Hateful Memes: An Analysis of Text Transformers in a Multimodal Context

Nicholas Farris
Georgia Institute of Technology
nicholas.farris@gatech.edu

Florian Pesce
Georgia Institute of Technology
florian.pesce@gatech.edu

Matthew Sandry
Georgia Institute of Technology
msandry3@gatech.edu

Gabriel Wang
Georgia Institute of Technology
gwang340@gatech.edu

Abstract

This paper performs an experiment on the usage of textual transformers in multimodal classification on the dataset from the Facebook hateful memes challenge. We use the transformers BERT, RoBERTa, XLM-R, and ERNIE as feature extractors within a multimodal model in combination with the ResNet image model in order to analyze the usage of textual transformers as feature extractors as well as compare their results across a variety of metrics. We performed two experiments, one where the gradients of the transformers were frozen and one where the gradients were allowed to update in order to train the encoders. The results of our experiments showed that our transformers did not perform significantly different from one another when used only as feature extractors, and that their overall performance for the task of classifying multimodal memes as hateful was not superior to simpler textual methods such as the bag-of-words method.

1. Introduction

The objective of this paper is to analyze the performance of different pre-trained textual encoders on the multimodal data problem of hateful meme classification proposed by Facebook AI. The Facebook AI hateful memes problem challenges teams to create a model with the highest accuracy in classifying multimodal internet memes which contain a combination of text and an image as either hateful or non-hateful. Due to the fact that classification of multimodal data requires more nuance and real-life context than most unimodal classification tasks, this problem proves to be especially difficult to solve with machine learning algorithms [5].

In our work, we aim to analyze the role of the textual encoder within the context of the hateful memes problem in order to determine which qualities of a text encoder lend themselves towards better classification of multimodal data when combined with an image encoder. More specifically, we use the pre-trained textual transformers BERT, RoBERTa, XLM-R, and ERNIE for this task. We use these text encoders as feature extractors for the textual portion of the data and combine these features with a visual feature extractor in order to directly compare the performance of the models across a variety of metrics.

If successful, the results of this work will provide useful information in the design process for multimodal models in the future. More specifically, it will help the designers of models to understand the merits of utilizing pre-trained transformers as feature extractors within the context of multimodal models. Additionally, it will inform design decisions surrounding which textual encoders are likely to perform well for a specific multimodal task. This is directly relevant for the task of identifying hateful content on the internet so that social media platforms can ensure that memes or other multimodal forms of content adhere to their platform policy. Additionally, the results of this work should extend to other classification tasks which utilize textual encoders in a multimodal setting.

If our experiments prove unsuccessful in creating a significant difference in performance either between models or compared to baseline methods, this work will still be beneficial by providing a better understanding of the proper use cases for transformers. Additionally, our work still will help direct further research directions and experiments into the subject of textual encoders for multimodal classification models.

The rest of this paper is structured as follows: In Section 2 we describe related work in the field relating to feature extraction and multimodal meme classification, as well as research into the differences between each of the transformer models that we use. This information will be directly relevant when analyzing the performance differences between models. In Section 3, we provide an in-depth summary of

the dataset that we use for our experiments. Section 4 details our approach when designing our experiments. Section 5 outlines the results of each experiment performed. In Section 6, we discuss our results and the consequences that they may have towards future work. Finally, Section 7 provides a summary of the contributions of each individual team member.

2. Related Work

2.1. Feature Extraction from Text

Traditional methods of feature extraction typically requires handcrafted features, especially when dealing with textual data. Designing an effective feature is a lengthy process, but deep learning has been shown to create new effective feature representation from the training data in a variety of applications. [7] Text feature extraction typically involves creating a vector space model, where a text is viewed in an N-dimensional space and each dimension of the text represents one feature, and is usually reliant on a keyword set. Comparatively, deep learning has the advantage as models typically don't require hand-designed features, as the models should be able to learn the features while they are training. Some notable models that have seen success in natural language processing (NLP) are autoencoders[9] and recurrent neural networks[10]. However, while many of these models have been widely applied in NLP, they are rarely used as text feature extractors with the reasoning that they typically rely on data with a time sequence. Similarly, generative adversarial networks (GANs) in theory should be capable of learning features from text and generating text, but are mainly applied to generate natural images and has not made significant progress in text feature extraction. [3] However, recently the concept of a transformer [13] has arisen, with a novel attention method and context vector representation that helps the models properly learn text representations without the weaknesses of RNNs. We seek to understand if these transformers, which have seen great success in NLP, will be able to extract features from the text in memes that can be applied to a multi-modal context.

