NTCIR 17 Conference: Proceedings of the 17th NTCIR Conference on Evaluation of Information Access Technologies, December 12-15, 2023, Tokyo, Japan
DOI: https://doi.org/10.20736/0002001314

MONETECH at the NTCIR-17 FinArg-1 Task: Layer Freezing, Data Augmentation, and Data Filtering for Argument Unit Identification

Supawich Jiarakul
Tokyo Institute of Technology
Japan
jiarakul.s.aa@m.titech.ac.jp

Hiroaki Yamada
Tokyo Institute of Technology
Japan
yamada@c.titech.ac.jp

Takenobu Tokunaga
Tokyo Institute of Technology
Japan
take@c.titech.ac.jp

ABSTRACT

This paper reports MONETECH's participation in FinArg-1's Argument Unit Identification in Earnings Conference Call subtask. Our experiments are based on the BERT and FinBERT models with additional experimentation on Large Language Model-based data augmentation, data filtering, and the model's layer freezing. Our best-performing submission, which is based on data filtering and the model's layer freezing, scores 75.54% in micro F1 evaluation. Results from additional runs also show that the model's layer freezing and data filtering could further improve model performance beyond our best submission.

KEYWORDS

argument mining, argument unit identification, data augmentation, layer freezing

TEAM NAME

MONETECH

SUBTASKS

FinArg-1: Argument Unit Identification

1 INTRODUCTION

One of the major factors in the financial market is investors' actions. It is essential to know investors' opinions for analyzing and predicting financial market movements. Therefore, financial opinion mining is crucial in analyzing and reasoning financial market movements. A financial opinion consists of many components, including the mentioned financial instrument entity, market sentiment, opinion holder, publishing time, opinion's validity period, and sets of elementary argumentative units like claims and premises [6]. The financial opinion mining task involves subtasks of mining each of the aforementioned components. Each component has its unique characteristics and hence requires different techniques and studies. In this research, we focus on argument mining in financial opinions.

Argument mining systems automatically identify elementary argumentative units, i.e., claims and premises, in an argument. Understanding the argumentative units of an argument could tremendously facilitate its analysis and comprehension process as the main points and supporting/attacking evidence are pointed out, and relations between each argumentative unit are made clear. Therefore, argument mining systems can be very beneficial for analyzing a large number of opinions by transforming unstructured texts into structured argument data, which, if done manually, would be extremely costly and time-consuming [8]. For that reason, argument mining has gained a lot of attention in the past several years, especially in the field of natural language processing (NLP). However, the studies with text in the financial domain as the target are still limited.

In this study, we participate in the argument unit identification task of FinArg-1 [5]. The task is defined as a binary classification of a given argumentative sentence from an earnings conference call into either a premise or a claim. Our experiment results are summarized as follows.

- Freezing the model's layers to some extent can help reduce the model's complexity and improve their generalization, resulting in better performance in argument unit classification.
- Our data augmentation scheme introduces noises that worsen the model performance. However, adjusting the augmentation-to-original data ratio could slightly improve the performance compared to our baseline methods.
- Data filtering based on length and based on the similarity of the augmented sentences to their original counterparts can both improve the model's performance.
- Our context augmentation scheme with both simple and hierarchical BERT architectures does not yield improvements over baseline models, hinting at the limitation of LLM-based training data augmentation in this task.
- Our best formal run is based on data filtering and the model's layer freezing. With additional runs, we can get further improved results on the released test dataset with the above-mentioned methods.

The remainder of this paper is structured as follows. Section 2 describes related work, including argument mining techniques in both financial and non-financial domains. Section 3 explains the techniques and approaches used in our study. Section 4 shows and discusses the experimental results based on each method described in section 3. Finally, section 5 concludes our study and gives an outlook for potential future work.

2 RELATED WORK

This section briefly discusses the past work in argument mining for both the financial domain and other domains. As mentioned in Section 1, the studies about argument mining in the financial domain are still limited. Therefore, we first discuss the past work in general domain and non-financial domains. Rinott et al. [13] proposed a task to automatically detect evidence that supports a given claim. By classifying evidence into either a study, expert, or anecdotal premise, their proposed system based on supervised learning

proved to perform best at detecting expert premises when compared to detecting study or anecdotal premises. Stab and Gurevych [14] showed that their proposed model of globally learning component classification, relation identification, and argumentation structure together improved the performance of each subtask over baseline methods. The results also showed that the component classification subtask was easier than relation linking. Chakrabarty et al. [3] proposed a Reddit corpus of opinionated claims using the acronyms IMO/IMHO (in my (humble) opinion) as a self-labeled signal for claims. Their results showed that fine-tuning a language model with this corpus improved the performance model on claim detection.

