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

ALLEN H. HUANG, *HKUST*[†]

HUI WANG, *Renmin University of China*

YI YANG, *HKUST*

ABSTRACT

We develop FinBERT, a state-of-the-art large language model that adapts to the finance domain. We show that FinBERT incorporates finance knowledge and can better summarize contextual information in financial texts. Using a sample of researcher-labeled sentences from analyst reports, we document that FinBERT substantially outperforms the Loughran and McDonald dictionary and other machine learning algorithms, including naïve Bayes, support vector machine, random forest, convolutional neural network, and long short-term memory, in sentiment classification. Our results show that FinBERT excels in identifying the positive or negative sentiment of sentences that other algorithms mislabel as neutral, likely because it uses contextual information in financial text. We find that FinBERT's advantage over other algorithms, and Google's original bidirectional encoder representations from transformers model, is especially salient when the training sample size is small and in texts containing financial words not frequently used in general texts. FinBERT also outperforms other models in identifying discussions related to environment, social, and governance issues. Last, we show that other approaches underestimate the textual informativeness of earnings conference calls by at least 18% compared to FinBERT. Our results have implications for academic researchers, investment professionals, and financial market regulators.

Keywords: deep learning, large language model, transfer learning, interpretable machine learning, sentiment classification, environment, social, and governance (ESG)

* Accepted by Jenny Tucker. We thank Jenny Tucker (editor), three anonymous referees, Stefano Bonini (discussant), Sean Cao (discussant), Diego Garcia (discussant), Louise Hayes (discussant), Alan Huang (discussant), Fuwei Jiang (discussant), Jaehoon Lee (discussant), Xiangyu Li, Brandon Lock (discussant), Tim Loughran, Amy Zang, and seminar participants at AllianceBernstein, Aqumon, Central University of Finance and Economics, Ernst & Young, Hong Kong Monetary Authority, HKUST, JP Morgan, Lancaster University, and Seoul National University, as well as the 2020 Bergen Fintech Conference, 2020 Conference on Asia-Pacific Financial Markets, 2021 AAA Annual Meeting, 2021 EAA Conference, 2021 EFMA Conference, 2021 FARS Midyear Conference, 2021 FMA Annual Meeting, 2021 Hawai'i Accounting Research Conference, 2021 Japanese Accounting Review Conference, 2021 Summer Institute of Finance, 2022 Fifth Annual Data Science in Finance: Frontiers in Investment Data Science, 2022 CAPANA conference, and 2022 China International Risk Forum for their comments. We gratefully acknowledge the financial support of the Hong Kong Research Grants Council (T31-604/18-N and T31-603/21-N). This paper was formally circulated under the title "FinBERT—A Deep Learning Approach to Extracting Textual Information."

[†] Corresponding author.

Vol. 40 No. 2 (Summer 2023) pp. 806–841
doi:10.1111/1911-3846.12832

FinBERT : un grand modèle de langage pour l'extraction d'informations à partir de textes financiers

RÉSUMÉ

Les auteurs développent FinBERT, un grand modèle de langage innovateur qui s'adapte au domaine de la finance. Ils montrent que FinBERT intègre des connaissances financières et peut mieux résumer les informations contextuelles dans les textes financiers. À l'aide d'un échantillon de phrases classées par les chercheurs et provenant de rapports d'analystes, les auteurs démontrent que FinBERT surpasse considérablement le dictionnaire de Loughran et McDonald et d'autres algorithmes d'apprentissage automatique, y compris naïve Bayes, Support Vector Machine, la forêt aléatoire, le réseau de neurones convolutifs et la mémoire à long et court terme, dans la classification des sentiments. Leurs résultats montrent que FinBERT excelle dans l'identification du sentiment positif ou négatif dans des phrases déchiffrées à tort comme neutres par d'autres algorithmes, probablement parce que le modèle utilise les informations contextuelles issues du texte financier. Les auteurs constatent que l'avantage de FinBERT sur les autres algorithmes, et sur le modèle original BERT (*bidirectional encoder representations from transformers*) développé par Google, est particulièrement important lorsque la taille de l'échantillon d'entrainement est petite et dans les textes contenant des mots utilisés en finance qui ne sont pas fréquemment utilisés dans les textes généraux. FinBERT surpasse également les autres modèles dans l'identification des discussions liées aux questions environnementales, sociales et de gouvernance. Enfin, les auteurs démontrent que les autres approches sous-estiment l'information textuelle issue des conférences téléphoniques sur les résultats, et ce d'au moins 18 % comparativement à FinBERT. Leurs résultats ont des implications pour les chercheurs universitaires, les professionnels de l'investissement et les autorités de règlementation des marchés financiers.

Mots-clés : apprentissage profond, grand modèle de langage, apprentissage par transfert, apprentissage automatique interprétable, classification des sentiments, environnement, social et gouvernance (ESG)

1. Introduction

A burgeoning finance and accounting literature has explored the use of natural language processing (NLP) algorithms to analyze financial texts (see reviews by Li 2010b; Das 2014; Loughran and McDonald 2016; Gentzkow et al. 2019; Bochkay et al. 2023). Most, if not all, of these studies rely on NLP algorithms that assume a bag-of-words structure.[1] That is, the algorithms ignore context and instead analyze text as a collection of individual words treated independently without regard for grammar or word order (Wallach 2006).[2] Research in finance and accounting usually cites early NLP studies which find these algorithms to be as effective as other contemporaneous algorithms that incorporate the internal structure of a document (Manning and Schütze 1999; Friedman et al. 2001).

In recent years, computational linguistic researchers have introduced deep-learning-based NLP algorithms, such as the embeddings from language model (ELMo), open AI generative

[1] These algorithms include dictionary approaches (Loughran and McDonald 2011; Li et al. 2013); the naïve Bayes (NB) classifications (Li 2010a; A. H. Huang et al. 2014; Buehlmaier and Whited 2018); topic modeling algorithms, such as the latent Dirichlet allocation and supervised latent Dirichlet allocation (Lang and Stice-Lawrence 2015; A. H. Huang et al. 2018; Donovan et al. 2021; Frankel et al. 2022); support vector machine (SVM) and random forest (RF) models (Purda and Skillicorn 2015; Frankel et al. 2016, 2022; Manela and Moreira 2017; Donovan et al. 2021); and others that measure textual features such as readability (Li 2008; Loughran and McDonald 2014; Bonsall et al. 2017), salience or concreteness (Lundholm et al. 2014; Elliott et al. 2015; X. Huang et al. 2018), and similarity (Brown and Tucker 2011; Hoberg and Phillips 2016; Brown et al. 2018). Appendix 1 provides a glossary of the NLP algorithms and key terms used in this paper.

[2] Some of these algorithms can accommodate limited (local) context in immediately adjacent words by using bigrams and trigrams as inputs.

pretraining model (OpenAI GPT), and bidirectional encoder representations from transformers (BERT algorithm) (Peters et al. 2018; Radford et al. 2018; Devlin et al. 2019).[3] These algorithms, often referred to as large language models (LLMs) due to their large number of parameters (up to billions), learn semantic and syntactic relations among words from a large volume of texts, which allows them to consider context (e.g., other words in the same text and word sequences) in summarizing texts. Computational linguistic studies show that LLMs can substantially outperform other NLP algorithms in language translation, named entity recognition, and sentiment classification of general text (Devlin et al. 2019).

However, LLMs also have several disadvantages compared with simpler NLP approaches. First, the enormous number of parameters can be difficult and costly to train and use, requiring substantial amounts of textual data, computing resources (e.g., storage and high-capacity processors such as a graphic processing unit (GPU) or cloud servers), energy consumption, and training time. Even computational linguistic researchers question whether recently developed LLMs are too complex with too little improvement to justify their increased implementation costs (Strubell et al. 2019; Schwartz et al. 2020; Bender et al. 2021). Second, similar to other deep learning algorithms, LLMs are notoriously opaque, often referred to as "black boxes" (Castelvecchi 2016; Lauriola et al. 2022). These algorithms' general mechanisms are straightforward, and one can observe the trained model's parameters. However, the large number of parameters make it extremely difficult, if not impossible, to know precisely how the trained model turns inputs into outputs or the relative importance of inputs in determining the outputs. This low interpretability can be an obstacle for researchers who want to use these models to test economic theories (Loughran and McDonald 2016).

