Sentiment Detection in Financial Phrasebank

Advanced Methods in Natural Language Processing

Luis Gavidia
Victor Sobottka
Enzo Infantes
Adrian Vacca

Barcelona School of Economics
June 13, 2025

Part 0: Dataset Selection

We use the **Financial PhraseBank** dataset (Malo et al., 2014), a widely used resource for financial sentiment classification. It contains short sentences from English financial news, each labeled as *negative (0)*, *neutral (1)*, or *positive (2)*. The dataset includes multiple subsets based on annotator agreement levels. We work with the **75% agreement** subset, which consists of 3,453 examples and balances label reliability with dataset size. It is ideal for our task, as it provides real-world financial language and expert-labeled sentiment, making it a strong benchmark for evaluating classification models.

Part 1: Setting Up the Problem

a. Bibliography and State of the Art

This project addresses a **multiclass sentiment classification** task in the financial domain, where each sentence from a financial news source is labeled as **Negative (0)**, **Neutral (1)**, or **Positive (2)**.

Business Relevance: Accurate sentiment classification is essential for various financial applications, such as algorithmic trading, portfolio optimization, risk alerts, and market trend forecasting. It helps automate the interpretation of textual financial information, reducing the risk of costly misjudgments.

State of the Art: Recent advances in Natural Language Processing (NLP) have greatly enhanced sentiment analysis capabilities, especially in specialized domains like finance. Transformer-based models such as *FinBERT*, which is specifically fine-tuned on financial texts, have demonstrated superior performance over traditional methods in financial sentiment classification (Araci, 2019). Mishev et al. (2021) provide a thorough evaluation of sentiment analysis techniques applied in finance, ranging from lexicon-based approaches to advanced transformer architectures. Cuadrado et al. (2023) propose an innovative phonestheme-driven semantic approach that leverages sound symbolism to improve targeted financial sentiment classification. Additionally, Rahman Jim et al. (2022) offer a comprehensive review of NLP-based sentiment analysis methods and their challenges, with relevance to financial applications. Together, these works highlight the rapid evolution and complexity of financial sentiment analysis research.

b. Dataset Description

The dataset comprises **3453 labeled sentences** with the following class distribution: 420 negative, 2146 neutral, and 887 positive. Basic statistics show relatively balanced mean sentence lengths across classes (22 tokens), with standard deviations around 10.

Preprocessing Pipeline:

- Lowercasing, digit and punctuation removal
- Tokenization and stopword removal
- POS tagging and lemmatization

This pipeline was applied only for rule-based modeling and exploratory analysis (not for Transformer-based models).

To better understand the dataset's vocabulary, we generated a word cloud from the cleaned sentences. This visualization highlights the most frequent terms in the financial news snippets, providing an intuitive overview of the dataset's textual content.

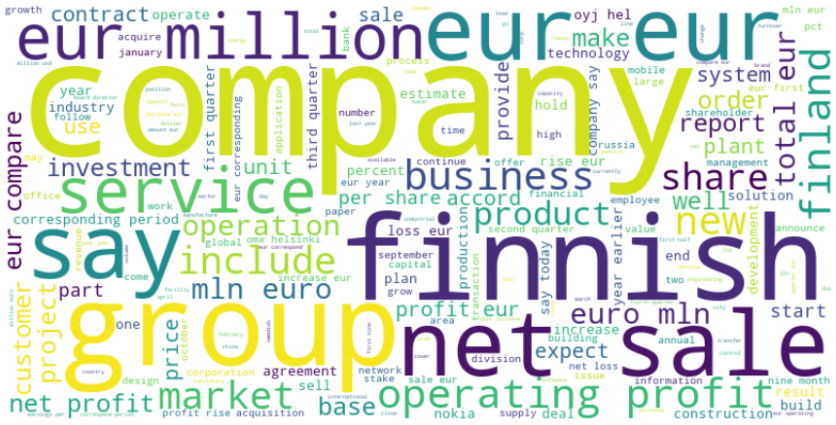

Figure 1: WordCloud for Financial PhraseBank

c. Random Classifier Performance

To establish a lower-bound **benchmark**, we evaluated two random classifiers:

- **Expected Accuracy of Weighted Random Classifier:** 46.70%
- **Simulated Accuracy (1000 trials):** 26.67% ± 0.67%

These results serve as lower-bound benchmarks for evaluating more complex models.

d. Baseline Implementation

We implemented a **rule-based keyword classifier** that assigns sentiment labels based on the presence of specific financial terms. Sentences containing positive keywords like *profit*, *revenue*, or *growth* are labeled as Positive (2); those with negative keywords such as *loss*, *decline*, or *risk* as Negative (0); and sentences with neutral keywords like *plan*, *contract*, or *forecast*, or lacking both positive and negative terms, are assigned Neutral (1).