In the financial domain, Chen et al. [4] proposed the FinNum-3 shared task consisting of investors' Chinese claim detection and managers' English claim detection subtasks. Onuma and Kadowaki [10] investigated different formats of numerical representations in manager's English claim detection. They showed that the optimal numerical representation format varies depending on the language model. Their submission based on FinBERT [2] and with Marker numerical representation format in the joint learning setting was the top performer among the subtask's submissions.

Papagari et al. [11] introduced hierarchical architectures for the long document classification tasks that extended the original BERT [7] architecture and allowed for longer text inputs. The architecture involves breaking the long document into smaller chunks of text, passing each chunk through the base models, and propagating the embedding output of each chunk through a recurrent layer or a transformer model. The proposed architectures showed improvement over the base model with truncation on a number of long-text classification tasks.

3 METHODS

3.1 Pretrained Language Models

We use BERT [7] and FinBERT [2] as the two main pre-trained language models in this task. BERT, which stands for Bidirectional Encoder Representations from Transformers, is a transformer-based language representation model. BERT's bidirectional nature enables it to take into consideration the context on both sides of each word, making it excel in various tasks when compared to once de-facto language representation methods, including word2vec [9] and GloVe [12] which are context-independent.

FinBERT is a variant of the original BERT, further pre-trained specifically for the financial domain. FinBERT's language model is a base-size BERT further pre-trained on the TRC-2 financial corpus which is a subset of Reuters' TRC2 corpus[1]. The model yields improved accuracy, cross-entropy loss, and macro F1 average results in financial sentiment analysis tasks compared to baseline models [2].

We fine-tune the pretrained language models on the provided training dataset. [1] All three of our formal runs (Table 8) are based on the original BERT, as it shows superior results in this task.

3.2 Layer Freezing

BERT and its variants contain a large number of parameters which can lead to overfitting even after a few epochs when data is scarce. Freezing some layers could help reduce the number of parameters the model needs to learn, reduce its complexity, and thus result in a better generalization of the model. It also provides the side benefits of boosting training speed and reducing required memory. This experiment is set to explore the effect of freezing the embedding layer as well as some of the 12 encoder layers on the Argument Unit Classification task. Note that in this experiment, models are frozen in a bottom-to-top manner, meaning that the freezing starts from the embedding layer and expands into some of the 12 encoder layers layer by layer.

3.3 Data Augmentation

Our experiment at an early stage shows that the pre-trained models show signs of overfitting after just a few epochs when we directly feed the training dataset to the model at fine-tuning. In addition to other techniques to reduce the model's complexity and improve generalization, we also use GPT-3.5 Turbo to rephrase all argumentative units in order to generate a more significant number of training data points for better generalization. GPT-3.5, which stands for Generative Pre-trained Transformer 3.5, is a 175-billion-parameters autoregressive large language model proposed by OpenAI. Being trained on massive cross-domain text data, it delivers strong performance on various NLP tasks and can understand and generate natural language across different domains. We focus our usage on the rephrase task in this experiment.

We use the following prompt and parameters for data augmentation.

Prompt: "Give me 6 rephrased statements of the following statement from an earnings conference call: { sentence }"

Parameters:
messages = { "role": "user", "content": { prompt } },
temperature = 0,
max_tokens = 2000,
request_timeout=20

3.4 Data Filtering

We conduct data filtering in order to reduce noise from the dataset fed to the models. The filtering is conducted based on the argumentative sentence's length and the BERTScore similarity between the original sentence and the augmented sentence when we apply data augmentation.

3.4.1 Based on Length. Our experiment at an early stage shows that the models tend to struggle to classify shorter argumentative units. In this experiment, we try to remove very short argumentative units from the training dataset in order to leave out the noise. We first tokenize argumentative units in the training dataset using tiktoken[2] tokenizer by OpenAI, sort the training dataset by argumentative unit's token length, and recreate two new training datasets by leaving out the 10% and 25% shortest argumentative units. For augmented data, we first filter the original dataset and use only augmented data from the original data points that are left after the filtering.

