Project Report : CS 7643

Annalisa Barbara
abarbara3@gatech.edu

Yoshiki Kakehi
ykakehi3@gatech.edu

Jaegook Alex Kim
jkim3415@gatech.edu

Karim Ahmed Abdel Sadek
ksadek3@gatech.edu

Abstract

Hateful content on social media has become a growing concern, and we believe it is crucial to address this issue. Our work aims to contribute to the ongoing efforts to detect and mitigate hateful memes by taking on the Hateful Meme Challenge proposed by Meta. Our primary focus is on exploring the effectiveness of both Multimodal and Unimodal models in detecting hateful content in memes, and we strive to match the benchmark results set by Meta. We also try to compare the predictions obtained by using classical sentiment analysis tools such as VADER (Valence Aware Dictionary and Sentiment Reasoner) with the results obtained by using pre-trained unimodal models on text detection. Our report provides a detailed account of the challenges we faced while working on this task and the strategies we employed to overcome them. We present our experimental results and analyze the factors that influenced our model's performance, aiming to offer insights on how to improve hateful content detection online.

1. Introduction/Background/Motivation

What problem are we trying to solve?

The internet has millions of new memes generated everyday. Whilst, the purpose of memes is to spread entertainment across multiple media platforms such as Reddit, Instagram, and Tiktok, not all memes are appropriate. There are memes that consist of slurs, racism, and fake news. There is limited human power to analyze all the memes on the internet and take out hateful memes. The problem we tried to solve is to automate the process of determining whether a meme contained hateful content or not with deep learning.

One of the main challenges of the hateful memes dataset is the necessity of simultaneously analyzing both the text and the image in order to understand if a meme has hateful content. We observed that most of the memes in the dataset were created in such a way that if we were to analyze just the text or image, it would not be able to accurately detect whether a meme was hateful or not. This is because memes can be hateful when both the image and text are taken into consideration jointly. For example, a harmless picture of a kid combined with a pedophilia reference together is inappropriate, yet text and picture separate, it may seem appropriate.

For this reason, we tried to address the problem using a multimodal approach. Our starting point relies on notebooks already presented at the Hateful Memes challenges. We took a pre-trained BERT model to extract text features and a pre-trained ResNet model to extract features from the images. We then joined them in a transformer (Fusion and Concat approach). In all these passages, we performed GridSearch to tune our hyperparameters.

How is it done today, and what are the limits of current practice?

Currently, identifying hateful memes is primarily done through manual content moderation by human moderators, which can be a time-consuming and costly process, particularly for large-scale social media platforms like Facebook, Instagram, or Reddit. Alternatively, some social media platforms use automated content moderation systems that use machine learning algorithms to flag potentially offensive content, but these systems are often not sophisticated enough to accurately identify hate speech in memes.

Hate speech detection using deep learning typically involves training a neural network on a labeled dataset of text samples, where each sample is labeled as either hate speech or not. The neural network can be a variety of architectures, such as a Convolutional Neural Network, a Recurrent Neural Network, or a Transformer-based architecture such as BERT or GPT-2.

The neural network takes the text input as a sequence of tokens and uses embedding layers to represent each token as a vector and then processes the embeddings with one or more hidden layers, which use non-linear transformations to capture complex patterns and features in the text.

The output layer of the network typically consists of a single neuron, which predicts the probability that the input

text is hate speech. The model is trained using an optimization algorithm, such as stochastic gradient descent, to minimize the cross-entropy loss between the predicted probabilities and the true labels.

To address the limitation of limited annotated data, transfer learning is often used, where a pre-trained language model is fine-tuned on a hate speech classification task. This fine-tuning allows the model to leverage the knowledge learned from large amounts of unlabeled text, which can improve the model's performance on the downstream task of hate speech detection.

Another limitation is the potential for bias in the labeled datasets used for training the models, which can result in models that are biased against certain groups or perspectives. To mitigate this, techniques such as adversarial training or data augmentation can be used to reduce bias and increase the model's robustness to different forms of hate speech.

Who cares? If you are successful, what difference will it make?

Hate speech detection in memes is an important area of research because memes have become an increasingly popular way for people to express their opinions and emotions online. Memes are also one of the main sources of online entertainment in contemporary times. With all good things, however, there is always a counter force. Memes can also be used to spread hate speech, misinformation, and propaganda, which can have serious consequences for individuals and society. We have seen the spread of fake news, and it is especially more critical to prevent the spread through memes as younger generations intake information through memes and other medium of entertainment.