The primary objective of this study is to introduce FinBERT, a state-of-the-art LLM customized for financial texts based on Google's BERT algorithm, and compare its performance with that of the dictionary approach and other machine learning algorithms. Ex ante, it is unclear whether and how much FinBERT outperforms algorithms used in recent studies, such as support vector machine (SVM), random forest (RF), convolutional neural network (CNN), long short-term memory (LSTM, a type of recurrent neural network), and Google's original BERT model. Though the computational linguistic literature finds that LLMs achieve superior results over earlier algorithms, the findings do not assess financial texts written for professional investors, which can differ considerably from general texts in vocabulary and writing style. Moreover, these studies focus on tasks with more commercial appeal, such as answering questions, identifying entities, and translating language. They do not examine NLP algorithm performance on outcomes of interest to finance and accounting researchers, such as informativeness to investors. Whether and how much LLMs, especially one customized for financial text, outperform simpler algorithms in finance- and accounting-related tasks is thus an empirical question.

Recent finance and accounting studies have documented that machine learning algorithms such as LSTM, CNN, SVM, RF, and the BERT model perform well on tasks involving financial text (Azimi and Agrawal 2021; Brown et al. 2021; Siano and Wysocki 2021; Frankel et al. 2022). However, these studies do not compare LLMs with other deep learning algorithms. They also do not investigate whether an LLM customized for financial texts leads to further improvement or benefits.

Training an LLM usually includes two steps: pretraining and fine-tuning. To train FinBERT, we follow Google's BERT algorithm procedure. During pretraining, FinBERT learns semantic and syntactic information from a large corpus of unlabeled financial texts including corporate filings, analyst reports, and earnings conference call transcripts. During fine-tuning, FinBERT learns downstream NLP tasks. For example, for sentiment classification, the most representative NLP task in financial economics research, we leverage a data set of 10,000 sentences from analyst

[3] To avoid confusion, we refer to the machine learning NLP technique introduced by Devlin et al. (2019) as the "BERT algorithm" and Google's publicly released English-language BERT pretrained on Wikipedia and BookCorpus as the "BERT model" (see section 2 for details).

reports previously classified by researchers. We then compare FinBERT's performance to that of the Loughran-McDonald (LM) dictionary and other machine learning models (NB, SVM, RF, CNN, and LSTM). We find that FinBERT substantially outperforms all of them. Specifically, FinBERT's out-of-sample sentiment classification accuracy rate is 88.2%, whereas the LM dictionary, NB, SVM, RF, CNN, and LSTM rates are 62.1%, 73.6%, 72.6%, 71.9%, 75.1%, and 76.3%, respectively. We find similar patterns using other performance measures, including precision, recall, and F_1 score.[4]

Further analyses show a distinct advantage in FinBERT's detection of negative sentiments (89.7% accuracy, compared to less than 60% accuracy for non-BERT algorithms), a point worth noting given that negative sentiments have a larger impact on investors than positive ones (Loughran and McDonald 2011; A. H. Huang et al. 2014). FinBERT's superior results over the deep learning algorithms CNN and LSTM suggest that unsupervised pretraining of LLM, though costly, provides substantial benefits. We also find that FinBERT classifies sentiments more accurately than the BERT model (85.0% accuracy), consistent with the incorporating of finance domain knowledge improving LLM performance.

Manually labeling training samples imposes considerable costs. We therefore explore how NLP models perform when using smaller training samples. We find that FinBERT retains its high accuracy even when using small training samples, whereas the other machine learning algorithms experience performance declines. For example, FinBERT retains 81.3% accuracy using only 10% of the training sample, which is higher than the two best-performing non-BERT-algorithm models (LSTM and CNN) using the full training sample (76.3% and 75.1%, respectively). More remarkably, the BERT model is 76.7% accurate when trained on a 20% subset but only 62.0% accurate when using a 10% subset, a sizable decline of 14.7%. Taken together, the results indicate that pretraining LLMs on domain-specific texts helps achieve state-of-the-art performance even when using small training samples, likely due to learning relevant semantic and syntactic information from pretraining texts.

We conduct two further analyses exploring the source of FinBERT's better performance. First, we examine the algorithms' performance when we randomize word sequence in sentences. We find that FinBERT's accuracy deteriorates considerably by 11.3% but that non-BERT algorithms, including those based on deep learning, either remain constant (NB) or drop slightly (less than 3.5% for SVM, RF, CNN and LSTM), consistent with FinBERT's main advantage over non-BERT algorithms being its ability to process contextual information.

Second, we investigate how differences in the pretraining texts contribute to FinBERT's advantage over the BERT model by exploring the role of finance vocabulary—that is, words and subwords that frequently appear in financial text but not in general text. We begin by using an interpretable model to identify which word in a sentence is the most important word for FinBERT to classify the sentence's sentiment. We find that such words are more likely to belong to the finance vocabulary among sentences that FinBERT correctly classifies, than among sentences that FinBERT incorrectly classifies, and that this discrepancy is largest in negative sentences, which our prior results show that FinBERT especially excels at classifying. Next, we sort the testing sample sentences into two groups based on their proportion of finance vocabulary and find that FinBERT's advantage over BERT is larger for the group with a higher proportion of finance vocabulary. Last, we show that the finance vocabulary helps FinBERT retain its performance (i.e., reduce its accuracy deterioration) when the training sample becomes smaller. In sum, results from the three tests suggest that FinBERT's outperformance is at least partially due to its familiarity with finance vocabulary.

LLMs can be fine-tuned using a relatively small sample of labeled texts to achieve high accuracy in NLP tasks. To corroborate our main analyses based on sentiment classification and further

[4] F_1 score is the harmonic mean of precision and recall, that is, 2 × (precision × recall) / (precision + recall).

demonstrate how to use FinBERT, we conduct an additional fine-tuning task of interest to researchers, practitioners, and regulators: labeling environment, social, and governance (ESG) discussions. Specifically, we label 2,000 sentences from firms' corporate social responsibility (CSR) reports and management discussion and analyses (MD&As) for this task. We find that FinBERT achieves 89.5% accuracy in labeling ESG discussions, higher than non-BERT algorithms. We also find that, consistent with results from previous analyses, FinBERT's advantage over other algorithms increases as the training sample size decreases.

Last, we examine whether FinBERT's superior performance in sentiment classification in analyst reports extends to other financial texts and quantify its economic significance. Specifically, we measure the sentiments of 28,873 earnings conference call transcripts from 2003 to 2020 using FinBERT and other NLP approaches and estimate associations with their respective market reactions. We use these associations as a noisier measure of algorithm performance because they capture both the accuracy of NLP algorithms' sentiment classification and how the market should react to earnings conference call sentiments. We find that although market reactions are positively associated with textual sentiments, as measured by all NLP algorithms, the economic significance of this association is greatest for sentiments measured by FinBERT. The other algorithms underestimate the economic magnitude by at least 18.1% (LSTM) and up to 48.6% (RF). Using both the Vuong (1989) test and the bootstrapping technique, we confirm that the model using textual sentiments generated by FinBERT has greater explanatory power than other models. These results are consistent with FinBERT better capturing the sentiment of earnings conference calls and supplement our main analyses based on the researcher-labeled analyst report sample.

This study primarily contributes to the growing body of literature that uses textual analysis in financial economics (for a recent review, see Bochkay et al. 2023). We use the pretraining procedure proposed by Google and a large corpus of financial texts to develop FinBERT, an LLM specific to the finance domain. We document that FinBERT outperforms other NLP algorithms in the most widely used tasks for analyzing financial texts (i.e., sentiment classification) and in a task of interest to financial economic researchers (ESG classification). Although other deep learning NLP algorithms, such as RF and LSTM, can dominate domain-specific word lists in analyzing financial texts, they underperform FinBERT, even with 10 times the training sample size. Our results using a smaller training sample have practical implications for practitioners and researchers because manual labeling of training samples is arguably one of the costliest tasks in supervised machine learning.

