Arthur Schopenhauer at Touché 2024: Multi-Lingual Text Classification Using Ensembles of Large Language Models

Notebook for the Touché Lab at CLEF 2024

Hamza Yunis

Abstract

This paper describes the submitted approach of Team Arthur Schopenhauer to Task 1 of the Touché lab at CLEF 2024. The goal of this task is twofold: detecting human values in texts (Subtask 1), and recognizing whether these values are attained or constrained (Subtask 2). The approach described in this paper simplifies Subtask 1 by restricting the detected values in a text to a maximum of one value. It also simplifies Subtask 2 by handling it separately from Subtask 1; that is, human values and attainment are detected independently of each other. This simplification strategy proved successful, as the submitted approach was ranked 2nd among the participating teams' best submissions (a single team can make multiple submissions) in Subtask 1 and was ranked 1st in Subtask 2. The described simplification results in two text-classification tasks, which are handled by fine-tuning and ensembling multiple BERT-based models.

Keywords

Touché, Human Value Detection, BERT, Large Language Models, Ensembling

1. Introduction

The decisions that a human individual makes are affected by the values that are held by this individual [1]. Human values also affect an individual's attitudes towards various issues and, by extension, the arguments that they express in writing [2]. Task 4 of SemEval-2023 [3] had the goal of identifying human values behind textual arguments.

The subject of this paper is Task 1 (Human Value Detection) of the Touché [4] lab (hosted at CLEF 2024), which in turn consists of two subtasks: Subtask 1 has the goal of identifying the human values that a specific piece of text references, while the goal of Subtask 2 is to recognize whether these values are attained or constrained in the text. For example, both of the following two texts (obtained form the task's dataset) reference the value "Universalism: concern". However, this value is attained in the first text and constrained in the second:

> *"Widely considered one of the darkest days of the Troubles, relatives of the victims have met regularly to mourn their loss and campaign for justice."*
> *"We were hoping that we would get recourse to justice for our dead family members and that hasn't happened."*

This paper describes the approach submitted by Team Arthur Schopenhauer to the aforementioned lab. The approach achieved the 9th best score in Subtask 1 (all higher-performing approaches were submitted by a single team, which means Team Arthur Schopenhauer was ranked 2nd), whereas it achieved the best score in Subtask 2.

As the previous example demonstrates, detecting human values in texts is very challenging and cannot be performed using classical NLP methods. For this reason, our approach relies on modern BERT-based architectures, which have demonstrated a high capability in natural-language-understanding tasks [5].

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France
EMAIL: hamza.uns88@gmail.com (H. Yunis)

2. Background

The dataset provided by the organizers consists of the training set (labeled), the validation set (labeled), and the test set (not labeled). It stems from the ValuesML project, itself part of a broad JRC initiative that aims for a deep insight into values and identities [6]. Once the models of our approach were developed, they were applied to the test set to predict its labels. The predicted labels were then submitted to the organizers via the TIRA platform [7] for evaluation.

The labeled part of the dataset contains zero-one labels for 19 human values, with two label columns for each value corresponding to *constrained* and *attained*, totaling 38 columns. A text may constrain or attain a specific value, but not both. However, there are texts where it is unclear whether the referenced value is attained or constrained, in which case both columns corresponding to the value are filled with 0.5.

The dataset contains texts from 9 languages. In addition, the organizers provide automated English translations for non-English texts. However, due to concerns regarding the accuracy of the translations, our approach uses the original texts and relies on multi-lingual language models.

3. System Overview

Our submitted approach tackles the two described subtasks independently. Furthermore, the labeled datasets were divided into English and non-English texts, for each of which a different set of models was fine-tuned. Upon applying the fine-tuned models to the test set, which was used for the final submission, the texts in the test set were split in the same manner and the appropriate models were applied to each part.

3.1. Task Simplification

This section describes how the original subtasks were transformed in order to simplify the model fine-tuning process.

3.1.1. Simplifying Subtask 1

Subtask 1, in its given form, corresponds to a multi-label classification problem, because a single text may refer to multiple human values; that is, a single data instance may belong to multiple classes simultaneously. However, preliminary data analysis showed that approximately 94% of the labeled texts have either one label or no label. Therefore, for the sake of simplicity, it was decided to restrict the fine-tuning process to these instances, which turns the problem into a single-label classification problem. This simplification required introducing the *no-label* class for texts that have no label.

