Intent Classification and Entity Extraction in financial Domain

Project-I (EE47009) report submitted to
Indian Institute of Technology Kharagpur
in partial fulfillment for the award of the degree of
Bachelor of Technology
in
Electrical Engineering

By
R.Jaisaikrishnan
(19EE10050)

Under the supervision of
Dr. Pawan Goyal
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Autumn Semester, 2022-23

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR
KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Intent Classification and Entity Extraction in Financial Domain**" submitted by **R.Jaisaikrishnan** (Roll No. 19EE10050) to Indian Institute of Technology Kharagpur towards partial fulfillment of requirements for the award of the degree of Bachelor of Technology in Electrical Engineering is a record of bona fide work carried out by him under my supervision and guidance during Autumn Semester, 2022-23.

Date: December 22, 2022
Place: Kharagpur

Dr. Pawan Goyal
Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur
Kharagpur – 721302, India

Abstract

Intent classification and entity extraction are essential techniques for analyzing and understanding financial texts in the financial domain. Intent classification involves identifying the purpose or intent behind a text, while entity extraction involves identifying and extracting specific pieces of information from a text. These techniques are crucial for automating financial document processing and analysis, as they allow financial texts to be accurately and efficiently interpreted by machine learning systems. This thesis presents a detailed study of intent classification and entity extraction techniques in the financial domain, focusing on the challenges and opportunities these techniques present. Our goal is to provide a comprehensive overview of the current state-of-the-art in this field and offer insights and guidance for researchers and practitioners interested in using these techniques for financial document analysis. We use an end-to-end framework called Multiple Novel Intent Detection (MNID) for identifying and classifying novel intents in conversational agents in the financial domain. The MNID framework combines clustering, annotation, and retraining to handle new, previously unknown intents efficiently. Our evaluation of the MNID framework on financial natural language understanding datasets (Finer-139) showed superior performance compared to competitive baselines.

Keywords: Natural Language Processing, Intent Classification, BERT, Novel Intent Detection

Acknowledgements

I would like to express my sincerest gratitude to my supervisor Dr. Pawan Goyal, for his constant support and guidance throughout this project. I was able to complete my work with his valuable feedback and suggestions. I also want to thank Mr. Ankan Mullick, Research Scholar, for his constant support and mentorship, providing continuous inputs that have aided in the successful completion of this project. Finally, I extend my thanks to the Department of Electrical Engineering for this opportunity to work on this project.

Contents

1 Introduction

Intent Classification and Entity Extraction in Financial Domain is a research topic that explores the use of natural language processing (NLP) techniques for automating the analysis of financial reports in the financial sector. The task of Intent Classification allows for the segregation of similar financial reports, which makes the analysis and understanding of the data in the reports much more accessible. However, the sheer amount of financial data in the reports makes it difficult to tag every single sentence in the documents manually. This study mainly focuses on solving this problem by using a framework to identify the various intents and tag the reports accordingly automatically. Overall, Intent Classification and Entity Extraction in Financial Domain represents a significant step forward in the development of intelligent systems for handling intent tagging and has the potential to significantly improve the efficiency and effectiveness of the classification in this domain.

One of the most used tagging systems in the financial domain is the XBRL or eXtensible Business Reporting Language method. It is a standardized method for tagging financial documents. It allows companies to electronically transmit financial data in a standardized format, which makes it easier for analysts, investors, and regulators to access and analyze the information. One of the main benefits of XBRL tagging is that it allows for more accurate and consistent data reporting. Since the tags are standardized, it eliminates the possibility of human error or inconsistency in how financial data is presented. This helps to ensure that financial statements are comparable and transparent, which is essential for investors to make informed decisions. XBRL tagging also saves time and resources by automating the process of data collection and analysis. Companies no longer have to enter financial data into spreadsheet programs manually, and analysts can quickly access the information they need without spending hours reviewing and organizing financial statements.

1.1 Motivation

Tagging financial documents with their respective intents/classes make it easy to analyze the accuracy, transparency, and efficiency of financial reporting and analysis.
Tagging financial documents with relevant tags can provide several benefits:

- Improved accuracy: The tags are standardized, which means they are used in the same way by all companies and organizations. This helps to ensure that financial data is accurately and consistently reported and reduces the risk of errors or misunderstandings.

