SemEval 2023. ValueEval: Identification of Human Values behind Arguments
NLP Course Project

Nicola Carrassi, Gabriele Colasuonno, and **Antonio Guerra**
Master's Degree in Artificial Intelligence, University of Bologna
{ nicola.carrassi, gabriele.colasuonno, antonio.guerra7 }@studio.unibo.it

Abstract

In this report we will assess the use of multiple technique to tackle the SemEval23 Human Value Detection challenge. This task consists of classifying whether or not a given textual argument belongs to a given human value category. This task uses a set of 20 value categories compiled from the social science literature. Given the fact that a textual argument could belong to multiple human value categories, it is a Multi-Label classification task. Two approaches have been tested: Transformer-based with multiple architectures and Extreme Gradient Boosting. Our results suggest that the Transformer-based approach outperforms the XGBoost method on this task, achieving better overall performance in terms of precision, recall and F1-score, obtaining in our best configuration (based on RoBERTa) a Macro F1-Score of 0.42.

1 Introduction

The demand for responsible NLP technologies which are more inclusive, fair and explainable has increased in the last years. Researchers have studied how to align machines with human values as one of the directions which allows to obtain responsible AI that understands human morals and values. Human values are of concern to most of social sciences and have also been integrated into computational framework of argumentations. In NLP, values have not been analyzed for argument mining, as we do in this work. We decided to address the problem of recognizing human values from a textual argument and detect which values the text draws upon. Since in real world an argument can refer to more categories, this problem is a multi-label problem.

Multi-Label classification is a class of problems which aims to assign multiple labels for each instance simultaneously. Traditionally, those tasks have been dealt with ensemble methods (Opitz and Maclin, 1999)(Korovkinas and Danenas, 2017). These kind of approaches outperform traditional machine learning classifiers in tasks similar to this one like multi-label sentiment analysis (Kanakaraj and Guddeti, 2015). Unfortunately, the labels are very sparse due to the fact that we need to consider the power set of the labels as the set of possible values assignment to a text and this makes the labels very sparse, which leads to the problem of the long tail distribution. Overall, the usage of ensemble method represents a good baseline which does not require a massive amount of computational resources to be fitted and stored, especially if compared with more complex techniques such as Transformer-based approaches (Vaswani et al., 2017). Approaches based on transformers represent the state-of-the-art in a lot of different NLP approaches, among them there is also multi-label sentiment analysis on different datasets.

In this work we decided to test and compare the results obtained with an ensemble method based on Extreme Gradient Boosting (Chen and Guestrin, 2016), and then we tried several Transformer-based approaches, using different architectures.

We decided to use an ensemble method as this method is an alternative which represents a more democratic approach in the research field. This could also be seen as a baseline result which we then tried to improve using more complex techniques.

As regards transformers, we decided to test 3 different models and inspect the results obtained with them. The models are: BERT (Devlin et al., 2018a), RoBERTa (Liu et al., 2019), and XLNet (Yang et al., 2019). We evaluated the models using as metric for the performances Macro F1-Score, as stated in the paper of the challenge (Kiesel et al., 2022), obtaining a score of 0.42 with our best model.

2 Background

Multi-label classification are problems in which the examples are associated with a set of labels $Y \subseteq L$. One of the most common strategies to solve these tasks is problem transformation. Problem transformation methods consist into turning the learning problem into traditional single-label classification. This can be done in several ways:

- Assigning only one label to each example and transforming the problem into a multi-class classification task;
- Discarding every multi-label instance from the original dataset;
- Learn a binary classifier per class;
- Transforming the different sets of multi-label data into a single label.

