University of Zurich[UZH]

**Department of Computational Linguistics**

**Student Project Report**

# Profiling Credit Repair Customer Complaints using Few Shot Learning

Computational Forensic Linguistics

Author: **Xiaojing Zhang**

Author: **Xiaotong Yang**

Submission Date: 21.01.2024

# 1 Introduction

The project aims to classify consumers based on their level of related knowledge by analyzing their narratives published in an up-to-date online database. This analysis is conducted in the context of credit reporting and credit repair scams.

## 1.1 Credit reporting and credit repairing

Credit reporting is a detailed record of an individual's credit history and financial behaviour. Late payments, defaults and bankruptcies can lower a customer's credit score. To overcome the difficulties of obtaining mortgages or other types of credit with a low credit score, some organisations have developed a scam that claims to repair customers' credit by promising quick fixes, charging upfront fees or using deceptive tactics. With the rapid growth of credit reporting and repair scams, it will be beneficial to investigate this issue and find patterns of existing scams.

## 1.2 Few-shot learning

In the traditional supervised machine learning method, a large amount of annotated data set is basically required for model accuracy. However, in some practical fields such as finance, it has always been difficult to collect and annotate the data due to the ever-changing nature of financial data. Few-shot learning (FSL) is a machine learning technique that aims to learn from scarce data, which was studied before the era of deep learning. The FSL technique may be a suitable solution in this context. It would be interesting to explore the potential of FSL in the rapidly changing world of finance.

# 2 Literature Review

FSL focuses on developing models that can accurately classify previously unseen classes with only a limited number of examples (Fu et al., 2022). In the context of natural language processing research, FSL shows improved performance in tasks (Wang et al., 2020). Bansal et al. (2019) address the challenge of learning to generalize to new tasks with a few examples in the context of natural language processing (NLP). The article introduces a new method called LEOPARD, which enables optimization-based meta-learning across tasks with different numbers of classes. It evaluates various methods on generalization to diverse NLP classification tasks. The study demonstrates that LEOPARD, trained with the state-of-the-art transformer architecture (BERT), learns better initial parameters for few-shot learning than self-supervised pre-training or pre-training followed by multi-task learning. The text evaluates the generalization to NLP tasks not seen during training and shows significant accuracy improvements with only a few examples per label. The shared task in PAN@CLEF 2023, 'Profiling Cryptocurrency Influencers with Few-shot Learning,' has gained extensive attention. This task aims to profile Cryptocurrency Influencers in social media from a low-resource perspective (Bevendorff et al., 2023). In their study, Girish et al. (2023) compared two feature extraction models, FSL-Word2Vec and FSL-ST, in terms of pre-processing and text representation. The vector representations from both models were then used as input for training and testing in LinearSVC to make predictions. The prediction accuracy was evaluated using the macro F1 score. The results indicate that neither model performed well in

subtasks 1 and 2, with FSL-Word2Vec performing slightly better than FSL-ST. The authors suggest that the small training sample size may be the reason for this. Both models achieve better F1 scores in subtask3. Overall, there are no significant differences in the performance of the two models. Villa-Cueva et al. (2023) propose an alternative framework for this PAN challenge. The authors of the text utilise pre-trained language models, including BERT, RoBERTa, and DeBERTa, to facilitate transfer learning, data augmentation, and fine-tuning of Natural Language Inference (NLI) models. They achieve this by adapting two distinct frameworks: traditional fine-tuning and entailment-based fine-tuning, as well as employing data augmentation through ChatGPT. The system's performance was evaluated using a 4-fold validation approach to ensure equal class representation and robust performance estimation. The results showed that the traditional fine-tuning approach had superior performance for subtask 1, while the entailment-based approach had slightly better performance for subtasks 2 and 3.

# 3 Our experiment

## 3.1 Initial design of the project

Our experiment was initially designed after being inspired by the usefulness and suitability of the FSL method in finding patterns from consumers' narratives of their credit reporting and credit repairing scam experiences. The aim was to classify consumers by their levels of related knowledge, so that the project method and results might be useful for certain educational organizations to target consumers with lower levels of knowledge for better training and education. The FSL technique emphasises 'K-way, N-shot', where K represents the number of classes and N represents the training sample size per class (Tian et al., 2020). The project was designed to train on two datasets, one for binary classification and the other for 4-class classification, in order to investigate the accuracy for different K values.

