

FinBERT: A Large Language Model for Extracting Information from Financial Text

Allen H. Huang
Hui Wang
Yi Yang

Contemporary Accounting Research, 2023
Presented by Byunghun Han

01/03/2025

Contents

- 1 Introduction
- 2 Deep Learning NLP Algorithms
- 3 Comparing Performance on Sentiment Classification
- 4 Comparing Performance in Labeling ESG-related Discussions
- 5 Conclusion

1. Introduction

Motivation

David Baker

"for computational protein design"



David Baker. Ill. Niklas Elmehed © Nobel Prize Outreach

Demis Hassabis

"for protein structure prediction"



Demis Hassabis. Ill. Niklas Elmehed © Nobel Prize Outreach

John Jumper

"for protein structure prediction"



John Jumper. Ill. Niklas Elmehed © Nobel Prize Outreach

Figure 1: The Nobel Prize Winners in Chemistry 2024 (Source: Nobelprize.org)

1. Introduction

Motivation

- Natural language processing (NLP) algorithms have been widely used to analyze financial texts in finance and accounting.
 - ▶ Dictionary
 - ▶ Machine learning
- Recently, deep-learning-based NLP algorithms have been developed (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019).
 - ▶ Open AI Generative PreTraining model (OpenAI GPT)
 - ▶ Bidirectional Encoder Representations from Transformers (BERT)
 - ▶ Embeddings from Language Model (ELMo)
- Large Language Models (LLMs)
 - ▶ A large number of parameters allows them to learn **semantic** and **syntactic** relations among words.
⇒ Context!
 - ▶ 264 papers related to ChatGPT and other LLMs are uploaded to SSRN within the accounting, finance, or economics networks January 2022 to March 2024 (M.M. Dong et al., 2024).

1. Introduction

Motivation

- This paper introduces **FinBERT**, an LLM customized for financial texts based on Google's BERT algorithm.
 - ▶ Whether and how much FinBERT outperforms simpler algorithms in finance- and accounting-related tasks

2. Deep Learning NLP Algorithms

Background: Embedding

- What is embedding?
 - ▶ Process of mapping words, sentences, or documents into **vectors** in a high-dimensional space
 - ▶ Semantically similar words are placed **closer together** in the embedding space.
 - ▶ Ex) "Dog" - "Cat" vs. "Dog" - "Car"
- It is necessary because text data cannot be directly understood or processed by computers:
 - ▶ Allows models to learn **semantic** relationships between words
 - ▶ Prepares text data for **input** into machine learning models
 - ▶ Enables quantitative evaluations of similarity through mathematical operations
- Trends of Embedding by time
 - 1 One-Hot Encoding
 - 2 Neural Word Embedding
 - 3 Contextual Embedding

2. Deep Learning NLP Algorithms

One-Hot Encoding

- Assumption: Words are **independent** of each other.
 - ▶ Captures whether a word exists in a document
 - ▶ "Bag-of-words" structure
 - Each word is represented by a unique vector, but it produces large dimensions and lacks semantic meaning.
 - ▶ "Dog" $\rightarrow [1, 0, 0]$
 - ▶ "Cat" $\rightarrow [0, 1, 0]$
 - ▶ "Car" $\rightarrow [0, 0, 1]$
- $\Rightarrow$ **No** semantic similarities between words
- Models
 - 1 Loughran-McDonald dictionary (LM)
 - 2 Naïve Bayes (NB)
 - 3 Support Vector Machine (SVM)
 - 4 Random Forest (RF)

2. Deep Learning NLP Algorithms

Neural Word Embeddings

- Neural word embedding leverages neural networks from **deep learning** to learn the relationships between words.
 - ▶ Neural word embedding works as a layer within a deep learning model.
 - ▶ Input layer: Accepts the word data
 - ▶ Hidden layer: Learns the word vectors → Neural word embedding
 - ▶ Output layer: Achieves specific learning objectives, such as predicting surrounding words.
- It determines each word's vector representation by learning their similarities from co-occurrences in texts.
 - ▶ Based on **surrounding** words in the training text
- Each word is represented by a **static vector** regardless of actual surrounding words in the text of NLP tasks.
 - ▶ Ex) A word "bank" is expressed as the same vector with two different meanings: "institution" vs. "river".