In the last case the problem learns a single function $H : X \rightarrow P(L)$ where $P(L)$ is the power set of the original labels. The main drawback of this aspect is that it may lead to a huge number of classes and few example per class (Tsoumakas and Katakis, 2007).
To overcome the limitations that traditional machine learning models may have, researchers had the intuition that training more learners and decide based on the results obtained by them could help learning more patterns in the data. This idea is referred in the literature as Ensemble learning and the main concept behind this is that weighting and aggregating several individual classifiers will be better than relying on the results of a single learner. This concepts in the context of supervised learning have been explored since the 1970s and had a significant increase in importance in the 90s when (Hansen and Salamon, 1990) showed that the generalization error of neural networks could be reduced using ensemble methods. Since the 90s, ensemble methods have become always more important in the literature and the number of published papers per year has been constantly growing in time.
Boosting is a type of ensemble learning in which the new predictors are added sequentially; each new predictor tries to correct the errors of the previous one. This is done assigning a weight value to each training sample, which initially have equal value and when an element contributes to error, its weight value is increased.

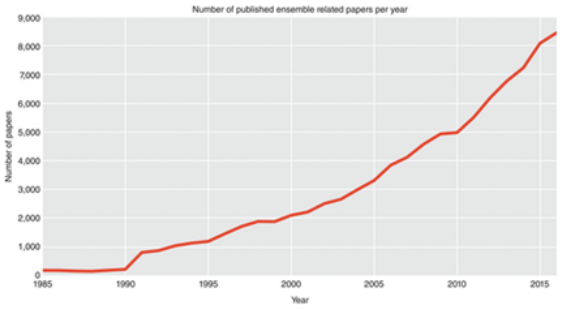

Figure 1: Graph showing the increase of paper on Ensemble methods, image taken from (Sagi and Rokach, 2018)

Among boosting techniques one of the most powerful alternatives is represented by Gradient Boosting, which is a system of machine learning boosting, that represents a decision tree for large and complex data. It relies on the presumption that the next possible model will minimize the gross prediction error if combined with the previous set of models.

3 System description

Our first architecture is based on an ensemble method, specifically we tried to implement an Extreme Gradient Boosting (XGBoost) classifier. In particular, XGBoost makes use of decision trees with boosted gradient, providing improved speed and performance over other ensemble methods.
To efficiently implement the classifier we relied on the XGBoost library (Chen and Guestrin, 2016), i.e. a distributed gradient boosting library that has been developed to be very effective, adaptable, and portable. It uses the Gradient Boosting framework to implement machine learning algorithms.
Furthermore, analyzing this first architecture in more detail, we have converted the problem into a multi-class problem using the *LabelPowerset* method implemented in the scikit-multilearn library (Szymański and Kajdanowicz, 2017). *LabelPowerset* is a problem transformation approach that transforms a multi-label problem into a multi-class problem by considering all combinations of unique labels found in the training data as classes. Although this problem transformation approach is recommended in the literature, it generates the so-called *long-tail* problem, i.e. there are many combinations of labels and some of them are very rare. In this particular case, it should be specified that given the high number of labels in our dataset *LabelPowerset* could generate up to 2^{20} different classes, i.e. the worst case is the power set of all

the original labels, if they were all present in the training set.
Then, to obtain the set of features that will be passed to the model for training, the stopwords and punctuation are first removed through preprocessing steps on the data and subsequently the latter are converted through the *TfidfVectorizer* method of the Scikit-Learn library (Buitinck et al., 2013) into their TF-IDF representation. Finally, the TF-IDF features are used to train the classifier to predict the correct element of the powerset of the labels.
An image representing this first proposed architecture can be seen by looking at figure 2.

In our second architecture we used and tested three different transformer models. For the implementation and training of these models, we relied on the *transformers* library provided by Huggingface (Wolf et al., 2020) using Pytorch as backend.
In detail, three different architectures were tested:

- **BERT base model (uncased)**(Devlin et al., 2018a): BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based neural network model for natural language processing tasks such as language understanding and sentiment analysis. It's trained to understand the context of a word based on the words that come before and after it in a sentence, allowing it to achieve state-of-the-art performance on a wide range of NLP tasks. BERT models can be fine-tuned for specific tasks such as question answering and named entity recognition, it was trained on a large corpus of text data such as Wikipedia and BooksCorpus, and it has been considered a breakthrough on many NLP tasks, as well as being able to handle multiple languages. The main advantage of BERT is its ability to understand the context of a word in a sentence, which enables it to be highly accurate in various NLP tasks.