## 3.2 Dataset description

To apply natural language processing technologies, this project utilises the dataset of consumer narratives from the Consumer Financial Protection Bureau (CFPB) who have been scammed. To reflect the core of FSL, which is to learn from scarce data, we filter the narratives to only include those that are one-year in duration and specify the narratives as 'fraud or scam' and 'credit repair services.' In total, there are 272 matches available on the CFPB database as the raw narrative inputs. The next step is to annotate the data. The data was manually annotated to improve time and cost efficiency. The binary classification dataset consists of two classes: 1 represents a higher level of relevant knowledge (50% or more knowledge contained), while 0 represents a lower level of relevant knowledge (less than 50% knowledge contained). The dataset for 4-class classification is divided into four groups: the first lower group (0) represents 25% and below, the second lower group (1) represents 25% - 50%, the first higher group (2) represents 50% - 75%, and the second higher group (3) represents 75% and above. The data annotation process began with agreeing on rough criteria, such as the mention of an article or item of law indicating a good base of related knowledge. The dataset was then split equally between two people. However, following the final presentation, it was pointed out during the annotation process that a more professional approach was required. We then collaboratively annotated the first 10 data inputs and agreed upon more specific and comprehensive criteria. Subsequently, we split the datasets in half and annotated them independently, without showing each other the annotations. Once all data

inputs were annotated, we calculated the number of each class. In FSL studies, it is crucial to ensure that each class has an equal number of inputs. After basic data cleaning, which involved deleting repetitive and irrelevant narratives, and annotation, we discovered an imbalance in the number of inputs between classes for both the binary classification and the 4-class classification datasets. To address this issue, we used ChatGPT to generate the missing inputs and satisfy the FSL method's requirement. The unified prompting method was utilised to generate more data. The prompt used was to invent a new victim of credit report or credit score repair scams, who possesses financial/legal knowledge, and to write a new complaint based on a given example in a concise and similar style and format. The ChatGPT outputs were thoroughly examined to ensure alignment with the natural narratives from the CFPB database. As a result, the binary classification dataset contains 220 data inputs, while the 4-class classification task contains 218. Until then, the dataset preparation was finished, and the annotation process was enhanced to be more professional after the final presentation.

## 3.3 Training process

In the initial phase of our few-shot learning project, we divided our datasets as follows: 70% allocated for the training set, 15% for the test set, and 15% for the validation set. For binary classification, this configuration resulted in 153 data points for training, 32 for testing, and 33 for validation. In the 4-class classification scenario, we had 154 data points for training, 33 for testing, and 33 for validation. After our final presentation, we received feedback suggesting that our test set was too small.

Few-shot learning typically focuses on training with a notably small number of examples, significantly fewer than what is usually used in machine learning models, highlighting the challenge of learning effectively from minimal data. This understanding led us to reconsider our dataset splitting approach to better align with the few-shot learning context.

Consequently, we revised our data splitting strategy. For binary classification, we designated 10% of the dataset for training, 80% for testing, and 10% for validation. For the 4-class classification task, the split was adjusted to 20% for training, 60% for testing, and 20% for validation. Under this new structure, our binary classification dataset included 22 training data points (11 per class), 174 for testing, and 22 for validation. For the 4-class classification, the dataset comprised 44 training data points (11 per class), 132 for testing, and 44 for validation.

| **Classification Type** | **Dataset Split** | **Training Data** | **Test Data** | **Validation Data** |
|---|---|---|---|---|
| Binary (Initial) | 70% / 15% /15% | 153 | 32 | 33 |
| 4-Class (Initial) | 70% / 15% /15% | 154 | 33 | 33 |
| Binary **(Revised)** | 10% / 80% /10% | 22 (11 per class) | 174 | 22 |
| 4-Class **(Revised)** | 20% / 60% /20% | 44 (11 per class) | 132 | 44 |

*Table 1 outlines the distribution of data points for training, testing, and validation in our few-shot learning project, with initial and revised splits to reflect adjustments based on feedback.*

# 4 Approaches & Experiments:

## 4.1 Feature extraction

In our approach to feature extraction for the classification task, we initially leveraged the built-in feature extraction capabilities of pre-trained models, namely, Word2Vec, BERT, and Sentence Transformers. For BERT and Sentence Transformers, we extracted the outputs of the last hidden layer corresponding to the special [CLS] token. In contrast, for the Word2Vec model, we utilized the average of word embeddings. These features provide a rich, pre-trained contextual basis for our classification model.

We complemented these pre-trained features with manually extracted ones, including average sentence length, domain terms, and TF-IDF features. Our underlying intuition was that individuals with financial or legal knowledge tend to use longer, more complex sentences and frequently employ specific jargon, such as “law”, “refund”, or particular regulations.

### 4.1.1 Average sentence length

In our classification task, the average sentence length per class emerges as a relevant factor, albeit with varying degrees of prominence across the four classes. Different types of complaints tend to manifest in narratives of varying lengths; some complaints are typically concise, while others require more extended explanations or details. This variation in sentence length can serve as a distinctive pattern, assisting the classifier in distinguishing between the different classes of complaints.

For our binary classification, the average sentence length for class 0 (i.e., people who have no financial or legal knowledge) is 22 while the average sentence length for class 1 (those who have financial or legal knowledge) is 25.

For 4-class classification, the average sentence length for class 0 is 22.08, the average sentence length for class 0 is 22.98, the average sentence length for class 2 is 25.14, and the average sentence length for class 3 is 25.39.

We can see that the average sentence length per class can indeed be a significant factor in our classification task, though not that pronounced for the 4-class task.