3.1.2. Simplifying Subtask 2

Subtask 2 was tackled independently of Subtask 1, which means the models of Subtask 2 were fine-tuned to predict a given text's *attainment*, regardless of the human value that the text references. Accordingly, the simplified version of Subtask 2 corresponds to a single-label classification problem with two classes, namely *attained* and *constrained*.

3.2. Data Preprocessing

The major steps of data preprocessing are shown in Figure 1. It begins by merging together the training and validation sets, then *unuseful data* is cleaned from the merged set, after which the dataset is reshaped to reflect the task simplification described in Section 3.1, and finally a new

validation set is created within which the different strata of the labeled data are proportionally represented.

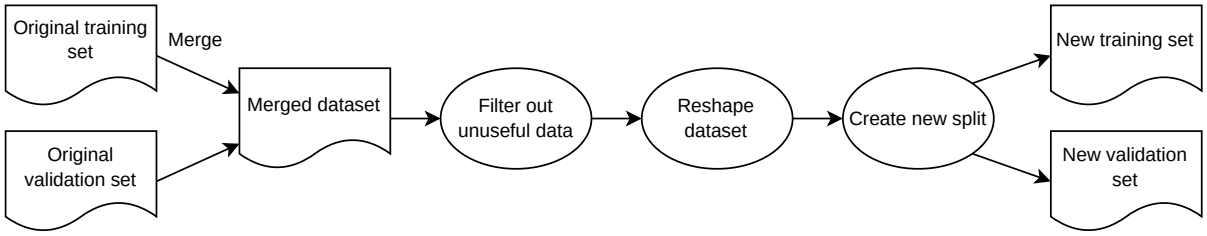

Figure 1: Data preprocessing pipeline.

3.2.1. Filtering Out Unuseful Data

After merging the training and validation sets, the following rows were removed from the merged set with the help of the pandas [8] library:

- Rows with duplicate texts (first occurrence kept).
- Rows with more than one label (in accordance with task simplification from Section 3.1.1).
- Rows with two words or less (believed to be noisy).

3.2.2. Reshaping the Dataset

To reflect the task simplification described in Section 3.1, the original 38 label columns were replaced by the following 2 columns:

hv_value A numeric code for the human value referenced by the text (including *no-label*).

attainment A numeric code for attainment (*constrained*, *attained*, or *unknown*). The *unknown* code is assigned to texts that do not have a human value label, or for which the attainment was *unclear* in the original dataset.

In addition, rows with the human value *Humility* were removed from the dataset. The reason for this additional filtering is that such rows are rare in the dataset and, after initial experiments, the fine-tuned models could not predict *Humility* with any accuracy.

3.2.3. Creating a New Split

The last step of data preprocessing was creating a new train-validation split. The validation set was created using the proportional allocation strategy with a sampling rate of 0.1, whereby each stratum is specified by a combination of language and label, for example all rows with language "EN" and label "Conformity: interpersonal" form one stratum. Splitting was achieved using the function `train_test_split` from scikit-learn [9] using the fixed random state 66 (not related to the random seed used when fine-tuning the models). This way, the validation set could be reproduced in different Python sessions.

3.3. Fine-Tuning the Models

For both subtasks, the approach relies on the pretrained models microsoft/deberta-v2-xxlarge [10] for English texts and FacebookAI/xlm-roberta-large [11] for non-English texts, both obtained from the Hugging Face Hub [12]. The process of producing the fine-tuned models is depicted in Figure 2.