- Enhanced transparency: The tagging allows financial data to be easily shared and compared across different companies and organizations. This can help to increase transparency and improve the reliability of financial information.

- Greater efficiency: Tagging can streamline the reporting process and review financial information, allowing data to be easily extracted and analyzed. This can save time and reduce the burden of manual data entry and review.

1.2 Objective

The process of classifying the reports based on their corresponding classes and segregating them manually often takes up a lot of resources. Hence the prime objective of this work is to design a pipeline to automate the classification part using the Multiple Novel Intent Detection (MNID) approach, which will detect the corresponding intent classes and improve the overall classification accuracy of the model. After the model training, the goal is to evaluate the Intent classification model, compare the results obtained and explore areas of improvement.

2 Methodology

In this section, we will try to understand the approach followed to create the pipeline for the Intent Classification. Our work mainly applies the Pretrained BERT [1] Model (bert-base-uncased). BERT (Bidirectional Encoder Representations from Transformers) is a pretrained language model. It is a deep learning model trained on a large text dataset and can be fine-tuned for a wide range of natural language processing tasks. This allows BERT to understand the meaning and context of words better and perform tasks such as language translation, question answering, and text classification more accurately. BERT has been widely used in various natural language processing tasks and has achieved state-of-the-art performance on many benchmarks. We also use an end-to-end framework to generate high-quality data points for intent detection known as MNID. This framework would allow us to identify new intent classes, achieve higher classification accuracy, and automatically classify the data points according to the identified intent classes.

2.1 BERT Models

BERT[1] is a transformers model pretrained on a large corpus of English data in a self-supervised fashion. This means it was pretrained on the raw texts, with no humans labeling them in any way (which is why it can use lots of publicly available data), with an automatic process to generate inputs and labels from those texts.

1. Masked language modeling (MLM): taking a sentence, the model randomly masks 15% of the words in the input then run the entire masked sentence through the model and has to predict the masked words. This differs from traditional recurrent neural networks (RNNs) that usually see the words one after the other, or from autoregressive models like GPT, which internally masks the future tokens. It allows the model to learn a bidirectional representation of the sentence.
2. Next sentence prediction (NSP): the model concatenates two masked sentences as inputs during pretraining. Sometimes they correspond to sentences next to each other in the

original text, and sometimes not. The model then has to predict whether the two sentences were following each other.

Our analysis will focus on four different BERT models, giving us a diverse perspective of the classification task and variation in the performances. They are namely:

1. BERT-BASE (uncased): The model was pre-trained on 4 cloud TPUs in Pod configuration (16 TPU chips total) for one million steps with a batch size of 256. The sequence length was limited to 128 tokens for 90% of the steps and 512 for the remaining 10%.
2. SEC-BERT: This model has the same architecture as BERT-BASE, and it is also trained on financial documents.
3. SEC-BERT-NUM: This model is similar to SEC-BERT-BASE, but we replace every number token with a [NUM] pseudo-token handling all numeric expressions uniformly, disallowing their fragmentation
4. SEC-BERT-SHAPE: This model is similar to SEC-BERT-BASE, but we replace numbers with pseudo-tokens that represent the number's shape, so numeric expressions (of known shapes) are no longer fragmented, e.g., '53.2' becomes '[XX.X],' and '40,200.5' becomes '[XX,XXX.X]'.

2.2 MNID Approach

Multiple Novel Intent Detection [2] approach uses a framework to detect new intent classes within a fixed annotation cost. The proposed algorithm uses clustering to identify different classes within a dataset and determine how points within each class are distributed. Depending on whether the points within a class are well-separated from other classes or are mixed in with points from other classes, the algorithm employs either a "silver" strategy to efficiently annotate many points or a "gold" strategy to annotate the most uncertain points with human assistance. Through rigorous experimentation, it has been shown that using this approach to create a labeled

dataset can result in better performance compared to a traditional "few-shot" setting where a fixed number of instances from each class are provided and labeled by humans.

3 Experimentation and Results

3.1 Dataset

FiNER-139 [3] is a dataset consisting of 1.1 million sentences from annual and quarterly reports of publicly-traded companies in the US, labeled with XBRL tags for entity extraction. Unlike NER or contract element extraction, which usually involves identifying a limited set of common entity types, FiNER-139 involves a much larger set of 139 entity types, many of which are numeric and require context to be correctly identified.