### 4.1. 2 Domain terms

Incorporating domain terms as features in text classification models profoundly enhances their effectiveness, particularly in context-rich domains like finance or law. We adopted slightly different strategies to select domain terms for binary and 4-class classification tasks.

**• Binary Classification**

We identified top 20 most frequent terms (domain terms) in the complaint narratives classified under label 1 (i.e., people with financial or legal knowledge):

*[('xxxx', 863), ('credit', 504), ('consumer', 170), ('information', 138), ('account', 129), ('report', 114), ('company', 104), ('services', 91), ('debt', 88), ('reporting', 85), ('card', 82), ('law', 77), ('refund', 75), ('financial', 73), ('act', 73), ('loan', 73), ('without', 68), ('also', 66), ('score', 63), ('would', 63), ('service', 61), ('money', 56), ('repair', 56), ('received', 53), ('complaint', 53), ('action', 52), ('unauthorized', 52), ('paid', 50), ('payments', 50), ('fair', 50), ('accounts', 47), ('business', 46), ('payment', 45), ('call', 45), ('right', 44), ('one', 44), ('section', 42), ('fee', 41), ('rights', 41), ('provided', 40), ('practices', 39), ('full', 38), ('email', 37), ('USC', 36), ('violation', 36), ('identity', 36), ('months', 35), ('get', 35), ('however', 35), ('experian', 35)]*

Then we filtered out the meaningless or less relevant words from the list by manually removing words that are either too common, irrelevant, or don't contribute significantly to the context of financial or legal knowledge, such as xxxx', 'without', 'also', and similar general terms. The domain terms that are finally used in our experiments are:
*['credit', 'consumer', 'information', 'account', 'report', 'company', 'services', 'debt', 'reporting', 'card', 'law', 'refund', 'financial', 'act', 'loan', 'score', 'service', 'money', 'repair', 'received', 'complaint', 'action', 'unauthorized', 'paid', 'payments', 'fair', 'accounts', 'business', 'payment', 'call', 'right', 'section', 'fee', 'rights', 'provided', 'practices', 'full', 'email', 'USC', 'violation','identity', 'months']*

**• 4-Class Classification**
As 4-class classification is more subtle, we decided to extract frequent words for each class. We first identified top 50 most frequent terms in the complaint narratives for each class:

| Class | Frequent Terms |
|---|---|
| 0 | *[('xxxx', 170), ('credit', 86), ('money', 38), ('account', 36), ('paid', 32), ('never', 23), ('help', 22), ('report', 21), ('company', 21), ('score', 19), ('get', 18), ('debt', 18), ('called', 17), ('said', 17), ('would', 17), ('month', 15), ('services', 15), ('days', 15), ('anything', 13), ('months', 13), ('asked', 13), ('refund', 13), ('repair', 13), ('nothing', 13), ('service', 13), ('information', 12), ('took', 12), ('made', 12), ('payments', 12), ('told', 12), ('back', 11), ('payment', 11), ('card', 10), ('cant', 10), ('sent', 10), ('law', 10), ('want', 9), ('amount', 9), ('could', 9), ('years', 9), ('email', 9), ('need', 9), ('charged', 9), ('still', 9), ('keep', 9), ('freedom', 9), ('relief', 9), ('time', 9), ('accounts', 8), ('signed', 8)]* |
| 1 | *[('xxxx', 413), ('credit', 156), ('company', 60), ('would', 59), ('account', 59), ('paid', 49), ('money', 48), ('report', 45), ('never', 39), ('received', 38), ('called', 32), ('get', 32), ('told', 31), ('back', 27), ('debt', 27), ('accounts', 26), ('one', 25), ('card', 25), ('even', 24), ('sent', 23), ('information', 22), ('services', 22), ('pay', 21), ('amount', 21), ('also', 21), ('still', 20), ('bank', 20), ('months', 19), ('score', 19), ('refund', 19), ('help', 18), ('give', 18), ('loan', 18), ('could', 18), ('service', 17), ('phone', 17), ('nothing', 17), ('email', 17), ('paying', 16), ('name', 16), ('want', 16), ('contacted', 15), ('like', 15), ('since', 15), ('payment', 15), ('charged', 15), ('time', 15), ('know', 15), ('take', 14), ('said', 14)]* |
| 2 | *[('xxxx', 343), ('credit', 270), ('company', 77), ('debt', 61), ('account', 60), ('loan', 56), ('report', 51), ('services', 49), ('would', 47), ('money', 45), ('score', 42), ('service', 42), ('card', 41), ('paid', 39), ('financial', 37), ('payment', 36), ('received', 36), ('consumer', 35), ('refund', 35), ('email', 34), ('repair', 34), ('business', 33), ('information', 33), ('get', 31), ('call', 29), ('payments', 28), ('complaint', 28), ('fee', 28), ('law', 27), ('months', 26), ('also', 26), ('like', 25), ('even', 25), ('month', 24), ('back', 24), ('time', 23), ('creditors', 23), ('never', 23), ('one', 23), ('full', 23), ('dispute', 22), ('without', 22), ('made', 22), ('pay', 21), ('told', 21), ('take', 20), ('contract', 20), ('promised', 20), ('accounts', 19), ('funds', 19)]* |
| 3 | *[('xxxx', 523), ('credit', 243), ('consumer', 135), ('information', 107), ('reporting', 72), ('account', 70), ('act', 67), ('report', 66), ('law', 50), ('without', 46), ('unauthorized', 44), ('services', 43), ('card', 41), ('refund', 41), ('also', 40), ('section', 39), ('financial', 37), ('fair', 37), ('USC', 35), ('rights', 35), ('company', 35), ('action', 34), ('identity', 34), ('violation', 32), ('right', 32), ('person', 30), ('agency', 30), ('theft', 29), ('debt', 29), ('complaint', 28), ('privacy', 28), ('accounts', 28), ('use', 26), ('provided', 26), ('payments', 25), ('may', 25), ('repair', 24), ('written', 23), ('score', 23), ('immediate', 23), ('states', 23), ('legal', 22), ('practices', 22), ('one', 22), ('charges', 22), ('experian', 22), ('lexington', 21), ('violated', 21), ('charge', 21), ('service', 20)]* |