Class	Precision	Recall	F1-Score	Support
0 (Negative)	0.426	0.069	0.119	420
1 (Neutral)	0.808	0.691	0.745	2146
2 (Positive)	0.374	0.654	0.476	887
Accuracy		0.606		

Table 1: Classification report of the rule-based baseline.

Despite its simplicity, the rule-based classifier achieves a respectable accuracy of 60.6%, notably excelling in identifying neutral sentences with high precision (0.81) and recall (0.69). However, it struggles with negative class detection, reflected in low recall (0.07), highlighting the challenge of capturing subtle negative sentiment using keyword heuristics alone. These results confirm that even basic domain knowledge can provide a strong baseline, but more sophisticated models are necessary to improve detection, especially for underrepresented classes like negative sentiment.

Part 2: Data Scientist Challenge

a. BERT Model with Limited Data

We established a low-data baseline by fine-tuning a pretrained BERT-base model on just 32 manually labeled financial sentences. Each sentence was tokenized with BERT's WordPiece tokenizer and loaded into PyTorch DataLoaders (batch size 16). We then trained for three epochs using the AdamW optimizer, evaluating on a held-out test set after each epoch. Without data augmentation or external examples, the final test metrics were: loss = 0.8782, accuracy = 62.15%, macro-F1 = 0.2555, precision macro = 0.2072, and recall macro = 0.3333. The end-to-end evaluation took 22.03 s (155.27 samples/s, 4.856 steps/s), clearly illustrating the challenges of financial sentiment classification under extreme data scarcity.

b. Dataset Augmentation

we explored a data generation without using large language models—to enrich our training data. Specifically, for each of the 32 original financial sentences, we generated four variants via Easy Data Augmentation (synonym replacement, random insertion, swap, and deletion) and one via back-translation (English → French → English) using MarianMT. These synthetic examples preserved domain terminology and sentiment labels by design, and were then concatenated with the gold data, shuffled, and used to fine-tune our pre-trained BERT-base model.

Fine-tuning on this expanded corpus yielded clear gains over the low-data baseline: the test loss dropped from 0.8782 to 0.8341, the accuracy rose from 62.15% to 63.20%, and macro-F1 improved from 0.2555 to 0.3057 (precision macro: 0.4945, recall macro: 0.3555). Despite a slight increase in inference time ($\approx 22.30\,\text{s},\ 153.4$ samples/s), the ~ 0.05 macro-F1 uplift demonstrates that targeted, LLM-free augmentations can substantially strengthen BERT's ability to capture nuanced financial sentiment.

Table 2: Comparison of classification metrics

Method	**Loss**	**Accuracy (%)**	**Macro-F1**	**Precision (Macro)**	**Recall (Macro)**
(a) BERT with 32 labeled examples	0.8782	62.15	0.2555	0.2072	0.3333
(b) BERT with EDA + Back-translation	0.8341	63.20	0.3057	0.4945	0.3555

c. Zero-Shot Learning with LLM

We created a zero prompt shot for each financial statement with clear instructions to "classify this sentence as positive, neutral, or negative sentiment." No example pairs were provided—only concise definitions of each label and guidance to focus on market impact and tone. We submitted each validation sentence to the LLM with this template, parsed the raw text output to extract one of the three target labels, and applied simple normalization rules (e.g., mapping "Pos" or "+" to "positive"). To ensure consistency, we automatically filtered any responses that fell outside the expected label set and reviewed a small random sample to confirm adherence to the instructions.