2. Deep Learning NLP Algorithms

Neural Word Embeddings

- Advantages over one-hot encoding

- 1 Learn **relations** between words that its vectors capture both semantic and syntactic information
- 2 Produce much **smaller** vectors of a few hundred dimensions
- 3 Possible to calculate words' similarities within vector space:
ex) "king" - "man" + "woman" $\approx$ "queen"

- Limitation

- ▶ It cannot capture the contexts of a word in a sentence if the word has several meanings.
- ▶ "I deposited money in the bank"
- ▶ "We walked along the river bank"

⇒ Contextualized Embedding

- Models

- 1 Convolutional Neural Network (CNN)
- 2 Long Short-Term Memory (LSTM)

2. Deep Learning NLP Algorithms

Contextual Embedding

- NLP researchers have introduced LLMs, which have contextualized embeddings (Peters et al. 2018; Radford et al. 2018; Devlin et al. 2019).
 - ▶ Dynamic → They represent words using different vectors depending on their context.
- Examples
 - ▶ OpenAI GPT
 - ▶ BERT model by Google
 - ▶ BERT processes inputs both left-to-right and right-to-left simultaneously and considers the entire sentence to derive contextual meaning for each word.
- Transfer learning
 - ▶ Transfer knowledge learned by a model in one domain or task to another domain or task
 - ① Pretraining ⇒ Train a model on a large dataset and a general task
 - ② Fine-tuning ⇒ Adapt the pretrained model to the specific target task with a smaller dataset

2. Deep Learning NLP Algorithms

Bidirectional Encoder Representations from Transformers (BERT) Algorithms

- BERT algorithms have two training objectives in the pretraining step.
 - 1 Masked Language Model (MLM)
 - 2 Next Sentence Prediction (NSP)
- Masked Language Model (MLM)
 - ▶ Predict masked words in a sentence, enabling it to understand semantic and syntactic relationships within text
 - ▶ Some percentage of tokens in the input are **randomly** masked (replaced with a [MASK] token).
 - ▶ Trained to predict the original word based on its surrounding context
 - ▶ Example:
 - ▶ Input: "The stock price of Apple [MASK] significantly yesterday."
 - ▶ Target: "The stock price of Apple rose significantly yesterday."

2. Deep Learning NLP Algorithms

Bidirectional Encoder Representations from Transformers (BERT) Algorithms

- Next Sentence Prediction (NSP)
 - ▶ Helps the model understand the relationship between two sentences, improving tasks like document classification or question answering
 - ▶ The model is given a pair of sentences and tasked to **predict** whether the second sentence logically follows the first.
 - ▶ Example:
 - ▶ Positive: The second sentence follows the first
→ "The company reported profits. Investors reacted positively."
 - ▶ Negative: The second sentence is unrelated to the first
→ "The company reported profits. The weather was sunny today."
- After pretraining, BERT can be fine-tuned for specific NLP tasks by adding a simple classification head (or task-specific layer).
 - ▶ Sentiment analysis

2. Deep Learning NLP Algorithms

Limitation in BERT model

- Issue: The BERT model may **not** work well for processing **financial texts** written for professional investors, because financial texts are **different** from general texts in vocabulary and writing style.
 - ▶ Ex) bank: "Lending" vs. "Fish" or "River"
- Therefore, **financial texts** are used in pretraining to create a context-specific vector space.
 - ① Corporate annual and quarterly filings from the SEC's EDGAR website
 - ▶ 60,490 10-Ks and 142,622 10-Qs of Russell 3000 firms
 - ▶ Between 1994 and 2019
 - ② Financial analyst reports from the Thomson Investext database
 - ▶ 476,633 reports issued for S&P 500 firms
 - ▶ Between 2003 and 2012
 - ③ Earnings conference call transcripts from the SeekingAlpha website
 - ▶ 136,578 transcripts of 7,740 public firms
 - ▶ Between 2004 and 2019

2. Deep Learning NLP Algorithms

FinBERT: A finance domain-adapted LLM

- Reasons for choosing the texts in pretraining FinBERT
 - ① They are the most **representative** and frequently used types of corporate financial communications (Loughran and McDonald, 2016).
 - ② They include comments on important stakeholders in financial communications, which is **informative** to investors (Li, 2010; Huang, 2014, 2022).
 - ③ They help FinBERT get used to **diverse** financial topics and communication styles.

