JRIRD at the NTCIR-16 FinNum-3 Task: Investigating the Effect of Numerical Representations in Manager's Claim Detection

Shunsuke Onuma
The Japan Research Institute, Limited
Japan
onuma.shunsuke@jri.co.jp

Kazuma Kadowaki
The Japan Research Institute, Limited
Japan
kadowaki.kazuma@jri.co.jp

ABSTRACT

This study presents JRIRD's work on the FinNum-3 Manager's Claim Detection subtask. Numeracy is essential in financial documents and some studies have focused on numerical information representations in natural language processing. For the FinNum-3 task, we tried four representations of numerical value in a text and experimented with joint learning using numerical category information. The results showed that the best format of numerical values depended on a pre-trained model. The joint learning with numerical categories improved the performance of some pre-trained models and numeral format settings.

KEYWORDS

claim detection, argument mining, category classification, joint learning, numerical representation

TEAM NAME

JRIRD

SUBTASKS

Manager's Claim Detection (English)

1 INTRODUCTION

In the financial domain, several studies have focused on financial tasks such as stock movement prediction or volatility forecasting using financial text such as financial reports [8, 13, 15, 16]. On the other hand, there have been studies on treatments of numeric values in natural language processing (NLP). Some studies have attempted exploring the abilities of language models to incorporate numerical information [10, 12].

As numerical information is essential owing to the nature of finance, Chen et al. [2] proposed a financial task focusing on numerals in a text. NTCIR-16 FinNum-3 task [3] focuses on reports written by professional stock analysts and companies' earnings call transcriptions. The main task of FinNum-3 is the claim detection task that determines whether the described numerals are in the investor's and/or manager's claims. In FinNum-3, a numerical category classification task is also an auxiliary task, which determines the financial category of the target numeric value (same as in FinNum-1 [4]). We participated in the English subtask in FinNum-3, which targets companies' earnings calls.

In our study, we focused on the format of numerical information in a text. To determine a better numeral format, we examined various numeral formats including the format ignoring the target numeral. We tried four formats of numerical values using pre-trained language models. We also verified whether joint learning of the numerical category classification along with the claim detection can improve the performance of the claim detection.

Our experiment results were summarized as follows.

- In the claim detection task, the model performance slightly improved in many cases by considering numerical information compared to ignoring it.
- The best format of numerical information was dependent on pre-trained models and settings. The formats that specially preprocess numerical information were better in the setting without joint learning. However, the best format depended on the pre-trained models in the joint learning setting.
- Joint learning with numerical category classification slightly improved the performance for the claim detection task on small language models. However, the performance of large language models worsened in some formats.

Note that, the differences in result scores were not significant. Further experiments to examine whether there is a statistically significant difference will be a future study.

The remainder of the study is as follows. Section 2 describes related works in aspects of numeracy in NLP and studies focusing on earning calls in the finance domain. Section 3 shows the task setting and our approach. Section 4 explains the implementation. Section 5 discusses the experiment results. Finally, section 6 describes our conclusion.

2 RELATED WORKS

We focused on encoding and representing numerical values in text because we believed recognizing numerical information was essential for this task. Thawani et al. [12] provided a survey of recent works related to numerical information in natural language processing (NLP). Regarding numeral notation, the survey suggested scientific notation. For example, in scientific notation, the number 80 could be written as 8e1. Also, the survey recommended char-level tokenization. Nogueira et al. [10] compared the performance of T5 model on a simple arithmetic task using various representations of numerical values. In the results, representations that allow the model to find the significance of a digit were better for large number calculations. These studies mainly focused on synthetic or numerical reasoning tasks to evaluate numeracy abilities. We focused on the input format of numerical information in the classification tasks based on the approaches in these studies.

In the financial domain, studies dealing with earnings calls to predict stock price movements and volatility have been conducted. Theil et al. [13] tackled volatility prediction of stock prices using the contents of earnings calls. Ye et al. [16] and Ma et al. [8] focused

Table 1: Formatted examples: the example text is "Fiscal Year 2018 Fourth Quarter" and the target numeral is "2018". [MASK], [NUM], and [EXP] are special tokens.