Our study also adds to the computational linguistic literature by documenting that FinBERT, a domain-adapted LLM, outperforms Google's BERT model in analyzing financial texts. Our results suggest that pretraining with domain-specific texts can improve LLM performance, especially when the fine-tuning sample is small, and that such improvement is likely due to domain-adapted LLMs' familiarity with words not frequently used in general texts. Our evidence from two classification tasks (sentiment and ESG) shows that LLMs perform well on domain-specific tasks that typical computational linguistic studies do not focus on.

Academic researchers, investment professionals, and financial market regulators are increasingly using NLP algorithms to extract insights from financial texts. Our finding that LLMs that incorporate finance domain-specific knowledge summarize textual information more accurately than other NLP algorithms has practical implications for these users. As textual information is notoriously difficult to measure and measurement errors can lead to spurious results, including both type I and type II errors, we suggest that researchers consider using LLMs for textual analysis. In addition to fine-tuning on researcher-labeled sentences, as in our analyses, researchers can fine-tune FinBERT on firm fundamentals (e.g., earnings, revenue) and capital market variables (e.g., stock price, returns, volatilities) to identify how text maps to valuation-relevant information and actions. We leave these explorations to future research. In addition, we recommend that researchers pay attention to developments in computational linguistic research as newer NLP algorithms will undoubtedly lead to further improvements in the accuracy, reliability, and efficiency of extracting textual information, which can help researchers design and conduct new tests of economic theories.

To facilitate adoption of FinBERT, we publicly post the following: (i) the source code to pretrain FinBERT, (ii) the pretrained FinBERT, and (iii) the FinBERT fine-tuned for sentiment classification and ESG-related discussion. We also include a tutorial on how to use FinBERT for sentiment classification in online Appendix A.[5]

2. Deep learning NLP algorithms

Neural network and deep learning algorithms

Deep learning algorithms belong to a class of machine learning algorithms called neural networks, which have an input layer and output layer connected by one or more hidden layers. The input layer takes in the initial raw data (e.g., text in NLP tasks) and outputs it to the hidden layer(s), which processes the data using nonlinear functions with certain parameters and then feeds it to the next layer. The output layer obtains data from the last hidden layer and presents the prediction outcome (e.g., positive, neutral, or negative for sentiment classification). Neural network algorithms usually specify a nonlinear function and use training data to find parameters that minimize prediction errors (i.e., the difference between the prediction from the output layer and the actual value). A logistic regression can be considered the simplest neural network with only input and output layers (i.e., no hidden layer) and a sigmoid activation function.

Neural networks with only one hidden layer are shallow, whereas those with multiple hidden layers are deep (LeCun et al. 2015). Deep neural networks (deep learning) can capture more complex relations than shallow ones, but they require more parameters to estimate and thus more training data. Several recent studies in finance and accounting show that deep learning algorithms (e.g., CNN, LSTM, and the BERT model) can outperform general and domain-specific (e.g., LM) dictionaries and simple machine learning models (e.g., NB) in processing financial text (Azimi and Agrawal 2021; Brown et al. 2021; De la Parra 2021; Siano and Wysocki 2021).

Neural word embedding

Machine learning NLP algorithms represent text using vectors and use these vectors as inputs for downstream tasks. Simple NLP approaches, such as NB, SVM, and RF, usually use one-hot encoding, which assumes that words are independent of each other (i.e., each pair of words has the same distance in the vector space).[6] In the early 2010s, NLP researchers introduced neural word embedding (also known as word representations) using vectors with fewer dimensions than one-hot encoding (e.g., word2vec by Mikolov et al. 2013). Intuitively, neural word embedding determines each word's vector representation by learning their similarities (i.e., word relations) from co-occurrences in texts.[7] As these algorithms learn from unlabeled texts, they are unsupervised (or self-supervised) machine learning algorithms.

Neural word embedding improves on one-hot encoding in two ways. First and most importantly, it learns relations between words from a corpus such that its vectors capture both semantic and syntactic information. For example, in word2vec, "Japan" and "sushi" are closer in the vector

[5] Please see supporting information, "Appendix A," as online supplemental material.

[6] One-hot encoding records whether a word exists in a document. Each document is represented by a $V \times 1$-dimensional vector, where V is the number of distinct words in the entire corpus's vocabulary (or a predefined word list) and each element in the vector corresponds to a unique word. If a word exists in a document, the vector that represents the document takes a value of one in the element that corresponds to that word and zero otherwise.

[7] Specifically, neural word embedding is based on language models that predict the probability of a word's occurrence given its context (surrounding words and sentences). Neural word embedding models differ in their training's objective functions. For example, the objective function in one version of Word2Vec (Skip-Gram model) is to predict the words immediately adjacent to a particular target word.

space than "Japan" and "pizza" (Mikolov et al. 2013). Second, compared with one-hot encoding, which results in vectors with dimensions of tens of thousands, neural word embeddings usually produce much smaller vectors of a few hundred dimensions, thereby reducing the number of parameters that the machine learning algorithms must estimate and improving their performance.

Large language models

Earlier neural word embedding models, including word2vec, GloVe, and Facebook's fastText, learn similarities of words based on surrounding words in the training text. However, when researchers use word embeddings from these models, each word is represented by a static vector regardless of actual surrounding words in the text of NLP tasks. In recent years, NLP researchers have introduced LLMs including ELMo, OpenAI GPT, and the BERT model, which have contextualized embeddings (i.e., they represent words using different vectors depending on their context). For example, in the BERT model, the word "bank" has different output vectors for the two sentences "I went to the bank to deposit a check" and "We walk along the river's bank." Indeed, recent studies confirm that the BERT model's output vectors encode both syntactic knowledge (including parts of speech) and semantic knowledge (see the review by Rogers et al. 2020). Incorporating context into representing texts has substantially improved NLP task performance (Peters et al. 2018; Devlin et al. 2019).

LLMs require a massive amount of data and time to train. As most texts contain some general semantic and syntactic information, such as grammar and common word meanings, LLMs can be pretrained on a large corpus of unlabeled text to learn these types of general information (unsupervised learning). Researchers who want to use LLMs for specific NLP tasks can fine-tune a pretrained LLM using a much smaller training sample (supervised learning). This two-step process, also referred to as transfer learning, can achieve high performance without incurring substantial implementation costs. We use financial texts to pretrain and fine-tune FinBERT, an LLM based on Google's BERT algorithm.

FinBERT: A finance domain-adapted LLM based on Google's BERT algorithm

In the pretraining step, the BERT algorithm uses two training objectives, the masked language model and next sentence prediction, to learn the relations between words in a text (see Appendix 2 for details). Google pretrained the BERT model on general text, including Wikipedia and BookCorpus, including 3.3 billion tokens, or words and subwords (Devlin et al. 2019).[8] In the fine-tuning step, the BERT algorithm can accommodate various NLP tasks. We focus on single sentence classification because it includes sentiment classification and other tasks popular in financial economics.[9]

As the BERT model is pretrained with general texts, it may not work well for processing financial texts written for professional investors, which can differ considerably in vocabulary and writing style. For example, the BERT model's output vector for the word "allowance" may encode multiple meanings, such as the money given by parents to a child or an amount of something that someone is allowed to have, whereas in financial texts, "allowance" typically refers to an amount planned for a future cost. We thus use financial texts in the pretraining to create a context-specific vector space (e.g., "bank" is learned as more closely related to "lending" rather than "fish" or "river"). Recent studies show that LLMs pretrained on biomedical and computer science domain texts perform better at NLP tasks in these domains than Google's BERT model (Alsentzer et al. 2019; Beltagy et al. 2019; Lee et al. 2019).

[8] This pretraining took four days to process on 16 Cloud Tensor Processing Units (AI accelerator application-specific integrated circuits developed specifically for neural network algorithms) (Devlin et al. 2019).