These 139 entities have been further divided into a broad category of a total of 30 Intents which will be used to train the models.

Labels	Intents	Main Intent
AccrualForEnvironmentalLossContingencies	Accrual, Loss	Accrual
AcquiredFiniteLivedIntangibleAssetsWeightedAverageUsefulLife	Assets, UsefulLife	Assets
AllocatedShareBasedCompensationExpense	Compensation, Expense, Shares	Expense
AmortizationOfFinancingCosts	Amortization, Costs	Amortization
AmortizationOfIntangibleAssets	Amortization, Assets	Amortization
AntidilutiveSecuritiesExcludedFromComputationOfEarningsPerShareAmount	Amount, Shares	Amount
AreaOfRealEstateProperty	Other	Other
AssetImpairmentCharges	Costs	Costs
BusinessAcquisitionEquityInterestsIssuedOrIssuableNumberOfSharesIssued	Acquisition, Shares, Equity	Shares
BusinessAcquisitionPercentageOfVotingInterestsAcquired	Acquisition	Acquisition
BusinessCombinationAcquisitionRelatedCosts	Combination, Acquisition, Costs	Costs
BusinessCombinationConsiderationTransferred1	Combination	Combination
BusinessCombinationContingentConsiderationLiability	Combination, Liability	Liability
BusinessCombinationRecognizedIdentifiableAssetsAcquiredAndLiabilitiesAssumedIntangibleAssetsOtherThanGoodwi	Combination, Assets	Assets
BusinessCombinationRecognizedIdentifiableAssetsAcquiredAndLiabilitiesAssumedIntangibles	Combination, Assets	Assets
CapitalizedContractCostAmortization	Amortization, Costs	Costs
CashAndCashEquivalentsFairValueDisclosure	Other	Other
ClassOfWarrantOrRightExercisePriceOfWarrantsOrRights1	Other	Other
CommonStockCapitalSharesReservedForFutureIssuance	Stock	Stock
CommonStockDividendsPerShareDeclared	Stock, Shares	Stock
CommonStockParOrStatedValuePerShare	Stock	Stock
CommonStockSharesAuthorized	Stock, Shares	Stock
CommonStockSharesOutstanding	Stock, Shares	Stock
ConcentrationRiskPercentage1	Other	Other
ContractWithCustomerLiability	Liability	Liability
ContractWithCustomerLiabilityRevenueRecognized	Liability	Liability
CumulativeEffectOfNewAccountingPrincipleInPeriodOfAdoption	Other	Other
DebtInstrumentBasisSpreadOnVariableRate1	Debt	Debt
DebtInstrumentCarryingAmount	Debt, Amount	Amount
DebtInstrumentConvertibleConversionPrice1	Price, Debt	Price
DebtInstrumentFaceAmount	Debt, Amount	Amount
DebtInstrumentFairValue	Debt	Debt
DebtInstrumentInterestRateEffectivePercentage	Debt, Interest	Interest
DebtInstrumentInterestRateStatedPercentage	Debt, Interest	Interest
DebtInstrumentMaturityDate	Debt	Debt
DebtInstrumentRedemptionPricePercentage	Price, Debt	Price
DebtInstrumentTerm	Debt	Debt
DebtInstrumentUnamortizedDiscount	Debt, Amortization	Debt
DebtWeightedAverageInterestRate	Debt, Interest	Interest
DeferredFinanceCostsGross	Costs, Gross	Costs
DeferredFinanceCostsNet	Costs, Net	Costs
DefinedBenefitPlanContributionsByEmployer	Other	Other
DefinedContributionPlanCostRecognized	Costs	Costs
Depreciation	Depreciation	Depreciation
DerivativeFixedInterestRate	Interest	Interest
DerivativeNotionalAmount	Amount	Amount
DisposalGroupIncludingDiscontinuedOperationConsideration	Other	Other
EffectiveIncomeTaxRateContinuingOperations	TaxRate	TaxRate

Figure 3.1: List of corresponding Intents for all the 139 entity types

3.2 Method

We initially do a preprocessing step that involves tokenizing the text using a BERT tokenizer. This will convert the text into a 768-sized tensor which will be used to train the model to classify the sentence based on the intent it belongs to. Once the tokenizing is done, we use the models mentioned above to perform the classification tasks on the dataset.

