Am I the Asshole: How Language Models Perceive Ethics and Morality

Pariyakorn Chuisakun, Eleanor Duffey

University of South Florida
CAP4641 F24

Abstract

Morality is a system of values and principles concerning the distinction between right and wrong behavior. It is a fundamental aspect of human behavior and ethics. Not only does everyone operate with a different set of morals, but also other elements such as biases, emotional factors, and situational factors can influence one's moral judgement at a given time. In this project, we aim to explore the application of natural language processing techniques to analyze, classify, and understand human behavior and sentiments. The goal is to understand how well language models are able to perform moral judgement tasks.

Using data sourced from popular "Am I the Asshole?" (AITA) subreddit, we trained the model to classify each post as "You're the Asshole" (YTA) or "Not the Asshole" (NTA). To evaluate the model's ability in moral judgment tasks, accuracy is based on how well the model aligns with the majority of votes for a particular post. By leveraging transformer-based models, specifically from the HuggingFace ecosystem, the project integrates tokenization, pretrained embeddings, and fine-tuning strategies to better make moral judgments.

1. Introduction

Everyone has been involved in a situation and questioned whether or not they were in the wrong. People often doubt if their behavior in a given situation aligns with socially accepted norms and customs. Since their inception, internet forums have been an outlet for people to express themselves, explain their train of thought, and seek validation. Among these outlets is the "Am I the Asshole?" subreddit, a forum where the writer describes a moralistic scenario, and readers provide feedback on whether the writer's actions were justified. The appeal of AITA lies in its participatory nature, blending entertainment, ethical discourse, and emotional connection. These aspects make AITA a valuable dataset for examining human behavior, sentiment, and moral reasoning.

Language models have been shown to perform well in a variety of sequencing, classification, and analysis tasks. This project aims to determine how well LMs are able to align with the majority of human judgement patterns on certain ethical narratives.

2. Related Work

2.1 NLP Research on Morality

NLP has seen extensive research in analyzing morality and ethics within textual data. For instance, studies such as Sap et al. (2020) explore frameworks for detecting moral judgments and social norms within language, providing insights into implicit biases and moral reasoning. Similarly, Ghadiri et al. (2022) present methodologies for classifying ethical content in online text, highlighting the potential of transformer-based models in uncovering nuanced moral dimensions. This project focuses on subjective moral judgment. We aim to leverage the AITA subreddit dataset to classify the ethical stances in a narrative context.

2.2 NLP Research on Reddit

Recent research in applying NLP to Reddit data has emphasized the platform's potential as a resource for analyzing conversational dynamics and societal attitudes. Data from subreddits has been employed in studies like Kaur and Singh (2020) to evaluate emotional tone, sentiment, and social interaction. Similarly, Bassignana et al. (2022) showcase the utility of Reddit as a corpus for tasks like stance detection and opinion mining. Our project builds upon this foundation, employing AITA subreddit posts to tune transformer models for ethical stance classification. These efforts illustrate the platform's value as a corpus for research in ethics and NLP.

3. Dataset Collection

We started by finding a subreddit database that would have the necessary verdict labels for our classification task. We found the AITA Database created by Elle O'Brien which contained over 100k reddit posts scraped from the r/AmItheAsshole subreddit (Elle O'Brien, 2020). In total, this database contained 97,628 individual posts.

3.1 Cleaning and Preprocessing

To obtain the data, we closed the repository and used the author's provided instructions. Running a dvc (Data Version Control) command, we downloaded the data file. The data downloaded in the form of a large csv file containing values "id" (a unique string to identify each post), "timestamp" (time of post creation in Unix format), "title" (post title), "body" (post text), "edited" (timestamp at time of edit, False otherwise), "verdict" ("asshole", "not the asshole", "everyone sucks", or "no assholes here"), "score" (numerical difference between upvotes and downvotes), "num_comments" (total number of comments), and "is_asshole" (whether the user was declared the asshole) (O'Brien, 2020).

When initially downloaded, the formatting of the data was affected due to the punctuation in the original reddit posts, including indentation, quotes, and newlines. To make the data usable, we wrote code to remove the newline characters in the body of each post, losing some contextual formatting, but making the data usable as a csv list.