[1] https://trec.nist.gov/data/reuters/reuters.html

[2] https://github.com/openai/tiktoken

3.4.2 Based on Similarity. We also explore the effect of filtering based on the similarity of the augmented sentences to the original sentences using BERTScore [15]. In this experiment, we recreate two new training datasets:

(1) Original training dataset + augmented argumentative units that have above-average BERTScore (filter for only augmented sentences that are similar to original ones)
(2) Original training dataset + augmented argumentative units that have below-average BERTScore (filter for only augmented sentences that are dissimilar to original ones)

3.5 GPT-3.5 Direct Prediction

To validate the performance of our proposed methods with that of Large Language Models (LLMs), we prompt GPT-3.5 Turbo to predict whether the given argumentative sentence is a premise or a claim. We conduct both zero-shot and ten-shot promptings with the following prompts and parameters. We pick five claim examples and five premise examples from the training dataset for the ten-shot prompt to incorporate in the prompts as context.

- Zero-shot prompting
  Prompt: "Classify the following statement from an earnings conference call into either a premise or a claim: { sentence }."
  Parameters:
  messages = { "role": "user", "content": { prompt } },
  temperature = 0,
  max_tokens = 100,
  request_timeout=20

- Ten-shot prompting
  Prompt: "Classify the given statement from an earnings conference call into either a premise or a claim.
  {{ context sentence } => { label }} * 10
  { target sentence } =>"
  Parameters:
  messages = { "role": "user", "content": { prompt } },
  temperature = 0,
  max_tokens = 100,
  request_timeout=20

3.6 Context Augmentation

Since the target text in the dataset lacks context, we conduct additional experiments augmenting preceding and following text to the target sentences using GPT-3.5 Turbo with the following prompt and parameters.

Prompt: "Please add preceding sentences and following sentences to the following statement from an earnings conference call: { sentence }"
Parameters:
messages = { "role": "user", "content": { prompt } },
temperature = 0,
max_tokens = 2000,
request_timeout=20

We then attach the preceding text, target text, and the following text together with [SEP] tokens as separators. The attached text is embedded through BERT or FinBERT. We take the mean pooling of the target text's tokens as the embedding representation and as input to the linear classifier. The architecture is also shown in Figure 1.

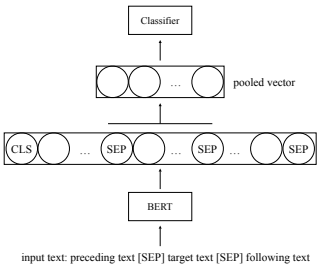

Figure 1: Model Architecture for Context-Augmented Text

3.7 Hierarchical Architectures

In addition to the architecture described in Section 3.6, we experiment with the following two hierarchical architectures on the context-augmented text. We modified the architectures proposed by Papagari et al. [11] since the input text in our experiment is already segmented into three chunks, i.e., preceding, target, and following text.

3.7.1 Pooling method. This architecture feeds each input text chunk into a base model. All embedding outputs are then pooled using either mean pooling or max pooling into one representative embedding vector, which will be fed into the classifier as input. The architecture is as shown in Figure 2.

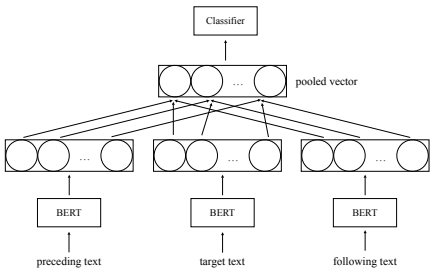

Figure 2: Pooling Method's Architecture

3.7.2 LSTM-based method. This architecture feeds each input chunk into a fine-tuned base model. The embedding output from each input chunk is then propagated through an LSTM layer, which yields a vector to be fed to the classifier. The architecture is as shown in Figure 3.