To perform the classification task, we initially train the model with the ideal scenario (i.e., where we are given data points for each of the new classes) and then compare the results against the MNID approach, where the data points for some classes will be missing.

In finer-139, the majority (91.2%) of the gold tagged spans are numeric expressions, which cannot all be included in Bert's finite vocabulary; e.g., the token '9,323.0' is split into five subword units, ['9', '##,' '##323', '##.', '##0'], while the token '12.78' is split into ['12', '##.', '##78']. Subword tokenization can lead to excessive fragmentation of numeric expressions, which can negatively impact the performance of subword-based models. This is because fragmentation increases the likelihood of generating nonsensical sequences of labels. So, we will use SEC-BERT-NUM & SEC-BERT-SHAPE to avoid such fragmentation of the sequences of labels.

For the MNID framework, we will be mainly using two major clustering algorithms: (i) K-Means [4] and (ii) Agglomerative Clustering [5]. In the first stage of the MNID framework, we will use the Novel Class Detection technique of incremental clustering to identify new classes. Then, in the second stage, we perform Cluster Quality Based Annotation, where we evaluate the quality of the obtained clusters and identify the good & bad clusters. Then, we will use gold and silver strategies to annotate the less confident points in each cluster until the budget is exhausted. Finally, we retrain the BERT models again, using the class/intent labels obtained from the MNID framework to find the classification accuracy.

The models are trained on a set of 3000 training sets and validated using 2000 sets for five epochs. The models are then tested on 2000 sets of the testing dataset.

3.3 Results

```
train(model, train_loader_set, valid_loader_set, LR, EPOCHS)

Utilizing GPU
Training Started
100%|██████████| 3000/3000 [06:30<00:00,  7.68it/s]
Epochs: 1 | Train Loss:  1.429 |
          Train Accuracy:  0.721 | Val Loss:  1.096 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:28<00:00,  7.72it/s]
Epochs: 2 | Train Loss:  0.962 |
          Train Accuracy:  0.805 | Val Loss:  1.036 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:28<00:00,  7.72it/s]
Epochs: 3 | Train Loss:  0.885 |
          Train Accuracy:  0.806 | Val Loss:  0.987 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:28<00:00,  7.72it/s]
Epochs: 4 | Train Loss:  0.822 |
          Train Accuracy:  0.810 | Val Loss:  1.022 |
          Val Accuracy:  0.774
100%|██████████| 3000/3000 [06:28<00:00,  7.72it/s]
Epochs: 5 | Train Loss:  0.770 |
          Train Accuracy:  0.815 | Val Loss:  1.000 |
          Val Accuracy:  0.789
Training Finished
```

Figure 3.2: BERT Base model training after 5 Epochs

```
train_loader_set = Dataset(train_sample, sec_base_tokenizer)
valid_loader_set = Dataset(valid_sample, sec_base_tokenizer)
train(sec_base_model, train_loader_set, valid_loader_set, LR, EPOCHS)

Utilizing GPU
Training Started
100%|██████████| 3000/3000 [06:28<00:00,  7.73it/s]
Epochs: 1 | Train Loss:  1.262 |
          Train Accuracy:  0.771 | Val Loss:  1.020 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:21<00:00,  7.86it/s]
Epochs: 2 | Train Loss:  0.887 |
          Train Accuracy:  0.804 | Val Loss:  0.941 |
          Val Accuracy:  0.790
100%|██████████| 3000/3000 [06:21<00:00,  7.87it/s]
Epochs: 3 | Train Loss:  0.791 |
          Train Accuracy:  0.813 | Val Loss:  0.912 |
          Val Accuracy:  0.793
100%|██████████| 3000/3000 [06:21<00:00,  7.87it/s]
Epochs: 4 | Train Loss:  0.722 |
          Train Accuracy:  0.823 | Val Loss:  0.902 |
          Val Accuracy:  0.798
100%|██████████| 3000/3000 [06:21<00:00,  7.87it/s]
Epochs: 5 | Train Loss:  0.654 |
          Train Accuracy:  0.840 | Val Loss:  0.886 |
          Val Accuracy:  0.800
Training Finished
```