In order to use a binary classification model, we discarded alternative verdicts such as "everyone sucks here" and "no assholes here", so only "asshole" and "not asshole" remained. Due to inconsistent text formatting, characters such as left (“) and right (”) quotation marks had to be replaced with straight quotes ("), and backslashes (\) had to be doubled (\\) to avoid issues with escape characters. Once fixed, we appended the verdict, post title, and post text into separate "AH" and "NTA" files and counted of each. There were approximately 59,000 "not the asshole" (NTA) posts compared to approximately 21,000 "asshole" (YTA) posts.

3.2 Data Analysis

Based on the data from the two categories, here's a breakdown of some statistics:

	YTA	*NTA*
Average number of comments	111.39	80.42
Average word count	316.41	348.64
Median word count	285	323
Average score (upvotes - downvotes)	266.21	370.91
Percent containing edits	34.18%	23.79%

Table 1: Statistics between sets of data.

When looking at the most common nouns in each category, "car," "mother," and "boyfriend" were the most common when the original user was declared "not the asshole," while "girlfriend," "job," and "relationship" hold the top three spots in the "asshole" category. Here's a breakdown of the disparity between common words:

	YTA	*NTA*	*Difference*
"car"	3906	4752	+846
"boyfriend"	4634	3815	-819
"girlfriend"	4692	4424	-268
"dog"	4572	4143	-429

Table 2: Common nouns between datasets.

This data does not account for acronyms like "bf" and "gf" when accounting for "boyfriend" and "girlfriend" counts. This is due to the fact that the user may have been using these acronyms for a different purpose and assuming their intention would inflict personal biases onto the data.

We leveraged spaCy library for tokenization and included a custom list of stop words to exclude non-context and narrative words. We implemented a function to extract the most relevant words (nouns, verbs, and adjectives) after filtering out high-frequency terms. The texts were grouped by their labels, and top words were identified for each group. This helped us uncover distinguishing patterns in the dataset and provided more insight into the context of most common ethical concerns within our datasets.

3.4 Balancing Data Set

To avoid bias in the training data, we shaved each data set down to 20,000 entries, and further divided each into a training, validation, and test set. We used an 80/10/10 split, resulting in 16,000 test posts, 2,000 validation posts, and 2,000 test posts per label. We then combined the “asshole” and “not the asshole” files for a total of 32,000 training samples, 4,000 validation samples, and 4,000 testing samples. When converting from csv to a jsonlist, we added “[TITLE]” and “[BODY]” before their respective categories before concatenating them into a single “body” section.

The inclusion of a validation set allows us to “pretest” our model on an unseen set and monitor the model’s performance. This prevents the model from overfitting on the training data before being exposed to the testing data.

4. Experiment

4.1 Environment

The primary platform for training and evaluation was Google Colab Pro, which provided access to advanced GPU resources. It is a cloud-hosted version of Jupyter Notebook, it enabled us to both write and execute code in a collaborative environment without establishing a local set-up.Colab Pro instances provide high-performance CPUs and up to 25GB of RAM, ensuring faster and more stable execution of preprocessing, training, and evaluation of tasks. Using Colab Pro, we could use NVIDIA A100-SXM4-40GB GPU (A100), an advanced GPU. Because of its high memory capacity and computational performance, with A100 GPU, we could work with larger language models.

The programming language used for developing and running the projection is Python 3.10. We also used PyTorch 2.0+, which provided a deep learning framework used for implementing and fine-tuning the modes, along with CUDA and GPU accelerations. Scikit-learn was employed to calculate evaluation metrics such as accuracy, precision, recall, and F1 score. We pulled our selected models from the HuggingFace database.

4.1 Models

4.1.1 DistilBERT

The first model we worked with on this project was DistilBERT. It is known to be a smaller, faster, lighter version of BERT. It is trained using knowledge distillation, where it learns to mimic the performance of BERT. We considered this model first because it requires a lower computational overhead while still maintaining a decent accuracy. However, its limitations led to undesirable outputs. It has a lower capacity, making it less capable of handling complex linguistic features, subtle context, or longer sequences (Sanh, 2020).

4.1.2 LongFormer

LongFormer is an adept model that introduces sliding window attention, allowing it to more efficiently process longer sequences. It maintains attention over the entire sequence, making it critical for nuanced decision making. In addition, its ability to handle long texts without truncating them makes it an option to explore for this task. However, this complex model requires significant resources, specifically on GPUs with limited memory. While it produced more favorable results, it made training and fine-tuning more challenging and time consuming (Beltagy, 2020).

