SemEval-2023 Task 4: ValEval – Identification of Human Values behind Arguments

Spencer Paulissen and **Caroline J. Wendt**
University of Colorado Boulder
Department of Computer Science
spencer.paulissen@colorado.edu and caroline.wendt@colorado.edu

Abstract

Identifying expressions of human values in textual data is a crucial albeit complicated challenge, not least because ethics are highly variable, often implicit, and transcend circumstance. Opinions, arguments, and the like are generally founded upon more than one guiding principle, which are not necessarily independent. As such, little is known about how to classify and predict moral undertones in natural language sequences. Here, we describe and present a solution to ValEval, our shared contribution to SemEval 2023 Task 4. Our research design focuses on investigating chain classifier architectures with pretrained contextualized embeddings to detect 20 different human values in written arguments. We show that our best model substantially surpasses the classification performance of the baseline method established in prior work. We discuss limitations to our approach and outline promising directions for future work.

1 Introduction

1.1 Motivation

Human values are inextricably linked to the formation of thoughts, opinions, and actions. They underlie an individual's moral, ethical, and philosophical convictions which are manifest in their worldview. In essence, public and private discourse are expressions of human values. As online communication platforms increasingly provide social alternatives to the "town square," understanding written content, as it relates to the human values contained within it, is an interesting and important NLP challenge. ValEval presents a unique opportunity to investigate the human values behind the formation and presentation of opinions on controversial topics, particularly those with social, economic, and political implications.

1.2 Task Description

The main objective of ValEval [1] is to detect human values in arguments. Given an argument with three attributes and a finite set of human value categories, the system must classify whether or not the argument is associated with a given value for all values in the set. ValEval is composed of a single task rather than sub-tasks; however, the committee specifies the option to focus on a subset of the 20 human value categories for classification. Here, we employ a classification model using all 20 human value categories.

1.3 Related Work

Since its debut, BERT (Devlin et al., 2018) has become a bedrock of many NLP applications (Tenney et al., 2019). BERT enables researchers to expedite the task of creating contextual word embeddings. More recently, newer alternatives to BERT have become increasingly popular, including the GPT models created by Google. Yet, BERT continues to transform the traditional architecture for NLP problems: provided a sample as input to a pre-trained transformer language model, one can generate contextualized embeddings to then be passed to downstream tasks. Following in the steps of many others, we too utilize BERT in this way in our proposed solution to the present multi-label classification problem.

Multi-label classification is not new, and there are numerous methods proposed in the literature, which typically span one of two categories: problem transformation and algorithm adaption (Zhang and Zhou, 2014). Problem transformation methods decompose the complexities of multi-class classification into more familiar approaches. The simplest method of doing so is to treat each possible class as a binary classification problem (Zhang and Zhou, 2014), but this tends to be rather crude and ignores

[1] https://touche.webis.de/semeval23/touche23-web/index.html#evaluation

Human Values	Conclusion	Stance	Premise
Achievement Power: resources Security: personal Conformity: rules Benevolence: dependability	We should ban algorithmic trading	in favor of	Algorithmic trading allows computers to buy and sell assets suddenly and in massive quantities, creating shifts in the market and making human trading difficult.

Table 1: A glimpse at the dataset. In this training example, 25% of the possible human value categories are represented in the given argument.

potential inter-class dependencies. On the other hand, algorithm adaption methods attempt to reshape existing methods to directly accommodate the multi-label data. This, however, requires extensive knowledge of the method to be adapted itself as well as the changes to the loss metric required to handle the new labels.

All things considered, we contend that chain classifiers are a particularly good algorithm to approach the present problem because they grant the simplicity of binary classifiers in tandem with the capability to incorporate inter-label dependencies (Read et al., 2011). Each classifier in the chain is, in addition to the original input, provided with all predictions from preceding classifiers. In a 10-class setting, for instance, the first classifier in the chain receives the base input, while the tenth (and final) classifier receives the base input and the predictions for the nine previous classes. This provides the model more information to discover inter-class dependencies and make informed choices. Notably, this approach significantly relies on the order of classifiers, the importance of which can be circumvented by building an ensemble of chains with different orderings (Zhang and Zhou, 2014).

2 Methods

2.1 Data

The data for this task [2] are adapted from the task committee's previous work (Kiesel et al., 2022). The dataset contains 5220 uniquely identified natural language arguments (presented in English) based on African, Chinese, Indian, and American cultures. Each argument is comprised of a conclusion, stance, and premise string. The conclusion of an argument concisely describes the ultimate judgement or decision, the stance indicates whether the conclusion is in the affirmative or the opposition, and the premise statement justifies the conclusion. Each argument is also annotated with a binary vector of 20 human value categories, such that a given argument can be founded upon more than one human value. Table 1 provides a sample of five of the human value categories relative to an example argument from the dataset, which contains sentiments that evoke those particular values.