- **RoBERTa model**(Liu et al., 2019): RoBERTa (Robustly Optimized BERT Pre-training) is a variation of the BERT model that is designed to improve its performance on various natural language processing (NLP) tasks. It makes changes to the original BERT model such as training on a larger dataset, longer training times, removing the Next Sentence Prediction objective, using dynamic masking and a lower learning rate during fine-tuning. These changes aims to allow the model to learn more robust representations and improve its performance on NLP tasks compared to the original BERT model.

- **XLNet**(Yang et al., 2019): XLNet (eXtreme Language Modeling) is a permutation-based language model introduced by Google AI in 2019. It differs from BERT and GPT-2, which are autoregressive models, by conditioning the probability of each word on all other words in the sentence. This allows it to overcome some of the limitations of autoregressive models and achieve better performance on natural language understanding and generation tasks. XLNet has been pre-trained on a large corpus of text data and fine-tuned on multiple natural language understanding tasks, and it has shown to outperform BERT and GPT-2 on several of those tasks, and it can be used on multiple languages.

Furthermore, all the models have been tested using the corresponding tokenizers provided by the Hugging Face transformers library. In particular, to adapt these three transformer models for multi-label classification, the *AutoModelForSequenceClassification* class from the transformers library was used. This class adds a sequence classification/regression head to the transformer architecture, allowing the performance of various downstream classification tasks, including multi-label classification.
To train our transformer-based model on the multi-label classification task, we employed an appropriate loss function. In particular, the default choice for this type of classification is the Binary Cross Entropy loss (BCE) combined with the Sigmoid activation function. Unlike multi-class classification, where the Softmax activation function is used and all probabilities must add up to 1, in the multi-label case, we must treat each probability individually and therefore Sigmoid activation is more suitable.
Moreover, in order to address the class imbalance present in our dataset, which will be discussed further in Section 4, we also experimented the Distribution Balanced Loss (DBloss) proposed by (Huang et al., 2021). We achieved this by downloading the Pytorch implementation from the official Github repository (source code available at: https://github.com/Roche/BalancedLossNLP).

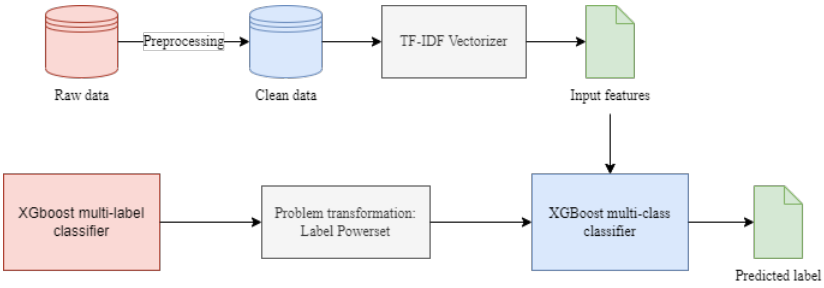

Figure 2: XGboost architecture

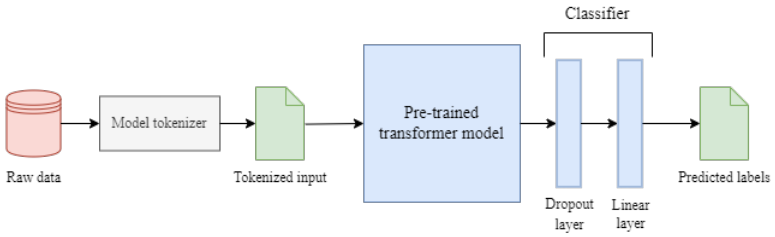

Figure 3: Transformer-based architecture. Note that pre-trained transformer model block can be either BERT, RoBERTa or XLNet.

Specifically, switching between different loss functions required overriding the "Forward" method of HuggingFace's *AutoModelForSequenceClassification* class, so we defined a custom class specifically for multi-label classification that allows us to switch between the two different losses.

Finally, to fine-tune the three pre-trained transformer-based models and train the added classifier, we used the Trainer APIs provided by the HuggingFace Transformers library.

A diagram representing the second proposed architecture can be found in Figure 3.