Table 2 shows top 50 most frequent terms (domain terms) for each class in the 4-class classification task

We also removed meaningless and irrelevant words from the list. Given the context of each class ranging from no financial or legal knowledge (Class 0) to profound knowledge (Class 3), we selected the domain terms for each class as follows to reflect the increasing complexity and specificity of financial and legal terminology:

Class 0 (No Financial or Legal Knowledge):
We focused on basic and general terms and selected: *'credit', 'money', 'account', 'paid', 'report', 'company', 'score', 'debt', 'services', 'card', 'refund', 'payment', 'months', 'email'.*

Class 1 (Some Knowledge):
We included terms that indicate a basic understanding of financial transactions and responsibilities and selected: *'credit', 'account', 'company', 'paid', 'money', 'report', 'debt', 'accounts', 'card', 'services', 'loan', 'score', 'refund', 'payment', 'bank'.*

Class 2 (Moderate Knowledge):
We chose terms that reflect a moderate understanding of financial services, loans, and consumer rights and selected: *'credit', 'debt', 'loan', 'account', 'services', 'score', 'financial', 'payment', 'consumer', 'refund', 'business', 'information', 'fee', 'creditors', 'complaint', 'contract'.*

Class 3 (Profound Knowledge):
We select words associated with legal, reporting, and advanced financial concepts: *'credit', 'consumer', 'information', 'reporting', 'account', 'act', 'law', 'financial', 'fair', 'USC', 'rights', 'violation', 'identity', 'agency', 'theft', 'privacy', 'legal', 'experian', 'lexington'.*

Then we removed the duplicates and used the following domain terms for 4-class task:
*['score', 'paid', 'business', 'months', 'consumer', 'financial', 'complaint', 'rights', 'violation', 'services', 'theft', 'account', 'agency', 'creditors', 'debt', 'law', 'company', 'information', 'money', 'act', 'contract', 'bank', 'reporting', 'legal', 'privacy', 'accounts', 'USC', 'card', 'identity', 'payment', 'refund', 'loan', 'fee', 'lexington', 'email', 'report', 'credit', 'fair']*

### 4.1.3 TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) features are pivotal in text classification tasks because they provide a nuanced understanding of the text's content. By balancing term frequency with the uniqueness of the term across the entire document corpus, TF-IDF effectively highlights words that are particularly relevant to each document. So we decided to extract TF-IDF as feature in our classification.

In our implementation, we streamlined the process of extracting TF-IDF features using Scikit-learn's TfidfVectorizer. This tool simplifies the calculation of TF-IDF scores for words in our documents. First, it breaks down the text into individual words, a process known as tokenization, making it ready for analysis. Then, it calculates two scores for each word: the Term Frequency (TF) and the Inverse Document Frequency (IDF). The TF score is a measure of how often a word appears in a document, adjusted by the document's total word count. The IDF score reflects the rarity of a word across all documents, giving higher scores to rarer words. The TF-IDF score for each word is then determined by multiplying its TF and IDF scores. After applying this vectorizer to our training data, it converts the text of our training, validation, and test sets into TF-IDF vectors. These vectors, which represent the importance of each word in the documents, are used as input features for our machine learning model.

We combined features from pre-trained models with manually extracted features, i.e., average sentence length, domain terms and TF-IDF features. We found that adding these manually extracted features was very useful. For binary classification, the integration of these features resulted in a substantial increase in accuracy, from 78.9% to 85.2%, and a similar pattern was observed for 4-class classification, with accuracy improving from 59.8% to 63.6%. The addition of manual features not only improved the model's accuracy but also its F1 score, indicating a more balanced precision and recall, which is crucial for models used in real-world applications where the cost of false positives and false negatives varies.