4.1.3 BART

BART is a denoising autoencoder for pretraining sequence to sequence models. It is best known for excelling in tasks that require text generation or reconstruction. We considered this model because of its ability to perform well in both versatility and discriminative tasks. AITA posts are usually long, so with smaller models, the sequence is often truncated. In BART, pretraining includes scenarios in which part of the context is removed, so it is more adaptable to truncated texts. Although the generative aspects of BART may not add significant value for classification problems like AITA, it is still a powerful encoder-decoder model that is effective at understanding nuanced texts. While it did produce desirable results, it is heavier and more resource intensive when compared to BERT (Lewis, 2020).

4.1.4 BERT

BERT is pretrained on masked LM and next sentence prediction tasks. It is a transformer model with bidirectional encoder representations, so it is effective at understanding context and relationships in text. It has been proven to perform well on a variety of NLP tasks and can be easily fine-tuned for more specific tasks (Devlin, 2019).

4.1.5 RoBERTa

An optimized version of BERT, RoBERTa; is trained with a larger corpus and can learn bidirectional context without NSP and masking. It has been observed to achieve better accuracy than BERT in classification tasks with a similar token length limitation of 512 tokens, which still makes it suitable for this experiment, as the average number of tokens in our training dataset is 433.287 (Liu, 2019).

4.2 Parameter Tuning

Once we determined that RoBERTa was the most suitable model, the training arguments needed to be tuned. For the number of epochs, which is the number of full passes through the training data, it was determined that three epochs was best suited. When trained on more epochs, the model exhibited signs of overfitting around epoch 4-5. The training loss was significantly decreasing, while validation loss was significantly increasing. This demonstrated that the model was not accurately predicting data unseen within the training data. Early stopping was also incorporated to assist in finding the optimal epochs. Regarding batch sizes, we determined that low batch sizes performed better. The number of batch sizes determines the number of training samples processed in one forward/backward pass per device; adjusting this parameter helps to balance memory usage and training efficiency. When we trained with high batch sizes, training was significantly faster but was more memory intensive. We also observed metrics to be better with low batch sizes, but we once again encountered an overfitting issue. To mitigate this, we enabled gradient accumulation step, which also helped stimulate a higher batch size.

For learning rate, we determined that a low learning rate, specifically 1e-5, was best suited for this project. Learning rate helps to control the step size at each iteration while moving toward a minimum of the loss function. This helps determine how quickly the model adapts to the data. We observed that higher learning rates tend to overshoot and fail to converge properly. Choosing a low learning rate helped to ensure stable training and avoid overfitting or underfitting. With lower learning rates, there was the concern that it would not be able to converge, so in order to discourage this we used cosine as our learning rate scheduler. A learning rate scheduler determines how learning rate changes during training. Using a cosine scheduler created a smooth reduction of learning rate and showed better convergence compared to linear scheduling.

Another integration to avoid overfitting was through the use of weight decay, which is a regularization technique that penalizes large weights. This was especially helpful to balance underfitting and overfitting since we were working with low batch sizes.

When we explored varying values of warmup steps and accumulation steps, there was no significant change in the resulting values to suggest further implementation of these changes. Each of the values remained within 0.02 of each other, which was not enough to conclusively say it had an impact on the performance.

Logging steps define how often training metrics are logged during training. This was another parameter we adjusted to evaluate how it changes the metrics. Though the changes resulted in negligible differences across all metrics.

4.3 Fine-Tuning

Prior to fine tuning, we attempted to incorporate transfer learning to further pretrain the model. We pretrained the model further on similar tasks, like sentiment classification, as an attempt to improve metrics. After its incorporation, the model performed worse. This could be because of task misalignment, as sentiment analysis generally focuses on determining positive, negative, and neutral emotions. The AITA tasks require assessing context, actions, and intent within narratives. Pretraining on sentiment data might not provide the nuanced understanding needed for ethical or social reasoning. The misalignment could be leading the model to learn irrelevant features that do not contribute to AITA classification.

To further enhance the model's performance, we accumulated a custom training function to explicitly use Cross-Entropy Loss (CE loss). CE loss is a widely used loss function for classification tasks. It measures the difference between the predicted probabilities and the true labels, guiding the model toward more accurate predictions. This showed improvements in accuracy and the F1 score. Recall also improved, while precision was slightly lower. When evaluating training and validation loss, an issue of overfitting arose. This was mitigated by applying label smoothing in the loss function to prevent the model from becoming overly confident in its predictions. This adjustment helped balance the model's learning process, allowing it to generalize better to unseen data while reducing the risk of overly confident yet incorrect predictions.