Format	Example "Fiscal Year 2018 Fourth Quarter"
Mask	Fiscal Year [MASK] Fourth Quarter
Marker	Fiscal Year [NUM] 2018 [NUM] Fourth Quarter
Digit	Fiscal Year [NUM] 2 0 1 8 [NUM] Fourth Quarter
Scientific (sig1)	Fiscal Year [NUM] 2 [EXP] 3 [NUM] Fourth Quarter
Scientific (sig4)	Fiscal Year [NUM] 2 . 0 1 8 [EXP] 3 [NUM] Fourth Quarter

on the Q&A sections in earnings calls. Yang et al. [15] tried to predict stock returns and financial risk using text and audio data. As shown in these related studies, a detailed analysis of the earning calls is helpful in financial tasks.

3 TASK AND OUR APPROACH

The FinNum-3 task is to determine whether the described numerals are in the investor's and/or manager's claims. Specifically, given a target numeric value and context text that contains it, the main task is to determine whether the target numeric value is in a claim. The FinNum-3 also provided a numerical category classification task as an auxiliary task. The auxiliary task is to determine the financial category of the target numeric value (same as in FinNum-1 [4]).

In FinNum-3, there is a Chinese subtask for stock analyst reports and an English subtask for companies' earnings calls. We participated in the English subtask. See the overview paper [3] for details of the task.

We tried four formats of numerical information in the input text and evaluated the performance of the main task in each format. In addition, we assessed the performance of joint learning with numerical category classification. We only fine-tuned the existing pre-trained language models rather than building a customized architecture for numeral information.

3.1 Numeracy Representing Formats

We tried the following four formats for representing the numerical information. First, to investigate the contribution of numerical information, we tried *Mask* format, which masked the target numerals in context text. Then, we tried three formats, *Marker*, *Digit*, and *Scientific* formats, to investigate the differences in formats. Table 1 shows example texts in each format. In the example, the context text is "Fiscal Year 2018 Fourth Quarter" and the target numeral is 2018. The explanation of these formats is as follows.

Based on the related works [10, 12], we expected that Digit and Scientific formats would facilitate recognizing the numerical features. For example, the number of a year mainly consists of four digits. In another example, the magnitude of absolute money expressions is often a million or billion scales. Although the magnitude of these numbers in these examples is informative, we assumed language models might not capture the information from subword tokens. Digit and Scientific formats may mitigate this issue.

Mask. In *Mask* format, we replaced the target number in the input text with a mask token. This setting prevented the model from considering the target numeral for claim detection and numerical category classification.

Marker. In *Marker* format, we used numerical information naturally without a special preprocess. As multiple numerals could exist in the context of the text, we put special tokens before and after the target numeral to distinguish it from others. Moreno et al. [9] used a similar preprocess step in the FinNum-2 task [5].

Digit. In *Digit* format, we split numerals into digits by using space. We aimed to recognize the numerals precisely in Digit format rather than digits in subwords. We put special tokens before and after the target numeral as in Marker. We also split other numerals in the context and inserted spaces before and after split numerals.

Scientific. In *Scientific* format, we converted numerals into scientific notation (e.g., 80 $\rightarrow$ 8e1). We expected the model to easily capture the most significant digits and the magnitude of the numerals. We put special tokens before and after the target numeral as in Marker. We also converted other numerals into a scientific notation in the text and inserted spaces before and after converted numerals.

In Scientific format, the number of significant digits of the mantissa is adjustable. We tried one and four digits for significant digits. When the number of significant digits is one, the format expresses the number in "d [EXP] e," when four, "d . d d d [EXP] e" (d is 0~9, e is the exponent value, and [EXP] is a special token). This paper refers to each setting as Scientific (sig1) and Scientific (sig4), respectively.

We split the numerals in the mantissa and exponent part into digits in Scientific form.

3.2 Task Setting

To clarify the effectiveness of joint learning with the numerical category classification task, we conducted the experiments in two settings: *Claim Detection Only setting*, which targeted only the claim detection task and *Joint Learning setting*, which combined the numerical category classification task with the claim detection task.