Figure 3.3: SEC-BERT model training after 5 Epochs

```
train_loader_set = Dataset(train_sample, sec_num_tokenizer)
valid_loader_set = Dataset(valid_sample, sec_num_tokenizer)
train(sec_num_model, train_loader_set, valid_loader_set, LR, EPOCHS)
```

```
Utilizing GPU
Training Started
100%|██████████| 3000/3000 [06:31<00:00,  7.67it/s]
Epochs: 1 | Train Loss:  1.255 |
          Train Accuracy:  0.771 | Val Loss:  1.029 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:27<00:00,  7.74it/s]
Epochs: 2 | Train Loss:  0.900 |
          Train Accuracy:  0.807 | Val Loss:  0.958 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:27<00:00,  7.75it/s]
Epochs: 3 | Train Loss:  0.806 |
          Train Accuracy:  0.813 | Val Loss:  0.918 |
          Val Accuracy:  0.794
100%|██████████| 3000/3000 [06:27<00:00,  7.75it/s]
Epochs: 4 | Train Loss:  0.727 |
          Train Accuracy:  0.829 | Val Loss:  0.887 |
          Val Accuracy:  0.803
100%|██████████| 3000/3000 [06:27<00:00,  7.74it/s]
Epochs: 5 | Train Loss:  0.637 |
          Train Accuracy:  0.852 | Val Loss:  0.874 |
          Val Accuracy:  0.811
Training Finished
```

Figure 3.4: SEC NUM BERT model training after 5 Epochs

```
train_loader_set = Dataset(train_sample, sec_shape_tokenizer)
valid_loader_set = Dataset(valid_sample, sec_shape_tokenizer)
train(sec_shape_model, train_loader_set, valid_loader_set, LR, EPOCHS)
```

```
Utilizing GPU
Training Started
100%|██████████| 3000/3000 [06:27<00:00,  7.75it/s]
Epochs: 1 | Train Loss:  1.342 |
          Train Accuracy:  0.768 | Val Loss:  1.062 |
          Val Accuracy:  0.789
100%|██████████| 3000/3000 [06:27<00:00,  7.74it/s]
Epochs: 2 | Train Loss:  0.895 |
          Train Accuracy:  0.807 | Val Loss:  0.957 |
          Val Accuracy:  0.790
100%|██████████| 3000/3000 [06:27<00:00,  7.74it/s]
Epochs: 3 | Train Loss:  0.784 |
          Train Accuracy:  0.823 | Val Loss:  0.874 |
          Val Accuracy:  0.809
100%|██████████| 3000/3000 [06:27<00:00,  7.75it/s]
Epochs: 4 | Train Loss:  0.680 |
          Train Accuracy:  0.841 | Val Loss:  0.824 |
          Val Accuracy:  0.820
100%|██████████| 3000/3000 [06:27<00:00,  7.75it/s]
Epochs: 5 | Train Loss:  0.597 |
          Train Accuracy:  0.855 | Val Loss:  0.807 |
          Val Accuracy:  0.812
Training Finished
```

Figure 3.5: SEC SHAPE BERT model training after 5 Epochs

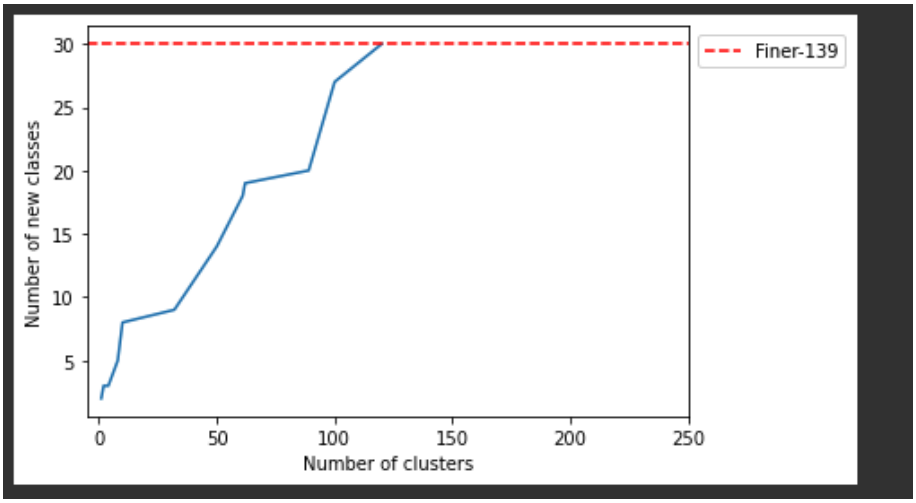

Figure 3.6: Class Discovery with the number of clusters

The MNID framework successfully identifies almost all the new intents present in the datasets. For the Finer-139 dataset, it was able to identify 29 out of the 30 intents (96.77%). Table 3 shows the accuracy and F1 scores of different models for the training and testing dataset. Figure 3.6 depicts the variation in the number of new classes identified with the number of clusters. The horizontal line represents the total number of new intents present. The plot shows an increasing trend in the number of classes with an increasing number of clusters.