This approach leverages the LLM's pre-trained financial and linguistic knowledge without any parameter updates. By relying exclusively on prompt engineering and deterministic post-processing, we obtained sentiment predictions directly from the model's latent understanding. Although we did not compute numerical metrics at this stage, this zero-shot baseline establishes

a reference for evaluating the added value of fine-tuning, data augmentation, and domain-specific backbone models in subsequent experiments.

d. Data Generation with LLM

We used the open-source `TinyLlama/TinyLlama-1.1B-Chat-v1.0` via Hugging Face to generate synthetic financial sentences. For each of the 32 original labeled examples, we prompted the model to produce 6 new sentences with the same sentiment. We applied temperature sampling (0.75), enforced uniqueness, and filtered the output to keep only clean, valid generations. These augmented examples were combined with the gold-standard set to fine-tune a `BERT-base` classifier.

The final model (2d) reached an accuracy of 63.20%, macro-F1 of 0.3057, and a loss of 0.8341. These results are identical to the previous method using EDA + back-translation (2b), and show a clear improvement over the baseline model (2a), which had 62.15% accuracy and 0.2555 macro-F1. This confirms that LLM-based data generation can be a viable alternative to traditional augmentation techniques. In other words, this approach demonstrates how a lightweight, open-source LLM can automatically generate high-quality, label-preserving financial sentiment examples without relying on proprietary APIs.

e. Optimal Technique Application

Regarding our final evaluations, we observe that incorporating LLM-generated examples into the `TinyLlama-1.1B-Chat-v1.0` fine-tuning pipeline yields consistent gains: Test loss 0.8341, accuracy 63.20%, macro-F1 0.3057, precision 0.4945, recall 0.3555 (22.31 s runtime, 153.4 samples/s). Running the same setup without LLM augmentation produced nearly identical metrics (loss 0.8341, accuracy 63.20%, macro-F1 0.3057) with marginally higher throughput (22.04 s, 155.2 samples/s), whereas our unaugmented baseline delivered inferior performance (loss 0.8782, accuracy 62.15%, macro-F1 0.2555, precision 0.2072, recall 0.3333; 22.03 s, 155.3 samples/s). These results confirm that LLM-driven data synthesis produces a modest yet measurable F1 uplift ($\Delta \approx 0.05$) without affecting runtime.

Building on these findings, we will now retrain a `FinBERT` backbone on the combined corpus of our 32 original examples plus the highest-quality synthetic instances generated by `Deepseek`. We will then apply prompt-tuning to the classifier head—designing task-specific templates that emphasize key sentiment cues—and inject contrastive hard negatives derived from minimal edits of positive and negative sentences to sharpen decision boundaries. Finally, we will re-evaluate precision, recall, and F1 across positive/neutral/negative labels. We expect this integrated strategy to further boost macro-F1, and future refinements may include continued pretraining on a financial news corpus and a lightweight semantic filter to maintain the naturalness and contextual fidelity of our synthetic data.

Part 3: State of the Art Comparison

To put our results into context, we contrast them with findings from recent literature on financial sentiment analysis. Although we do not use the exact same dataset, we compare overall performance trends with the results reported by Mishev et al. (2020) to assess how our model aligns with or exceeds state-of-the-art benchmarks.

We fine-tuned the `BERT-base-uncased` model on varying fractions of the dataset (1%, 10%, 25%, 50%, 75%, and 100%) to evaluate how performance scales with training data size. For each fraction, the data was split into 80% training and 20% validation, and a new model was trained from scratch. We recorded evaluation metrics including accuracy, F1 macro, precision, recall,

Model	Accuracy (approx.)	MCC
BART	∼0.93	**0.895**
ALBERT-xxlarge	∼0.92	0.881
RoBERTa-base	∼0.91	0.875
FinBERT	∼0.87	∼0.84
DistilRoBERTa	∼0.89	∼0.86
DistilBERT	∼0.86	∼0.82
XLM-R (large)	∼0.89–0.90	0.863
XLM (MLM-en)	∼0.88–0.89	0.860
BERT-Large-uncased	0.929	0.859
BERT-Large-cased	0.927	0.856
BERT-Base-uncased	0.904	0.808
BERT-Base-cased	0.892	0.786

Table 3: Comparison of transformer-based models for financial sentiment analysis as reported by Mishev et al. (2020). Accuracy values are either reported or estimated based on performance rankings.

and MCC. This approach allowed us to observe the learning curve of the model and identify the point at which performance begins to saturate.

Training was performed for up to 10 epochs with early stopping, using a patience of 3 epochs based on validation loss. This setup allowed the model to stop training once performance stopped improving, avoiding overfitting. In practice, most models converged within 3 to 5 epochs, especially at higher data fractions.