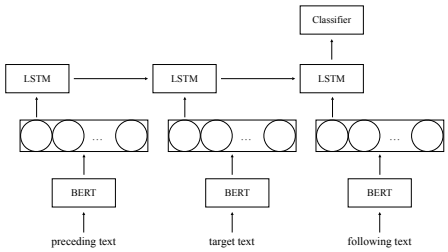

Figure 3: LSTM-based Method's Architecture

Table 1: Experiment Parameters

Parameters	Values
Models	BERT (base) or FinBERT
Batch size	8
Max length	512
Epochs	5
Optimizer	AdamW
Weight Decay	0.01
Learning Rate	2e-5
Classification Layer	Linear
Dropout	0.5

Table 2: Experimental results of baseline methods

Models	Micro-F1	Macro-F1	Weighted-F1
BERT (base)	75.03%	74.98%	75.03%
FinBERT	75.23%	74.98%	75.10%
GPT Prediction (zero-shot)	53.46%	36.14%	54.24%
GPT Prediction (ten-shot)	56.14%	34.40%	52.33%

4 EXPERIMENTS RESULTS AND DISCUSSION

This section discusses the evaluation results of the baseline methods on the task, experiments, formal runs, and additional runs. The parameters of the experiments are shown in Table 1.

4.1 Baselines

We use BERT (base) and FinBERT as our baseline models. As discussed in Section 3.5, we also try prompting GPT-3.5 Turbo to give a prediction to investigate whether our proposed methods can outperform the performance of an LLM. The F-1 score evaluation results are shown in Table 2. From the results, FinBERT slightly outperforms BERT in this task. Moreover, the performance of GPT-3.5 on this particular task is still lacking compared to BERT and FinBERT.

4.2 Layer Freezing

The F-1 score evaluation results of altering the number of frozen layers in the model as described in Section 3.2 are shown in Table 3. The results show that freezing layers, to some extent, improves the model's generalization and thus yield better evaluation scores. However, freezing too many layers could result in an oversimplified and underfitted model. The optimal number of frozen layers varies between BERT and FinBERT. In the case of BERT, freezing the embedding layers up until the eight encoder layers yield the best result in the F-1 score evaluation, whereas the optimal first unfrozen layer for FinBERT is the sixth encoder layer.

4.3 Data Augmentation

In this experiment, we use GPT-3.5 Turbo to perform data augmentation to extend the training data seven times (augmented data: original data at 6:1 ratio) with the approach described in Section 3.3. The results of fine-tuning with the augmented data are shown in Table 4. Compared to the results of using the original training dataset, the augmented data tends to introduce noise to the dataset and yield lower F-1 scores with exceptions in some configurations.

We also conduct experiments with different augmented data to original data ratios, namely 0.5, 1, 2, 3, 4, 5. The augmented data are picked randomly from the generated pool to match each ratio in each experiment. The experimental results are shown in Table 4. The 1:1 augmentation ratio yields the best results with a slightly better F-1 score than the non-augmented case for BERT. The 2:1 ratio yields the best results among other augmented cases for FinBERT, but the results are still lower than the non-augmented case.

4.4 Data Filtering

Expecting to leave out noises, we filter out some data from the dataset based on the argumentative sentence length and the similarity of the augmented sentence to the original argumentative sentence from the training dataset for the augmented dataset. The filtered dataset is then used as the training dataset to fine-tune the models.

4.4.1 Based on Length. From the observation that the models tend to struggle or yield random outputs when dealing with very short argumentative sentences, we perform data filtering based on the argumentative sentence length as described in Section 3.4.1. The experimental results are shown in Table 5. The results show that leaving out very short sentences could help reduce noise and improve the models' performance on the task. Removing the shortest 25% gives the best results for BERT. For FinBERT, the best result is achieved when removing the shortest 25% and freezing the embedding layer of the model while leaving just the shortest 10% gives better results under the no-layer-freezing setting.

4.4.2 Based on Similarity. As described in Section 3.4.2, we use BERTScore to filter for the similar augmented dataset, i.e., dataset augmented with only rephrased sentences that are similar to the original ones, as well as for dissimilar augmented dataset, i.e., dataset augmented with only rephrased sentences that are dissimilar to the original ones. The experimental results are shown in Table 6. The results show that filtering augmented datasets improves the performance of both BERT and FinBERT. For BERT, removing similar sentences yields better results than removing dissimilar sentences. On the contrary, removing dissimilar sentences yields better performance for FinBERT than removing similar sentences.