Detecting hate speech in memes is therefore crucial for social media platforms, governments, and organizations that want to prevent the spread of harmful content online.

What data did you use?

The dataset contains 10k samples, each consisting of an image and a corresponding text caption. The dataset was preprocessed to remove duplicates and irrelevant samples and was labeled as either "hateful" or "not hateful" by human annotators. The dataset was designed to capture the nuanced and complex nature of hate speech in memes, which often involves the use of humor, sarcasm, and cultural references.

To ensure the quality of the dataset, the authors employed a team of trained annotators to label each sample as "hateful" or "not hateful". The annotators underwent rigorous training and calibration to ensure consistency and accuracy in their labeling.

The annotators were instructed to identify samples that contained hate speech, which can take various forms, such as derogatory language, offensive stereotypes, or incitement to violence. The labeling process involved multiple rounds of annotation and quality control, and each sample was independently labeled by at least three annotators. Discrepancies in labeling were resolved through a consensus process, in which the annotators discussed and agreed upon the appropriate label for each sample.

The dataset was then split into training, validation, and test sets, and the authors provided baseline models and evaluation metrics for the challenge

2. Approach

We decided to use two different approaches: A unimodal approach, i.e just analyzing the text of the memes, and a multimodal approach, joining two pre-existing models. Specifically, for the multimodal approach, we used 3 different type of model combinations: XLNET with ResNet, GPT2 with ResNet and Bert with ResNet.

For the unimodal analysis, we used the pre-trained DeHate Bert model and compared the results with the ones obtained by using functionalities from the vaderSentiment library, the offers specific built-in functions to detect the sentiment on social media textual data.

What emerged after several experiments, is that the DeHate Bert model achieves an accuracy of 0.58 after 10 epochs while using VADER, we have been able to reach an accuracy of 0.6. The predictions using the vaderSentiment package have been obtained by using the pre-built SentimentIntensityAnalyzer fuction and using the compound score of a sentence in order to detect if the sentiment was negative or positive. Since the compound score is normalized between -1 and 1, -1 if the overall sentiment of the sentence is very negative and 1 if instead is very positive, a threshold of -0.05 has been set in order to classify the sentence as negative and positive otherwise. Remarkably, the majority of misclassified sentences utilizing VADER were also misclassified by DeHate Bert. These sentences only became hateful when paired with an accompanying image, which are precisely the types of sentences that are challenging for unimodal models to accurately classify. We also tried to use the pre-trained unimodal models Bert and XLNET but with none of them we have been able to reach an accuracy over 0.6 since all of them started to overfit after the first few epochs. Given that hate can be conveyed through both textual and visual modalities, we hypothesized that a multimodal approach would be a more effective and promising means of addressing this issue. By the nature of the project itself, we identified different possible problems that could have arisen. The first one regards the context of the memes themselves and their labeling. Being a context-dependent evaluation, both from a historical and a cultural point of view, the labeling could be ambiguous or incorrect. It is context dependent since hateful memes often arise

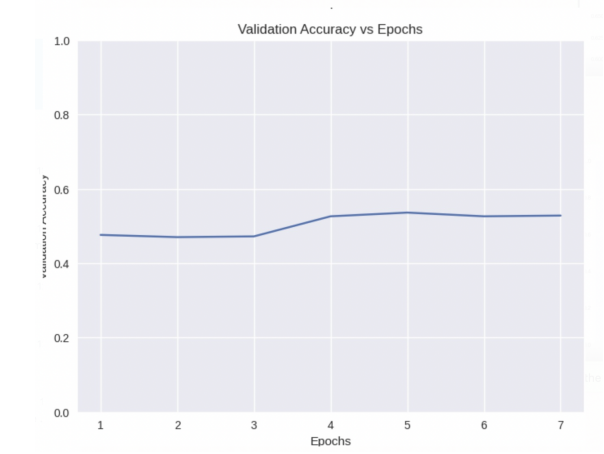
given an event occurring in a specific moment, such that a political election or similar. We could also anticipate more classical problems that can arise in a deep learning setting. For example, being the nature of each meme being unique and subjective, could lead to a easy overfitting of our model.