4 IMPLEMENTATION

4.1 Data Preprocess and Split

We found that some annotated target numerals differed from what we expected by observing the dataset. We preprocessed the dataset

Table 2: Data preprocessing in our implementation. The underline in the examples mean the annotation span.

Example	Preprocessed Example	Preprocess description
Slide 7.	Slide 7.	The annotation included a period at the end of the sentence. We excluded the periods from annotations when the target numeral ends with it.
LEAP-1	LEAP-1	The annotation included a hyphen in proper nouns. We excluded the leading hyphens from the annotations because there were more hyphens in proper nouns than those representing negative numbers. Note that this rule also affects representations of negative numbers (e.g., "-1" to "-1").
1800	1800	The annotation was on the part of a single numeral (1800). We included the numbers before or after the target numeral without spaces into the target numeral.
1 118	1118	Two annotations were on a single numeral (1118). We merged two target numerals when one follows the other without a space.
300 million	300000000	The annotation did not include a numeric scale. We multiplied the target numeral by the power of 10 according to the following numeric scale. We checked hundred, thousand, million, billion, trillion, and quadrillion for numeric scales.
1/3	1/3	Two annotations were on both sides of a slash or colon. These symbols were mainly fraction notation and time notation signals and both sides were the same type of number in such cases. We merged such two annotations.

as we described in Table 2. Our model treated the merged annotation as a single case of data, and after prediction, we attached the prediction for the original data.

Note that Scientific format converted numbers on both sides of a symbol in cases such as "1/3" or "10:30". For example, we converted "1/3" into "1 [EXP] 0 / 3 [EXP] 0" in Scientific (sig1) format.

The training dataset was divided into five consecutive folds without shuffling. We used four folds as training data, while leaving one for validation for each model. The best performing model was selected based on the validation data using grid search with the hyperparameters described later. Five different models were created depending on each validation fold[1]. The final prediction model was an ensemble model that averaged the predictions of these five models.

As a performance metric, macro-f1 was employed in the main claim detection task.

4.2 Pretrained Language Models

We used popular pre-trained language models such as BERT [6], RoBERTa [7], and T5 [11]. We used the large model for each of these[2]. Since the task is in a financial domain, we also tried FinBERT [1] tuned with texts from the financial domain. Because FinBERT is based on the base size of BERT, we also tried the base size of BERT as well for comparison.

We used HuggingFace's Transformers [14] implementation for the experiments.

4.3 Learning and Prediction

We first explain our learning and prediction using Encoder-based language models, i.e., BERT, RoBERTa, and FinBERT. We will explain later about T5 due to its different architecture.

Our input for BERT, RoBERTa, and FinBERT was the preprocessed text containing the target numeral in each format. We tokenized the input text by default tokenizer of each model.

We fed the output of the language model into a classification layer in the Claim Detection Only setting. In the Joint Learning setting, we fed the output into two classification layers for each task and averaged losses from these outputs. The ensemble model predicted the class label with the highest probability averaged over the predicted probabilities for each class from five fold models.

We fine-tuned these models using cross-entropy loss for claim detection and numerical category classification[3]. The data size for each class was imbalanced, but we did not use the weighted average loss for simplicity. The treatments for imbalanced data will be our future work.

T5 is an Encoder-Decoder model that produces text output from text input. For classification tasks, T5 typically takes a prefix text indicating a task type with the original text and we need to transform each class label into text output.

[1] The best hyperparameters of these models could be different.

[2] All models are case sensitive. We used the Whole Word Masking model for BERT-Large.

[3] We updated the parameters of language models during fine-tuning in addition to the parameters of classification layers.

Table 3: Hyperparameters searched with grid search.