Table 3: Experimental results on altering first layers to be unfreezed

Models	First unfreeze layer	Micro-F1	Macro-F1	Weighted-F1
BERT (base)	No Freezing	75.03%	74.98%	75.03%
	Layer 1	76.06%	76.01%	76.06%
	Layer 2	76.16%	76.16%	76.16%
	Layer 4	75.13%	75.01%	74.92%
	Layer 6	75.03%	74.99%	75.04%
	Layer 8	76.57%	76.52%	76.57%
	Layer 12	69.66%	69.63%	69.68%
FinBERT	No Freezing	75.23%	74.98%	75.10%
	Layer 1	75.23%	75.02%	75.13%
	Layer 2	75.54%	75.54%	75.53%
	Layer 4	75.13%	75.00%	75.09%
	Layer 6	75.75%	75.75%	75.74%
	Layer 8	74.92%	74.90%	74.87%
	Layer 12	68.83%	68.79%	68.85%

Table 4: Experimental results on altering augmented-to-original data ratio

Models	Ratio	Micro-F1	Macro-F1	Weighted-F1
BERT (base)	No Augmentation	75.03%	74.98%	75.03%
	2:1	74.61%	74.55%	74.61%
	1:1	75.13%	75.04%	75.11%
	1:2	73.37%	73.30%	73.37%
	1:3	73.89%	73.81%	73.88%
	1:4	72.86%	72.84%	72.81%
	1:5	73.27%	73.15%	73.24%
	1:6	72.34%	72.32%	72.28%
FinBERT	No Augmentation	75.23%	74.98%	75.10%
	2:1	74.92%	74.91%	74.89%
	1:1	73.37%	73.35%	73.39%
	1:2	72.86%	72.80%	72.86%
	1:3	74.82%	74.78%	74.83%
	1:4	73.68%	73.46%	73.34%
	1:5	71.93%	71.88%	71.83%
	1:6	71.93%	71.57%	71.42%

Table 5: Experimental results on filtering dataset based on sentence length

Models	Percentage Filtered Out	Micro-F1	Macro-F1	Weighted-F1
BERT (base)	No Filtering	75.03%	74.98%	75.03%
	10%	75.23%	75.21%	75.25%
	25%	75.54%	75.50%	75.55%
FinBERT	No Filtering	71.93%	71.57%	71.42%
	10%	74.41%	74.41%	74.42%
	25%	76.16%	76.16%	76.17%

Table 6: Experimental results on filtering augmented dataset based on similarity to original sentence

Models	Filtering Condition	Micro-F1	Macro-F1	Weighted-F1
BERT (base)	No Filtering	72.34%	72.32%	72.28%
	Above-average BERTScore	72.86%	72.64%	72.76%
	Below-average BERTScore	73.99%	73.94%	74.00%
FinBERT	No Filtering	71.93%	71.57%	71.42%
	Above-average BERTScore	73.99%	73.99%	74.01%
	Below-average BERTScore	72.96%	72.86%	72.94%

Table 7: Experimental results on hierarchical model architectures

Model	Method	Micro-F1	Macro-F1	Weighted-F1
BERT(base)	Simple	73.48%	73.35%	73.44%
	Mean Pooling	73.68%	73.56%	73.65%
	Max Pooling	74.72%	74.52%	74.46%
	LSTM	72.86%	72.86%	72.87%
FinBERT	Simple	74.82%	74.73%	74.80%
	Mean Pooling	74.30%	74.30%	74.31%
	Max Pooling	73.37%	73.35%	73.31%
	LSTM	74.41%	74.13%	74.26%

Table 8: Released formal run test results

Formal Runs	Micro-F1	Macro-F1	Weighted-F1
MONETECH-1	75.13%	75.13%	75.12%
MONETECH-2	75.03%	75.02%	75.04%
MONETECH-3	75.54%	75.53%	75.56%

4.5 Context Augmentation and Hierarchical Architectures

As described in Section 3.6 and Section 3.7, we generate the preceding and following context to each target text in the dataset using GPT-3.5 Turbo. The generated text is fed to the Simple architecture as in Section 3.6 and to mean-pooling, max-pooling, and LSTM-based architectures as in Section 3.7. The experimental results are shown in Table 7. Overall, results from context augmentation with the abovementioned four architectures do not outperform those of baseline models, with the max pooling method yielding the best results for BERT and the simple method yielding the best result for FinBERT. This is in line with the results of the data augmentation experiment in that the incorporation of augmented training data generally does not improve this task.