4 Data

The SemEval23 dataset (Kiesel et al., 2023) is a resource for researchers working in the field of natural language processing, sentiment analysis, and multi-label classification. The dataset contains more than 5000 English texts that are labeled with 20 human values. These values include principles such as humilty, benevolence and tolerance, among others.

The dataset is divided into 3 parts: a training set, a validation set, and a test set. All these sets are provided as tab-separated values files (tsv files) with one header line. Moreover, for both training and validation sets, an additional tsv file is provided containing the ground truth labels.

In order to better evaluate how our models generalize on unseen data we discarded the original test set as it has no labels. Instead, we considered the validation set as test set and then split the training set considering 20% of the training data as validation data.

Next, we have analyzed the training data. Initially, we assessed the number of labels assigned to each row of our training dataset and found that the average number of labels per row is 3. However, we discovered that there are a few samples that have either 0 or more than 6 labels assigned. To enhance the effectiveness of our training data, we resolved to eliminate these rows. Subsequently, we removed any duplicates present in the original dataset.

As a final analysis of our training data, we evaluated the distribution of individual labels and discovered that there was an imbalance in the distribution of classes among them. Class imbalance can lead to poor performance of the model on the under-represented classes, particularly in applications where those labels are important. This can also cause an inclination of the model to predict the majority class instead of accurately predicting the minor class.

After several attempts to augment data for the highly under-represented classes which resulted in models exhibiting bias towards certain specific sentences, we determined that the following classes, which had fewer than 200 samples: Hedonism, Stimulation, and Conformity: interpersonal, would

not be included in the training phase.
Finally, to further enhance the performance of our models, we decided to implement data augmentation techniques on the training set. Specifically, we utilized TextGenie (Pandya, 2022), a library that aids in generating new samples through paraphrase generation with a generative model. The model employed to generate the new data was t5-small-tapaco (Pandya, 2021). As a result of this process, the number of training samples increased from approximately 4,000 to 12,540.
After completing the data augmentation phase, we moved to the data preprocessing stage. This step is particularly crucial for the XGboost-based approach, whereas the use of raw input is preferable when working with transformers.
Here is the list of the data preprocessing steps performed for the XGboost approach:

- Conversion to lowercase;
- Removal of contracted forms using the Contractions python library (Con);
- Punctuation removal;
- Stopwords removal using the NLTK library (Bird et al., 2009);
- Lemmatization using spaCy (Honnibal et al., 2020);

5 Experimental setup and results

As already described in the previous sections we considered two different approches to tackle this challenge: we exploited different Transformers model and an Ensemble approach trough XGBoost framework. For the evaluation of each approach we refer, as stated on the offical page of the challenge, to Macro F1-score, Precision and Recall averaged over all value categories and for each category individually. Moreover, we decided to evaluate also Micro F1-score because it is more suited for dataset with imbalanced classes, which is our case.
Starting with the Ensemble approach, the XGBoost framework provides different Machine Learning classifier algorithms, in our tests we tried the following: XGBClassifier and XGBRFClassifier which are XGBoost implementation of an Ensemble Tree Classifier and a Random Forest Classifier, the best results were provided by the second one.
Furthermore, to transform the problem into a Multi-Class Classification, as already explained in the previous section, we tried different approaches, all provided by Scikit Multilearn library:

- **Binary Relevance:** treats each label as a separate single-class classification problem;
- **Classifier Chain:** treats each label as a part of a conditioned chain of single-class classification problems;
- **Label Powerset:** treats each label combination as a separate class with one multi-class classification problem.

The last approach has provided the best results. Finally we have also tested different values for the following hyperparameters: max depth of each tree and number of estimators, the best results were provided by the first set to 7 and the second to 65, in particular with this approach we reached a Micro F1-Score of 0.24 and a Macro F1-Score of 0.10.
Concerning the Transformers approach, in order to obtain better performance, we have managed the input sentence in different ways exploiting the transformer special tokens. Summing up, for the first test we just concatenated the input columns without any special token, then we used two Separator tokens to divide the "Stance" column and finally we decided to use only one Separator token to divide the "Premise" column from the concatenation of the "Stance" and "Conclusion" columns. In the last case in order to add more natural language sense to the sentence we decided to add "This is" before the "Stance" column. Moreover, beyond the default Separator token, we tried also to add a custom special token without getting any improvement. Among the three approaches described the best results were provided using only one default Separator token so henceforth all tests with transformers will refer to this approach.