[9] Other fine-tuning NLP tasks include sentence pair classification (e.g., determining whether the second sentence contradicts the first or if they are semantically similar), question answering (e.g., selecting a sentence that best answers the question), and single sentence tagging (e.g., recognizing name entities).

We pretrain FinBERT using three types of financial texts: (i) corporate annual and quarterly filings—60,490 10-Ks and 142,622 10-Qs of Russell 3000 firms between 1994 and 2019 from the SEC's EDGAR website; (ii) financial analyst reports—476,633 reports issued for S&P 500 firms between 2003 and 2012 from the Thomson Investext database; and (iii) earnings conference call transcripts—136,578 transcripts of 7,740 public firms between 2004 and 2019 from the SeekingAlpha website.[10]

We choose these texts for three reasons. First, they are the most representative and frequently used types of corporate financial communications in financial economics research (Li 2010b; Loughran and McDonald 2016; Bochkay et al. 2023). Including them in pretraining helps FinBERT analyze textual information most relevant to research in these areas. Second, they include comments from important stakeholders in financial communications (e.g., managers, financial analysts). Prior studies have consistently documented that such information is informative to investors (Li 2010a; A. H. Huang et al. 2014, 2022). Third, because these texts are subject to different regulatory requirements and have different intended audiences and disclosure media, including them in the pretraining stage can expose FinBERT to diverse financial topics and communication styles.

In total, our sample includes 4.9 billion tokens, around 50% larger than the 3.3 billion tokens that Google used to pretrain the BERT model. Following prior domain adaptation of BERT, we pretrain FinBERT with the same configuration as the $\text{BERT}_{\text{BASE}}$ model using Google's code. We use an Nvidia DGX-1 server with four Tesla P100 GPUs and 128 gigabytes of GPU memory and Horovod, Uber's distributed training framework. Overall, pretraining FinBERT takes approximately two days (see Appendix 2 for more details).

3. Comparing performance on sentiment classification

We first compare sentiment classification for FinBERT, the LM dictionary, and NLP algorithms (e.g., NB, SVM, RF, CNN, and LSTM). NB assigns textual documents to the most likely category based on the "naïve" assumption that words are independent of each other, given a document's category (for more details, see Li 2010a; A. H. Huang et al. 2014). SVM constructs hyperplanes in the vector space to separate categories (see Frankel et al. 2016; Manela and Moreira 2017). RF combines the outputs of multiple decision trees into a single result (see Frankel et al. 2022). CNN and LSTM are deep neural network models that capture complex relations between input and output, and LSTM also processes sequential data (see Azimi and Agrawal 2021; Brown et al. 2021). We discuss how we implement non-BERT algorithms in Appendix 3.

To ensure comparability, we use the same training and testing samples for different machine learning algorithms. Specifically, we use the financial analyst reports with researcher-labeled sentiments. An alternative approach to using researcher labels is to use noisier, "naturally occurring" labels to train the algorithms, often referred to as weakly supervised learning. For example, Frankel et al. (2022) and Siano and Wysocki (2021) use firm fundamentals and stock returns as sentiment labels for 10-K and earnings announcement texts to train machine learning algorithms. This weakly supervised learning avoids the cost of manual labeling but sacrifices accuracy because operating performance and stock returns are affected by information and events not in the text and the text may include information not contained in the current period's operation or stock returns. As we are primarily interested in comparing NLP algorithms' performance on sentiment classification, we use the researcher-labeled approach to train the algorithms.

[10] We parse the 10-Ks and 10-Qs into different items and include only those items most relevant for investors that contain the fewest tables. We include three sections from the 10-Ks, Item 1 (Business), Item 1A (Risk Factors), and Item 7 (Management's Discussion and Analysis), and two from the 10-Qs, Item 1A (Risk Factors) and Item 2 (Management's Discussion and Analysis).

We classify analyst report text sentiments at the sentence level for two reasons. First and most importantly, the sentence is the natural unit in language for expressing an opinion (Ivers 1991; Li 2010a; A. H. Huang et al. 2014). Second, the BERT algorithm only takes input of 512 tokens, which limits the input text size. The sample includes 10,000 sentences, of which 3,577 are positive, 4,586 are neutral, and 1,837 are negative.[11]

Comparing performance on the full training sample

To fine-tune FinBERT and the BERT model and train the NB, SVM, RF, CNN, and LSTM models, we randomly split the full sample into three parts comprising 81% for training (8,100 sentences), 9% for validation (900 sentences), and the remaining 10% (1,000 sentences) for testing.[12] We use the training sample to fit the parameters of the machine learning models, the validation sample to set the hyperparameters of the models and monitor model performance, and the testing sample to evaluate the final model.[13] For all machine learning algorithms, we assign each sentence to the sentiment category with the highest likelihood predicted by the algorithm. For the LM dictionary, we define a sentence as negative if it contains at least one word from the LM Fin-Neg list, positive if it does not contain any word from the LM Fin-Neg list and contains at least one word from the LM Fin-Pos list, and neutral otherwise.[14]

In panel A of Table 1, we tabulate the sentiment classification performance of the NLP algorithms in the testing sample. For each algorithm, we report its overall *Accuracy* (the number of sentences correctly classified scaled by the total number of sentences in the testing sample), mean *Precision* (the number of sentences correctly classified into a category scaled by the total number of sentences classified into that category), mean *Recall* (the number of sentences correctly classified into a category scaled by the number of sentences researchers label as that category), and mean F_1 *score* (the harmonic mean of *Precision* and *Recall*). We also separately report *Recall* for sentences in each sentiment category as labeled by the researcher.[15]

The results show that FinBERT performs best (highest *Accuracy* of 88.2% and F_1 *score* of 87.8%), followed by BERT (*Accuracy* of 85.0% and F_1 *score* of 84.2%).[16] Among the remaining approaches, the two other deep learning algorithms (CNN and LSTM, with *Accuracy* of 75.1% and 76.3%, respectively, and F_1 *score* of 72.5% and 73.3%, respectively) outperform the machine learning algorithms NB, SVM, and RF (*Accuracy* of 73.6%, 72.6%, and 71.9%, respectively; F_1 *score* of 71.1%, 69.6%, and 66.8%, respectively). We also find that machine learning algorithms outperform the LM dictionary in sentiment classification (62.1% *Accuracy* and 58.1% F_1 *score*).[17]

[11] We obtain this sample from A. H. Huang et al. (2014), in which the authors read sentences and label their sentiments. Although these sentences are also included in the pretraining text of FinBERT, it is unlikely to introduce bias that favors FinBERT in sentiment classification because the pretraining sample does not include sentiment labels. Nonetheless, in a robustness test (untabulated), we document that the performance gap between FinBERT and the BERT model in the Financial PhraseBank (a public data set for financial sentiment classification that does not overlap with FinBERT's pretraining data, Malo et al. 2014) is similar to that reported in our main analyses.

[12] For brevity, we refer to the BERT model fine-tuned for NLP tasks as BERT in the following sections.

[13] We discuss the hyperparameters for the FinBERT and BERT models in Appendix 2 and those of NB, SVM, RF, CNN, and LSTM in Appendix 3.

[14] In a sensitivity test (untabulated), we define a sentence's LM sentiment as negative when it has more negative than positive words, positive when it has more positive than negative words, and neutral otherwise. We find almost identical results (*Accuracy* of 62.2% vs. 62.1% in the testing sample, Table 1, panel A).

[15] We do not report *Precision* and F_1 *score* for each sentiment category separately because they use sentences not labeled in the category by researchers as denominators. We report the full confusion matrices in Table 1, panel B.

[16] In a robustness test (untabulated), we use 10-fold cross-validation following Li (2010a) and find that FinBERT on average outperforms BERT by 3.4%.

[17] In an additional test (untabulated), we find that supervised latent Dirichlet allocation achieves *Accuracy* (F_1 *score*) of 59.4% (43.1%) in the testing sample, underperforming SVM and RF.