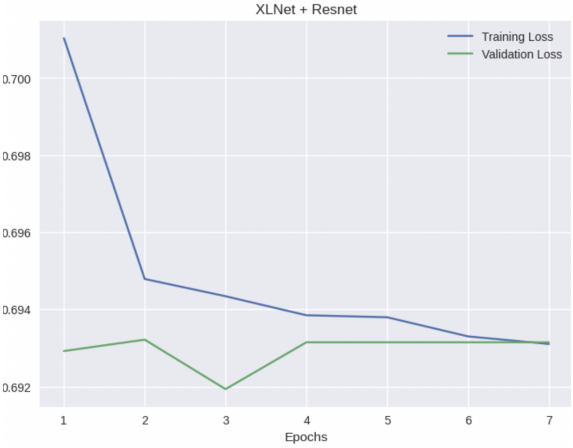
3. Experiments and Results

As in most of the classical deep learning settings, we used the validation accuracy on an ad-hoc designed test set to measure our performance. We implemented different models, and we can see different results depending on the model used.

XL Net+ResNet

For the Multimodal XLNET+ResNet, an accuracy of 55.1 has been reached. This is slightly below the Meta benchmark for the multimodal models. We can justify this by considering a few aspects. This multimodal model we implemented is different from the model considered by Meta for the benchmark, and we also lack the necessary computing power to fine-tune it appropriately. Here below, we are attaching the plots that produced the best results for this model.





Those, have been reaching using the following hyperparameters:

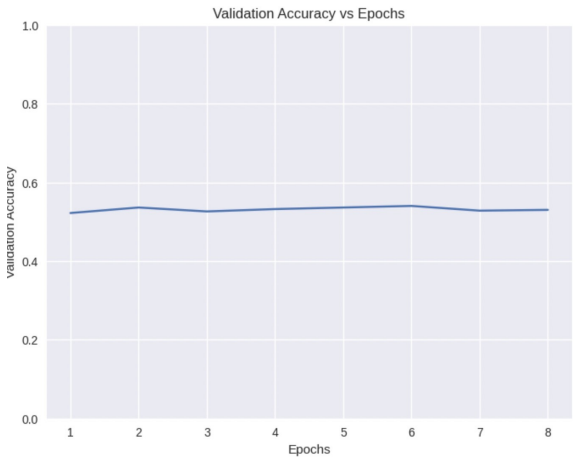
- Learning Rate: 0.08
- Learning Rate Scheduler: Step size = 2, $\gamma = 0.1$
- Optimizer: SGD
- Weight decay = 0.0001
- momentum = 0.99
- epochs = 8

Those results were obtained by performing a own-implemented GridSearch, searching 28 different combinations of hyperparameters.

BERT+ResNet

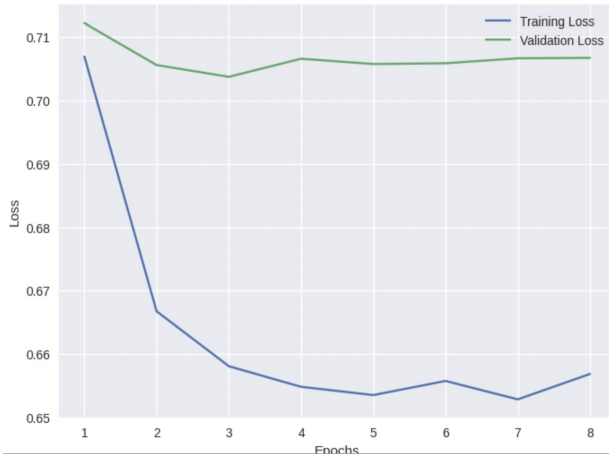
The BERT+Resnet model implements a fusion of ResNet to classify images and BERT to classify the texts in the memes.

The result for the BERT+ResNet can be seen in the following graphs.



As observed, the validation increases per epoch. It plateaus to around 0.575 after 4 epochs. The BERT+ResNet model peaks and plateaus with 0.58 accuracy. Meta released a simple BERT (unimodal) with an accuracy of 0.59. We were able to reach up to par with the BERT trained by Meta, however, our model also implements ResNet for image classification. As aforementioned, meme require the combination of text and image to clearly determine whether it is hateful.

We recognize that the model should have had a higher accuracy, however, we must note that the lack of computing power limits us from testing out a vast variety of hyperparameters like Meta.