The use of public defenders should be mandatory ... **[SEP]** this is in favor of The use of public defenders should be mandatory

---

Figure 4: Example of an input after the special tokens addition.

We have fine-tuned the three aformentioned models for a maximum of five epochs stopping the training earlier in case of overfitting, on average this happened after the first 2/3 epochs as also described in the following article (Devlin et al., 2018b). We

conducted some tests to choose the optimal hyperparameters, in detail we tried different batch sizes, learning rates, optimizers and loss functions. With regard to the latter we decided to use Binary Cross Entropy (BCE) loss which is commonly used for multi-label text classification (Bengio et al., 2013) and Distribution-Balanced (DB) loss which provided the best results for multi-label text classification task as shown in (Huang et al., 2021).

The following table will show the best configuration for each model.

	BERT Base	RoBERTa Base	XLNet
Batch Size	64	32	64
Learning Rate	5e-5	2e-5	2e-5
Optimizer	Adamw	Adamw	Adamw
Loss	DB	BCE	DB

Table 1: Comparison of the different set of hyperparameters for each model.

For the evaluation on the test set we have operated two different thresholding methods as they directly impact the choice of a label for the multi-label problem (Fallah et al., 2022). The threshold can be adjusted in several ways, either to optimize all the labels: a global threshold, or to optimize each label individually so the number of thresholds is equal to the number of labels. For the global threshold we have simply used a general 0.5 value while for the individual ones we have computed for each label the threshold that maximizes the F1-Score, obviously all the computations are made on the training set.
The best results for each model are presented in the tables below, one for global and one for individual thresholding.

	BERT Base	RoBERTa Base	XLNet
Macro F1-Score	0.33	0.34	0.31
Micro F1-Score	0.50	0.52	0.49
Precision	0.66	0.68	0.69
Recall	0.28	0.29	0.25

Table 2: Comparison of the different results obtained from each model with global thresholding.

	BERT Base	RoBERTa Base	XLNet
Macro F1-Score	0.39	0.42	0.40
Micro F1-Score	0.49	0.47	0.50
Precision	0.42	0.43	0.41
Recall	0.38	0.48	0.41

Table 3: Comparison of the different results obtained from each model with individual thresholding.

6 Discussion

As shown in the previous section the Transformer approach outperforms the Ensemble one with XGBoost, in fact if with the former we achieve a maximum Macro F1-Score of 0.42, with the latter only 0.10.
Moreover, fine-tuning a Transformer model in this specific case with a relatively small dataset requires a lot less amount of time compared to the one needed to train from scratch a Random Forest classifier with XGBoost: in particular we are talking about three times longer training.
One of the reasons of the bad scores achieved with XGBoost framework may be that when exploiting the Label Powerset approach we have up to 2^{20} possible output that, with a small sized dataset like the one of SemEval23 challenge, is not the best possible scenario since the majority of possible output is not even present a single time in the dataset.
Concerning the three different Transformer model, RoBERTa provided the best result in almost every metric with both global and individual thresholding as we showed in the tables in the previous section. Actually we quite expected this as in (Liu et al., 2019) it is shown how it provided better results than BERT and XLNet in almost every downstream tasks.
Looking at our scores from a broader perspective we have not achieved fully satisfactionary results as the same models that we used performs much better on the same multi-label text classification task but on different datasets. So this gives us reason to believe that the imbalanced distribution of the SemEval23 challenge's dataset may not help to achieve higher scores and better performance generally speaking. Despite implementing various data balancing techniques, our scores did not see significant improvement. This is due to the class imbalance present in the dataset, where certain classes have a significantly lower distribution compared to others. Attempting to balance these classes through oversampling leads to overfitting and does not effectively address the issue of class imbalance.
The class imbalance in the dataset results in a lower score for classes with low distribution. This is because the model is less likely to correctly assign these labels to sentences due to the limited number of examples available for training. Conversely, the model is more likely to correctly classify labels with higher distribution as it has been trained on a

larger number of examples for these classes.