Parameter	BERT, RoBERTa, FinBERT	T5
learning rate	2e-5, 3e-5, 5e-5	2e-5, 3e-5, 5e-5
num train epochs	3, 5, 10, 15, 20	3, 5, 10, 15, 20
train batch size	16, 32	16, 32 (32, 64 in the Joint Learning setting)
max seq length	512	512

Table 4: Macro-f1 results of the claim detection task in the Claim Detection Only setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	**0.884**	0.883	0.886	0.885	0.887	0.887	0.883	0.903	0.873	0.898
Marker	0.879	0.892	0.892	0.895	0.883	0.893	0.887	0.901	0.877	0.898
Digit	0.876	**0.902**	**0.892**	0.899	**0.892**	0.893	0.888	0.902	**0.885**	**0.901**
Scientific (sig1)	0.882	0.886	0.889	**0.901**	0.890	0.891	**0.895**	**0.908**	0.882	0.898
Scientific (sig4)	0.876	0.895	0.886	0.900	0.880	**0.894**	0.884	**0.908**	0.876	0.897

Table 5: Micro-f1 results of the claim detection task in the Claim Detection Only setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	**0.953**	0.965	0.955	0.965	0.955	0.965	0.953	0.970	0.949	0.967
Marker	0.950	0.968	0.957	0.966	0.954	0.967	0.955	0.969	0.950	0.967
Digit	0.950	**0.970**	**0.958**	0.969	**0.958**	**0.968**	0.956	0.969	**0.955**	**0.968**
Scientific (sig1)	**0.953**	0.966	0.956	**0.969**	0.957	0.967	**0.959**	**0.972**	0.953	0.966
Scientific (sig4)	0.951	0.969	0.955	0.969	0.952	**0.968**	0.955	**0.972**	0.950	0.966

In the Claim Detection Only setting, we attached the prefix text "claim classification :" to the original text in the input and transformed labels for the output text, i.e., in-claim and out-claim labels into "in claim" and "out claim" for the output text, respectively. In the numerical category classification, we attached the prefix text "category classification :" in the input and used the label name of each category as the output text. We tokenized the input text by default tokenizer of T5 and fine-tuned the model using cross entropy loss for each generated token.

In the Joint Learning setting, T5 is unable to predict labels for two tasks in a single inference because of different prefixes for each task. We created two instances from an original instance for each task and input these sequentially within a batch for joint learning. This process doubled the batch size compared with other models.

The ensemble model of T5 predicted a label by a majority vote of the predictions of the five fold models. When there was a tie, we chose the majority label based on the training dataset.

For each model, we searched best hyperparameters with grid search within the range shown in Table 3 [4].

We used Microsoft Azure and AI Bridging Cloud Infrastructure (ABCI)[5] provided by the National Institute of Advanced Industrial Science and Technology (AIST). We use an NVIDIA A100 GPU for training each model. The fine-tuning process at one epoch took around 5 minutes for T5 with 64 batch size.

5 RESULTS

5.1 Claim Detection Only Setting

Tables 4 and 5 show the macro-f1 and micro-f1 scores of the claim detection task in the Claim Detection Only setting, respectively.

5.2 Joint Learning Setting

Tables 6 and 7 show the claim detection task's macro-f1/micro-f1 scores in the Joint Learning setting. Tables 8 and 9 show the macro-f1/micro-f1 scores in the numerical category classification task in the Joint Learning setting.

5.3 Submit Models

We chose the submitted models from those trained in the Joint Learning setting. In each model based on BERT (large), RoBERTa,

[4]We trained models with half-precision format (e.g., FP16) except for T5. We could not train T5 stably with FP16.

[5]https://abci.ai

Table 6: Macro-f1 results of the claim detection task in the Joint Learning setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	0.881	0.895	0.884	0.899	0.888	0.893	0.885	**0.904**	0.873	0.896
Marker	0.890	0.903	**0.898**	**0.908**	**0.896**	0.910	0.887	0.904	**0.879**	0.893
Digit	**0.892**	**0.911**	0.893	0.902	0.885	0.901	0.887	0.897	0.876	0.900
Scientific (sig1)	0.881	0.900	0.891	0.897	0.892	0.899	0.876	0.901	0.871	**0.903**
Scientific (sig4)	0.890	0.904	0.889	0.903	0.889	**0.911**	**0.888**	0.895	0.871	0.901