2.2. Comparing the Transformers

While there is no prior work directly related to comparing transformers being used as feature extractors within the context of multimodal classification, understanding the differences between the transformers is important in order to understand the differences in their results.

BERT, RoBERTa, XLM-R and ERNIE use different kinds of tokenizers, which have different vocabulary sizes and different tokenization rules. Similarities between tokenizers are usually between 40 and 80%. The fact that different models use different tokenizers and different subword tokens makes it difficult to compare models at the tokenizer level. This means more appropriate comparisons can be made by comparing the architectures, and by comparing model results on the appropriate metrics.

The main important difference between our pre-trained models that may impact their performance on the hateful memes dataset is the way that they were trained. BERT is trained using two different tasks: Masked Language Model and Next Sequence Prediction. Masked Language Model pre-training means that a random subset of the generated tokens are hidden during training, and the objective function is to determine correctly predict the hidden tokens. When performing Next Sequence Prediction, BERT receives two sentences, and must determine if the second sentence is the true continuation of text after the first sentence. The Masked Language Model training allows BERT to learn relationships between words or tokens within a sentence, while Next Sequence Prediction trains towards learning relationships between sentences [2].

RoBERTa is the most similar model to BERT, and utilizes a similar Masked Language Model for training. The key difference in RoBERTa's training is that it performs dynamic masking, which means that the masked tokens change during training epochs. Additionally, RoBERTa drops the Next Sequence Prediction completely from its training and utilies significantly larger batch sizes as compared to BERT. Finally, RoBERTa trains on approximately ten times the amount of data as BERT, and requires more computing power as well [8].

XLM-R differs significantly from our other models in that its main purpose is cross-language text modeling. Instead of using words or characters as token inputs, XLM-R utilizes Byte-Pair Encoding which splits the inputs into the most common sub-words across languages, which creates a shared vocabulary between languages and allows different languages to be compared easier. For XLM-R's training, each sample contains the text in two different languages, and masks are randomly chosen for each language separately. During training, XLM-R uses the text context from the other language in order to properly predict the identity of each masked token. XLM-R also differs from BERT in that it receives the language ID and positional encodings separately, whereas BERT receives them at the same time [6] [1].

ERNIE utilizes a significantly different training methodology as compared to the other three models. The significant difference in ERNIE's training is that it continually pre-trains with a variable amount of tasks in sequence. The goal of this is to allow ERNIE to remember information about previous tasks while it is training for a new task. Additionally, new customized tasks can be introduced to ERNIE at any time with this architecture [11].

2.3. Multi-modal Meme Classification

The original problem [5] is to be able to classify memes as hateful when given the image and the text within the image. This challenge is particularly interesting as the data type is multi-modal, and we believe that having a strong understanding of the textual component of the data will be important for the success of any models. Other models that have seen success on this challenge used a multi-modal transformer and ensemble learning, where the transformers were designed to be trained on both text and images at the same time, then multiple of these models formed an ensemble to vote on the meme's classification. [14]

3. The Dataset

The data that we used consists of 9140 internet memes from the the Facebook AI hateful memes challenge dataset. The memes contain an image combined with a short line of text, both of which combine to create a particular message for the meme. Additionally, all data cases contain a label that identifies the message of the meme as being either hateful or non-hateful. This dataset contains all data examples from the hateful memes challenge which contain labels and excludes all data which was unlabeled.

The data is split to 80 percent training data, 10 percent development data, and 10 percent testing data. This split was chosen to conserve the approximate percentages between groups of the complete hateful memes dataset, however the complete dataset does not contain labels for test data which were necessary for our purposes in order to create our metrics. We therefore used all data examples from the training and development splits of the complete dataset and re-split the data according to our percentages.

The hateful memes dataset was made and labeled by Facebook in conjunction with third party annotators. It was made to be intentionally difficult for a machine to classify the images as either hateful or non-hateful without utilization of both textual and visual elements. This was done by duplicating labeled images and altering them in a way that only one of the two modalities was changed and the resultant image's label was the opposite of the original. This way, there are many example within the dataset where the image or text is the same between two memes, but the ground truth classification is different.

Due to the nature of the problem of identifying hateful content on the internet, this dataset contains memes that attack a wide range of different groups, and many of the memes may be offensive or triggering to certain audiences.

3.1. Related Dataset

Another multi-modal hateful meme dataset was also released to the public [12]. While the contents of the dataset were very similar to the Facebook challenge [5] we did not utilize this dataset due to our time constraints. However, the authors did discuss various baselines and approaches for this type of multi-modal classification. In particular, "stacking" two models together to create a fusion of the representation it learns, such as an RNN to learn the text which is stacked on top of a visual transformer, which can learn from the combination of text and images.

