Multi-Task Learning Improves Performance in Deep Argument Mining Models

Amirhossein Farzam
Duke University
a.farzam@duke.edu

Shashank Shekar
New York University
shashank.shekhar@nyu.edu

Isaac D. Mehlhaff
Texas A&M University
imehlhaff@tamu.edu

Marco Morucci
New York University
marco.morucci@nyu.edu

Abstract

The successful analysis of argumentative techniques in user-generated text is central to many downstream tasks such as political and market analysis. Recent argument mining tools use state-of-the-art deep learning methods to extract and annotate argumentative techniques from various online text corpora, but each task is treated as separate and different bespoke models are fine-tuned for each dataset. We show that different argument mining tasks share common semantic and logical structure by implementing a multi-task approach to argument mining that meets or exceeds performance from existing methods for the same problems. Our model builds a shared representation of the input and exploits similarities between tasks in order to further boost performance via parameter-sharing. Our results are important for argument mining as they show that different tasks share substantial similarities and suggest a holistic approach to the extraction of argumentative techniques from text.

1 Introduction

Text content generated by online users is a fundamental source of information for understanding the ideas, feelings, and behavior of large populations of interest for social scientists. Within these texts, it is important to be able to recognize ideas and worldviews expressed by individuals on a large scale. To this end, argument mining (AM) has emerged in recent years as a sub-field of natural language processing (NLP) focusing on creating language models capable of detecting and classifying argumentative strategies in online texts.

Within AM, several different sub-tasks have been proposed. For example, Misra and Walker (2013) focus on identifying agreement and disagreement in online texts, Oraby et al. (2017) propose a method to distinguish factual from emotional argumentation techniques, Lawrence et al. (2017) detect the presence of certain rhetorical figures in arguments, and Wachsmuth et al. (2017a,b) produce measures of argument quality. These are only some examples of the many distinct classification tasks that have been identified in AM, not to mention a wide range of work on span identification (e.g. Morio et al., 2022). In this paper, we suggest that all these AM sub-tasks share substantial similarity and use this idea to formulate a model that achieves high accuracy in several of these problems.

More specifically, existing work in AM treats many of the classification tasks within the field as separate problems and focuses on fine-tuning bespoke models for each task (e.g. Abbott et al., 2011; Stab and Gurevych, 2014, 2017; Sheng et al., 2020). While this approach has been demonstrated to work in many settings, it fails to take advantage of the substantial similarities between AM tasks.

In this paper we propose to take advantage of the similarities across AM tasks by constructing a multi-task model (Caruana, 1997; Zhang et al., 2014; Zhao et al., 2018; Liu et al., 2015), that is trained on all tasks at once and builds a shared latent representation of the inputs for each task, and uses this representation to make more accurate predictions for each individual task. Our models also provide evidence that AM sub-tasks do indeed share substantial conceptual overlap (Schulz et al., 2018); the latent representations of different tasks output by our model depicted in Figure 1 clearly depict clusters of individual tasks as well as substantial overlap between these clusters in representation space, indicating that the same latent features are informative for multiple tasks.

The model we propose achieves performance similar to or greater than existing models on all tasks for which we had information on previous metrics, and it also surpasses individual-task models fine-tuned on similar architectures for these tasks. In addition, our models allow for substantial

Proceedings of the 11th Workshop on Argument Mining, pages 46–58

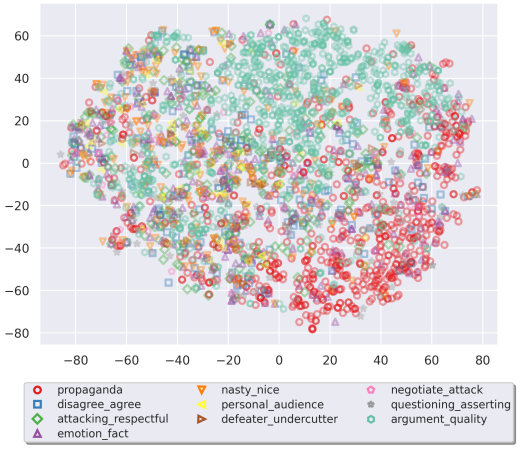

Figure 1: t-SNE projection of the BERT embedding included as the first layer in our model. Points are color-coded according to their task.

computational gains over individual-task models as they permit training inference for many outputs at once, instead of training and evaluating an individual model for each desired task.

Overall, our results have the important implication for AM as a field that further research and model-building should not only focus on taking advantage of the structure of the specific task of interest (e.g. Jin et al., 2022), but also on incorporating information from similar tasks into the model for better performance.

2 Related Work

We build on an active research agenda in argument mining (AM)—the automated extraction of argumentative structure, reasoning, and features from text (Habernal and Gurevych, 2017). Cabrio and Villata (2018) identify two stages in AM: identifying arguments within documents and classifying those arguments on their characteristics, such as supporting, attacking, or background information. Our work is situated in the second stage, involving the identification of features or typologies of arguments.

Much computational work in AM has investigated argumentation in online interactions (Abbott et al., 2011; Rosenthal and McKeown, 2015; Swanson et al., 2015), due in part to the vast amounts of available data and the ease of collecting it. But some scholars have used news articles to construct corpora of propaganda and fact-checking (e.g., Da San Martino et al., 2019; Rashkin et al., 2017). Still others have leveraged monologues like persuasive essays or legal decisions (Stab and Gurevych, 2014; Walker et al., 2019). We incorporate all three types of data into our models to further show that tasks with different data-generating processes and textual characteristics nevertheless exhibit common semantic structure.