Table 7: Micro-f1 results of the claim detection task in the Joint Learning setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	0.952	0.967	0.954	0.969	0.956	0.967	0.955	**0.971**	0.949	0.966
Marker	0.956	0.970	**0.961**	**0.971**	**0.960**	**0.973**	0.955	0.969	**0.951**	0.966
Digit	**0.957**	**0.972**	0.958	0.969	0.955	0.970	0.955	0.968	0.950	0.968
Scientific (sig1)	0.953	0.969	0.957	0.968	0.958	0.969	0.951	0.969	0.949	**0.968**
Scientific (sig4)	**0.957**	0.971	0.956	0.969	0.956	0.972	**0.956**	0.967	0.948	0.968

Table 8: Macro-f1 results of the numerical category classification task in the Joint Learning setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	0.777	0.712	0.798	0.703	0.783	**0.699**	0.800	0.736	0.777	0.715
Marker	0.783	0.727	0.797	0.729	0.778	0.691	0.799	**0.745**	**0.796**	**0.719**
Digit	0.776	0.721	0.788	0.728	0.787	0.683	**0.804**	0.740	0.772	0.713
Scientific (sig1)	0.763	0.720	**0.799**	**0.740**	**0.816**	0.688	0.794	0.728	0.766	0.715
Scientific (sig4)	**0.786**	**0.731**	0.797	0.722	0.780	0.692	0.787	0.728	0.750	0.703

Table 9: Micro-f1 results of the numerical category classification task in the Joint Learning setting. The bold score is best in the pre-trained model for each dataset.

Pre-trained model	BERT (base)		BERT (large)		FinBERT		RoBERTa		T5	
Dataset	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Mask	0.858	0.877	0.877	0.888	0.862	0.885	0.865	0.891	0.863	0.888
Marker	0.867	0.890	0.878	**0.897**	0.872	0.893	**0.872**	0.896	**0.869**	0.895
Digit	0.870	**0.893**	0.876	0.891	0.870	0.889	**0.872**	**0.898**	0.861	0.898
Scientific (sig1)	0.862	0.885	**0.880**	0.895	**0.882**	0.888	0.867	0.896	0.864	**0.900**
Scientific (sig4)	**0.872**	0.888	0.877	0.893	0.870	**0.893**	0.868	**0.898**	0.858	0.896

and FinBERT[6], we submitted the best model in the macro-f1 of the Claim detection task on the dev set. The submitted models are as follows.

JRIRD_1 BERT (large) model with Marker format in the Joint Learning setting.

JRIRD_2 RoBERTa model with Scientific (sig4) format in the Joint Learning setting.

JRIRD_3 FinBERT model with Marker format in the Joint Learning setting.

[6]We carried out the experiments of T5 in the Joint Learning setting after the deadline and thus excluded from the submission candidates.

Table 10: Hyperparameters of submitted models.

Model	Fold	train batch size	learning rate	num train epochs
JRIRD_1	1	32	3e-5	5
	2	16	3e-5	15
	3	32	2e-5	5
	4	32	5e-5	3
	5	16	2e-5	10
JRIRD_2	1	32	2e-5	5
	2	16	3e-5	10
	3	32	3e-5	5
	4	32	3e-5	15
	5	32	5e-5	20
JRIRD_3	1	32	5e-5	10
	2	32	5e-5	3
	3	32	3e-5	5
	4	32	5e-5	10
	5	32	5e-5	3

Table 10 shows the selected hyperparameters of each submitted models.

5.4 Discussion

We examined the performances of our models, including models that were not submitted, from the following perspective, mainly focusing on the macro-f1 results.

(1) The effectiveness of numerical information
(2) The best format of numerical information
(3) The effectiveness of joint learning with numerical category classification

5.4.1 Effectiveness of Numerical Information. Tables 4 and 6 show that the best formats in each pre-trained model were other than Mask format in most of our settings, except for BERT (base) on the dev set in the Claim Detection Only setting and RoBERTa on the test set in the Joint Learning setting. Therefore, the model performance might have improved slightly by considering numerical information in the task.

5.4.2 Comparing Formats of Numerical Information. The effectiveness of the input format of numerical information showed no common trend among the results in each pre-trained model. The best formats depended on the model and task setting.