In both the binary and 4-class classification tasks, adding domain terms to the model made the biggest difference, reflecting the importance of contextual and domain-relevant information in text classification. For the 4-class classification, just using domain terms raised the accuracy to 62.7%, which matches the pattern seen in binary classification. When we combined all three features — average sentence length, domain terms, and TF-IDF — the model did the best in both tasks. This shows that even though each feature helps on its own, using them all together gives the model a full set of tools to better understand and sort the text, which is especially helpful when the task gets more complicated with more classes.

**• Binary Classification**

| | **Accuracy (%)** | **F1 Score (%)** |
|---|---|---|
| Best-performing model without manual features | 78.9 | 78.6 |
| Best-performing model with average sentence length | 80.7 | 80.5 |
| Best-performing model with domain terms | 84.7 | 84.4 |
| Best-performing model with TF-IDF | 82.4 | 82.4 |
| Best-performing model with all three features | **85.2** | **85.2** |

Table 3 compares the performance of our best-performing model with varying combinations of manual features, showing that adding average sentence length, domain terms, and TF-IDF features together yields the highest accuracy and F1 score.

**• 4-Class Classification**

| | **Accuracy (%)** | **F1 Score (%)** |
|---|---|---|
| Best-performing model without manual features | 59.8 | 60.4 |
| Best-performing model with average sentence length | 62.1 | 61.8 |
| Best-performing model with domain terms | 62.7 | 62.1 |
| Best-performing model with TF-IDF | 60.2 | 60.0 |
| Best-performing model with all three features | **63.6** | **63.7** |

Table 4 presents a comparison of accuracy and F1 scores for a 4-class classification model, illustrating incremental improvements with the addition of average sentence length and domain terms individually, and the highest performance when combining these with TF-IDF features.

## 4.2 Hyperparameter tweaking

We employed a variety of hyperparameter tuning strategies to optimize our models, ranging from manual adjustments to systematic grid searches. Each model required a unique approach to ensure maximum performance and generalization.

**• BERT-Based Classification:**

For models integrating BERT or Sentence Transformers with logistic regression, MLP Classifier, and BertForSequenceClassification, we adopted a manual approach to hyperparameter tuning. The learning rate is a critical hyperparameter. For binary classification tasks, we set the learning rate between 2e-5 and 5e-3, allowing for quick convergence without overshooting the optimal solution. In contrast, for the more complex 4-class classification, we used a lower range of 1e-5 to 5e-4. This adjustment acknowledges the increased complexity and potential for overfitting in multi-class scenarios, necessitating a more cautious optimization approach.

**• Sentence Transformer with LinearSVC:**
For the LinearSVC model, we implemented a comprehensive grid search strategy through GridSearchCV. This method systematically explored a range of hyperparameters, including 'C' for regularization strength, 'tol' for the tolerance for stopping criteria, and 'max_iter' for the maximum number of iterations, to determine the most effective combination for our classification task. By adjusting these parameters, we aimed to strike a balance between model complexity and generalization ability, ensuring robust performance without overfitting.

**• KNN Classification:**
We leveraged GridSearchCV to methodically tune the hyperparameters of the KNN model. Our grid search focused on the 'n_neighbors' parameter, with candidates ranging from 3 to 9. This parameter is fundamental to the KNN algorithm, as it determines the balance between noise sensitivity and the ability to generalize. We coupled this with the use of StratifiedKFold in cross-validation, ensuring each fold maintained representative class proportions. This strategy is particularly crucial in text classification tasks, where class imbalances can significantly skew model performance.

So, our approach to hyperparameter tuning was multifaceted and tailored to the specific demands of each model. This process is integral to achieving high-performing, robust machine learning models that effectively generalize to new data. The diversity in our tuning strategies reflects the nuanced nature of hyperparameter optimization in machine learning, highlighting the importance of context-specific approaches for different model types.

## 4.3 Training strategies

The training strategy for our models is carefully designed to address the challenges posed by a small dataset and the complexity of the model architecture. We have set the training to run for a maximum of 500 epochs. This extensive range of epochs is chosen to ensure thorough learning, but with the small dataset size of only 11 samples per class, there is a significant risk of overfitting. To counteract this, we implement an early stopping mechanism, a crucial regularization technique. Our early stopping is configured with a patience parameter of 10, meaning the training will be halted if there is no improvement in validation loss for ten consecutive epochs. This approach not only prevents overfitting but also optimizes the training time, ensuring that the model does not waste resources on learning that does not contribute to performance improvements.

In our training process, we employ the Adam optimizer due to its effectiveness in handling sparse gradients and its adaptability with variable learning rates. It maintains separate learning rates for each parameter and adjusts them throughout the training, which is particularly advantageous for handling the sparse and varied features typically seen in natural language processing tasks. Our choice of optimizer, especially beneficial in our limited data

scenario, helps navigate the challenges of deep learning models more effectively, mitigating issues like vanishing or exploding gradients.