There is evidence that many natural language tasks share a common core (Radford et al., 2019), and models trained on one task tend to also perform well on others. Halder et al. (2020) demonstrate that multi-task approaches benefit model performance in several natural language tasks such as topic detection and sentiment analysis. Multi-task approaches have been more rare within AM, but two existing works suggest the framework may offer benefits to these unique tasks.

Schulz et al. (2018) apply multi-task learning to token-level tagging for AM tasks. They consider six datasets each with different token annotations and train a recurrent model to learn all of them at once, providing initial evidence that gains in model performance can be attained via multi-task learning in AM. Similar results are echoed in Schiller et al., who focus on stance detection instead. Morio et al. (2022) present an end-to-end multi-task architecture for identifying argument components in unstructured text. Our task differs from theirs in two ways: First, Morio et al.'s model is especially focused on span identification and relation classification, wherein the model links, for example, premise to claim. Our primary objective in this paper would more accurately be described as component classification. Second, the component classification part of their model is focused on more traditional AM tasks like classifying texts as claims for or against. We focus instead on argument characteristics that are more complex or subjective and, in some cases, can be understood as being nested within overarching concepts.

Cheng et al. (2020) propose a model trained simultaneously on two tasks: argument identification within texts and argument-rebuttal pair matching across texts. Again, our approach is focused on argument classification, not span identification, and we aim to classify argument *types*. Moreover, our use of multi-task learning differs slightly. Whereas Cheng et al. train a model to perform two complementary but quite distinct tasks, we show that a single model can perform multiple argument classification tasks simultaneously. Accordingly, our proposed architecture differs from both Morio et al. and Cheng et al.

A prevalent model architecture for multi-task learning within computer vision involves segre-

gating the network into shared and task-specific components. This conventional structure, termed a "shared trunk" (Crawshaw, 2020), typically comprises a universal feature extractor, constructed of convolutional layers that are employed by all tasks, and a distinct output branch for each task (Zhang et al., 2014; Dai et al., 2016; Zhao et al., 2018; Liu et al., 2019). Further enhancements on this shared trunk template have been made by (Zhao et al., 2018) and (Liu et al., 2019), who incorporated task-specific modules into the original framework.

This template is not confined to computer vision but is also prevalent in multi-task learning models in NLP. Traditional feed-forward architectures, using the shared trunk template in combination with task-specific modules, have been utilized for multi-task NLP by a variety of researchers (Collobert and Weston, 2008; Collobert et al., 2011; Liu et al., 2015, 2016). These architectures bear a structural resemblance to their counterparts in computer vision, featuring a shared, global feature extractor followed by task-specific output branches. However, in the context of NLP, the features in question are text representations.

3 Data

We draw on three benchmark corpora to create a dataset with a diverse number of argument characteristics. We take eight tasks from the Internet Argument Corpus (IAC), a collection of posts extracted from several online debate and discussion fora (Abbott et al., 2016; Walker et al., 2012). Each post is annotated on a variety of characteristics, such as whether the post expresses disagreement or uses an emotion- or fact-based argument, with a value in $[-5, 5]$ on each characteristic. Some researchers have dichotomized these data by removing observations around the midpoint Oraby et al. (2015). This practice is not appropriate in the multi-task setting, however, as it would remove too much information that the model could use to build shared representations across tasks. Instead, we dichotomize the data by simply cutting on the scale midpoint.

A wide array of studies have used the IAC to construct unique tasks (Galitsky et al., 2018; Hartmann et al., 2019; Misra et al., 2016) and train single-task models (Lukin et al., 2017; Misra and Walker, 2013; Oraby et al., 2016). Three tasks have received notable attention: the classification of disagreement Abbott et al. (2011); Wang and Cardie (2014), emotional or factual arguments Oraby et al. (2015), and nasty or nice tone Lukin and Walker (2013).

The second benchmark corpus we draw on is IBM-Rank-30k, a corpus of crowd-sourced arguments Gretz et al. (2020). Two quality scoring functions then translated binary annotations into a continuous value of argument quality in $[0, 1]$. We use scores produced by the authors' weighted average scoring function because it accounts for coder reliability, leading to less noisy annotations. As with the IAC labels, we dichotomize the data by cutting on the scale midpoint.

The final corpus is introduced by Da San Martino et al. (2019), who collect articles from both propagandistic and non-propagandistic news sources and annotate sentences within each article that contain one or more of eighteen different propaganda techniques, such as loaded language or causal oversimplification. We extract all sentences from each article, including those that are annotated as containing no propaganda techniques. Data from all three corpora are combined to create our final dataset. We use 80% of the data for training and set aside 10% each for validation and test sets.

Finally, to help guard against overfitting, we conduct four types of data augmentation on the training set (Shorten et al., 2021). In back-translation, we translate the text into a different language, then translate it back to the original language. We choose German as the target language for its high lexical similarity to English. In contextual word embedding, we randomly choose thirty percent of tokens, feed the surrounding words to BERT (Devlin et al., 2018), and substitute the predicted word in for the original. In synonym augmentation, we randomly choose thirty percent of tokens and substitute the most similar word from the WordNet lexical database (Fellbaum, 1998). Finally, in random cropping, we randomly delete thirty percent of tokens. Table 1 shows the total number of observations in the training set as well as the class balance for each task.

4 Methodology

Our methodology is based on a multi-task learning approach which leverages the shared information across tasks corresponding to different sources of data, leading to improved performance on each task. The model architecture and the loss function are the two key components of our methodology.