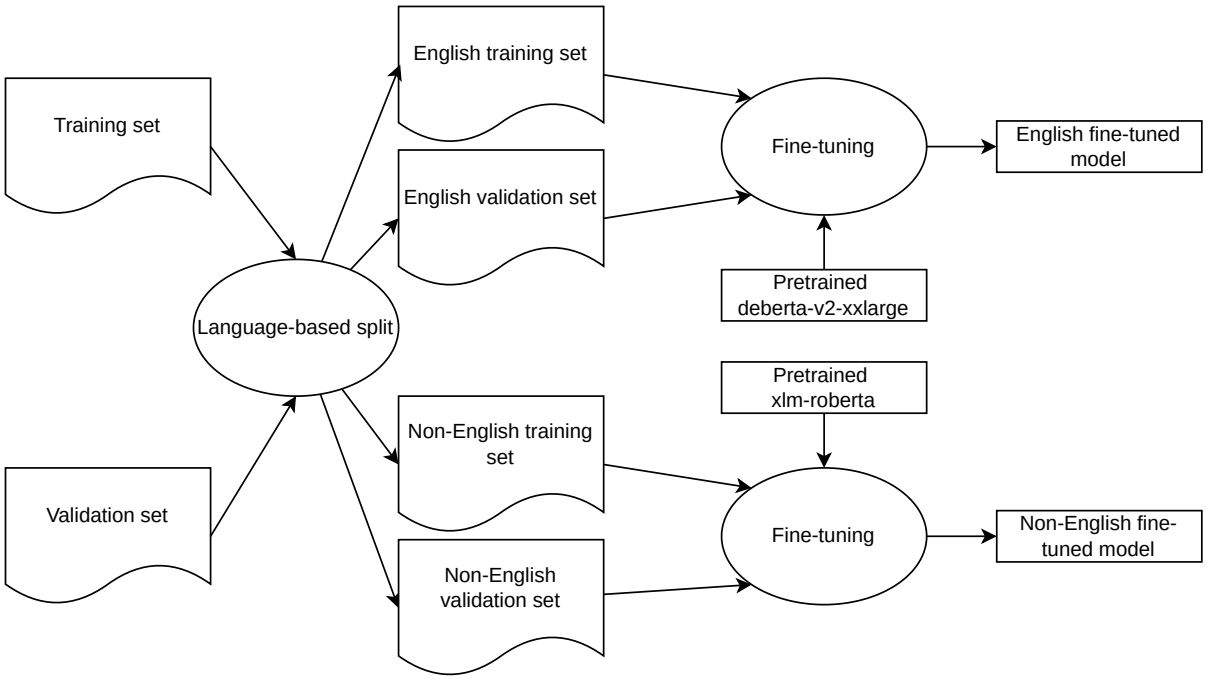

Figure 2: Conceptual overview of the fine-tuning process.

For Subtask 1, bagging [13] was applied using two four-model ensembles, one for each language subset. For Subtask 2, only one model was fine-tuned for each language subset, because using multiple models offered no improvement of predicative performance during experimentation. The classification heads of the fine-tuned models had 19 outputs[1] for Subtask 1 and 2 outputs for Subtask 2. It should be noted that the models for Subtask 2 were fine-tuned only on the data with known attainment; that is, rows with the *unknown* value in the *attainment* column were excluded.

Our approach applies the commonly used cross-entropy loss function [14]. However, due to observed class imbalance in Subtask 1, the use of the weighted cross-entropy loss function [15] was contemplated. Our experiments showed that using the weighted cross-entropy loss function (using inverse class frequencies as weights) delivers higher performance for some low-frequency classes, but lower performance overall; therefore, a combination of both weighted and non-weighted cross-entropy loss functions was used in each ensemble.

All ten models of our approach were fine-tuned using the same train-validation split, but with different hyperparameters, as described in Table 1. The remaining hyperparameters are described in Appendix A.

Fine-tuning was performed using PyTorch [16] directly, rather than the Hugging Face Trainer API. During fine-tuning, checkpointing was used, so the model checkpoint with the best F1-score (macro) was kept.

3.4. Ensembling Strategy

Ensembling is relevant only to Subtask 1, because for Subtask 2, only one model is used with each language subset.

Each of the models in Table 1 produces a predicted label[2], along with a probability of that

[1]The *Humility* class was removed from the original 19 classes and the *no-label* class was added.

[2]For details on extracting predictions from neural network outputs, see https://www.learnpytorch.io/02_pytorch_classification/

Table 1
Overview of the fine-tuned models used in the submitted approach. For the remaining hyperparameters, see Appendix A.