2. Deep Learning NLP Algorithms

FinBERT: A finance domain-adapted LLM

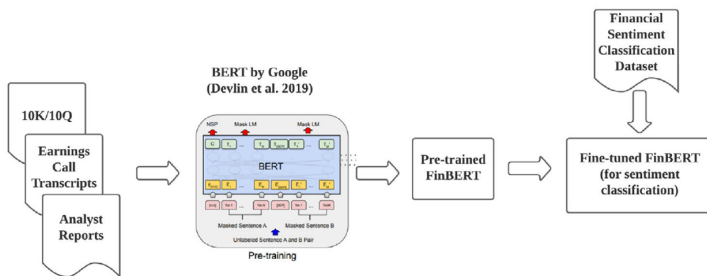


Figure 2: FinBERT pretraining and fine-tuning (for sentiment classification)

- FinBERT can be fine-tuned for a specific task.
 - ▶ Here, sentiment classification
 - ▶ ESG-related classification
 - ▶ But, it requires labeled data.

3. Comparing Performance on Sentiment Classification

- Sentiment classifications: FinBERT vs. non-FinBERT algorithms
 - ① Loughran-McDonald (LM) dictionary
 - ② Naïve Bayes (NB) classification
 - ③ Support Vector Machine (SVM)
 - ④ Random Forest (RF)
 - ⑤ Convolutional Neural Network (CNN)
 - ⑥ Long Short-Term Memory (LSTM)
- The same training and testing samples are used for different algorithms.
 - ▶ Financial analyst reports with researcher-labeled sentiments at the sentence level (Huang et al., 2014)

3. Comparing Performance on Sentiment Classification

Comparing performance on the full training sample

- The sample includes 10,000 sentences.
 - ▶ 3,577 Positive
 - ▶ 4,586 Neutral
 - ▶ 1,837 Negative
 - ▶ In this step, sample imbalance needs to be considered (Lemaître, Nogueira, and Aridas, 2017; He et al., 2024).
- Construction of different sample sets
 - ▶ To **fine-tune** FinBERT and BERT and **train** the NLP algorithms
 - ▶ Training: 8,100 sentences (81%)
 - ▶ Validation: 900 sentences (9%)
 - ▶ Testing: 1,000 sentences (10%)

3. Comparing Performance on Sentiment Classification

Comparing performance on the full training sample

- For LM dictionary,
 - ▶ A sentence is **negative** if at least one word from the LM Fin-Neg list.
 - ▶ A sentence is **positive** if it does not have any word from the LM Fin-Neg list and contains at least one word from the LM Fin-Pos list.
 - Neutral, otherwise
- Measures for performance
 - ▶ Using # sentences **correctly** classified into a category
 - ① *Accuracy*: scaled by the total sentences in the testing sample
 - ② *Precision*: scaled by the total sentences classified into that category
 - ③ *Recall*: scaled by the total sentences researcher's label as that category
 - ④ *F₁ score*: Harmonic mean of *Precision* and *Recall*

3. Comparing Performance on Sentiment Classification

Comparing performance on the full training sample

	Overall (%)				Positive (358 sentences) Recall (%)	Neutral (458 sentences) Recall (%)	Negative (184 sentences) Recall (%)
	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>F₁score</i>			
FinBERT	88.2	87.2	88.5	87.8	88.5	87.3	89.7
BERT	85.0	83.6	85.1	84.2	86.3	84.1	84.8
LM dictionary	62.1	64.4	58.0	58.1	40.2	84.3	49.5
NB	73.6	71.8	70.7	71.1	75.4	78.4	58.2
SVM	72.6	71.0	68.8	69.6	68.7	82.8	54.9
RF	71.9	75.4	64.8	66.8	70.7	86.9	37.0
CNN	75.1	74.4	71.4	72.5	73.5	83.6	57.1
LSTM	76.3	74.8	72.5	73.3	79.3	82.1	56.0

Figure 3: NLP algorithms' performance in sentiment classification in the testing sample