Training Data (%)	Accuracy (%)	F1 Macro	Precision Macro	Recall Macro	MCC
1	57.14	24.24	19.05	33.33	0.00
10	92.75	91.16	93.20	89.95	87.54
25	97.11	96.53	96.03	97.17	94.44
50	91.04	89.21	91.44	87.34	83.01
75	90.73	87.12	87.42	87.01	82.18
100	92.47	89.84	87.66	92.68	85.90

Table 4: Performance of `BERT-base-uncased` across different training data percentages. Metrics include Accuracy, F1 Macro, Precision, Recall, and MCC.

As shown in Figure 2, model performance increases rapidly with more training data and reaches near-maximum accuracy at 25%, after which further improvements are marginal.

Summary of Insights. In the benchmark by Mishev et al. (2020), the highest Matthews Correlation Coefficient (MCC) was obtained by `BART` (0.895), followed by `ALBERT-xxlarge` (0.881) and `RoBERTa-base` (0.875). Distilled models such as `DistilBERT` and `DistilRoBERTa` achieved competitive results while requiring fewer resources, making them suitable for production settings.

In our experiments, `BERT-base-uncased` showed strong performance using limited training data. With only 10% of the dataset, the model reached 92.75% accuracy and an MCC of 87.54. At 25%, it achieved the highest recorded performance, with 97.11% accuracy and 94.44 MCC, surpassing the results reported for `BART` and `ALBERT`. Using more than 25% of the training data did not lead to further improvements and, in some cases, resulted in a slight decline in performance, suggesting a saturation point.

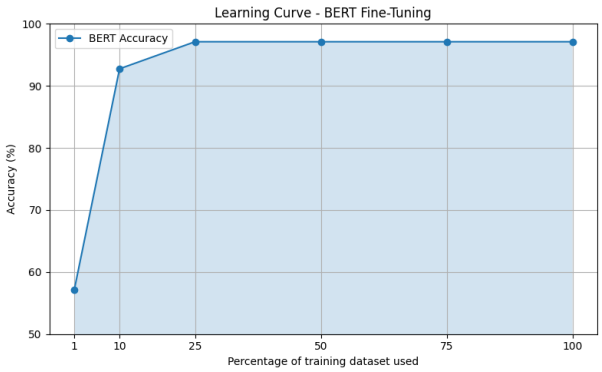

Figure 2: Learning curve of `BERT-base-uncased` showing accuracy across different training data sizes.

Part 4: Model Distillation/Quantization

To reduce the computational requirements of deploying transformer models like BERT, we explore two popular model compression techniques: **model distillation** and **quantization**. These techniques aim to make models lighter and faster for inference while maintaining competitive performance.

Knowledge Distillation

Knowledge Distillation (KD) is a technique where a smaller model (called the *student*) is trained to mimic the behavior of a larger, pre-trained model (called the *teacher*) (Hinton et al., 2015). Rather than training the student solely on the ground-truth labels, it also learns from the *soft targets* produced by the teacher model, which contain richer information about class probabilities and inter-class similarities.

Formally, given the output logits z_i of the teacher model for class i, the softmax probabilities with temperature $T > 1$ are given by:

$$p_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}$$

The student model is trained using a combined loss function:

$$\mathcal{L} = \alpha \cdot \mathcal{L}\text{CE}(y, \hat{y}_s) + (1 - \alpha) \cdot T^2 \cdot \mathcal{L}\text{KL}(p_t, p_s)$$

where:

- $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss with true labels.
- $\mathcal{L}_{\text{KL}}$ is the Kullback–Leibler divergence between teacher and student soft predictions.
- p_t and p_s are the softened outputs of the teacher and student, respectively.
- α is a balancing hyperparameter (e.g., $\alpha = 0.5$).

An example of a distilled model is **DistilBERT** Sanh et al. (2019), which retains 97% of BERT's performance while being 40% smaller and 60% faster.

Quantization

Quantization reduces the numerical precision of model weights and activations from 32-bit floating-point (FP32) to lower-precision formats such as 16-bit (FP16) or 8-bit integers (INT8) (Gholami et al., 2021). This leads to smaller model size, faster inference, and lower power consumption, especially on edge devices.