4. Approach

Our approach was to run a series of experiments in order to better understand the performance of the following pre-trained textual encoders: BERT, ERNIE, RoBERTa and XLM-R. Our main focus was on making sure that all the experiments relate to each other by keeping differences between our different runs to a minimum. More precisely, the model architecture and tokenizers are the only variable across our different runs. For the rest of the model architecture, we used a state-of-the-art Resnet-152 model for the Vision Module, and concatenated mid-level its output with the output of the Language Module. [4]

In order to make sure that we were comparing similar implementations, we ran the classification models on both the large and base architectures for each text encoder. One of our first experiments was to compute different metrics (accuracy, F1-score, recall, etc.) for each different text encoder in order to quantify their performance and to detect whether some text encoders are suitable to different tasks depending on the metric that is being optimized. The text encoders are used as feature extractors for the text portion of the data, and they are all pre-trained. The only gradient that is being updated is the one that pertains to the final Linear layer. In another experiment, we also unfroze the text layers, in order to train the encoders as well. Our approach should provide a thorough comparison of text encoders and their performance in the context of multimodal tasks. What is new about our approach is that we are really working on understanding and comparing text encoders. On the contrary, existing papers focus more either on the merits of a given model, or on finding the best model for a certain task.

In terms of difficulties, we had anticipated running into issues related to learning how PyTorch Lightning works, and learning how to work with transformers and their related libraries. We also encountered other issues we had not anticipated. First of all, we had to deal with many runtime issues and the slow iteration time made it arduous to make corrections to our code. Most of us worked on Google Collab, which despite its ease of use, was not practical for us because it would often time out, thus making us lose hours of runtime. With more dedicated resources to run our code on personal GPUs, our work would have been much easier. Furthermore, we ran into issues because the PyTorch Lightning version we were using was deprecated (upgrading was generating other issues so we decided to keep the

model	accuracy	auc	auroc	cohen kappa	f1	hamming dist	hinge	iou	roc	precision	recall
notext	56.9	0.466	**0.595**	0.137	0.554	0.431	0.967	0.397	0.137	0.556	0.553
bag-of-words	56.3	0.459	0.585	0.124	0.528	0.437	0.973	0.391	0.124	0.555	0.503
bert-based-case	**57.0**	0.461	0.586	**0.140**	**0.559**	**0.430**	0.978	**0.398**	**0.140**	0.556	**0.562**
bert-large	55.7	0.471	0.581	0.113	0.541	0.443	0.978	0.386	0.113	0.543	0.540
roberta	55.3	0.488	0.575	0.097	0.468	0.447	0.964	0.374	0.101	0.552	0.406
roberta-large	53.4	0.443	0.550	0.063	0.484	0.466	0.982	0.361	0.064	0.522	0.451
ernie	56.0	**0.511**	0.577	0.114	0.495	0.440	**0.959**	0.384	0.116	**0.558**	0.445
ernie-large	54.5	0.459	0.566	0.089	0.532	0.455	0.984	0.374	0.089	0.530	0.533
xlm-r	52.2	0.451	0.541	0.042	0.498	0.478	0.995	0.352	0.042	0.507	0.490

Table 1. Text Feature Extraction Models Vs Performance

old Lightning version). Because of this, we had to write our own code to generate metrics and loss logging. Another difficulty was that in order to avoid data contamination, we had to re-split the data in a 80-10-10 split, balance it, while still making sure that the testing data had labels to use for our metrics computations.

5. Experiments and Results

5.1. Experiment 1

In experiment 1, we analyze the contribution of transformer models for language understanding when the transformer is implemented as a feature extractor. In this experiment, we evaluated the accuracies, F1-scores, precision, recall, and many other metrics achieved by implementing bag of words, BERT, BERT-Large, RoBERTa, RoBERTa-Large, ERNIE, ERNIE-Large, and XLM-R as the text feature extractor prior to multi-modal training. We hypothesized that the pretrained transformers would serve as better feature extractors than traditional approaches such as bag of words or models without text encoding. We decided to explore this hypothesis as if a particular pretrained transformer demonstrates a greater ability to serve as a feature extractor in a multimodal context, then we would be able to use this information to inform further architecture development decisions.