Model Name	Languages	Architecture	Random Seed	Loss Function
Subtask 1				
Model 1	English	deberta-v2-xxlarge	66	Cross-Entropy
Model 2	English	deberta-v2-xxlarge	66	Weighted Cross-Entropy
Model 3	English	deberta-v2-xxlarge	67	Cross-Entropy
Model 4	English	deberta-v2-xxlarge	67	Weighted Cross-Entropy
Model 5	Non-English	xlm-roberta	66	Cross-Entropy
Model 6	Non-English	xlm-roberta	66	Weighted Cross-Entropy
Model 7	Non-English	xlm-roberta	67	Cross-Entropy
Model 8	Non-English	xlm-roberta	67	Weighted Cross-Entropy
Subtask 2				
Model 9	English	deberta-v2-xxlarge	66	Cross-Entropy
Model 10	Non-English	xlm-roberta	66	Cross-Entropy

label. One common way to ensemble predictions is soft voting[3]. Our approach adjusts the original soft voting strategy by employing the concept of *safe prediction*, for want of a better term, which will be used to denote a prediction whose probability exceeds a certain threshold. With this definition of a safe prediction, ensembling was achieved using Algorithm 1 (**pruned soft voting**). The rationale behind this algorithm is as follows: If one of the predictions is *safe*, while the others are not, then it should be chosen as the final prediction, regardless of the remaining predictions.

The threshold used in the final submission was obtained by repeatedly applying Algorithm 1 with a different threshold to the validation set and selecting the threshold that produced the best macro F1-score. For the English ensemble, the optimal threshold was 0.44, for the non-English ensemble: 0.49.

Table 2 displays a performance comparison between pruned soft voting and ordinary soft voting using the validation set and shows that pruned soft voting offers a marginal improvement. However, it should be noted that, since the threshold for pruned soft voting was optimized using the validation set itself, the evaluation scores of pruned soft voting will be at least as high as those of soft voting, because soft voting is equivalent to pruned soft voting with threshold 0.

One point to consider when evaluating the ensembling strategies is that the *no-label* class (see 3.1.1) is included in the calculation of the F1-score (macro). This class will not be used in the final evaluation of the approach by the organizers, so for each evaluation that we performed, a corresponding *adjusted* F1-score which does not include the *no-label* class was calculated.

Table 2
Results of applying the final trained ensembles to the validation set using different voting strategies. The *adjusted* F1-scores correspond to F1-scores that do not include the *no-label* class (see 3.1.1).

Languages	F1-score (soft voting)	F1-score (pruned soft voting)	Adjusted F1-score (soft voting)	Adjusted F1-score (pruned soft voting)
English	0.4012	0.4405	0.3799	0.4211
Non-English	0.3963	0.4036	0.3788	0.3867
Combined	0.3989	0.4077	0.3807	0.3902

[3]For details on soft voting, see https://machinelearningmastery.com/voting-ensembles-with-python/

Algorithm 1 Pruned Soft Voting

Input:
Sequence S of pairs $(l_1, p_1), \ldots, (l_n, p_n)$ of predicted values for the label of one data instance, coupled with their prediction probabilities
Probability threshold T

Output:
One final label prediction

if exists at least one probability p_i in S such that $p_i \geq T$ **then**
return the final label prediction by applying soft voting only to those pairs (l_i, p_i) in S with $p_i \geq T$
else
return the final label prediction by applying soft voting to the entire sequence S
end if

4. Results

Upon submitting the approach, the fine-tuned models were applied to the test set and the results were exported to a .tsv file in the required format, which was submitted to the organizers. As texts with two words or less are believed to be noisy, the models of Subtask 1 were not applied to these texts; rather, *no-label* was manually predicted.

The evaluation results that were reported by the organizers are shown in Table 3 and Table 4. The reported F1-score for Subtask 1 (0.35) is significantly lower than the adjusted F1-score (0.3902) produced during our evaluation using the validation set (Table 2). This was expected for three reasons:

1. The the *Humility* label was not included when calculating the F1-scores in our evaluations. Since our submission never predicts the *Humility* label, the F1-score for this label was 0 when our submission was evaluated by the organizers, thus reducing the overall macro-averaged F1-score.
2. The filtering described in Section 3.2.1 was not applied to the test set. In particular, the test set does contain texts with multiple labels, which our approach cannot cope with.
3. In the process of fine-tuning the models, the validation set was used for checkpointing; therefore, the models have a degree of overfitness to the validation set.