Task	Training N	Balance
Propaganda	61,909	63/37
Disagree/Agree	66,684	21/79
Emotion/Fact	76,403	41/59
Attacking/Respectful	65,998	66/34
Nasty/Nice	65,829	73/27
Personal/Audience	24,749	25/75
Defeater/Undercutter	24,357	38/62
Negotiate/Attack	26,604	44/56
Questioning/Asserting	29,791	66/34
Argument Quality	96,036	6/94

Table 1: Size and class balance of training data.

Additionally, we make use of several standard training and optimization techniques, described in this section, in order to improve performance.

4.1 Model Architecture

Our model architecture shares a key similarity to network templates comprising a shared trunk feeding task-specific modules, common to multi-task learning architectures proposed in previous works (e.g. Zhao et al., 2018; Liu et al., 2015). Morio et al. (2022) also use an architecture with corpus-specific branches, but this portion of their architecture is only used in pre-training; they then fine-tune the model on each corpus individually. Our model is fully multi-task, thus showing the benefit of a multi-branch architecture that needs no fine-tuning.

This architecture aims to utilize shared information across tasks through the shared trunk while learning distinct task features through the task-specific modules. Following the same principle, we use a network with double-branching in layers following the shared trunk, in order to make use of commonalities across different types of tasks as well as more fine-grained information about each individual task.

We therefore use a feed-forward neural network with four sequential sets of layers: a base text embedding model shared across all tasks, followed by a shared encoder, which is followed by a double branching structure feeding two sets of task-specific modules. The main results we report use small BERT as the base embedding model (Devlin et al., 2018), but any base model can be used, and we report results across five such models below.

The base embedding model is followed by three dense layers, each followed by dropout. These layers help in learning features that are shared across tasks. The architecture then branches out to learn task-type and task-specific features. In particular, the architecture consists of four sets of layers, described below, and visualized in Figure 2. Each dense layer in the network uses a ReLU activation, with the exception of the final activation layer, which is a sigmoid for binary classification.

- **Shared embedding layers:** We use a BERT model (Devlin et al., 2018) to obtain an embedding of the text input. In order to keep the model size small and training practical, we use small BERT (Turc et al., 2019), which outputs a 128-dimensional embedding. The embedding model, shared across all tasks, is fine-tuned through our training.

- **Shared encoding layers:** In addition to the base embedding model, all tasks share an encoder, consisting of two sequential dense layers each followed by a dropout layer. This helps learn a shared representation, used by all tasks, while allowing for sparsity and reducing the problem to learning our target features.

- **Task-type Layers:** The first branching in the network architecture follows the shared layers aiming to learn coarse-grained task-specific features which are expected to share logical structures across tasks within each type. This is particularly suitable for multi-task learning on data consisting of a mixture of datasets, where the number of labels exceeds the number of sources in the mixture. Given such input data, in the first step towards learning the shared representation, the task-type layers learn dataset-specific features, while still utilizing commonalities between individual tasks sharing a dataset. For each task-type branch, we use two sequential dense layers each followed by dropout. Since we have three sets of target labels each corresponding to their own dataset, we use three main branches.

- **Task-specific Layers:** Each main branch further branches out into individual task layers. These layers aim to learn more fine-grained features from the representations produced through the main branches, and output a vector representation for each task. Each task-specific branch contains two sequential dense and dropout layers, which feed a sigmoid activation layer for predicting labels. The number of these sub-branches equals the number of individual features in the combined dataset. In the branch corresponding to propaganda

techniques, we additionally use a maximum pooling layer to reduce the eighteen individual propaganda technique labels to a single binary propaganda classification, predicting whether a propaganda technique is used.

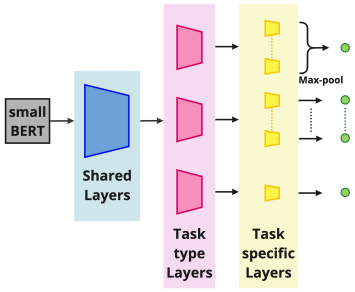

Figure 2: Model architecture.

The full architecture is illustrated in Figure 2. Using this architecture, we obtain a vector representation of the size of the fine-grained features described in the dataset. Note that this need not be the same as the size of the target output. It is not in this case, as we apply max-pooling to eighteen entries of the output corresponding to propaganda techniques in order to obtain a binary label. The network outputs a real-valued 10-dimensional vector which is then mapped to a binary vector of size 10 using individual thresholds for each label. For the results produced in the main text of this paper, we use a model with 32536 trainable parameters in addition to the parameters in small BERT.[1]

4.2 Loss Function

The loss function plays a crucial role in our multi-task learning approach, which relies on a mixed corpora corresponding to different task types. The custom loss function is designed to handle the data size imbalance across task types, in addition to class imbalance. This helps the model capture the contribution of each prediction to the overall loss, while task types are randomly shuffled in the input data. Considering this, given predicted labels $\hat{y}$ and true labels y, the total loss $\mathcal{L}$ used in our gradient descent optimization is:

$$\mathcal{L}\left(\hat{y}|y\right) = \sum_{k} \nu_k \mathcal{L}(\hat{y}|y, \mathcal{D}_k),$$

where D_k denotes the set of data point indices corresponding to task-type k, and $\nu_k \sim 1/|D_k|$ are the task-type weights. The loss for each task type k, which accounts for the class imbalance across output labels, is:

$$\mathcal{L}(\hat{y}|y, \mathcal{D}_k) \sim \frac{1}{|T_k|} \sum_{j \in D_k} \sum_{t \in T_k} \sum_{c \in \mathcal{C}_t} w_t^c \, l(\hat{y}_j | y_j = c),$$

where $l(.)$ is the loss function, T_k denotes the set of tasks within task type k, and $\mathcal{C}_t$ is the corresponding set of classes. The class weights w_t^c, which are proportional to the inverse of the enrichment of class c in task t within dataset k, counter the impact of class imbalance. We use the binary cross-entropy loss for the loss function l throughout our computations. In the implementation, the loss computation is vectorized using masked matrices to filter entries by task.