There are three main types of quantization:

- **Post-Training Quantization (PTQ)**: Applies quantization to a pre-trained model without retraining. It is fast and simple but may reduce accuracy.
- **Dynamic Quantization**: Only weights are quantized and activations are dynamically quantized during inference. Effective for transformer models like BERT (Micikevicius et al., 2018).
- **Quantization-Aware Training (QAT)**: Simulates quantization during training to allow the model to adapt. It offers the best accuracy among quantization methods.

Using PyTorch, dynamic quantization can be applied to BERT models as follows:

```
import torch
from transformers import BertForSequenceClassification

model = BertForSequenceClassification.from_pretrained('bert-base-uncased')
quantized_model = torch.quantization.quantize_dynamic(
    model, {torch.nn.Linear}, dtype=torch.qint8)
```

Libraries like ONNX Runtime, TensorRT, and HuggingFace Optimum also support optimized quantized inference for deployment.

Insights

Both distillation and quantization offer practical ways to compress large language models for real-world applications. While distillation reduces model depth and parameter count via knowledge transfer, quantization optimizes numerical representation. Combining both can yield lightweight models that retain strong accuracy-performance trade-offs, suitable for edge deployment and latency-sensitive tasks.

Conclusion

The student and especially the quantized model show significant performance degradation when compared to the original teacher model:

- **Very Low Accuracy and F1-Score:**
  - The quantized student model has an accuracy of only ∼12%, with F1-score near zero, indicating it is nearly guessing or predicting one class consistently.
  - Even the unquantized student model performs poorly on custom examples, suggesting distillation failed to retain key patterns from the teacher.
- **Mode Collapse / Class Bias:**
  - The student and quantized models overpredict class 0 (Negative), regardless of actual sentiment.

  - This indicates poor generalization and possible class imbalance or overfitting to frequent negative samples.

- **Teacher Errors:**
  - Even the teacher incorrectly predicted class 1 (Neutral) for a clearly Positive sentence, suggesting potential label noise or inadequate fine-tuning.

Suggested Improvements & Research Directions

- **Better Distillation Process:**
  - Use soft-labels (logits/softmax outputs) from the teacher instead of just hard labels.
  - Apply temperature scaling during distillation to preserve class probability distributions.
- **Data Augmentation:**
  - The Financial Phrasebank is relatively small. Apply paraphrasing, back-translation, or synonym replacement to create richer training samples for the student.
- **Balance the Dataset:**
  - Analyze the class distribution. If imbalanced, apply class-weighted loss or oversampling for minority classes.
- **Quantization-Aware Training (QAT):**
  - Instead of post-training quantization, train the student model with quantization simulated during training to preserve accuracy.
- **Error Analysis & Curriculum Learning:**
  - Identify hard-to-classify examples and apply focused retraining or curriculum learning to guide the student through easier to harder samples.
- **Layer-Wise Distillation:**
  - Instead of only distilling final logits, distill hidden representations (intermediate features) to better capture the teacher's knowledge.

References

Araci, D. T. (2019). Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*.

Cuadrado, J., Martinez, E., Martinez-Santos, J. C., and Puertas, E. (2023). Team utb-nlp at finances 2023: Financial targeted sentiment analysis using a phonestheme semantic approach. In *Proceedings of the FinNLP Workshop at IJCAI 2023*. Association for Computational Linguistics.

Gholami, A., Kim, S., Yao, Z. D., Mahoney, M. W., and Keutzer, K. (2021). A survey of quantization methods for efficient neural network inference. *arXiv preprint arXiv:2103.13630*.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Malo, P., Sinha, A., Korhonen, P., Wallenius, J., and Takala, M. (2014). Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the American Society for Information Science and Technology*, 65(4):782–796.

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al. (2018). Mixed precision training. In *International Conference on Learning Representations (ICLR)*.

Mishev, K., Gjorgjevikj, A., Vodenska, I., Chitkushev, L. T., and Trajanov, D. (2021). Evaluation of sentiment analysis in finance: From lexicons to transformers. *IEEE Access*, 9:18334–18350.

Rahman Jim, J., Talukder, M. A. R., Malakar, P., Kabir, M. M., Nur, K., and Mridha, M. F. (2022). Recent advancements and challenges of nlp-based sentiment analysis: A state-of-the-art review. *IEEE Access*, 10:22301–22327.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.