In preparation for our model, we sample the data uniformly at random to create a training set (80%) and development set (20%). Figure 1 displays the distribution of all possible human value labels across the training and development sets. While the representation of each human value is not balanced across the entire dataset, our training/development split is proportionally balanced within each human value category. We preprocess the data using word-based tokenization and encoding from a small pretrained BERT model variant intended for use when computational resources are limited (Bhargava et al., 2021; Turc et al., 2019). We adapt the traditional pairwise input method which is often used for question-answering models, to create an input representation suitable for both the BERT model and the input triples in our data (i.e., conclusion, stance, premise).

2.2 Model

While we tested multiple architectures, our final model is divided into two main components, as illustrated in Figure 2 in Appendix A. First, we use a pre-trained BERT model to generate contextualized embeddings for each of the conclusion, stance, and premise of a given sample. We then extract the pooling layer output of BERT, which corresponds to BERT's <CLS> token and serves as a contextualized embedding of the entire input. We pass that output to a hidden layer, comprised of one or more linear layers, to project to a specific dimension.

The output of the hidden layer is then connected to the head of the classifier chain. Each classifier in the chain is relatively simple; it includes a dropout

[2] https://zenodo.org/record/6818093#.Y4pqGuzMLIA

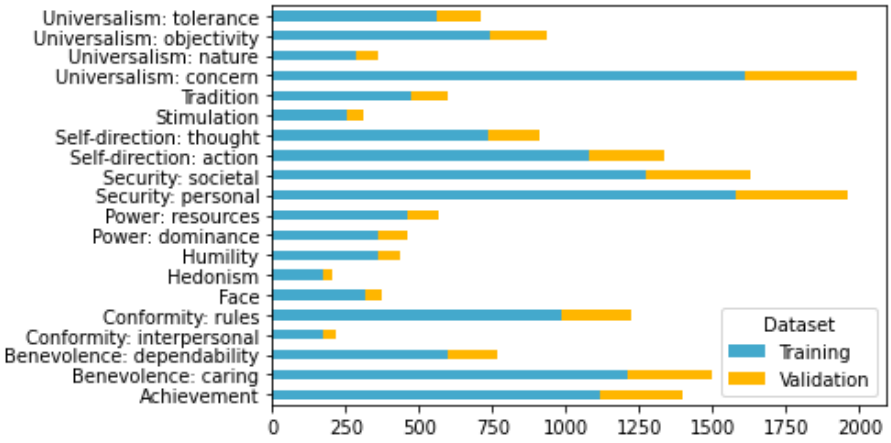

Figure 1: The distribution of human value categories across the training and development sets. Each value category appears to be proportionally well-represented in the training and development data.

layer for regularization, a linear layer for the prediction, and a sigmoid activation function. Each classifier makes a prediction on its assigned class, which is then appended to the input and passed to the next classifier in the chain. This process repeats for all 20 classifiers in the chain, at which point all generated predictions on each class are returned.

2.3 Experimental Design

Each model is trained over 100 epochs with the AdamW optimizer. Preliminary results show more epochs yield no improvements. Since we perform 20 binary classifications, we train our model using Binary Cross-Entropy (BCE) loss. For each training session, we change one or two of a number of parameters that define the model. These parameters are either architecture choices or hyperparameters. For our architecture, we experiment with unfreezing BERT weights, adjusting the number of layers in each classifier in the chain, and varying the number of hidden layers between BERT and the classifier chain. In terms of hyperparameters, we experiment with varying the dropout probability, the hidden size to project to, batch size, learning rate, and the classification threshold. As will be discussed in 2.4, we compare models using two different F1-scores.

The number of hidden layers and the prediction threshold appear to be the two particular parameters with the largest impact on performance. We observe that due to the complexity of the task, adding an additional layer between BERT and the classifier chain improves performance; yet, a third layer slightly decreases performance. We use the prediction threshold to determine the binary prediction status of an output. We initially define a threshold of 0.5, but discover that associating the threshold with the label density of the dataset significantly improves performance. That is, we incorporate the label density into the predictions in order to leverage the general class distribution. Our best results are associated with a threshold of 0.3426, which is twice the label density of the entire dataset. The results from our various training sessions are listed in Table 3 in Appendix A.