4.3 Model Training

For training the parameters in our model, we take advantage of an array of optimization and training enhancement techniques. We use an AdamW optimizer (Loshchilov and Hutter, 2017) for the stochastic gradient descent with an initial learning rate of 0.0003. To help avoid overfitting, we employ a weight decay rate of 0.01 and 40% dropout. We use 5% of data for warmup, a batch size of 256, and stop training after two epochs without a decrease in loss. We also incorporate threshold tuning, maximizing true positive rate while minimizing false positive rate, for optimal mapping of the sigmoid layer's output to binary labels. All training hyperparameters are tuned through a standard grid search over 72 sets of hyperparameters and selected based on validation F1 score.

5 Empirical Results

We evaluate our multi-task model's performance in terms of prediction metrics, computational efficiency, and comparison against existing metrics on the target labels. We also offer evidence that the tasks we combine do indeed share important similarities by presenting text embeddings and intermediate layer representations, in Figure 1 and Figure 3. We show that our model performs favorably in comparison to previously published models (Table 2), while being substantially more computationally efficient than single-task counterparts (Figure 4).

[1] Including bias terms, there are $64 \times 32 + 32$ learnable parameters in the shared layer, $32 \times 32 + 32$ between the shared layer and each of the task-type branches, $32 \times 16 + 16$ in each task-type branch, $16 \times 16 + 16$ between each task-type branch and each task-specific branch, and $16 \times 8 + 8$ in each task-specific branch.

5.1 Commonalities Across Tasks

Our model was trained on three different corpora, described in section 3, which we argue possess important semantic similarities. To provide evidence of our ten tasks existing within a common representation space, we present t-SNE projections (Van der Maaten and Hinton, 2008) of the input text embeddings corresponding to each label at three different locations within the neural network. Figure 1, discussed in section 1, shows the t-SNE projection from the output of the BERT model we use as our base encoder. Points are color-coded according to their task. If our text data carried mutually exclusive information applicable only to the particular task for which it was labeled, we would see distinct clusters of representations in Figure 1.

There is some minor evidence of clustering, particularly with respect to the propaganda and argument quality tasks, but even those tasks have observations spanning the entire representation space, and they clearly mix with other task representations. This suggests the fine-tuned BERT model is learning representations that reflect similar semantic and logical structures across tasks. We also highlight that the clustering behavior within tasks observable in the figure shows that our model's embeddings are not completely discarding task-specific structure. Rather, our model learns task-specific representations, and those representations exist within a common space with other task-specific representations, thus further lending evidence to the theory behind our approach.

This pattern is preserved throughout the layers of our model. Figure 3 presents similar t-SNE projections of two other intermediate layers: a shared layer (before any model branching occurs) and the final task-specific layer before the sigmoid activation (after the double-branching). Following the BERT model, each successive layer in the neural network gradually becomes more task-specific, and encodes information that is more relevant to distinguishing among tasks and among labels within tasks. It is notable, then, that we observe similar levels of clustering in the t-SNE projections regardless of model layer. Propaganda and argument quality tasks appear to inhabit more discernible regions of the representation space, but their clusters are neither well-defined nor tightly constrained.

We take this consistent pattern as evidence that AM tasks share a common semantic space. Enabling a model to learn these fine-grained similarities and differences between tasks and across task types is therefore likely to improve performance relative to models that rely solely on shared features or no sharing at all. We test this conjecture in the next section.

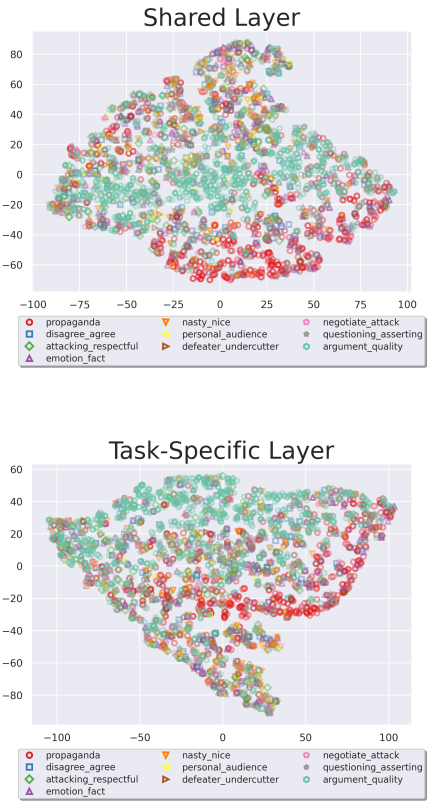

Figure 3: t-SNE projections of representations from the shared layers (top) and the task-specific layers (bottom).

5.2 Performance Evaluation

We evaluate the performance of our model primarily in terms of weighted F1 scores, which account for the class imbalances noted in Table 1. Our proposed method represents a significant departure from common approaches to argument classification, so we want to be sure our model is performing favorably relative to other models.In comparison with previous metrics (Table 2), our model shows superior performance in predicting all of the tasks for which we had previous information available. This indicates that effectively leveraging shared features improves performance across multiple tasks.