TABLE 1
NLP algorithms' performance in sentiment classification in the testing sample

Panel A: Sentiment classification performance of FinBERT, BERT, LM dictionary, NB, SVM, RF, CNN, and LSTM

	Overall (%)				Positive (358 sentences)	Neutral (458 sentences)	Negative (184 sentences)
	Accuracy	*Precision*	*Recall*	F_1*score*	*Recall* (%)	*Recall* (%)	*Recall* (%)
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

Panel B: Confusion matrices of FinBERT, BERT, LM dictionary, NB, SVM, RF, CNN, and LSTM

	Researcher label			
	Positive	Neutral	Negative	Row total
FinBERT				
Positive	**317**	36	6	359
	(88.3, 88.5)	(10.0, 7.9)	(1.7, 3.3)	(100, 35.9)
Neutral	28	**400**	13	441
	(6.3, 7.8)	**(90.7, 87.3)**	(2.9, 7.1)	(100, 44.1)
Negative	13	22	**165**	200
	(6.5, 3.6)	(11.0, 4.8)	**(82.5, 89.7)**	(100, 20.0)
BERT				
Positive	**309**	42	10	361
	(85.6, 86.3)	(11.6, 9.2)	(2.8, 5.4)	(100, 36.1)
Neutral	33	**385**	18	436
	(7.6, 9.2)	**(88.3, 84.1)**	(4.1, 9.8)	(100, 43.6)
Negative	16	31	**156**	203
	(7.9, 4.5)	(15.3, 6.8)	**(76.8, 84.8)**	(100, 20.3)
LM dictionary				
Positive	**144**	25	7	176
	(81.8, 40.2)	(14.2, 5.5)	(4.0, 3.8)	(100, 17.6)
Neutral	177	**386**	86	649
	(27.3, 49.4)	**(59.5, 84.3)**	(13.3, 46.7)	(100, 64.9)
Negative	37	47	**91**	175
	(21.1, 10.3)	(26.9, 10.3)	**(52, 49.5)**	(100, 17.5)
NB				
Positive	**270**	67	41	378
	(71.4, 75.4)	(17.7, 14.6)	(10.8, 22.3)	(100, 37.8)
Neutral	64	**359**	36	459
	(13.9, 17.9)	**(78.2, 78.4)**	(7.8, 19.6)	(100, 45.9)
Negative	24	32	**107**	163
	(14.7, 6.7)	(19.6, 7.0)	**(65.6, 58.2)**	(100, 16.3)
Column total	358	458	184	1,000
	(35.8, 100)	(45.8, 100)	(18.4, 100)	(100, 100)

(The table is continued on the next page.)

TABLE 1 (continued)

Panel B: Confusion matrices of FinBERT, BERT, LM dictionary, NB, SVM, RF, CNN, and LSTM

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

Notes: This table tabulates the sentiment classification performance of NLP algorithms in the testing sample. The testing sample contains 1,000 sentences, randomly selected from a sample of 10,000 sentences from financial analyst reports, with sentiments labeled by researchers (358 positive sentences, 458 neutral sentences, and 184 negative sentences). Panel A reports each NLP algorithm's overall *Accuracy*, *Precision*, *Recall*, F_1 *score*, and its *Recall* in each sentiment category. *Accuracy* is the number of correctly classified sentences divided by the total number of sentences in the testing sample. The overall *Precision*, *Recall*, and F_1 *score* are the means of the *Precision, Recall*, and F_1 *score* in the three sentiment categories. For each sentiment category, *Precision* equals the number of sentences that are correctly classified into that category divided by the number of sentences classified into that category by the algorithm; *Recall* equals the number of sentences that are correctly classified into that category divided by the number of sentences classified into that category by researchers; and the F_1 *score* equals the harmonic mean of *Precision* and *Recall*, $F_1\,score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$. Panel B tabulates the sentiment classification of the testing sample by each NLP algorithm and the researchers in 3×3 metrics. Each cell reports the number of sentences with the corresponding label of the algorithm and of the researchers. In the parentheses in each cell, the first (second) number reports the percentage of all sentences in the row (column) in that cell. Cells in bold indicate correctly classified sentences (i.e., the algorithm assigns the same classification as the researchers).

TABLE 2
NLP algorithms' performance in sentiment classification across different training sample sizes

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

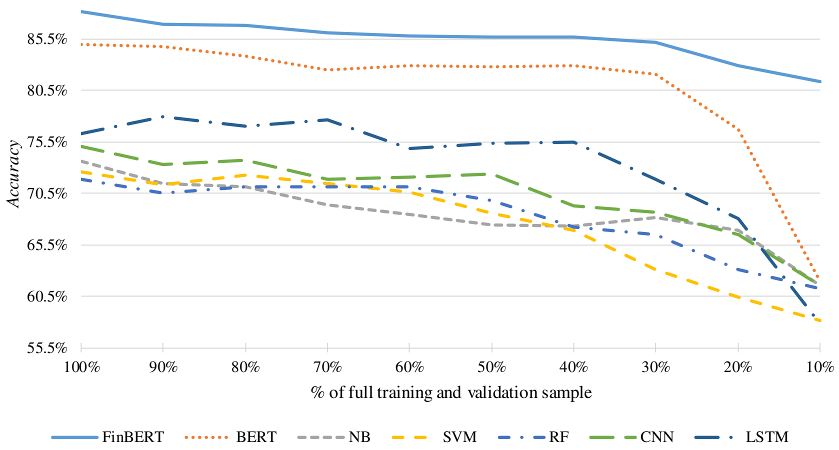
Notes: This table presents the sentiment classification accuracy rates of NLP algorithms using different training sample size. We use the full sample (100%) and subsets (90%, 80%, 70%, 60%, 50%, 40%, 30%, 20%, and 10%). For example, at 100% (10%), we use 8,100 (810) as the training sample and 900 (90) as the validation sample. We use a constant testing sample size of 1,000 sentences, same as Table 1. See notes to Table 1 for variable definitions.

Performance breakdown by sentiment type further demonstrates FinBERT's advantage. FinBERT performs well across researcher-labeled positive, neutral, and negative sentences (*Recall* of 88.5%, 87.3%, and 89.7%, respectively). While BERT also produces stable performance across the three sentiment types (*Recall* of 86.3%, 84.1%, and 84.8%, respectively), it underperforms FinBERT in all three sentiment types, especially negative sentences (89.7% − 84.8% = 4.9% lower). The non-BERT algorithms are far less consistent, showing satisfactory performance for neutral sentences, with *Recall* ranging from 82.1% to 84.3% (except for NB's 78.4% and RF's 86.9%), but notably lower *Recall* for positive and negative sentences (ranging from 37% to 79.3%). Non-BERT machine learning algorithms also have substantially weaker performance for negative versus positive sentences, with *Recall* differences of 17.2%, 13.8%, 33.7%, 16.4%, and 23.3% for NB, SVM, RF, CNN, and LSTM, respectively. FinBERT's large advantage over other NLP algorithms in classifying negative sentences (ranging from 31.5% to 52.7% for non-BERT algorithms and 4.9% for BERT) is important because prior studies document that investors often perceive negative information as more informative than positive information (Loughran and McDonald 2011; A. H. Huang et al. 2014).

In an additional analysis, we explore whether FinBERT's advantage over BERT widens when there are more negative sentences in the training and testing sample. Recall that in our main sample, the proportion of sentences researchers label as positive, neutral, and negative are 36%, 46%, and 18%, respectively. In this analysis, we form a sample using equal numbers of positive, neutral, and negative sentences from the full sample. The sample comprises only 40% of the full sample due to the limited number of negative sentences. Using this sample to train the algorithms, we find that FinBERT's *Accuracy* is 85.8% and BERT's is 80.7% (untabulated). As benchmarks, FinBERT's and BERT's *Accuracy* in the 40% sample with the same proportions as the full sample is 85.7% and 82.9%, respectively (reported in Table 2). That is, FinBERT's advantage over BERT increases from 2.8% to 5.1% when there are more negative sentences in the sample.