3. Comparing Performance on Sentiment Classification

Comparing performance on the full training sample

	Researcher label			Row total
	Positive	Neutral	Negative	
FinBERT				
Positive	317 (88.3, 88.5)	36 (10.0, 7.9)	6 (1.7, 3.3)	359 (100, 35.9)
Neutral	28 (6.3, 7.8)	400 (90.7, 87.3)	13 (2.9, 7.1)	441 (100, 44.1)
Negative	13 (6.5, 3.6)	22 (11.0, 4.8)	165 (82.5, 89.7)	200 (100, 20.0)
BERT				
Positive	309 (85.6, 86.3)	42 (11.6, 9.2)	10 (2.8, 5.4)	361 (100, 36.1)
Neutral	33 (7.6, 9.2)	385 (88.3, 84.1)	18 (4.1, 9.8)	436 (100, 43.6)
Negative	16 (7.9, 4.5)	31 (15.3, 6.8)	156 (76.8, 84.8)	203 (100, 20.3)
LM dictionary				
Positive	144 (81.8, 40.2)	25 (14.2, 5.5)	7 (4.0, 3.8)	176 (100, 17.6)
Neutral	177 (27.3, 49.4)	386 (59.5, 84.3)	86 (13.3, 46.7)	649 (100, 64.9)
Negative	37 (21.1, 10.3)	47 (26.9, 10.3)	91 (52, 49.5)	175 (100, 17.5)

Figure 4: Confusion matrices of algorithms

3. Comparing Performance on Sentiment Classification

Comparing performance on the full training sample

	Researcher label			Row total
	Positive	Neutral	Negative	
SVM				
Positive	246 (74.8, 68.7)	51 (15.5, 11.1)	32 (9.7, 17.4)	329 (100, 32.9)
Neutral	85 (16.5, 23.7)	379 (73.6, 82.8)	51 (9.9, 27.7)	515 (100, 51.5)
Negative	27 (17.3, 7.5)	28 (17.9, 6.1)	101 (64.7, 54.9)	156 (100, 15.6)
RF				
Positive	253 (74.6, 70.7)	50 (14.7, 10.9)	36 (10.6, 19.6)	339 (100, 33.9)
Neutral	101 (17.4, 28.2)	398 (68.7, 86.9)	80 (13.8, 43.5)	579 (100, 57.9)
Negative	4 (4.9, 1.1)	10 (12.2, 2.2)	68 (82.9, 37.0)	82 (100, 8.2)
CNN				
Positive	263 (75.6, 73.5)	53 (15.2, 11.6)	32 (9.2, 17.4)	348 (100, 34.8)
Neutral	76 (15.0, 21.2)	383 (75.7, 83.6)	47 (9.3, 25.5)	506 (100, 50.6)
Negative	19 (13.0, 5.3)	22 (15.1, 4.8)	105 (71.9, 57.1)	146 (100, 14.6)
LSTM				
Positive	284 (74.9, 79.3)	57 (15.0, 12.4)	38 (10.0, 20.7)	379 (100, 37.9)
Neutral	55 (11.6, 15.4)	376 (79.3, 82.1)	43 (9.1, 23.4)	474 (100, 47.4)
Negative	19 (12.9, 5.3)	25 (17.0, 5.5)	103 (70.1, 56.0)	147 (100, 14.7)
Column total	358 (35.8, 100)	458 (45.8, 100)	184 (18.4, 100)	1,000 (100, 100)

Figure 5: Confusion matrices of algorithms

3. Comparing Performance on Sentiment Classification

Additional analyses comparing performance on sentiment classification

① Smaller training sample

- ▶ Whether LLMs and their domain adaption can effectively reduce costs from **manual labeling** of the training sample
- ▶ Using subsets of the full training and validation sample

② Randomized word orders

- ▶ Whether FinBERT's superior performance is by better summarizing **contextual** information in financial texts
- ▶ How much the NLP algorithms consider **syntactic** information and **logical** structure in text