Table 3 shows a comparison of the predictive performance (as measured by the class-weighted F1-score) between baselines, single-task, and multi-task versions of the same model. The baseline metrics represent random guessing and the unigram metrics are produced by a naive Bayes classifier. As may be expected, baselines underperform all

Task	Citation	Metric	Previous	New	Absolute Gain	Relative Gain
Propaganda	Da San Martino et al. (2019)	F1	60.98	61.74	0.76	1.25
Disagree/Agree	Wang and Cardie (2014)	F1	63.57	71.38	7.81	12.29
Disagree/Agree	Abbott et al. (2011)	Acc.	68.20	70.73	2.53	3.71
Emotion/Fact	Oraby et al. (2015)	F1	46.20	63.93	17.73	38.38
Nasty/Nice	Lukin and Walker (2013)	F1	69.00	73.69	4.69	6.80

Table 2: Comparison to previously published metrics.

deep-network-based approaches.

Morio et al.'s (2022) multi-task model outperforms their single-task benchmarks about 80% of the time, and we see similar results here. Our multi-task model outperforms single-task models based on the same encoder architecture in six of our nine tasks. Ablating some layers brings that number up to seven. Again, we take this as evidence that our multi-task model is capable of exploiting the common structure between tasks in order to improve predictions. In Table 6 in the Appendix, we show that this performance gain is not merely due to adding additional trainable parameters; multi-task models of various sizes perform comparably.

We further investigated the impact of changing the base encoding model from small BERT to small ELECTRA (Clark et al., 2020) and base ALBERT (Lan et al., 2019), as well as freezing all BERT layers to prevent the pre-trained weights from being fine-tuned. In addition, we examined the effect of removing the base encoder entirely and using embeddings from two decoder-only architectures—Llama 2 (Touvron et al., 2023) and GPT-3 (Brown et al., 2020)—as input to the model. Table 4 shows a comparison of performance across these different variants of our multi-task model. All models have the architecture described in Section 4, however, the base encoder differs each time. Generally, multi-task models trained on different encoders or embeddings seem to display similar performance, indicating that the gain in performance due to the adoption of our framework is not necessarily due to the specific architecture of the encoder chosen. This is further demonstrated by the comparison of performance for each model variant across individual tasks, which is offered in Table 5 of the Appendix.

5.3 Ablation Study

We executed an ablation study to dissect the contributions of each component of our proposed multi-task architecture to its performance. To this end, we omit each of the shared, task-type, and task-specific layers to obtain the ablated neural networks. The results of this ablation analysis are detailed in Table 3, which compares the performances of the full multi-task model ('Multi-Task') against the counterparts with the shared ('Multi-Task-s'), task-type ('Multi-Task-p'), and task-specific ('Multi-Task-t') layers removed.

Removing any part of the model leads to a decline in task performance for a majority of tasks. Ablating the task-specific layers ('Multi-Task-t') causes the most extreme performance drops, with decreases of up to 39.99 F_1 points compared to the full model. This points to the significance of the task-specific branches for learning fine-grained representations. No ablated model surpasses the complete multi-task architecture on more than 2 tasks, suggesting that each element of the model structure enables gains in generalization. The results in Table 3 further show that ablation of the task-type and task-specific layers could lead to marginal improvement on two tasks at the cost of significant decline in performance on a few other tasks. This suggests that while the ablated model could better fit a minority of the tasks, different components of the full model facilitate simultaneous learning of *all* tasks toward consistently strong performance. Moreover, the model with shared layers ablated does not surpass the full model on any task, reinforcing the importance of the shared representations contained in those layers. Overall, these findings affirm the hierarchical design of our multi-task learning framework, where each layer contributes uniquely to the model's overall success.

5.4 Computational Efficiency

A key consideration, particularly when adding more trainable parameters as our model does, is whether the performance gain comes at the cost of more costly computation. We evaluate the peak GPU RAM usage and time to train our multi-task model and compare them to the same metrics from training the full set of single-task models. We conduct this evaluation by randomly sampling 5%,

Task	Baseline	Unigrams	Single-Task	Multi-Task	Multi-Task-s	Multi-Task-p	Multi-Task-t
Propaganda	55.47	38.46	**63.07**	61.74	23.21	47.35	21.75
Disagree/Agree	47.29	7.49	71.15	**71.38**	44.88	52.68	65.17
Emotion/Fact	45.80	21.91	**68.11**	63.93	59.78	62.10	64.00
Attacking/Respectful	56.47	51.16	67.46	**68.07**	55.42	56.20	53.37
Nasty/Nice	59.35	61.03	66.90	**73.69**	55.54	53.16	50.01
Personal/Audience	39.90	9.23	63.25	**65.69**	61.13	58.54	58.67
Defeater/Undercutter	53.4	45.21	45.97	55.65	50.23	**56.13**	41.68
Negotiate/Attack	36.93	55.31	64.76	**64.81**	62.13	61.33	61.68
Questioning/Asserting	50.57	57.47	59.61	**63.23**	55.36	62.71	60.75
Argument Quality	76.54	0.76	80.93	79.17	75.91	79.52	**84.14**

Table 3: Weighted F1 scores comparing baselines, single-task, and multi-task models, as well as multi-task model with ablated layers. Baseline metrics are produced by random guessing and unigram metrics by a naive Bayes classifier. The single-task and multi-task models fine-tune a small-BERT encoder as their embedding layer. "Multi-Task-s/p/t" refer to the multi-task model without the shared/task-type/task-specific layers.