In Table 1, panel B, we tabulate the full confusion matrices of each NLP algorithm, shedding further light on their performance. As shown, both FinBERT and BERT have low levels of type I and type II errors, and FinBERT performs consistently better than BERT. Non-BERT machine learning algorithms, including deep learning algorithms, often incorrectly label positive and negative sentences as neutral. For example, SVM, RF, and CNN incorrectly label 27.7%, 43.5%, and

Figure 1 Sentiment classification accuracy across sample sizes



Notes: This figure plots the sentiment classification accuracy rates of NLP algorithms using different training sample sizes from full (100%) to subsets of 90%, 80%, 70%, 60%, 50%, 40%, 30%, 20%, and 10%. For example, at 100% (10%), we use 8,100 (810) as the training sample and 900 (90) as the validation sample. We use a constant testing sample size of 1,000 sentences, same as in Table 1. See notes to Table 1 for variable definitions.

25.5% of negative sentences as neutral, respectively. Even the most accurate non-BERT algorithm, LSTM, mislabels 23.4% of negative sentences as neutral.[18]

Taken together, these results suggest four main conclusions. First, machine learning algorithms substantially outperform domain-specific finance dictionaries in identifying sentiment. Second, among machine learning algorithms, the deep learning models (FinBERT, BERT, CNN, and LSTM) perform better than non-deep-learning models (NB, SVM, and RF). Third, LLMs (FinBERT and BERT) outperform earlier deep learning models (CNN and LSTM), especially in identifying sentences with negative sentiments. Last, LLMs that incorporate domain-specific knowledge (FinBERT) outperform those that rely on generic texts (BERT).

Additional analyses comparing performance on sentiment classification

Performance with smaller training samples

One of the largest costs of training a supervised machine learning model is manual labeling of the training sample.[19] To assess whether LLMs and their domain adaptation can effectively reduce such costs, we examine how training sample size affects the performance of machine learning models. Empirically, we use subsets (i.e., 90%, 80%, . . ., 10%) of the full training and

[18] For the LM dictionary, we observe that although it has a low rate of type I errors (false positive) for positive sentiments (18.2%), it fails to identify more than half of the sentences with positive and negative sentiments (59.7% and 50.5%, respectively). Instead, the LM dictionary tends to label these sentences as neutral (49.4% and 46.7%, respectively). We list some examples of sentences correctly labeled by FinBERT but mislabeled by the LM dictionary in online Appendix B.

[19] Assembling a training sample includes extracting and preparing text, hiring labelers and instructing them, and the actual labeling task. In supervised machine learning, researchers often undergo a reiterative process, during which they label data, investigate data quality and class imbalances, calibrate the model, and assess the model's performance. Requiring a smaller training sample likely reduces the number of data labeling cycles and the costs associated with all these tasks. See Bochkay et al. (2023) for a detailed discussion on the implementation guideline of training data for supervised machine learning.

TABLE 3
Effect of word randomization on NLP algorithms' performance in sentiment classification

	Overall performance in randomized-words sample (%)				Positive (358 sentences)	Neutral (458 sentences)	Negative (184 sentences)
	Accuracy	Δ*Accuracy*	F_1 *score*	ΔF_1 *score*	*Recall* (%)	*Recall* (%)	*Recall* (%)
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

Notes: This table reports the classification performance of different NLP algorithms in the testing sample with words randomized. The testing sample contains 1,000 sentences, randomly selected from a sample of 10,000 sentences from financial analyst reports, with sentiments labeled by researchers (358 positive sentences, 458 neutral sentences, and 184 negative sentences). For each NLP algorithm, we report its overall *Accuracy* (F_1 *score*) in the randomized-words testing sample and the decrease in overall *Accuracy* (F_1 *score*) compared to non-randomized-words testing sample (from Table 1, panel A), as well as its *Recall* in each sentiment category in the randomized-words testing sample. See notes to Table 1 for variable definitions.

validation sample (hereafter, training sample for brevity) to train the models and compare their performance in a full-size testing sample containing 1,000 sentences.[20]

The results, tabulated in Table 2 and plotted in Figure 1, show that FinBERT performs consistently well with smaller training samples, but the other machine learning algorithms perform substantially worse. Reducing the training sample size from 100% to 10% reduces FinBERT's *Accuracy* by only 6.9%, whereas it reduces the *Accuracy* of BERT, NB, SVM, RF, CNN, and LSTM by 23.0%, 12.0%, 14.4%, 10.6%, 13.5%, and 18.5%, respectively. At 10% of the training sample, none of the non-FinBERT algorithms outperform the LM dictionary. Moreover, FinBERT's *Accuracy* using only 10% of the training sample (81.3%) is higher than the best-performing non-BERT-algorithm model, LSTM, using a full training sample (76.3%). We also observe that although BERT performs well when the training sample is at least 20% (76.7%, which is higher than non-BERT algorithms using a full training sample), its *Accuracy* deteriorates by 14.7% in the 10% training sample (62.0%). Untabulated summary statistics show that BERT demonstrates especially low performance with the 10% training sample on researcher-labeled positive and negative sentences; *Recall* of 56.4% and 2.2%, respectively, compared to 90.4% for neutral sentences. In comparison, FinBERT performs consistently well across researcher-labeled positive, neutral, and negative sentences (*Recall* of 79.6%, 83.8%, and 78.3%, respectively) with the 10% training sample.

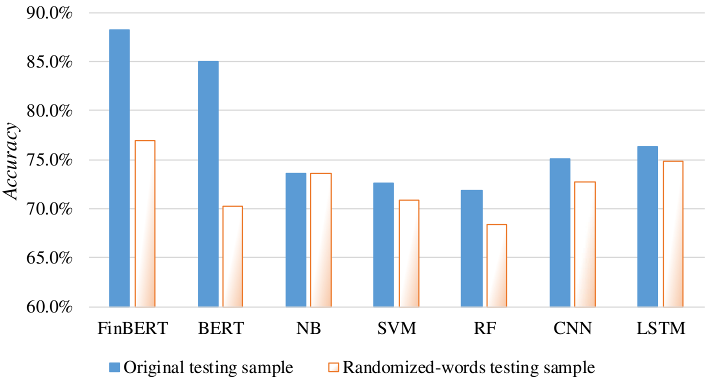
Taken together, our analyses find that LLMs, especially those pretrained on domain-specific texts, can achieve state-of-the-art performance even with a small training data set, which is consistent with the semantic and syntactic information they learn from pretraining texts helping them with downstream NLP tasks.

Performance in sample with randomized word orders

We explore whether FinBERT outperforms other NLP algorithms because it can better summarize contextual information in financial texts. Specifically, we examine how NLP algorithms'

[20] For example, when we use 90% of the original training and validation samples, the number of sentences drops to 7,290 (or 8,100 × 90%) and 810 (or 900 × 90%), respectively.

Figure 2 Effect of word randomization on NLP algorithms' sentiment classification



Notes: This figure presents the sentiment classification accuracy rates of NLP algorithms in the original testing sample and the randomized-words testing sample. The testing sample contains 1,000 sentences, randomly selected from a sample of 10,000 sentences from financial analyst reports, with sentiments labeled by researchers (358 positive sentences, 458 neutral sentences, and 184 negative sentences). See notes to Table 1 for variable definitions.

performance changes when we randomize word orders in the testing sample sentences, which can indicate how much these algorithms consider syntactic information and logical structure in text. We tabulate the results in Table 3 and plot them along with *Accuracy* in the original testing sample in Figure 2.

We observe several patterns. As expected, no change occurs in the NB's *Accuracy* because it assumes that words are independent of each other. Other non-BERT algorithms (e.g., SVM, RF, CNN, and LSTM) experience moderate decreases in *Accuracy* from 1.5% for LSTM to 3.5% for RF, consistent with these methods considering only limited contextual information.[21] The LLMs' performance deteriorates considerably, with FinBERT and BERT's *Accuracy* dropping by 11.3% and 14.8%, respectively.[22] In fact, in the randomized-words testing sample, FinBERT retains only a slight accuracy edge (76.9%) over other algorithms (LSTM is second best at 74.8%), and BERT (70.2%) underperforms most algorithms. We also tabulate the algorithms' *Recall* in each sentiment category and find that LLMs suffer most in their ability to classify non-neutral sentences. For example, FinBERT's *Recall* for positive and negative sentiments are only 67% and 54.3% (21.5% and 35.4% lower than those in the original testing sample), respectively, which are on par or lower than those of the non-BERT algorithms. These results echo those in Table 1 indicating that LLMs' ability to consider contextual information contributes substantially to their high accuracy in classifying sentiments, especially non-neutral ones.