## 4.4 Models

We undertook a comprehensive comparison of various model combinations to ascertain their effectiveness in classification tasks. The models we compared were:

• **Word2Vec-Google-News-300 + KNeighborsClassifier:**
This combination leverages the pre-trained Word2Vec embeddings from Google News, consisting of 300-dimensional vectors, with the KNeighborsClassifier. We chose this for its simplicity and effectiveness in capturing semantic relationships in text data.

• **BERT with various classifiers:**
We experimented with the BERT base model in conjunction with several classifiers - BERTForSequenceClassification, Logistic Regression, and MLPClassifier. BERT's powerful contextual embeddings were expected to significantly boost the performance of these classifiers.

• **Sentence Transformers with different classifiers:**
We also tested Sentence Transformers, known for making high-quality sentence embeddings, with classifiers like Logistic Regression, MLPClassifier, and LinearSVC. These combinations were expected to harness the nuanced representation of Sentence Transformers for effective classification.

**Expectations vs. Observations:**
For the binary classification subtask, our hypothesis was that BERT or Sentence Transformers combined with Logistic Regression would yield the best results. This expectation was based on the assumption that the sophisticated feature extraction capability of BERT and Sentence Transformers would align well with the simplicity yet effectiveness of Logistic Regression in a binary setup.

In the 4-class classification task, we anticipated that combinations of BERT or Sentence Transformers with MLPClassifier or LinearSVC would emerge as top performers. This was due to the assumption that the multi-layered structure of MLPClassifier and the margin-based optimization of LinearSVC would better handle the increased complexity of multi-class categorization.

Contrary to our expectations, the BERT base model with its native classifier, BERTForSequenceClassification, outperformed the other combinations in both binary and 4-class classification tasks. This outcome highlights the intrinsic strength of BERT in understanding context and nuances in text data, which apparently becomes more pronounced when used with its dedicated classifier. The design of BERTForSequenceClassification, which is finely tuned to work seamlessly with BERT's embeddings, likely contributed to its superior performance across both tasks.

## 4.5 Evaluation

Our evaluation strategy for comparing our models on both classification tasks was carefully structured to ensure robustness and fairness. Given the limited size of our dataset, there was a

potential risk of overfitting and biased performance estimations. To mitigate this, we implemented a train-validation-test partition approach. This method was crucial in obtaining an accurate performance assessment, as it minimized the risk of overfitting and provided a more generalized view of each model's capabilities.

In assessing the models, we employed a comprehensive set of metrics: accuracy and F1 score. Accuracy offered a straightforward measure of the overall effectiveness of our models. The F1 score provided a more nuanced view, especially valuable in scenarios where class imbalance might skew the accuracy metric. Together, these two metrics allow us to thoroughly evaluate and compare the performance of different models.

Our evaluation yielded compelling insights. The fine-tuned BERT base model, using its own classifier, excelled in both binary and 4-class classification tasks, highlighting its superior capability to understand complex text. This was evident in our results, where BERT consistently led in accuracy and F1 score.

Confusion matrices further illustrated model performances, pinpointing misclassification trends. By examining these incorrect predictions, we gained deeper understanding of model limitations, informing our future model optimization strategies.

**Performance Comparison of Models on Binary Classification Task**

| Model | Accuracy (%) | F1 Score (%) |
|---|---|---|
| BERT + BertForSequenceClassification | **85.2** | **85.2** |
| BERT + Logistic Regression | 76.7 | 76.7 |
| BERT + MLPClassifier | 77.8 | 77.8 |
| Sentence Transformers + Logistic Regression | 77.3 | 77.2 |
| Sentence Transformers + MLPClassifier | 70.8 | 70.7 |
| Sentence Transformers + Linear SVC | 72.7 | 72.7 |
| Word2Vec + KNN | 75.3 | 75.3 |

Table 5 presents the performance results of various models for the binary classification task, with each model evaluated based on its Accuracy and F1 Score percentages.

**Performance Comparison of Models on 4-Class Classification Task**

| Model | Accuracy (%) | F1 Score (%) |
|---|---|---|
| BERT + BertForSequenceClassification | **63.6** | **63.7** |
| BERT + Logistic Regression | 53.8 | 51.5 |
| BERT + MLPClassifier | 59.8 | 56.1 |
| Sentence Transformers + Logistic Regression | 50.9 | 50.6 |
| Sentence Transformers + MLPClassifier | 51.5 | 51.4 |
| Sentence Transformers + Linear SVC | 56.8 | 56.0 |
| Word2Vec + KNN | 50.0 | 48.8 |

Table 6 showcases the performance outcomes for various models on the 4-class classification task, with each model's performance quantified by Accuracy and F1 Score percentages.