The training loss and validation loss comparisons show that the model is over fitting to the dataset as training loss decreases exponentially, while validation loss converges to around 0.715. When the validation loss does not decrease with respect to the training loss and converges shown in the graph, it means that the model is having trouble generalizing to the test dataset.

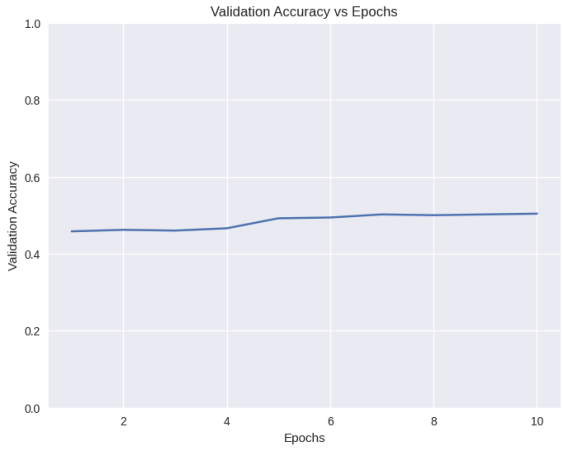
To find the best validation accuracy, we worked with two main optimizers Adam and SGD as they resulted in the highest accuracy. We quickly realized that epochs tend to play a significantly less factor after 8 epochs as it converges. We had a set number of epochs and mostly worked around the betas, weight decay, and learning rate.

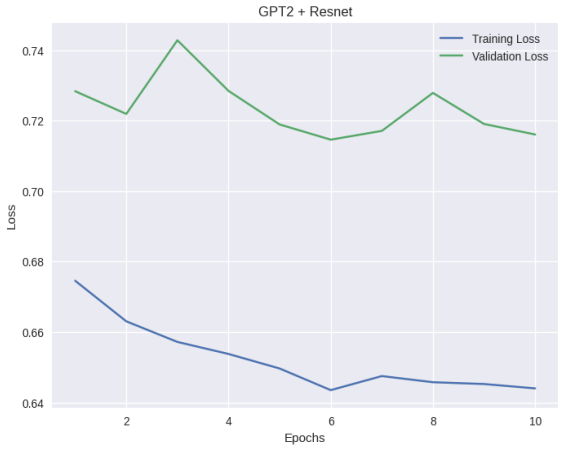
To reproduce the results for the BERT+ResNet, the following hyperparameters were used.

- Loss Function: Cross Entropy Loss
- Optimizer: Adam
- Learning Rate Scheduler: Step size = 5, $\gamma = 0.1$
- Learning Rate: 1e-4
- Betas: (0.9, 0.999)
- Weight Decay: 0.05
- Epoch: 10

GPT2 + ResNet

The GPT2 + Resnet model implements a fusion model that utilizes GPT2 to classify text in the meme image data and Resnet to classify the images. For our Multimodal GPT2 + ResNet, an accuracy of 50.40 was achieved. We can justify this accuracy by considering that we lacked the computing power to fine-tune the model appropriately. Below, we have attached the plots that produced the best results for this model.





Those, have been reached using the following hyperparameters:

- Loss Function: Cross Entropy Loss
- Optimizer: Adam
- Learning Rate Scheduler: Step Size = 4, $\gamma = 0.1$
- Learning Rate: 0.0001
- Betas = (0.9, 0.999)
- Weight decay = 0.001
- Epochs = 10

We found that the above hyperparameters produced the best results after performing a GridSearch algorithm that we implemented, which searched 27 different combinations of hyperparameters.

Something to note about our results is that our model seems to plateau in accuracy at around 50, with very minimal change or improvement. Additionally, while we observe an overall decrease in the training loss, there is some volatility in the validation loss that may be attributed to some overfitting.

4. Final Considerations

Although our validation loss is considerably high during the first epochs, our model does not completely overfit. We are able to determine overfitting by checking to see if the validation loss increases after a certain number of epochs as this indicates that it has trouble classifying the test dataset. However, for our models, every epoch results in a lower validation loss or fluctuates. Although the volatility in validation loss may attribute to some overfitting, it is not conclusive.