[21] As discussed in Appendix 3, we use unigrams and bigrams (only unigrams) as inputs to SVM and RF (NB). If we use only unigrams for SVM and RF, their *Accuracy* does not change (71.9% and 72.4%, respectively) when we randomize the word order. In an additional test (untabulated), we also implement NB using both unigrams and bigrams as inputs and find that its *Accuracy* drops by 3.2% when we randomize word order, in line with the SVM and RF models.

[22] One potential explanation of FinBERT's lower sensitivity to word randomization than that of BERT is that FinBERT's pretraining texts have word distribution that is more similar to the downstream tasks than the pretraining texts of BERT. Specifically, BERT-based algorithms are well-adapted to perform downstream tasks due to their ability to model higher-order word co-occurrence statistics (Sinha, Jia, et al. 2021; Sinha, Parthasarathi, et al. 2021). That is, they can perform well when the distribution of the words in the downstream tasks is similar to that in the pretraining text, even if word orders are randomized.

TABLE 4
The importance of Unique-FinVocab in FinBERT's classification

	FinBERT classification		
Subgroups	(1) Correct	(2) Incorrect	Difference (Correct – Incorrect)
All sentences (702 sentences)	7.01 (613)	4.49 (89)	2.52
Researcher labels			
Positive (258 sentences)	6.55 (229)	6.90 (29)	−0.35
Neutral (313 sentences)	6.52 (276)	2.70 (37)	3.82
Negative (131 sentences)	9.26 (108)	4.35 (23)	4.91
BERT classification			
Correct (599 sentences)	6.94 (576)	4.35 (23)	2.59
Incorrect (103 sentences)	8.11 (37)	4.55 (66)	3.56
Difference (Incorrect – Correct)	1.17	0.20	

Notes: This table reports the importance of Unique-FinVocab in FinBERT's classification of 702 sentences from the testing sample that contain Unique-FinVocab. We use LIME (Ribeiro et al. 2016) to identify the most important word in a sentence for FinBERT's classification. Columns (1) and (2) report statistics for subgroups of sentences for which FinBERT's classifications are correct and incorrect (i.e., the same as and different from researcher labels), respectively. In each cell, we report the percentage of sentences for which the most important word belongs to Unique-FinVocab on the top and the number of sentences of the subgroup in parentheses below.

Further analyses of FinBERT's advantage over BERT

We use three sets of analyses to explore what drives FinBERT's superiority over the BERT model. We focus on the two models' main difference, their pretraining text. Specifically, we investigate how finance vocabulary, which are the words and subwords that appear frequently in FinBERT's pretraining text but not in the BERT model's pretraining text (hereafter Unique-FinVocab), affect FinBERT's classification accuracy.[23]

First, we use Local Interpretable Model-Agnostic Explanations (hereafter, LIME) to study the importance of Unique-FinVocab in FinBERT's classification (Ribeiro et al. 2016). Intuitively, LIME measures a word's importance to a model using the change in the model's classification when the word is absent.[24] For example, consider a sentence that FinBERT classifies as positive. LIME first generates a group of sentences by randomly omitting some words from the original sentence and observes FinBERT's classification of these sentences. LIME then assigns a higher importance to a word if FinBERT is more likely to classify sentences without this word as negative or neutral (i.e., a different classification from the original sentence).[25]

Using the 702 sentences that contain Unique-FinVocab in the testing sample, we find two patterns consistent with this vocabulary's importance to FinBERT (tabulated in Table 4). First, we find that among the 613 sentences that FinBERT correctly classifies, the most important word to FinBERT's classification belongs to Unique-FinVocab 7.01% of the time (43 sentences). In

[23] We discuss the BERT algorithm's pretraining details, including vocabulary construction, in Appendix 2.

[24] LIME assigns importance to whole words rather than subwords because only words are understandable to humans (Ribeiro et al. 2016).

[25] Note that because interpretability models such as LIME rely on a linear relation to approximate local interpretations, they may not faithfully explain predictions made by LLMs, which have nonlinear mapping between high-dimensional input text and outputs (Alvarez-Melis and Jaakkola 2018). Nonetheless, they produce outputs that are easily interpretable by humans and are thus the most widely used methods to generate explanations in deep learning (Adadi and Berrada 2018).

sentences that FinBERT incorrectly classifies, the most important word belongs to Unique-FinVocab only 4.49% of the time (4 out of 89 sentences). Second, we find that within the subset of sentences that only FinBERT but not BERT correctly classifies, the most important word for FinBERT belongs to Unique-FinVocab 8.11% of the time (3 out of 37 sentences). In sentences that both FinBERT and BERT correctly classify, the ratio is only 6.94% (40 out of 576 sentences).

We also explore how Unique-FinVocab's importance varies with sentiment categories by separately examining the likelihood that the most important words for FinBERT belong to Unique-FinVocab in sentences that researchers label as positive, neutral, or negative. The results in Table 1 show that FinBERT's advantage over BERT is especially large for correctly identifying negative sentences. We thus expect Unique-FinVocab to play a more important role in FinBERT's accuracy in negative sentences than in positive and neutral sentences. We find results consistent with this intuition. Specifically, among sentences that FinBERT correctly classifies as positive, neutral, or negative, the most important word for FinBERT's prediction belongs to Unique-FinVocab in 6.55%, 6.52%, and 9.26% of the sentences, respectively, suggesting that Unique-FinVocab is especially instrumental when FinBERT labels negative sentiments.[26]

Second, we sort the testing sample sentences into two groups based on whether their proportions of Unique-FinVocab are above or below the sample median. If finance vocabulary contributes to FinBERT's advantage, we expect FinBERT to show better accuracy in the high Unique-FinVocab group than in the low Unique-FinVocab group. We again find consistent results (untabulated): FinBERT and BERT's *Accuracy* are 88.4% and 84.4%, respectively, in the high group (a 4% difference) and 88.0% and 85.6% in the low group (a 2.4% difference).

Last, we investigate whether Unique-FinVocab helps FinBERT retain its performance in small samples. To do so, we select 10% of the training sample (810 and 90 sentences for training and validation, respectively) with Unique-FinVocab and use it to fine-tune the models. Untabulated statistics show that FinBERT achieves *Accuracy* of 82% on the testing sample (1,000 sentences, same as Table 2), higher than its *Accuracy* in a same-size sample that does not require the presence of Unique-FinVocab (81.3%, reported in Table 2). In comparison, BERT's *Accuracy* in the two samples are almost identical (62.1% and 62%, respectively). Thus, the results suggest that Unique-FinVocab helps FinBERT reduce its accuracy deterioration when the training sample becomes smaller.

In sum, we find that exposure to financial vocabulary during pretraining likely contributes to FinBERT's advantage over BERT in sentiment classification in financial text. This insight also suggests that to the extent that words most indicative of sentiments in financial and general texts overlap, using the performance in sentiment classification task to compare FinBERT with BERT may underestimate FinBERT's advantage. That is, FinBERT's outperformance over BERT can be stronger in downstream tasks that rely more on finance vocabulary, such as identifying sentences related to financial constraints.[27] We leave such comparisons to future research.

4. Comparing performance in labeling ESG-related discussions

As explained in section 3, researchers can fine-tune a pretrained LLM for an NLP task using a relatively small sample of labeled texts and still achieve high performance. In this section, we provide additional examples by fine-tuning FinBERT for other tasks and then compare its performance with other NLP algorithms. We choose a task of interest to researchers, practitioners and regulators: labeling ESG-related discussions.