5. Conclusion and Future Work

This paper presented the approach of Team Arthur Schopenhauer to Task 1 of the Touché lab at CLEF 2024. The main idea of the approach is simplifying the given subtasks. It simplifies Subtask 1 by eliminating the possibility of detecting multiple human values in a single text, and simplifies Subtask 2 by eliminating the possibility of detecting any dependence between referenced human values and attainment in a text. The source code for the approach is available under the following like: https://github.com/h-uns/clef2024-human-value-detection.

For future work, there are two notable areas of experimentation for improving the submitted approach:

- Using larger or newer model architectures than the ones used in the approach.
- Developing separate, specialized models that detect only certain subsets of human values, rather than all 19 values. Reducing the number of detectable human values is expected to improve the training efficiency of each model. In addition, combining such models can facilitate detecting multiple human values in a single text.

Table 3
Achieved F_1-score of each submission on the test dataset for Subtask 1. A ✓ indicates that the submission used the automatic translation to English. Baseline submissions shown in gray.

		F_1-score																		
Submission	**EN**	**All**	**Self-direction: thought**	**Self-direction: action**	**Stimulation**	**Hedonism**	**Achievement**	**Power: dominance**	**Power: resources**	**Face**	**Security: personal**	**Security: societal**	**Tradition**	**Conformity: rules**	**Conformity: interpersonal**	**Humility**	**Benevolence: caring**	**Benevolence: dependability**	**Universalism: concern**	**Universalism: nature**	**Universalism: tolerance**
valueeval24-arthur-schopenhauer		35	12	24	33	35	40	37	47	24	38	46	49	50	19	00	32	31	46	60	27
valueeval24-bert-baseline-en	✓	24	00	13	24	16	32	27	35	08	24	40	46	42	00	00	18	22	37	55	02
valueeval24-random-baseline	✓	06	02	07	05	02	11	08	10	03	04	14	03	11	03	00	05	04	09	04	02

Table 4
Achieved F_1-score of each submission on the test dataset for Subtask 2. A ✓ indicates that the submission used the automatic translation to English. Baseline submissions shown in gray.

		F_1-score																		
Submission	**EN**	**All**	**Self-direction: thought**	**Self-direction: action**	**Stimulation**	**Hedonism**	**Achievement**	**Power: dominance**	**Power: resources**	**Face**	**Security: personal**	**Security: societal**	**Tradition**	**Conformity: rules**	**Conformity: interpersonal**	**Humility**	**Benevolence: caring**	**Benevolence: dependability**	**Universalism: concern**	**Universalism: nature**	**Universalism: tolerance**
valueeval24-arthur-schopenhauer		83	77	83	85	88	87	73	84	80	82	84	78	80	79	74	91	89	86	85	81
valueeval24-bert-baseline-en	✓	81	83	79	86	88	84	77	80	74	84	81	78	78	79	87	89	86	85	81	78
valueeval24-random-baseline	✓	52	51	47	54	52	53	55	53	52	52	50	54	53	49	45	53	56	52	49	56

Acknowledgments

The approaches [17] and [18] from SemEval-2023 provided a valuable kickstart for this approach.

References

[1] S. H. Schwartz, J. Cieciuch, M. Vecchione, E. Davidov, R. Fischer, C. Beierlein, A. Ramos, M. Verkasalo, J.-E. Lönnqvist, K. Demirutku, et al., Refining the Theory of Basic Individual Values, Journal of personality and social psychology 103 (2012). doi:10.1037/a0029393.

[2] J. Kiesel, M. Alshomary, N. Handke, X. Cai, H. Wachsmuth, B. Stein, Identifying the Human Values behind Arguments, in: S. Muresan, P. Nakov, A. Villavicencio (Eds.), 60th Annual Meeting of the Association for Computational Linguistics (ACL 2022), Association for Computational Linguistics, 2022, pp. 4459–4471. doi:10.18653/v1/2022.acl-long.306.

[3] J. Kiesel, M. Alshomary, N. Mirzakhmedova, M. Heinrich, N. Handke, H. Wachsmuth, B. Stein, SemEval-2023 Task 4: ValueEval: Identification of Human Values behind

Arguments, in: R. Kumar, A. K. Ojha, A. S. Doğruöz, G. D. S. Martino, H. T. Madabushi (Eds.), 17th International Workshop on Semantic Evaluation (SemEval 2023), Association for Computational Linguistics, Toronto, Canada, 2023, pp. 2287–2303. doi:10.18653/v1/2023.semeval-1.313.