2.4 Evaluation Metrics

To gauge our model's performance during training, we calculate a flat F1-score on the development set after every epoch. This flat F1-score does not calculate an F1-score per class; rather, it assumes an entry-by-entry approach. After training, we use the evaluation method provided by the task committee, which calculates a per-class F1-score before averaging. In most cases, we observe our flat score to be reasonably close to the official score, so we continue to use our method for simplicity.

3 Results

We reference the original task paper to establish a baseline (Kiesel et al., 2022). The authors separate tasks into different levels of various granularity. We focus on the highest granularity (Level 1). Kiesel et al. (2022) test three Level 1 baseline models: BERT, an SVM, and a simple model that always outputs 1. The highest F1-score they achieve in this context is 0.25, with BERT. Our model achieves an F1-score of 0.67, a significant improvement from the baseline. We compare our result to theirs in terms of the F1 evaluation metric in Table 2.

Model	F1-Score
BERT	0.25
SVM	0.20
Baseline-1	0.16
Our Model	**0.67**

Table 2: Results comparison with the original baseline models described in Kiesel et al. (2022). Our model significantly outperforms the baseline for the Level 1 task, which includes all 20 separate labels.

4 Discussion

Overall, our model offers a vastly improved alternative to the relatively simplistic baseline models. The class labels associated with human values in this context are, evidently, not independent. We observe values to often co-occur in groupings, and thus emphasize that additional functionality to account for inter-class dependencies within the model is a key component to optimize classification performance. We believe this to be our main model attribute that contributes to improvement over the baselines, none of which contain special features to leverage inter-class dependencies. Moreover, we note that our model trains quickly, only requiring 20 to 30 minutes to complete each training session, which we deem to be another major strength relative to the complex nature of the task. We acknowledge in Section 5 that a larger model may outperform ours, but we underscore that the reduction in training time for small models is highly beneficial for development. All code and data for our implementation are available on GitHub [3].

5 Limitations

As mentioned in Section 4, our most notable limitation is model size. We did not observe evidence of overfitting during training, which suggests that a bigger model could potentially learn more. Further, we note that additional layers and parameters in our model are associated with top results. Even still, as model size increases, so do the demands of computing resources and training time, both of which posed strict limitations on our ability to expand. Accordingly, we did not explore other models such as GPT or ALBERT to generate contextualized embeddings, which may improve performance, at the cost of training speed and computation complexity.

[3] https://github.com/srpauliscu/nlp-shared-task

There are also many other multi-label classification methods to explore in this context. Classifier chains, while better than independent classifiers, are still relatively limited in their exploitation of inter-class dependencies. Other approaches, such as ensemble methods, may more intelligently incorporate inter-class dependencies to identify samples correctly. Even our simple chain classifier could be augmented with additional chains using different permutations of class ordering, if training time and complexity are not highly limiting factors. Given its simplicity and size, the chain classifier approach is well suited for our needs and resources, though a more complex model may perform better.

Lastly, data augmentation is another direction worthy of future investigation. The provided training set is not substantial, despite the relatively small size of our model. A model like GPT could generate additional samples from those given, which is not ideal, though slightly noisy data could still improve training performance. It is likely that a more expansive and complex model would be required to exploit the advantages of a larger dataset, which, again, imposes considerable tradeoffs.

6 Conclusion

Human value classification is a difficult task, but a worthwhile one to solve. Our approach utilizes a simple but effective model that significantly improves upon established baselines. With more time and resources, our idea could be expanded upon to perhaps further improve performance. We are excited to see what solutions other teams are able to develop for ValEval.

References

Prajjwal Bhargava, Aleksandr Drozd, and Anna Rogers. 2021. Generalization in nli: Ways (not) to go beyond simple heuristics.

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

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.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. 2011. Classifier chains for multi-label classification. *Machine Learning*, 85(3):333–359.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962.

Min-Ling Zhang and Zhi-Hua Zhou. 2014. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837.