Models	Accuracy(%) [Train] (Ideal, MNID)	Accuracy (%) [Test] (Ideal, MNID)	F1 [Train] (Ideal, MNID)	F1 [Test] (Ideal, MNID)
BERT Base	(81.5, **83.2**)	(83.3, **83.8**)	(**77.4**, 77.2)	(74.5, **75.3**)
SEC-BERT	(84.0, **84.2**)	(**84.8**, 84.3)	(78.5, **79.8**)	(75.6, **77.6**)
SEC-BERT-NUM	(85.2, **87.5**)	(85.0, **88.2**)	(82.2, **84.5**)	(80.1, **82.3**)
SEC-BERT-SHAPE	(85.5, **89.4**)	(85.7, **90.1**)	(83.7, **85.2**)	(81.3, **83.3**)

Table 3: Overall accuracy and F1 scores in (%) across the dataset under ideal and MNID approaches.

4 Discussions

4.1 Conclusion

We can conclude from the above results that the MNID approach outperforms the baseline models. This is because of the high-quality silver annotated points obtained from the MNID framework. It gives an advantage to the models trained with the MNID approach, thus improving the performance overall. Intent classification and entity extraction are essential tasks in the financial domain, as they allow us to accurately understand the context related to financial information and services. A novel framework called Multiple Novel Intent Detection (MNID) was used for identifying and classifying novel intents in conversational agents. The MNID framework used a combination of clustering, annotation, and retraining to efficiently and effectively handle new, previously unknown intents. The used framework was evaluated on a natural language understanding (NLU) dataset (Finer-139) and, demonstrated superior performance compared to competitive baselines (i.e., ideal scenario where we are given data points for each new class). Thus, to conclude, the MNID framework provides a valuable solution for handling novel intents in the financial domain and has the potential to improve the effectiveness of in this domain.

4.2 Future Works

In our current approach, we are considering only one intent (Main Intent) for each corresponding entity type in the dataset. However, a particular entity can belong to multiple Intent classes. Future work could consider multiple intents per entity and then predict all possible intents to which an entity could belong.

Bibliography

[1] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

[2] A Framework to Generate High-Quality Datapoints for Multiple Novel Intent Detection

[3] FiNER: Financial Numeric Entity Recognition for XBRL Tagging

[4] MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In L. M. Le Cam & J. Neyman (Eds.), Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, pp. 281–297). California: University of California Press.

[5] Modern hierarchical, agglomerative clustering algorithms

[6] Md Shad Akhtar, Abhishek Kumar, Deepanway Ghosal, Asif Ekbal, and Pushpak Bhattacharyya. 2017. A multilayer perceptron-based ensemble technique for fine-grained financial sentiment analysis. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 540–546, Copenhagen, Denmark.

[7] Emily Alsentzer, John Murphy, William Boag, WeiHung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In Proceedings of the 2nd Clinical Natural Language Processing Workshop, pages 72–78, Minneapolis, USA.

[8] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. CoRR, abs/1508.01991.

[9] Dan Hendrycks and Kevin Gimpel. 2018. A baseline for detecting misclassified and out-of-distribution examples in neural networks.

[10] Bo Yang, Xiao Fu, Nicholas D Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In international conference on machine learning, pages 3861–3870. PMLR.

[11] Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. 2019. Out-of domain detection for low-resource text classification tasks. arXiv preprint arXiv:1909.05357.

[12] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. 2019. Multi-class classification without multi-class labels. arXiv preprint arXiv:1901.00544.

[13] Xin Mu, Feida Zhu, Juan Du, Ee-Peng Lim, and ZhiHua Zhou. 2017b. Streaming classification with emerging new class by class matrix sketching. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 31.
[14] Min Wang, Ke Fu, Fan Min, and Xiuyi Jia. 2020. Active learning through label error statistical methods. Knowledge-Based Systems, 189:105140.