3. Comparing Performance on Sentiment Classification

Performance with smaller training samples

Training sample	100%	90%	80%	70%	60%	50%	40%	30%	20%	10%	Accuracy diff. (100% – 10%)
	Accuracy in the testing sample (%)										
FinBERT	88.2	86.9	86.8	86.1	85.8	85.7	85.7	85.2	82.9	81.3	6.9
BERT	85.0	84.7	83.8	82.5	82.9	82.8	82.9	82.1	76.7	62.0	23.0
NB	73.6	71.5	71.1	69.4	68.5	67.4	67.3	68.2	66.9	61.6	12.0
SVM	72.6	71.4	72.3	71.5	70.6	68.6	66.9	63.1	60.4	58.2	14.4
RF	71.9	70.5	71.1	71.1	71.1	69.8	67.2	66.5	63.1	61.3	10.6
CNN	75.1	73.3	73.7	71.9	72.1	72.4	69.3	68.7	66.5	61.6	13.5
LSTM	76.3	77.9	77.0	77.6	74.9	75.4	75.5	71.9	68.1	57.8	18.5

Figure 6: Sentiment classification accuracy across sample sizes

3. Comparing Performance on Sentiment Classification

Performance with smaller training samples

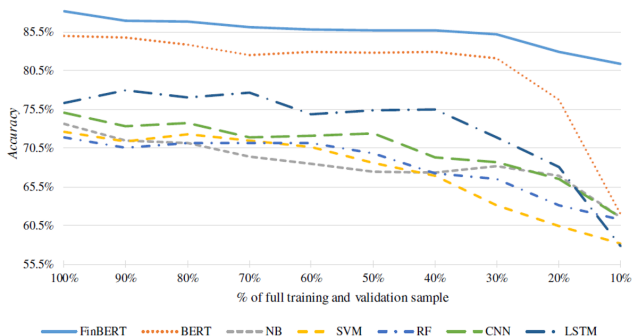


Figure 7: Sentiment classification accuracy across sample sizes

3. Comparing Performance on Sentiment Classification

Performance in sample with randomized word orders

	Overall performance in randomized-words sample (%)				Positive (358 sentences) Recall (%)	Neutral (458 sentences) Recall (%)	Negative (184 sentences) Recall (%)
	<i>Accuracy</i>	Δ <i>Accuracy</i>	<i>F₁ score</i>	Δ <i>F₁ score</i>			
FinBERT	76.9	-11.3	74.0	-13.8	67.0	93.7	54.3
BERT	70.2	-14.8	65.8	-18.4	56.4	91.9	42.9
NB	73.6	0.0	71.1	-0.0	75.4	78.4	58.2
SVM	70.9	-1.7	67.5	-2.1	64.0	85.2	48.9
RF	68.4	-3.5	60.1	-6.7	62.0	91.3	23.9
CNN	72.8	-2.3	69.8	-2.7	71.2	81.4	54.3
LSTM	74.8	-1.5	71.9	-1.4	76.3	81.4	55.4

Figure 8: Effect of word randomization on NLP algorithms' sentiment classification

3. Comparing Performance on Sentiment Classification

Performance in sample with randomized word orders

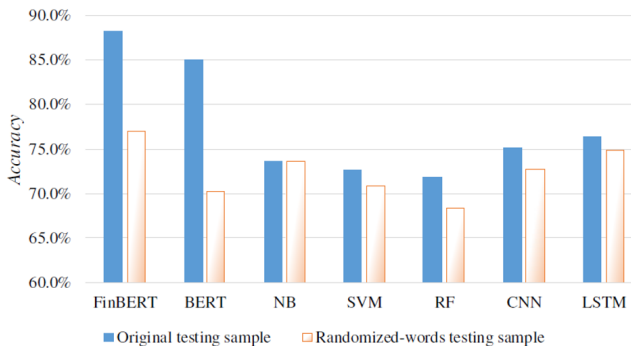


Figure 9: Effect of word randomization on NLP algorithms' sentiment classification

4. Comparing Performance in Labeling ESG-related Discussions

- Researchers can fine-tune a pretrained LLM for various tasks.
 - ▶ Here, labeling ESG-related discussions!
- How to construct a sample
 - ▶ Sentences from firms' CSR reports and MD&A sections of 10-Ks
 - ▶ Manually label sentences into one of the four categories: E, S, G, and non-ESG
 - ▶ In total, 2,000 sentences are used: 500 sentences each
 - ▶ 81% for training, 9% for validation, and 10% for testing