In the test set result of the Claim Detection Only setting (Table 4), either Digit or Scientific format was the best for each model.

However, in the test set result of the Joint Learning setting (Table 6), Marker and Mask were best for BERT (large) and RoBERTa, respectively. Either Digit or Scientific formats were best for BERT (base), FinBERT, and T5.

In the numerical category classification results (Table 8), the best numerical format was different for each pre-trained model. Furthermore, comparing the numerical category classification results with the claim detection results, the best formats in each model differed. From the result, we may conclude that the average loss of two tasks in the Joint Learning setting is not optimal. The investigation of better settings for joint learning can be a potential future study.

Related research [12] described the effectiveness of digit and scientific notation in simple arithmetic and numerical reasoning tasks. In our experiments, the Marker format was better than Digit or Scientific in some cases. In other words, it was difficult to determine whether Digit and/or Scientific notation were more effective than Marker in the task.

5.4.3 Effectiveness of Joint Learning. To evaluate the effect of joint learning, we compared the macro-f1 scores of the dev and test sets for each model and numerical format in Table 11.

For the test set, the joint learning improved the performance of BERT (base) and FinBERT in all formats. However, the effect depended on the formats for BERT (large), RoBERTa, and T5. In detail, for BERT (large), the joint learning worsened the performance of Scientific (sig1). For RoBERTa, the results of joint learning became worse in Digit and Scientific formats. For T5, the results degraded in Mask, Marker, and Digit formats.

While the joint learning was effective regardless of the format for the base size models, it sometimes affected the performance of the large models. In particular, RoBERTa struggled with Digit and Scientific formats. We need further investigations for the large models because these models may not be optimal in the Joint Learning setting.

5.5 Future Research Direction

Based on the above discussions, our potential future research question are as follows.

First, further investigation of learning settings, especially, in the joint learning approach. The best numerical formats differed between the Claim Detection Only and the Joint Learning settings. Furthermore, joint learning failed to improve the performance of large size models in some cases. Therefore, the method for joint learning can be improved. In addition, the numbers of instances in each class are imbalanced in both the claim detection task and

Table 11: Improvement of the macro-f1 scores for the claim detection task comparing the Claim Detection Only setting and the Joint Learning setting.

Dataset		Dev			Test		
Pre-trained model	Format	Claim Detection Only	Joint Learning	Improvement	Claim Detection Only	Joint Learning	Improvement
BERT (base)	Mask	0.884	0.881	−0.002	0.883	0.895	0.011
	Marker	0.879	0.890	0.010	0.892	0.903	0.011
	Digit	0.876	0.892	0.015	0.902	0.911	0.009
	Scientific (sig1)	0.882	0.881	−0.002	0.886	0.900	0.014
	Scientific (sig4)	0.876	0.890	0.014	0.895	0.904	0.009
BERT (large)	Mask	0.886	0.884	−0.002	0.885	0.899	0.014
	Marker	0.892	0.898	0.006	0.895	0.908	0.013
	Digit	0.892	0.893	0.001	0.899	0.902	0.003
	Scientific (sig1)	0.889	0.891	0.002	0.901	0.897	−0.004
	Scientific (sig4)	0.886	0.889	0.003	0.900	0.903	0.002
FinBERT	Mask	0.887	0.888	0.001	0.887	0.893	0.006
	Marker	0.883	0.896	0.013	0.893	0.910	0.017
	Digit	0.892	0.885	−0.007	0.893	0.901	0.008
	Scientific (sig1)	0.890	0.892	0.002	0.891	0.899	0.008
	Scientific (sig4)	0.880	0.889	0.009	0.894	0.911	0.017
RoBERTa	Mask	0.883	0.885	0.002	0.903	0.904	0.001
	Marker	0.887	0.887	0.000	0.901	0.904	0.003
	Digit	0.888	0.887	−0.001	0.902	0.897	−0.005
	Scientific (sig1)	0.895	0.876	−0.018	0.908	0.901	−0.008
	Scientific (sig4)	0.884	0.888	0.004	0.908	0.895	−0.013
T5	Mask	0.873	0.873	0.000	0.898	0.896	−0.002
	Marker	0.877	0.879	0.002	0.898	0.893	−0.005
	Digit	0.885	0.876	−0.009	0.901	0.900	−0.002
	Scientific (sig1)	0.882	0.871	−0.011	0.898	0.903	0.005
	Scientific (sig4)	0.876	0.871	−0.005	0.897	0.901	0.004

the numerical category classification task. The weighted loss might help the situation.