[4] J. Kiesel, Ç. Çöltekin, M. Heinrich, M. Fröbe, M. Alshomary, B. D. Longueville, T. Erjavec, N. Handke, M. Kopp, N. Ljubešić, K. Meden, N. Mirzakhmedova, V. Morkevičius, T. Reitis-Münstermann, M. Scharfbillig, N. Stefanovitch, H. Wachsmuth, M. Potthast, B. Stein, Overview of Touché 2024: Argumentation Systems, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2024.

[5] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805 (2018). URL: http://arxiv.org/abs/1810.04805. arXiv:1810.04805.

[6] M. Scharfbillig, L. Smillie, D. Mair, M. Sienkiewicz, J. Keimer, R. Pinho Dos Santos, H. Vinagreiro Alves, E. Vecchione, L. Scheunemann, Values and Identities - a Policymaker's Guide, Technical Report KJ-NA-30800-EN-N, European Commission's Joint Research Centre, Luxembourg, 2021. doi:10.2760/349527.

[7] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. doi:10.1007/978-3-031-28241-6_20.

[8] T. pandas development team, pandas-dev/pandas: Pandas, 2020. URL: https://doi.org/10.5281/zenodo.3509134. doi:10.5281/zenodo.3509134.

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

[10] P. He, X. Liu, J. Gao, W. Chen, Deberta: decoding-enhanced bert with disentangled attention, in: 9th International Conference on Learning Representations (ICLR 2021), OpenReview.net, 2021. URL: https://openreview.net/forum?id=XPZIaotutsD.

[11] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, V. Stoyanov, Unsupervised cross-lingual representation learning at scale, in: D. Jurafsky, J. Chai, N. Schluter, J. R. Tetreault (Eds.), Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL 2020), ACL, 2020, pp. 8440–8451. doi:10.18653/V1/2020.ACL-MAIN.747.

[12] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Huggingface's transformers: State-of-the-art natural language processing, 2020. arXiv:1910.03771.

[13] L. Breiman, Bagging predictors, Machine learning 24 (1996) 123–140.

[14] A. Mao, M. Mohri, Y. Zhong, Cross-entropy loss functions: Theoretical analysis and applications, 2023. URL: https://arxiv.org/abs/2304.07288. arXiv:2304.07288.

[15] T. H. Phan, K. Yamamoto, Resolving class imbalance in object detection with weighted cross entropy losses, arXiv preprint arXiv:2006.01413 (2020).

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, 2019. arXiv:1912.01703.

[17] D. Schroter, D. Dementieva, G. Groh, Adam-smith at SemEval-2023 task 4: Discovering human values in arguments with ensembles of transformer-based models, in: A. K. Ojha, A. S. Doğruöz, G. Da San Martino, H. Tayyar Madabushi, R. Kumar, E. Sartori (Eds.), Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023), Association for Computational Linguistics, Toronto, Canada, 2023, pp. 532–541. URL: https://aclanthology.org/2023.semeval-1.74. doi:10.18653/v1/2023.semeval-1.74.

[18] G. Balikas, John-arthur at SemEval-2023 task 4: Fine-tuning large language models for arguments classification, in: A. K. Ojha, A. S. Doğruöz, G. Da San Martino, H. Tayyar Madabushi, R. Kumar, E. Sartori (Eds.), Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023), Association for Computational Linguistics, Toronto, Canada, 2023, pp. 1428–1432. URL: https://aclanthology.org/2023.semeval-1.197. doi:10.18653/v1/2023.semeval-1.197.

A. Hyperparameters

Table 5
Overview of hyperparameters used in fine-tuning the models.

Hyperparameter	**Value**
Number of Epochs	10 for Subtask 1 (non-English); 12 for the rest
Batch Size	8 for deberta-v2-xxlarge; 16 for xlm-roberta
Optimizer	AdamW
Learning Rate	1×10^{-6}
Learning Rate Scheduler	Constant
Weight Decay	0.0 for base models; 0.01 for classification heads