Input sentence: social media gives it users a place to seek support when in need whether emotional or financially, things that would be more difficult if not impossible to do outside of their home. this is against Social media brings more harm than good
Ground Truth Labels: ['Self-direction: action', 'Face', 'Security: personal', 'Benevolence: caring', 'Benevolence: dependability']
Predicted Labels: ['Security: personal', 'Benevolence: caring']

Figure 5: Example of a generated output with the missed classification of rare labels.

7 Conclusion

In conclusion, the results from this experiment show that the transformer-based approach, specifically RoBERTa, significantly outperforms the ensemble approach with XGBoost in the multi-label text classification task. The transformer-based approach not only achieved a higher maximum macro F1-Score of 0.42, but also required less training time compared to the XGBoost. The poor performance of the XGBoost approach can be attributed to the Label Powerset approach that generates a large number of possible outputs which may not be well-suited for a small-sized dataset like the one used in the SemEval23 challenge. Furthermore, the imbalanced distribution of the dataset likely contributed to the poor performance of the models, as many of the classes with low distribution were not well-represented in the training data. Despite some attempts to balance the data, the results still fell short of expectations, highlighting the need for a more comprehensive solution to address class imbalance in multi-label text classification tasks in the future.

References

Contractions - pypi.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. Bert: Pre-training of deep bidirectional transformers for language understanding.

Haytame Fallah, Patrice Bellot, Emmanuel Bruno, and Elisabeth Murisasco. 2022. Adapting transformers for multi-label text classification. In *Proceedings of the 2nd Joint Conference of the Information Retrieval Communities in Europe (CIRCLE 2022), Samatan, Gers, France, July 4-7, 2022*, volume 3178 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Lars Kai Hansen and Peter Salamon. 1990. Neural network ensembles. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(10):993–1001.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Yi Huang, Buse Giledereli, Abdullatif Köksal, Arzucan Özgür, and Elif Ozkirimli. 2021. Balancing methods for multi-label text classification with long-tailed class distribution.

Monisha Kanakaraj and Ram Mohana Reddy Guddeti. 2015. Nlp based sentiment analysis on twitter data using ensemble classifiers. In *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pages 1–5.

Johannes Kiesel, Milad Alshomary, Nicolas Handke, Xiaoni Cai, Henning Wachsmuth, and Benno Stein. 2022. Identifying the Human Values behind Arguments. In *60th Annual Meeting of the Association for Computational Linguistics (ACL 2022)*, pages 4459–4471. Association for Computational Linguistics.

Johannes Kiesel, Nailia Mirzakhmedova, Milad Alshomary, Maximilian Heinrich, Nicolas Handke, Xiaoni Cai, Valentin Barriere, Doratossadat Dastgheib, Omid Ghahroodi, Mohammad Ali Sadraei, Ehsaneddin Asgari, Henning Wachsmuth, and Benno Stein. 2023. Touché23-human-value-detection.

Konstantinas Korovkinas and Paulius Danenas. 2017. SVM and naïve bayes classification ensemble method for sentiment analysis. *Balt. J. Mod. Comput.*, 5(4).

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

David W. Opitz and Richard Maclin. 1999. Popular ensemble methods: An empirical study. *J. Artif. Intell. Res.*, 11:169–198.

Het Pandya. 2021. Paraphrase datasets and pretrained models. `https://github.com/hetpandya/textgenie`.

Het Pandya. 2022. Textgenie. `https://github.com/hetpandya/textgenie`.

Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *WIREs Data Mining Knowl. Discov.*, 8(4).

P. Szymański and T. Kajdanowicz. 2017. A scikit-based Python environment for performing multi-label classification. *ArXiv e-prints*.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-label classification: An overview. *Int. J. Data Warehous. Min.*, 3(3):1–13.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

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. 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.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.