Second, the differences in scores were not significant in our experiments. Therefore, further experiments to examine whether there is a statistically significant difference will be our future study.

The following approaches are beyond the scope of this paper but may improve the performance.

From numerical aspect, data augmentation may also be effective. Due to the nature of numerical data, unknown values can appear during testing. Data augmentation by changing numerical values in a possible range can lead to the models becoming more robust.

Though we tried to use pre-trained language models with four different numerical formats, it is possible to design a specific architecture that incorporates numerical expressions.

6 CONCLUSION

This paper reports our experiments in the Manager's Claim Detection task in FinNum-3. We focused on the formats of numerical values in a text and joint learning with numerical categories.

In our experiments, the effectiveness of the numerical information was shown by comparing the results of masking the numeral information with other settings.

Comparing the numerical formats, the formats that specially preprocessed numerical information were the best for each pre-trained model in the Claim Detection Only setting. However, this tendency was not always observed in the Joint Learning setting.

Joint learning improved the performance of the claim detection task using BERT (base) and FinBERT. However, the improvement depended on the formats for large size models.

Our experiments confirmed that numerical information is essential for the claim detection task. Furthermore, preprocessing the numerical representation in text and incorporating the numerical category information might improve the performance of the task. We need a further research on numerical formats and a better approach to joint learning.

REFERENCES

[1] Dogu Araci. 2019. FinBERT: Financial Sentiment Analysis with Pre-trained Language Models. *arXiv preprint arXiv:1908.10063* (2019).

[2] Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. 2020. NumClaim: Investor's Fine-grained Claim Detection. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1973–1976.

[3] 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. *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo Japan*.

[4] Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2019. Overview of the NTCIR-14 FinNum Task: Fine-Grained Numeral Understanding in Financial Social Media Data. In *Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies*.

[5] Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2020. Overview of the NTCIR-15 FinNum-2 Task: Numeral Attachment in Financial Tweets. *Proceedings of the 15th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo Japan* (2020).

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 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)*. 4171–4186.

[7] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* (2019).

[8] Zhiqiang Ma, Grace Bang, Chong Wang, and Xiaomo Liu. 2020. Towards Earnings Call and Stock Price Movement. *arXiv preprint arXiv:2009.01317* (2020).

[9] Jose G Moreno, Emanuela Boros, and Antoine Doucet. 2020. TLR at the NTCIR-15 FinNum-2 Task: Improving Text Classifiers for Numeral Attachment in Financial Social Data. In *Proceedings of the 15th NTCIR Conference on Evaluation of Information Access Technologies, Tokyo Japan*. 8–11.

[10] Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2021. Investigating the Limitations of Transformers with Simple Arithmetic Tasks. In *Proceedings of the 1st Mathematical Reasoning in General Artificial Intelligence Workshop at ICLR 2021*.

[11] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

[12] Avijit Thawani, Jay Pujara, Filip Ilievski, and Pedro Szekely. 2021. Representing Numbers in NLP: a Survey and a Vision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 644–656.

[13] Christoph Kilian Theil, Samuel Broscheit, and Heiner Stuckenschmidt. 2019. PRoFET: Predicting the Risk of Firms from Event Transcripts. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. 5211–5217.

[14] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online, 38–45.

[15] Linyi Yang, Jiazheng Li, Ruihai Dong, Yue Zhang, and Barry Smyth. 2022. NumHTML: Numeric-Oriented Hierarchical Transformer Model for Multi-task Financial Forecasting. *arXiv preprint arXiv:2201.01770* (2022).

[16] Zhen Ye, Yu Qin, and Wei Xu. 2020. Financial Risk Prediction with Multi-Round Q&A Attention Network. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*. 4576–4582. Special Track on AI in FinTech.