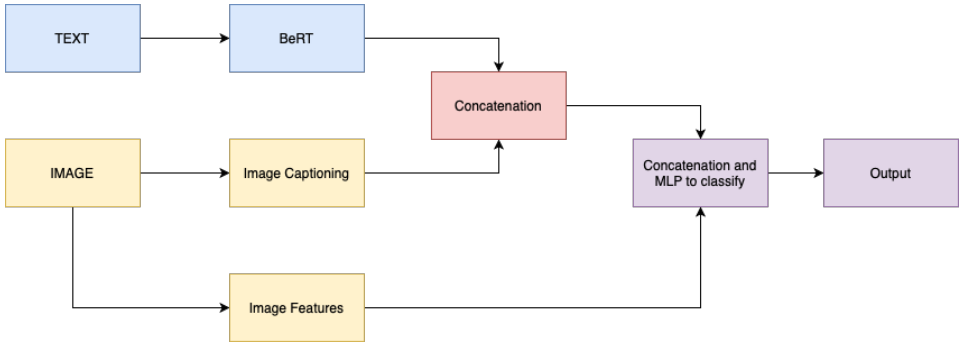

Figure 5.1: Image Captioning

- To detect what objects/subjects are in the images, we propose using a **detectron**. The detectron can provide us with more information and context and can be used similarly to the image captioning approach.

References

[1] Douwe Kiela, Hamed Firooz, Aravind Mohan, Vedanuj Goswami, Amanpreet Singh, Pratik Ringshia, and Davide Testuggine. The hateful memes challenge: Detecting hate speech in multimodal memes, 2020.

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

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[5] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019.

[6] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks, 2019.