# 5 Error Analysis

## 5.1 Confusion matrix & qualitative analysis

**• Binary Classification**

In the binary classification task, our models demonstrate a consistent error pattern: a tendency to misclassify actual positive instances (class '1') as negative (class '0'). This issue suggests a difficulty in capturing complex textual subtleties which could be due to the insufficient representation of such intricate cases in the training set. Alternatively, two models show an overclassification of negatives as positives.

A qualitative review of incorrect predictions highlights the models' struggle to interpret complex, lengthy narratives that include financial jargon and nuanced personal experiences. For instance, a complaint about Lexington Law was wrongly categorized as '0' (no knowledge) when evidence of legal knowledge suggested it should be '1' (some knowledge).

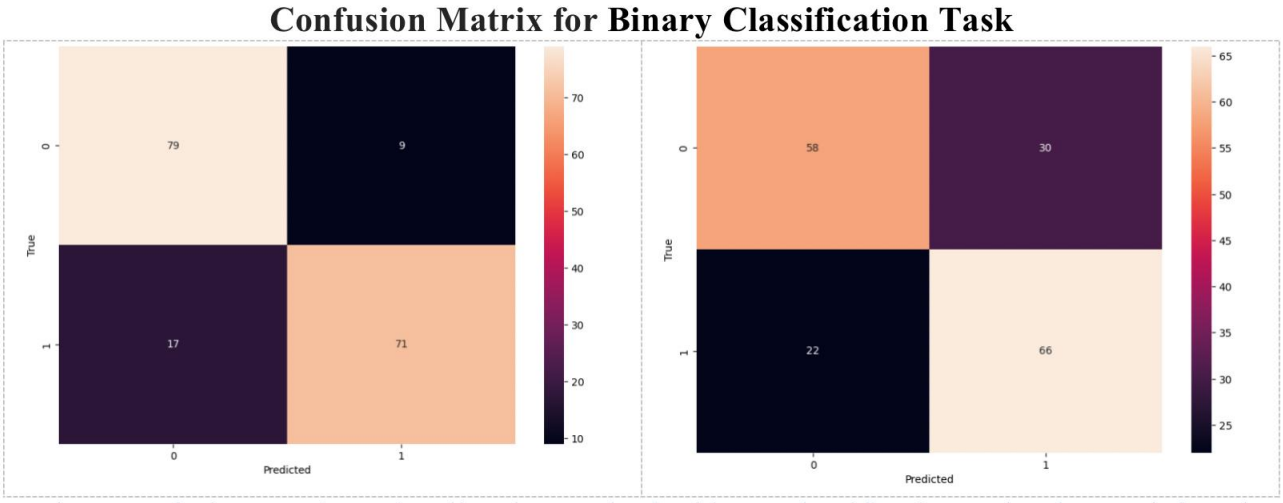


Figure 1: these confusion matrices visualize the results for binary classification task. The top-left and bottom-right cells display the number of correct predictions for the negative (0) and positive (1) classes, respectively, known as true negatives and true positives. Conversely, the top-right cell indicates instances wrongly classified as positive and the bottom-left cell represents instances incorrectly predicted as negative outcomes for actual positive instances.

**• 4-Class Classification**

For the 4-class classification, models falter when differentiating between adjacent classes of financial or legal knowledge, such as between 'some' and 'moderate' or 'moderate' and 'profound'. Errors in predictions imply an overestimation of language sophistication or a failure to recognize detailed financial or legal language indicative of higher knowledge levels. For example, an articulate grievance about credit report issues may be inaccurately assigned to a lower knowledge category due to the model's limited understanding of specialized terminology and contextual cues.