Model	Prec.	Rec.	F1
Baselines			
Baseline	62.26	52.43	52.17
Unigrams	33.65	44.55	34.80
Multi-Task Models			
BERT	**69.37**	**65.76**	**66.73**
BERT (frozen)	57.54	45.98	43.64
ELECTRA	69.19	63.98	65.16
ALBERT	58.65	63.10	58.34
Llama 2	64.55	55.57	56.72
GPT 3	64.56	62.13	60.23

Table 4: Class-weighted metrics, averaged across tasks, for various base encoders and embedding models.

10%, 20%, and 40% of the training data to assess how computational load increases with data size. All models for this analysis were trained on one NVIDIA A100 GPU for one epoch. Figure 4 displays the results.

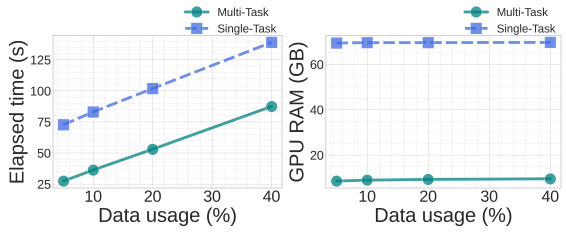

Figure 4: Computational efficiency of the multi-task model (green) compared against the single-task model (blue) in terms of elapsed training time (left) and peak GPU RAM usage (right) as the data usage increases. Both models were run on one NVIDIA A100 GPU for one epoch.

Our multi-task model achieves better performance using substantially lower computational resources overall, proving the branched task-specific modules in our model architecture to be an effective, yet practical, strategy for learning fine-grained features for label prediction. Comparing our model's performance with single-task classification on individual tasks (Table 3), we observe that it achieves comparable performance while decreasing the computation time by at least 31%.

Put together, these observations indicate that this multi-task learning approach simultaneously has a performance and computational efficiency advantage over single-task models. Computational efficiency plots for different multi-task model sizes are included in the Appendix for comparison.

6 Conclusion

Natural language tasks share substantial semantic and structural similarities, and deep learning models have been shown to be able to take advantage of these similarities in order to achieve better performance (Radford et al., 2019). In this paper, we further extend this result to the field of argument mining. We show that AM tasks do indeed share a substantial amount of features, and that these shared features can be used to boost model performance across previously unrelated tasks. We combine three data sources and propose models that outperform existing models on several of these tasks. Our models are also more computationally efficient and have better overall predictive accuracy than single-task models with comparable architectures. Aside from the practical usefulness of our models, our results are important for argument mining as a field, as they suggest that further research and model building should focus on exploiting commonalities between different tasks to boost performance.

In future work, we propose to extend our analysis to several other AM tasks that share commonalities with those studied here (e.g. Jin et al., 2022), as well as other language tasks such as topic modeling. We also propose devising improved model architectures for our multi-task setting. In particular, we propose to take advantage of frameworks such as contrastive learning (e.g. Chen et al., 2020) to encode known similarities between tasks within the representations learned by the model.

7 Limitations

As with all proposed models, ours carries important limitations. Although we show in the Appendix that the choice of base encoder does not have a drastic effect on performance, we suspect that the performance of our models is largely dependent on the ability to fine-tune a base encoder. Indeed, baseline models using unigram features performed quite poorly. Fine-tuning large base encoders—not to mention training one from scratch—can be computationally expensive. However, transfer learning may be able to help. Common semantic and logical structures across tasks point to opportunities for using transfer learning or pre-trained models from warm start to re-train on new tasks.

Multi-task models also depend on data quality and sufficient semantic overlap across tasks. This is especially challenging in AM, as argument annotation is often highly subjective (e.g. Walker et al., 2012), which can lead to noisy training data. Combining one low-quality dataset with other higher-quality ones may have a detrimental effect on model performance, as the model is unable to learn a shared representation space from noisy annotations, thus degrading performance on all tasks.

References

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn A. Walker. 2016. Internet Argument Corpus 2.0: An SQL Schema for Dialogic Social Media and the Corpora to Go With It. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation*, pages 4445–4452, Portorož, Slovenia.

Rob Abbott, Marilyn Walker, Pranav Anand, Jean E. Fox Tree, Robeson Bowmani, and Joseph King. 2011. How Can You Say Such Things?!? Recognizing Disagreement in Informal Political Argument. In *Proceedings of the Workshop on Language in Social Media*, pages 2–11, Portland, OR. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners.

Elena Cabrio and Serena Villata. 2018. Five Years of Argument Mining: A Data-Driven Analysis. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 5427–5433, Stockholm. International Joint Conferences on Artificial Intelligence Organization.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28:41–75.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Liying Cheng, Lidong Bing, Qian Yu, Wei Lu, and Luo Si. 2020. Ape: Argument pair extraction from peer review and rebuttal via multi-task learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7000–7011.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537.

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

Giovanni Da San Martino, Seunghak Yu, Alberto Barrón-Cedeno, Rostislav Petrov, and Preslav Nakov. 2019. Fine-grained analysis of propaganda in news article. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 5636–5646.

Jifeng Dai, Kaiming He, and Jian Sun. 2016. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3150–3158.

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

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, MA.

Boris Galitsky, Dmitry Ilvovsky, and Dina Pisarevskaya. 2018. Argumentation in Text: Discourse Structure Matters. In *19th International Conference on Computational Linguistics and Intelligent Text Processing*, Hanoi.