We attempted freezing the lower layers of the model during fine-tuning to focus updates on higher, task specific layers. Though after implementation, it improved loss but worsened metrics elsewhere. While fine-tuning, we encountered an exploding gradient problem, where the gradients became too large and the losses fluctuated rapidly. We mitigated this through the use of gradient clipping, which is meant to limit the size of the gradients during backpropagation to stabilize training. While this helped resolve the issue, it was not included in the final model as we discovered that the main issue was that the learning rate was too high.

5. Results

5.1 Evaluation Metrics

We examined seven main evaluation metrics, with the first two pertaining to the training process. Training Loss measures how well the model performs on the training data. A low training loss is desirable; however too low of a loss can be an indicator of overfitting.

The second training metric is Validation Loss, which is a measure of how well the model performs on the validation set. The performance of this value is measured in comparison to the Training Loss. A low Training Loss with a high Validation loss signal that the model has overfitted to the training data and underperforms unseen data. If both values are high, this signals underfitting, but if both values are low, this is a more desirable state (Baeldung, 2024). Validation Loss and Training Loss were used in evaluating how well the model was learning. The metrics detailed in the following paragraphs were used when evaluating how well the model performed after pre-training.

First, we examined the Evaluation Loss. Much like the previous aforementioned two metrics, the Evaluation Loss describes how well the model performed on the evaluation (testing) set. An optimal value is one equivalent to validation loss, as it demonstrates that the model performs well on unseen data.

The second metric is the Evaluation Accuracy, which measures how often the model was correct in its predictions.

The third metric is the Precision, which is a measure of how accurately the model predicted that a story belonged to the “not the asshole” class. This is of how we established the binary classification labels: “not the asshole” corresponding to a 1 and “asshole” corresponded to a 0. Precision calculates the number of correctly assigned 1’s over the total number of assigned 1’s.

The fourth metric is Recall, which is the number of times the model was able to identify the “not the asshole” cases over the total number of “not the asshole” cases it was presented.

The fifth metric is the F1 score, which is the weighted average of the precision and recall. If the F1 score is low, it may signify that precision or recall is low. If F1 is high, it reflects a well-tuned model that is able to identify the correct class (Jurafsky, 2024).

5.2 Results

This is how each of the models performed after 3 epochs:

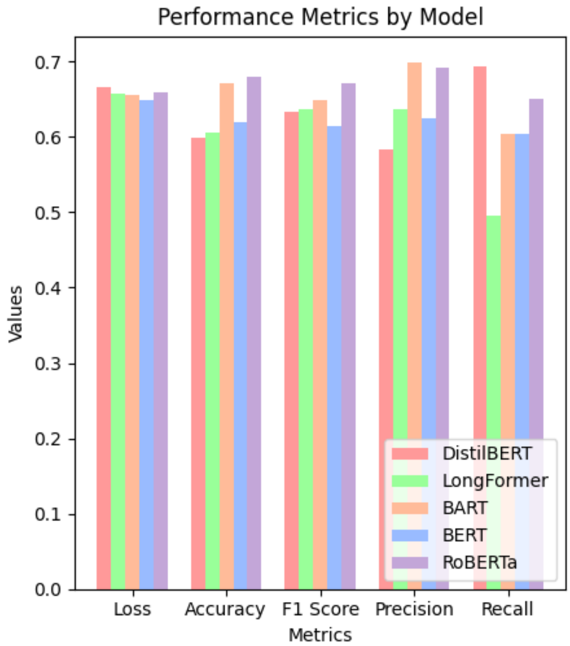

Figure 1: Bar graph showing the difference between model performance.

Although there is a negligible difference between the models, we decided to move forward with RoBERTa, because it produced the highest accuracy alongside higher precision, recall, and higher F1 scores.

After our previous exploration of fine-tuning parameters, we settled on a train and evaluation batch size of 4, 50 logging steps, weight decay of 0.01, 500 warmup steps, and 1 gradient accumulation step.

	Precision	*Recall*	*F1-Score*
YTA	0.65	0.69	0.67
NTA	0.67	0.62	0.65
macro average	0.66	0.66	0.66

Table 3: Metrics from the Classification Report of the Final Model

The classification results from the RoBERTa model show a moderate performance with an accuracy of 65.87%. The precision, recall, and F1 scores for both classes (YTA and NTA) are relatively balanced. This indicates that the model performs similarly on both labels, which is expected with a balanced dataset. The macro average scores confirm that the performance across the two classes is consistent. The results show that the model captures some relevant features for the task, but the smaller size of the model and token limit ultimately inhibit the performance. Further tuning and exploring the use of larger models may improve outcomes in the future.