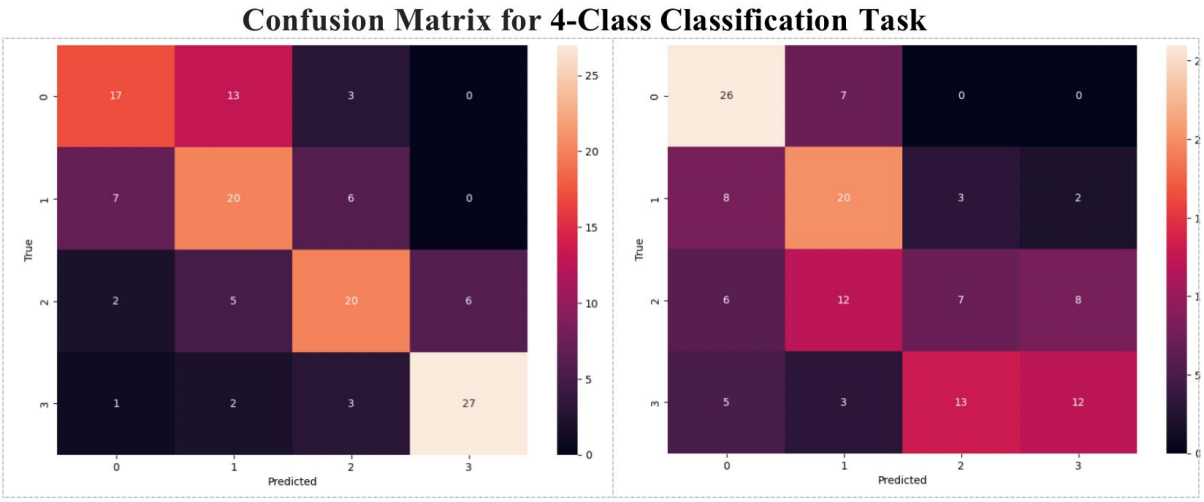


Figure 2: these confusion matrices depict the performance of two models on a 4-class classification task, categorizing levels of financial and legal knowledge. The left matrix, representing the best-performing model, shows a pattern of misclassification between classes with no or some knowledge (0 and 1) and some and moderate knowledge (1 and 2). The right matrix reveals difficulty distinguishing between moderate and profound knowledge (2 and 3).

## 5.2 Explanations

Such mistakes are likely influenced by two main factors: labeling process and data limitations.

Given that labeling was performed by team members rather than experts, inconsistencies could have been introduced despite efforts to standardize the process through internal agreement. This lack of professional consistency in labeling could lead to models that are uncertain about the characteristics that define each class, as the examples it learns from may not perfectly represent the true nature of each category.

The small dataset size exacerbates this problem. With only 11 samples per class in a few-shot learning scenario, the models have a limited context for learning. This constraint can lead to both overfitting, where the model learns noise and idiosyncrasies of the small training set rather than general patterns, and underfitting, where the model fails to capture the complexity of the classes due to insufficient examples.

## 5.3 Possible improvements

To address the above issues, we can do the following as our future directions:

One of the primary steps would be to involve financial professionals in the labeling process. These experts bring a nuanced understanding of financial language and concepts, which can significantly enhance the accuracy and reliability of labeled data. Their domain-specific knowledge ensures that the subtleties and complexities of financial jargon are correctly interpreted and categorized, which will lead to a more informed and precise training dataset.

We would also like to implement regularization techniques in the model to combat overfitting, which is especially important given the small dataset size. Techniques like L1 or L2 regularization, or dropout can penalize complexity in the model, encouraging it to learn the most important patterns and maintain simplicity. This can help the model generalize better to new, unseen data.

In addition, we can employ cross-validation techniques to provide a more accurate assessment of the model's performance. Since the dataset is small, techniques like k-fold cross-validation can maximize the training data used and provide insights into how the model might perform on independent datasets.

## 6. Limitations and Conclusion

In conclusion, our exploratory study, under the challenging context of few-shot learning, ventured into the domain of financial text classification. Utilizing a range of machine learning models, we navigated the complexities of this scenario. BERT's exceptional performance across both binary and 4-class tasks stood out, demonstrating its adaptability and profound semantic understanding in a limited data environment. Despite facing challenges such as labeling inconsistencies and the inherent constraints of few-shot learning, our findings provide a solid foundation for future research. This study serves as a significant step towards refining text classification techniques in specialized domains, paving the way for more advanced, context-sensitive models in the field of financial narrative analysis.

**References**

Bansal, T., Jha, R., & McCallum, A. (2019). Learning to few-shot learn across diverse natural language classification tasks. *arXiv preprint arXiv:1911.03863*.

Fu, M., Cao, Y., & Wu, J. (2022). Worst case matters for few-shot recognition.. https://doi.org/10.48550/arxiv.2203.06574

Girish, K., Hegdev, A., Balouchzahi, F., & Lakshmaiah, S. H. (2023). Profiling cryptocurrency influencers with sentence transformers. *Working Notes of CLEF*.

J. Bevendorff, I. Borrego-Obrador, M. Chinea-Ríos, M. Franco-Salvador, M. Fröbe, A. Heini, K. Kredens, M. Mayerl, P. Pęzik, M. Potthast, F. Rangel, P. Rosso, E. Stamatatos, B. Stein, M. Wiegmann, M. Wolska, , E. Zangerle, Overview of PAN 2023: Authorship Verification, Multi-Author Writing Style Analysis, Profiling Cryptocurrency Influencers, and Trigger Detection, in: A. Arampatzis, E. Kanoulas, T. Tsikrika, A. G. Stefanos Vrochidis, D. Li, M. Aliannejadi, M. Vlachos, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multi-linguality, Multimodality, and Interaction. Proceedings of the Fourteenth International Conference of the CLEF Association (CLEF 2023), Lecture Notes in Computer Science, Springer, 2023.

Tian, P., Wu, Z., Qi, L., Wang, L., Shi, Y., & Gao, Y. (2020, April). Differentiable meta-learning model for few-shot semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 07, pp. 12087-12094).

Villa-Cueva, E., Valles-Silva, J. M., López-Monroy, A. P., Sanchez-Vega, F., & Lopez-Santillan, R. (2023). Few Shot Profiling of Cryptocurrency Influencers using Natural Language Inference & Large Language Models.

Wang, Y., Yao, Q., Kwok, J. T., & Ni, L. M. (2020). Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, *53*(3), 1-34.