With the results, we observe that the models were able to approach generalization to a moderate degree. Although the target validation accuracy that we aimed to achieve was approximately 80, we were able to reach subpar of human accuracy of determining whether a meme was hateful or not. We were able to have the BERT model output 0.58 validation accuracy, XL_Net model has a 0.55 validation accuracy, and GPT2 has a 0.55 validation accuracy.

The team believes that if we had more computing power, we would have been able to work with a variety of hyperparameters, trying to get the best results for the models.

Additional Hyperparameter Tuning Notes

We tuned several parameters for the models including loss function, type of optimizer and number of epochs. To optimize the hyperparameter tuning we also used lr_scheduler, which adjusts the learning rate based on the number of epochs.

We have chosen the cross entropy loss function because our objective was to classify the memes into two categories of hateful or not hateful. We believed the cross entropy loss function was best for classifying models.

We worked around multiple optimizers from Gradient, Descent, Stochastic Gradient Descent (SGD), Adaptive Moment Estimation (Adam), and Adaptive Gradient optimization (Adagrad). After multiple attempts in training our models, we came to the conclusion that Adam consistently gave us the best validation accuracy compared to the other optimizers. Adam optimizer converges faster, even if it may not be the most optimal, which helps us as we are doing a binary classification of whether a meme is hateful or not.

The hyperparameters were chosen with the implementation of a random grid search. We worked with a range of weight decay and momentum while also tuning the learning rate optimally as mentioned previously. This allowed us to better train our models as we were able to experiment with multiple values improving also our validation accuracy.

What Deep Learning framework did you use?

We utilized PyTorch as our deep learning framework to implement and train our models.

What existing models did you start with and what did those starting points provide?

We started with existing code for the unimodal models, including the models for XLNet, BERT, GPT, and ResNet. These were starting points that we expanded upon to create multimodal models.

We combined a text and image algorithm together to analyze the memes from two points rather than just solely images or texts. We used ResNet as the main model for image classification while we used different text algorithms to analyze hateful text in the memes.

Potential future work

There is always room for improvement. The accuracy we were able to achieve is significant, but not good enough to be deployed without human assistance. Humans have a 84.7 percent accuracy of determining whether a meme is hateful, so human error is also pretty significant.

Ideally, we wish to create a model that is better than humans in classifying hateful memes to minimize the fake news, propaganda, and hates that is spread through social media platforms. This is possible as the architecture for neural networks is always improving. Artificial intelligence is a constantly growing field and the detection of hateful memes is one among the incredible amount of challenges that multimodal deep learning models and transfer learning are today facing.

Student Name	Contributed Aspects	Details
Annalisa Barbara	Unimodal Analysis and Hyperparameter Tuning	Imported the unimodal pre-trained models and implementation of the classification model using VADER. Performed hyperparameter tuning. Wrote part of the Introduction, Approach, and Final Considerations.
Yoshiki Kakehi	Implementation and Analysis	Trained the multimodal GPT2 + Resnet model and analyzed results. Performed hyperparameter tuning.
Jaegook Alex Kim	Implementation, Analysis, and Report write up	Trained the multimodal BERT + Resnet model and analyzed results. Performed hyperparameter tuning. Wrote Introduction, Approach, part of Experiments, and Final Considerations.
Karim Ahmed Abdel Sadek	Import of the dataset, Hyperparameter tuning	Imported the dataset and the models, created the backbone of the notebook. Implemented the function GridSearch from scratch. Performed hyperparameter tuning. Wrote abstract, and part of sections 2 and 3.

Table 1. Contributions of team members.

References

[1] https://github.com/rizavelioglu/hateful_memes-hate_detectron.

[2] https://github.com/facebookresearch/mmf/tree/main/projects/hateful_memes.

[3] https://github.com/drivendataorg/hateful-memes.

[4] Aggarwal et al. *HateProof: Are Hateful Meme Detection Systems really Robust?*

[5] Aijing Gao et al. *Hateful Memes Challenge: An Enhanced Multimodal Framework.*

[6] Chen et al. *Multimodal detection of hateful memes by applying a vision-language pre-training model.*

[7] Douwe Kiela et al. *The Hateful Memes Challenge: Detecting Hate Speech in Multimodal Memes.*

[8] Thakur et al. *Multimodal and Explainable Internet Meme Classification.*

[9] Velioglu et al. *Detecting Hate Speech in Memes Using Multimodal Deep Learning Approaches: Prize-winning solution to Hateful Memes Challenge.*