6. Conclusion

This project explored the application of NLP to classify moral judgments in AITA subreddit posts. Through rigorous parameter tuning and fine-tuning of transformer models, we identified key configurations to optimize model performance. Despite the limitations of computational resources, we were able to achieve a balanced accuracy of approximately 66%. This value suggests some learning, as a baseline value of 50% would be achieved from random guessing.

7. Discussion

7.1 Limitations

A major limitation we faced was a lack of resources. Many of the larger models, such as LongFormer and Llama, required more RAM than was available on Google Colab. As a result, we had to use smaller models that couldn't achieve higher levels of accuracy. Additionally, we were limited by the amount of time these models take to train. Some models were predicted to take hours to train, and with limited number of CPU tokens on Colab, we ultimately decided to focus on fine-tuning smaller models that took around 5 minutes per epoch.

7.2 Ethics Statement

This project explored the use of a language model to mimic human ethics in the form of Am I The Asshole posts from the subreddit of the same name. The data was taken from a public forum, with individual usernames removed to ensure user privacy. Additionally, the data was used for educational purposes and was not redistributed.

To mitigate numerical biases in the dataset, we ensured an equal distribution between the two classes. However, the verdict of each post may reflect the personal biases of the commenters. Our model attempts to mimic these judgements and may also reflect these biases. Our model is not being used to make serious judgements and is only for educational and research purposes.

8. Future Work

We believe we could further improve accuracy with greater resources. Because of limited RAM, we were only able to dedicate our time to smaller models that handled smaller token sequences. With more computational power, we could run models such as LongFormer or Llama, which can handle inputs of up to 1024 tokens.

Additionally, it would be interesting to see how well this model could be adapted into a multi-class classification model. In the original dataset, there were also labels "everyone sucks here" and "no assholes here" which pass judgements on both the original poster and the subjects of the story. One limitation of this would be data availability, as there are significantly fewer instances of these verdicts in both the dataset and the subreddit in general.

References

Baeldung. (2024). "Training and Validation Loss in Deep Learning". Baeldung. https://www.baeldung.com/cs/training-validation-loss-deep-learning

Bassignana, E., Platanios, E. A., & Espinosa Anke, L. (2022). "Stance Detection in Reddit Discussions." *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis.*

Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The Long-Document Transformer. arXiv preprint arXiv:2004.05150. https://arxiv.org/abs/2004.05150

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

Ghadiri, P., Moini, R., Yazdavar, A. H., & Sheth, A. (2022). "Ethics of AI in NLP: Detecting Moral Dimensions in Language." In *Proceedings of the Language Resources and Evaluation Conference (LREC).* https://aclanthology.org/2022.lrec-1.28.pdf.

Google Research. (2023). *Colaboratory: A Cloud-Based Jupyter Notebook Environment.* https://colab.research.google.com/

Honnibal, M., & Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. https://spacy.io

Jurafsky, D. & Martin, J. H. (2024). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*, 3rd edition. Online manuscript released August 20, 2024. https://web.stanford.edu/~jurafsky/slp3.

Kaur, A., & Singh, M. (2020). “Emotion Analysis of Reddit Data using NLP and Deep Learning.” Stanford CS230: Deep Learning Final Project Report. https://cs230.stanford.edu/projects_spring_2020/reports/38963762.pdf.

Kubota Ando, R. & Zhang, T. (2005). A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.703

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692. https://arxiv.org/abs/1907.11692

O’Brien, E. (2020). AITA Dataset. *Github*. https://github.com/iterative/aita_dataset.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library.* In Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc. https://pytorch.org/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research, 12:2825–2830. https://scikit-learn.org/

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2020). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108. https://arxiv.org/abs/1910.01108

Sap, M., Gabriel, S., Qin, L., Jurafsky, D., Smith, N. A., & Choi, Y. (2020). "Social Bias Frames: Reasoning about Social and Power Implications of Language." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). *Rethinking the Inception Architecture for Computer Vision.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://arxiv.org/abs/1512.00567

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv preprint arXiv:2302.13971. https://arxiv.org/abs/2302.13971

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., & Brew, J. (2020). 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. Association for Computational Linguistics.* https://doi.org/10.18653/v1/2020.emnlp-demos.6