A Appendix

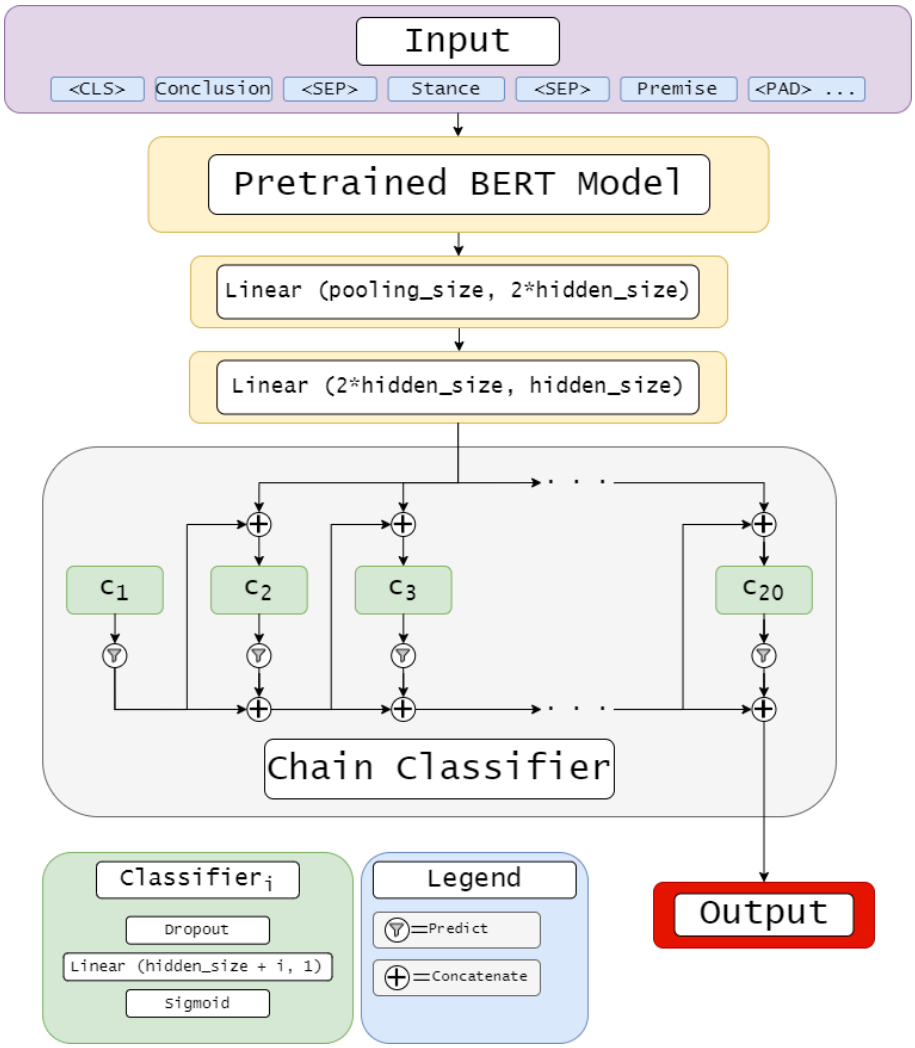

Figure 2: Overview of our model architecture. Each classifier in the classifier chain is comprised of a dropout layer, one linear layer, and a sigmoid activation function. Predictions are made using the threshold before appending each classifier's output to the input of the next classifier in the chain.

Architecture		**Hyperparameters**					**Scores**	
BERT Frozen?	Hidden Layers	Dropout	Hidden Size	Batch Size	Learning Rate	Threshold	Dev F1	Official F1
Yes	1	0.2	512	128	1x10-4	0.5	0.4104	0.3386
Yes	1	0.2	512	128	1x10-5	0.5	0.3031	0.1867
Yes	1	0.2	512	128	1x10-3	0.5	0.4468	0.4134
Yes	1	0.2	512	64	1x10-3	0.5	0.4502	0.4256
Yes	1	0.2	512	64	1x10-3	0.1713	0.4816	0.4840
Yes	1	0.2	512	64	1x10-3	0.3426	0.5034	0.5058
Yes	1	0.1	512	64	1x10-3	0.6574	0.3243	0.2919
Yes	1	0.05	512	64	1x10-3	0.3426	0.5247	0.5358
No	1	0.05	512	64	1x10-3	0.3426	0.2714	0.0884
Yes	1	0	512	32	1x10-3	0.3426	0.5343	0.5557
Yes	1	0	512	32	1x10-2	0.3426	0.5078	0.5084
Yes	1	0	512	32	5x10-3	0.3426	0.5239	0.5517
Yes	1	0	1024	32	1x10-3	0.3426	0.5423	0.5856
Yes	**2**	**0**	**1024**	**32**	**1x10-3**	**0.3426**	**0.5481**	**0.6659**
Yes	3	0	1024	32	1x10-3	0.3426	0.5438	0.6209

Table 3: Summary of our results during training, with our best model bolded. While our flat F1-score did not perfectly match with the official F1-score, it still served as a good indicator for increases or decreases in model performance.