In this experiment, we implemented early stopping based on average validation loss with a patience of 3. The text encoders all had the following hyper parameters: embedding-dimension=300, language-feature-dim=300, vision-feature-dim=300, fusion-output-size=256, learning-rate=0.0005, max-epochs=10, batch-size=4, and accumulated-grad-batches=16. All transformers except for ERNIE come from the transformers python library [15].

We measured the performance of these models with the metrics outlined in Table 1. Here you can see each model's accuracy, area under the curve using the trapezoidal rule, area under the receiver operating characteristic curve, cohen's kappa score (which measures inter-annotator agreement), f1 score, hamming distance (also known as hamming loss), mean hinge loss, intersection over union (Jaccard index calculation), receiver operating characteristic, precision, and recall.

Contrary to our hypothesis, we found no significant difference between multimodal binary classification performance between the various model architectures using different text transformers for text encoding. In fact, the model without any text encoding (trained only on images), performed with the second highest accuracy of all models evaluated in experiment 1.

[1]

5.2. Experiment 2

In experiment 2, we expand upon the work of experiment 1 by allowing the transformer models to update their gradients during training. We evaluate the same array of metrics as in experiment 1 on XLM-R and Bert when these models are unfrozen during training. We hypothesize that leveraging the pretrained transformers combined with gradient decent on our meme dataset will allow us to improve the performance of the overall multimodal model. The transformers were both trained for 10 Epochs each, and were not subject to early stopping in order to allow for better learning of the meme dataset. All other hyper parameters remain consistent from experiment 1.

Table 2 shows the resulting metrics from experiment 2. Notice how Bert and XLM-R had almost opposite reactions to training. Bert drastically increased recall, but overall loss in accuracy as this model labels almost all memes as hateful. XLM-R by contrast slightly improved in recall and improved modestly in precision leading to a modest overall improvement.

[1] Boilerplate torch lightning code provided by Drivendata and Facebook at: https://www.drivendata.co/blog/hateful-memes-benchmark/

model	accuracy	auc	auroc	cohen kappa	f1	hamming dist	hinge	iou	roc	precision	recall
bert-frozen	57.0	0.461	0.586	0.140	0.559	0.430	0.978	0.398	0.140	0.556	0.562
bert-trained	47.8	0.265	0.490	-0.018	**0.624**	0.522	1.018	0.266	-0.032	0.479	**0.896**
xlm-r-frozen	52.2	0.451	0.541	0.042	0.498	0.478	0.995	0.352	0.042	0.507	0.490
xlm-r-trained	**57.7**	**0.471**	**0.603**	**0.149**	0.535	**0.423**	**0.964**	**0.403**	**0.150**	**0.572**	0.503

Table 2. Comparison of Frozen vs Unfrozen Text Transformers

6. Discussion

From the experiment results and analysis we can see that simply utilizing different transformers and text encoders do not always lead to better results. While these transformers have seen great successes in other applications, especially with natural language processing, they don't appear to be strong feature extractors for text. They likely suffer the same weaknesses as other models, where the dependency on time and positioning of words in the text is limiting what features that can be extracted. While the attention aspect of the transformers allows for better representation learning, and overcomes some of the shortcomings of other models especially with longer texts, the context vectors and representations created at the end of the transformers do not appear to be useful features for other models. Therefore, transformers cannot be relied upon as feature extractors for textual data, despite their successes. We can also conclude that meme text is significantly different from traditional text, likely due to their shorter nature and differing meanings. When utilizing the pretrained weights of the various transformers, they did not appear to perform well on classifying the memes, which may be a sign that meme text is different enough from traditional text and that further fine-tuning of the transformer weights would be necessary to improve performance instead of relying on purely transfer learning.

We also believe that the most important and difficult aspect of the hateful memes problem is the multi-modal nature of the data. Simply improving upon one aspect, such as just text or just images does not appear to significantly effect the results. While models such as ResNet or BERT perform extremely well on images or text alone, that does not mean they can succeed at this problem. While some phrases and pictures are automatically hateful and are easier to classify, like we stated during our dataset exploration, the more difficult task is classifying when a meme becomes hateful when the text applied to the image changes the meaning or context. Understanding how to fuse these two different data types together will be key for future work, as the end goal is a model that can properly understand the context and meaning behind the image and text together.

However, ultimately we were able to succeed in understanding the impacts that various text transformers have on this hateful meme classification problem. We have come to the understanding that they are also not great feature extractors, and that state of the art models likely rely on different techniques to perform well.