4.6 Formal Runs

The configurations for each of the three formal runs are as follows. The three submissions are selected based on the micro F-1 evaluation of the validation dataset. The released results of the formal runs are shown in Table 8.

MONETECH-1. This submission is a BERT (base) model fine-tuned on the augmented dataset with the augmented-to-original data ratio of 1:1. The BERT model is fine-tuned with the embedding layer frozen.

MONETECH-2. This submission is a BERT (base) model fine-tuned on the training dataset that has the shortest 10% of the data removed.

MONETECH-3. This submission is a BERT (base) model fine-tuned on the training dataset that has the shortest 25% of the data removed. The BERT model is fine-tuned with the embedding layer frozen.

5 CONCLUSIONS

We described our approach to the earnings conference call's argument unit classification. Our submissions are based on the BERT model with methods including data augmentation, data filtering, freezing a part of model layers, and context augmentation with more complex architectures. Our best submission, BERT with frozen embedding layer and data filtering, ranked seventh among 21 submissions in terms of F1 score. Our additional runs show that the model's layer freezing and data filtering could further improve model performance beyond our best submission based on the evaluation of the released test dataset.

Although having surrounding context can be extremely insightful when identifying and classifying an argument unit, our experimental results still hint at the limitation of using artificial contexts generated by large language models. Given that the dataset lacks surrounding context, more concise context augmentation techniques and architectures could hold significant potential in further improving the model's performance.

REFERENCES

[1] Alaa Alhamzeh, Romain Fonck, Erwan Versmée, Elöd Egyed-Zsigmond, Harald Kosch, and Lionel Brunie. 2022. It's Time to Reason: Annotating Argumentation Structures in Financial Earnings Calls: The FinArg Dataset. In *Proceedings of the Fourth Workshop on Financial Technology and Natural Language Processing (FinNLP)*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), 163–169. https://doi.org/10.18653/v1/2022.finnlp-1.22

[2] Dogu Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. arXiv:1908.10063 [cs.CL]

[3] Tuhin Chakrabarty, Christopher Hidey, and Kathy McKeown. 2019. IMHO Fine-Tuning Improves Claim Detection. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 558–563. https://doi.org/10.18653/v1/N19-1054

[4] Chung-Chi Chen, Hen-Hsen Huang, Yu-Lieh Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2022. Overview of the NTCIR-16 FinNum-3 Task: Investor's and Manager's Fine-grained Claim Detection. In *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies* (2022). Tokyo, Japan, 87–91.

[5] Chung-Chi Chen, Chin-Yi Lin, Chr-Jr Chiu, Hen-Hsen Huang, Alaa Alhamzeh, Yu-Lieh Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2023. Overview of the NTCIR-17 FinArg-1 Task: Fine-Grained Argument Understanding in Financial Analysis. In *Proceedings of the 17th NTCIR Conference on Evaluation of Information Access Technologies* (2023). Tokyo, Japan.

[6] Hsin-Hsi Chen Chung-Chi Chen, Hen-Hsen Huang. 2021. *From Opinion Mining to Financial Argument Mining*. Springer Singapore, Singapore, Chapter 1, 1–8.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[8] John Lawrence and Chris Reed. 2019. Argument Mining: A Survey. *Computational Linguistics* 45, 4 (Dec. 2019), 765–818. https://doi.org/10.1162/coli_a_00364

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL]

[10] Shunsuke Onuma and Kazuma Kadowaki. 2022. JRIRD at the NTCIR-16 FinNum-3 Task: Investigating the Effect of Numerical Representations in Manager's Claim Detection. In *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies* (2022). Tokyo, Japan, 108–115.

[11] Raghavendra Pappagari, Piotr Żelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical Transformers for Long Document Classification. arXiv:1910.10781 [cs.CL]

[12] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[13] Ruty Rinott, Lena Dankin, Carlos Alzate Perez, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. 2015. Show Me Your Evidence - an Automatic Method for Context Dependent Evidence Detection. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 440–450. https://doi.org/10.18653/v1/D15-1050

[14] Christian Stab and Iryna Gurevych. 2017. Parsing Argumentation Structures in Persuasive Essays. *Computational Linguistics* 43, 3 (Sept. 2017), 619–659. https://doi.org/10.1162/COLI_a_00295

[15] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. arXiv:1904.09675 [cs.CL]