4. Comparing Performance in Labeling ESG-related Discussions

	Overall (%)				Environmental (50 sentences)	Social (50 sentences)	Governance (50 sentences)	Non-ESG (50 sentences)
	<i>Accuracy</i>	<i>Precision</i>	<i>Recall</i>	<i>F₁ score</i>	<i>Recall (%)</i>	<i>Recall (%)</i>	<i>Recall (%)</i>	<i>Recall (%)</i>
FinBERT	89.5	90.0	89.5	89.6	90.0	90.0	92.0	86.0
BERT	87.0	87.0	87.0	86.9	96.0	86.0	86.0	80.0
NB	80.0	80.1	80.0	80.0	78.0	76.0	88.0	78.0
SVM	75.0	75.3	75.0	75.0	78.0	66.0	74.0	82.0
RF	78.0	79.9	78.0	78.3	72.0	84.0	78.0	78.0
CNN	82.0	82.2	82.0	81.9	86.0	70.0	88.0	84.0
LSTM	86.0	86.1	86.0	86.0	90.0	82.0	86.0	86.0

Figure 10: Performance in classification of ESG-related discussion: Full sample

4. Comparing Performance in Labeling ESG-related Discussions

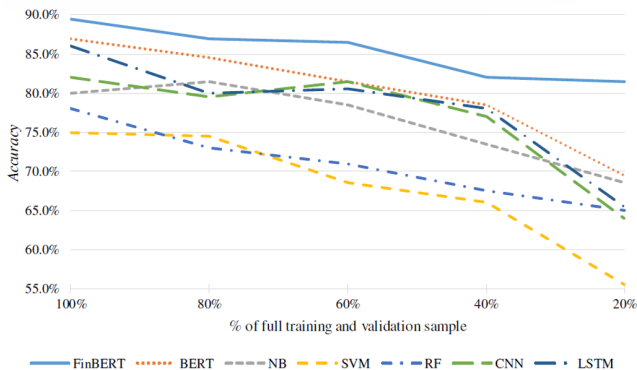


Figure 11: ESG classification accuracy across different sample sizes

5. Conclusion

- This paper examines whether a domain-specific LLM (FinBERT) improves the accuracy and efficiency of extracting sentiment and contextual information from financial texts compared to general NLP models and traditional algorithms.
- FinBERT **outperforms** other models in sentiment classification and ESG labeling tasks, especially in handling financial domain-specific vocabulary:
 - ▶ Even with smaller training sample sizes
 - ▶ In identifying negative sentiments
- FinBERT provides valuable tools for academic researchers, investment professionals, and regulators to better analyze and extract **insights** from financial texts, enabling more informed decision-making.

FinBERT

**** June 2, 2022: More fine-tuned FinBERT models available****

Visit [FinBERT.AI](#) for more details on the recent development of FinBERT.

We have fine-tuned FinBERT pretrained model on several financial NLP tasks, all outperforming traditional machine learning models, deep learning models, and fine-tuned BERT models. All the fine-tuned FinBERT models are publicly hosted at Huggingface 🤗. Specifically, we have the following:

- **FinBERT-Pretrained:** The pretrained FinBERT model on large-scale financial text. [link](#)
- **FinBERT-Sentiment:** for sentiment classification task. [link](#)
- **FinBERT-ESG:** for ESG classification task. [link](#)
- **FinBERT-FLS:** for forward-looking statement (FLS) classification task. [link](#)

In this Github repo,

- [FinBERT-demo.ipynb](#) demonstrates how to apply fine-tuned FinBERT model on specific NLP tasks.
- [finetune.ipynb](#) illustrates the process of fine-tuning FinBERT.

Figure 12: Intro to FinBERT

- Website: <https://github.com/yya518/FinBERT>

Appendix

Resources

How to use

You can use this model with Transformers pipeline for sentiment analysis.

```
from transformers import BertTokenizer, BertForSequenceClassification
from transformers import pipeline

finbert = BertForSequenceClassification.from_pretrained('yiyanghkust/finbert-tone')
tokenizer = BertTokenizer.from_pretrained('yiyanghkust/finbert-token')

nlp = pipeline("sentiment-analysis", model=finbert, tokenizer=tokenizer)

sentences = ["there is a shortage of capital, and we need extra financing",
             "growth is strong and we have plenty of liquidity",
             "there are doubts about our finances",
             "profits are flat"]
results = nlp(sentences)
print(results)  #LABEL_0: neutral; LABEL_1: positive; LABEL_2: negative
```

Figure 13: Coding example of using FinBERT