Shai Gretz, Roni Friedman, Edo Cohen-Karlik, Assaf Toledo, Dan Lahav, Ranit Aharonov, and Noam Slonim. 2020. A Large-Scale Dataset for Argument Quality Ranking: Construction and Analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(5):7805–7813.

Ivan Habernal and Iryna Gurevych. 2017. Argumentation Mining in User-Generated Web Discourse. *Computational Linguistics*, 43(1):125–179.

Kishaloy Halder, Alan Akbik, Josip Krapac, and Roland Vollgraf. 2020. Task-aware representation of sentences for generic text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3202–3213.

Mareike Hartmann, Tallulah Jansen, Isabelle Augenstein, and Anders Søgaard. 2019. Issue Framing in Online Discussion Fora. In *2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, arXiv:1904.03969, Minneapolis, MN. Association for Computational Linguistics.

Zhijing Jin, Abhinav Lalwani, Tejas Vaidhya, Xiaoyu Shen, Yiwen Ding, Zhiheng Lyu, Mrinmaya Sachan, Rada Mihalcea, and Bernhard Schölkopf. 2022. Logical fallacy detection. *arXiv preprint arXiv:2202.13758*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

John Lawrence, Jacky Visser, and Chris Reed. 2017. Harnessing rhetorical figures for argument mining. *Argument & Computation*, 8(3):289–310.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

Shikun Liu, Edward Johns, and Andrew J Davison. 2019. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880.

Xiaodong Liu, Jianfeng Gao, Xiaodong He, Li Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Stephanie Lukin and Marilyn Walker. 2013. Really? Well. Apparently Bootstrapping Improves the Performance of Sarcasm and Nastiness Classifiers for Online Dialogue. In *Proceedings of the Workshop on Language in Social Media*, pages 3–40, Atlanta. Association for Computational Linguistics.

Stephanie M. Lukin, Pranav Anand, Marilyn Walker, and Steve Whittaker. 2017. Argument Strength is in the Eye of the Beholder: Audience Effects in Persuasion.

Amita Misra, Brian Ecker, and Marilyn A. Walker. 2016. Measuring the Similarity of Sentential Arguments in Dialog. In *Proceedings of the SIGDIAL 2016 Conference*, pages 276–287, Los Angeles. Association for Computational Linguistics.

Amita Misra and Marilyn Walker. 2013. Topic Independent Identification of Agreement and Disagreement in Social Media Dialogue. In *Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 41–50, Metz, France. Association for Computational Linguistics.

Gaku Morio, Hiroaki Ozaki, Terufumi Morishita, and Kohsuke Yanai. 2022. End-to-End Argument Mining with Cross-Corpora Multi-Task Learning. 10:639–658.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. 2016. Creating and Characterizing a Diverse Corpus of Sarcasm in Dialogue. In *Proceedings of the SIGDIAL 2016 Conference*, arXiv:1709.05404, Los Angeles. Association for Computational Linguistics.

Shereen Oraby, Lena Reed, Ryan Compton, Ellen Riloff, Marilyn Walker, and Steve Whittaker. 2015. And That's A Fact: Distinguishing Factual and Emotional Argumentation in Online Dialogue. In *Proceedings of the 2nd Workshop on Argumentation Mining*, Denver.

Shereen Oraby, Lena Reed, Ryan Compton, Ellen Riloff, Marilyn Walker, and Steve Whittaker. 2017. And that's a fact: Distinguishing factual and emotional argumentation in online dialogue. *arXiv preprint arXiv:1709.05295*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen. Association for Computational Linguistics.

Sara Rosenthal and Kathy McKeown. 2015. I Couldn't Agree More: The Role of Conversational Structure

in Agreement and Disagreement Detection in Online Discussions. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 168–177, Prague. Association for Computational Linguistics.

Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. Stance detection benchmark: How robust is your stance detection? *KI-Künstliche Intelligenz*, pages 1–13.

Claudia Schulz, Steffen Eger, Johannes Daxenberger, Tobias Kahse, and Iryna Gurevych. 2018. Multi-task learning for argumentation mining in low-resource settings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 35–41.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2020. " nice try, kiddo": Investigating ad hominems in dialogue responses. *arXiv preprint arXiv:2010.12820*.

Connor Shorten, Taghi M. Khoshgoftaar, and Borko Furht. 2021. Text Data Augmentation for Deep Learning. *Journal of Big Data*, 8(1):101.

Christian Stab and Iryna Gurevych. 2014. Identifying Argumentative Discourse Structures in Persuasive Essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 46–56, Doha. Association for Computational Linguistics.

Christian Stab and Iryna Gurevych. 2017. Recognizing insufficiently supported arguments in argumentative essays. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 980–990.

Reid Swanson, Brian Ecker, and Marilyn Walker. 2015. Argument Mining: Extracting Arguments from Online Dialogue. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 217–226, Prague. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Henning Wachsmuth, Nona Naderi, Ivan Habernal, Yufang Hou, Graeme Hirst, Iryna Gurevych, and Benno Stein. 2017a. Argumentation quality assessment: Theory vs. practice. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 250–255.

Henning Wachsmuth, Nona Naderi, Yufang Hou, Yonatan Bilu, Vinodkumar Prabhakaran, Tim Alberdingk Thijm, Graeme Hirst, and Benno Stein. 2017b. Computational argumentation quality assessment in natural language. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 176–187.

Marilyn A. Walker, Pranav Anand, Jean E. Fox Tree, Rob Abbott, and Joseph King. 2012. A Corpus for Research on Deliberation and Debate. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation*, pages 812–817, Istanbul.