6.1. Multi-modal Data Fusion

There are multiple types of potential ways for multi-modal data fusion. For our work, we focused on data and feature level fusions, where we extracted the features from both the text and the image, flattened and concatenated them together, then trained a linear classifier on the fused features. We saw that our approach may have been a bit naive, as the transformers we used to extract features from the text did not to be helpful for the overall model, even though transformers have been shown to work very well on text. Our results were comparable to basic models that just relied on one type of data. This may be a sign that meme text is inherently simple, and complex models such as these novel transformers are not necessary. Or it could also mean that the transformers themselves do not prove to be good feature extractors, and users should be aware of this shortcoming when trying to utilize transfer learning.

We could have also attempted decision level fusion, where the models are trained to predict whether the text is hateful, then separately trained on if the image is hateful, then use some other model, such as a Bayesian classifier that can better utilize the existing predictions and information to predict if the meme is hateful. However, like we stated we expect that this method would likely suffer similar issues, where it would not properly learn the true context of the combined image and text.

Finally, we could attempted to use an entirely multi-modal model, instead of utilizing separate language and vision modules, have the model be designed in such a way to train on the combination of text and images without concatenation or other "fusion" methods.

6.2. Future Work

Ultimately our work was still limited due to time constraints and hardware capabilities. However, there were many other research areas that we also wanted to explore to expand upon our conclusions. Basic continuations of our work could involve more hyperparameter tuning, training the linear portions of our models for longer until convergence, finding the optimal amount of fine-tuning for each

of the transformers, and running the models for a higher number of epochs in order to investigate the role of early stopping.

More in-depth approaches could be similar to what other benchmark models have done, with dataset expansions, data augmentations, and ensemble methods [14]. We could also compare our performance on the related hateful meme dataset. [12]

7. Work Division

Our team worked quite well together with all members contributing to the project. All 4 team members implemented, trained, and analyzed the models. Model implementation was split up as follows: Gabriel implemented XLM models and gradient updating models, Nic implemented Bert models and metric scripts, Florian implemented Ernie models, and Matthew implemented Roberta models. All members contributed to the writing of the report. Matthew wrote the Abstract, Intro and part of Related Work, Gabriel wrote part of the related work, and the discussion, Nic wrote the experiment section, and Florian wrote the approach section. All team members maintained strong communication in group chat and stand up meetings throughout the duration of the project.

References

[1] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv:1911.02116 [cs]*, 2020. 2

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[3] Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. Challenges in representation learning: A report on three machine learning contests. In *International conference on neural information processing*, pages 117–124. Springer, 2013. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[5] 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. *arXiv preprint arXiv:2005.04790*, 2020. 1, 3

[6] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019. 2

[7] Hong Liang, Xiao Sun, Yunlei Sun, and Yuan Gao. Text feature extraction based on deep learning: a review. *Eurasip Journal on Wireless Communications and Networking*, 2017. 2

[8] 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. *arXiv preprint arXiv:1907.11692*, 2019. 2

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*, page 318–362. MIT Press, Jan 1986. 2

[10] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, Mar 2020. arXiv: 1808.03314. 2

[11] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019. 2

[12] Shardul Suryawanshi, Bharathi Raja Chakravarthi, Mihael Arcan, and Paul Buitelaar. Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, page 32–41. European Language Resources Association (ELRA), 2020. 3, 6

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv:1706.03762 [cs]*, 2017. 2

[14] Riza Velioglu and Jewgeni Rose. Detecting hate speech in memes using multimodal deep learning approaches: Prize-winning solution to hateful memes challenge. *arXiv:2012.12975 [cs]*, 2020. 3, 6

[15] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. 4

Student Name	Contributed Aspects	Details
Gabriel Wang	Literature review, Implementation, and Analysis	Researched related meme classification tasks and text feature extraction, see section 2 and 3. Implemented XLM-R with pretrained weights, and BERT and XLM-R with proper training for classification. See section 4 and 5, and the xlm-r branch in the code. Ran experiments and analyzed results, see section 6.
Florian Pesce	Tokenizer Research, Implementation and Analysis	Researched differences in text tokenizers. Implemented ERNIE-base and ERNIE-large with pre-trained weights. Ran experiments and analyzed results. Wrote code to investigate the effect of early stopping.
Matthew Sandry	Transformer Research, Data Restructuring, Implementation and Analysis	Researched differences in transformer training. Restructured data from the original dataset to fit project needs. Implemented RoBERTa model with pre-trained weights. Ran experiments and analyzed results.
Nicholas Farris	PM, Implementation and Analysis	Served as Project Manager. Implemented notext, boilerplate, and bert-frozen models. Implemented predictions and metrics scripts. Ran experiments and analyzed results

Table 3. Contributions of team members.