Vern R. Walker, Krishnan Pillaipakkamnatt, Alexandra M. Davidson, Marysa Linares, and Domenick J. Pesce. 2019. Automatic Classification of Rhetorical Roles for Sentences: Comparing Rule-Based Scripts with Machine Learning. In *Proceedings of the Third Workshop on Automated Semantic Analysis of Information in Legal Text*, Montréal.

Lu Wang and Claire Cardie. 2014. Improving Agreement and Disagreement Identification in Online Discussions with A Socially-Tuned Sentiment Lexicon. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 97–106, Baltimore. Association for Computational Linguistics.

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. 2014. Facial landmark detection by deep multi-task learning. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 94–108. Springer.

Xiangyun Zhao, Haoxiang Li, Xiaohui Shen, Xiaodan Liang, and Ying Wu. 2018. A modulation module for multi-task learning with applications in image retrieval. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 401–416.

A Additional Results

In this appendix, we compare the performance of our multi-task model with alternative designs and configurations for multi-task learning, in terms of model architecture, network size, and the base encoder.

A.1 Model Architecture

Table 5 compares the performance of our multi-task model—which incorporates branched task-type and task-specific modules—with a standard "shared-trunk" alternative, which consists of only a small BERT encoder followed by a sigmoid activation layer. This comparison shows the utility of our model architecture. Our multi-task model outperforms the shared-trunk model on all but two tasks, where the F1 metric is within 1 percentage point of that of the shared-trunk model. This performance gain comes at a negligible memory cost and a small increase in computation time (Figure 5).

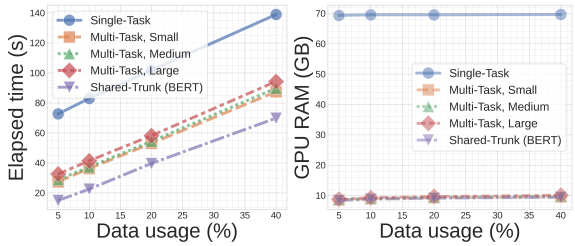

Figure 5: Computational efficiency for the single-task model as well as multi-task models with three different sizes of layers following the small BERT embedding. The small model contains 17024, medium 272384, and large 438784 trainable parameters in addition to the base encoder.

A.2 Network Size

We also compare the performance of the small multi-task model we presented in the main text with alternative networks that preserve the same architectural design but increase the sizes of the layers, from 17024 to 272384 and 438784 trainable parameters, following the base encoder. This comparison shows that the superiority in performance, due to the task-type and task-specific modules, is consistent across various network sizes and is not simply due to adding more trainable parameters on top of the shared trunk (Table 5). Moreover, Figure 5 further confirms that the layers following the BERT encoder are responsible only for a negligible increase in usage of computational resources, as multiplying the combined size of those layers by 16 (Multi-Task, Medium) and 32 (Multi-Task, Large) does not result in a substantial increase in elapsed time for training or peak GPU memory usage.

A.3 Alternative Embedding Models

In addition to comparing our model with other multi-task models, we also compare it to other base encoders. In particular, we deploy base ALBERT (Lan et al., 2019) and small ELECTRA (Clark et al., 2020), replacing the small BERT encoder with each of these other base encoders in our multi-task model. Although small BERT achieves the best average performance across different tasks, as the results in Table 6 suggest, using ELECTRA yields an average F1 score within 2 percentage points of that of small BERT, while ALBERT shows more variability across tasks with a lower average F1 score.

Task	Shared Trunk (BERT)	Multi-Task (17,024)	Multi-Task (272,384)	Multi-Task (438,784)
Propaganda	45.16	61.74	62.62	61.64
Disagree/Agree	37.96	71.38	62.07	66.74
Emotion/Fact	55.00	63.93	64.46	66.61
Attacking/Respectful	52.52	68.07	68.37	68.83
Nasty/Nice	55.62	73.69	73.04	73.38
Personal/Audience	66.51	65.69	70.17	65.24
Defeater/Undercutter	54.50	55.65	51.61	54.14
Negotiate/Attack	58.71	64.81	63.78	64.72
Questioning/Asserting	61.69	63.23	60.12	60.55
Argument Quality	79.34	79.17	68.36	81.28
Average	56.70	66.73	64.46	66.33

Table 5: Weighted F1 scores across shared layer sizes (with small BERT as base encoder). Number of trainable parameters in parentheses, not including base encoder.

Task	BERT	BERT (frozen)	ELECTRA	ALBERT	Llama 2	GPT 3
Propaganda	61.74	49.32	62.8	53.3	50.5	51.7
Disagree/Agree	71.38	62.1	59.4	69.2	65.8	68.9
Emotion/Fact	63.93	64.54	65.4	21.9	63.7	66.6
Attacking/Respectful	68.07	53.58	67.4	58.6	57.9	63.7
Nasty/Nice	73.69	52.28	71.5	61.1	59.4	66.4
Personal/Audience	65.69	10.36	68.5	64.1	59.4	65.8
Defeater/Undercutter	55.65	38.13	53.1	49.8	31.9	44.3
Negotiate/Attack	64.81	62.77	63.8	56.7	59.0	66.0
Questioning/Asserting	63.23	42.52	58.5	58.7	57.1	57.5
Argument Quality	79.17	0.76	81.2	90.0	62.5	51.4
Average	66.73	43.64	65.16	58.34	56.72	60.23

Table 6: Weighted multi-task F1 scores across base encoders and embedding models.