[26] We also compare the difference in the likelihoods that the most important word for FinBERT belongs to Unique-FinVocab when FinBERT's classification is correct versus when it is incorrect, across the three sentiment classes. We find that difference to be larger in negative sentences (4.91% = 9.26% − 4.35%) than in positive (−0.35% = 6.55% – 6.90%) and neutral (3.82% = 6.52% – 2.70%) sentences.

[27] This result also suggests that the BERT model may be more appropriate than FinBERT for analyzing general (non-financial) texts, such as Wikipedia or social media, because these texts may be less similar to the pretraining text used in FinBERT.

TABLE 5
Comparison of NLP algorithms' performance in classification of ESG-related discussion

Panel A: ESG classification—full sample

	Overall (%)				Environmental (50 sentences)	Social (50 sentences)	Governance (50 sentences)	Non-ESG (50 sentences)
	Accuracy	*Precision*	*Recall*	F_1 *score*	*Recall* (%)	*Recall* (%)	*Recall* (%)	*Recall* (%)
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

Panel B: ESG classification across different sample sizes

Training sample	100%	80%	60%	40%	20%	*Accuracy* diff.
			Accuracy in the testing sample (%)			(100% – 20%)
FinBERT	89.5	87.0	86.5	82.0	81.5	8.0
BERT	87.0	84.5	81.5	78.5	69.5	17.5
NB	80.0	81.5	78.5	73.5	68.5	11.5
SVM	75.0	74.5	68.5	66.0	55.5	19.5
RF	78.0	73.0	71.0	67.5	65.0	13.0
CNN	82.0	79.5	81.5	77.0	64.0	18.0
LSTM	86.0	80.0	80.5	78.0	65.5	20.5

Notes: This table tabulates the ESG classification performance of different NLP algorithms in the testing sample. The testing sample contains 200 sentences (50 environmental, 50 social, 50 governance, and 50 non-ESG), randomly selected from a sample of 2,000 sentences labeled by researchers. Panel A reports each NLP algorithm's overall *Accuracy*, *Precision*, *Recall*, F_1 *score*, and its *Recall* in each ESG category. The overall *Precision*, *Recall*, and F_1 *score* are the means of the *Precision*, *Recall*, and F_1 *score* in the four ESG categories. Panel B presents the ESG classification accuracy rates of NLP algorithms using different training sample sizes. We use the full sample (100%) and subsets (80%, 60%, 40%, and 20%). For example, at 100% (20%), we use 1,620 (324) as the training sample and 180 (36) as the validation sample. See notes to Table 1 for variable definitions.

To train the machine learning models to classify ESG-related discussions, we manually label sentences from firm disclosure into one of four categories: (i) environmental (e.g., climate change, natural capital, pollution and waste, and environmental opportunities); (ii) social (e.g., human capital, product liability, stakeholder opposition, and social opportunities); (iii) governance (e.g., ownership and control, board composition and duties, executive compensation, external audits, and internal controls); or (iv) non-ESG (MSCI 2022). To ensure sufficient representation of each category, we obtain sentences from firms' CSR reports and MD&A sections of 10-Ks. Because size and industry membership may affect firms' ESG activities and disclosure, we select the largest and smallest S&P 500 firms in each 2-digit GICS sector (20 total firms from 10 sectors).[28] We obtain CSR reports from firm websites and extract MD&A sections from 10-K filings.[29] We label 500 sentences each for ESG categories from CSR reports and 500 MD&A sentences that do not discuss ESG. Our final ESG-related discussion sample comprises 2,000 sentences. Similar to how we train machine learning models to classify sentiments, we use 81% of the sample for training, 9% for validation, and the remaining 10% for testing performance.

Table 5, panel A, reports NLP algorithms' performance in the testing sample. Consistent with results for the sentiment classification task, we find that FinBERT is better at classifying ESG sentences than other machine learning methods. Specifically, FinBERT achieves *Accuracy* of 89.5%, whereas *Accuracy* for NB, SVM, RF, CNN, and LSTM are 80.0%, 75.0%, 78.0%, 82.0%, and 86.0%, respectively. BERT has the second-best *Accuracy* at 87.0%, a slight advantage over LSTM.

We also analyze how different NLP algorithms perform with smaller training samples (results shown in Table 5, panel B and plotted in Figure 3) and find similar results as for the sentiment classification task (reported in Table 2). Specifically, FinBERT retains the best performance (a decrease of 8.0% in *Accuracy*) when we reduce the training sample to 20% of its full size, and other algorithms have larger decreases ranging from 11.5% for NB to 20.5% for LSTM. BERT's *Accuracy* is comparable to FinBERT in the full training sample, but it underperforms FinBERT by 12% in the smallest (20%) training sample.

In sum, our results reinforce our conclusion from the sentiment classification that domain-adapted LLMs can outperform other machine learning algorithms in NLP tasks in financial texts, especially with smaller training samples.

5. Earnings conference call tone and market reaction

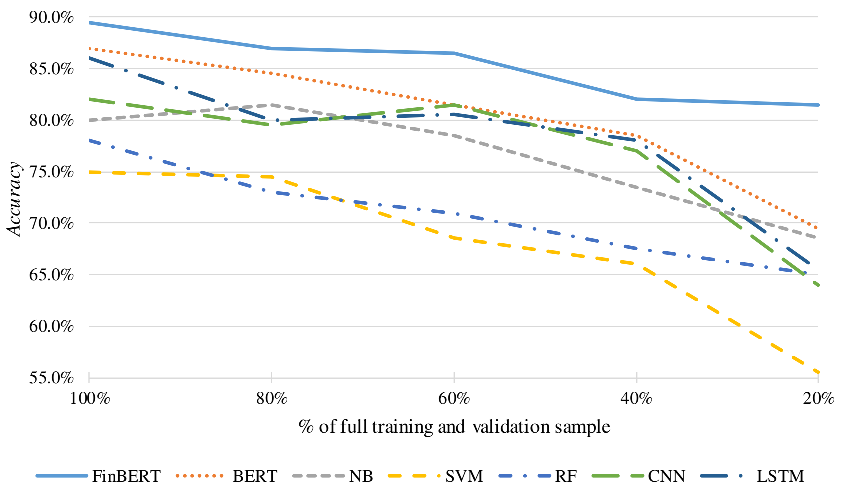
Last, we examine the association between market reaction to earnings conference calls and the tone of these calls, as measured by different NLP algorithms, to provide an alternative measurement of algorithm performance. In this test, we assume that an algorithm performs better if the tone is more highly correlated with the market reaction (i.e., it better explains how investors interpret earnings conference calls).

Using the association between tone and market reaction as the benchmark can be more objective and help quantify the benefit of using FinBERT to summarize textual sentiments in economic significance. However, this test is a joint test of whether NLP algorithms accurately capture textual sentiments and how the market should react to such sentiments as conveyed in earnings conference calls. For example, investors may not react strongly to textual sentiments, regardless of how accurately these sentiments are measured, if they consider the information in the text to be irrelevant, if they have already learned the information from other sources, or if they respond instead to quantitative information in the calls (Lansford et al. 2009; Keskek et al. 2014). Due to these limitations, we view the test as providing additional, albeit noisier, evidence to supplement our main analyses using researcher-labeled samples.

[28] We include the real estate sector in the financial sector because the former only moved out of the latter in 2016.

[29] These reports have various titles, such as Sustainability Report (eight firms), Corporate Accountability/Responsibility Report (four firms), ESG Report (two firms), and Environmental Responsibility Report (one firm). Five firms do not provide ESG disclosure before 2019.

Figure 3 ESG classification accuracy across different sample sizes



Notes: This figure presents the ESG classification accuracy rates of NLP algorithms using different training sample sizes. We use the full sample (100%) and subsets (80%, 60%, 40%, and 20%) of training samples. For example, at 100% (20%), we use 1,620 (324) as the training sample and 180 (36) as the validation sample. We use a constant testing sample size of 200 sentences, same as in Table 5, panel A. See notes to Table 1 for variable definitions.

We start with 31,592 earnings conference call transcripts of S&P 500 firms during 2003–2020 from the Thomson Reuters StreetEvents database. After removing firm-quarters without sufficient data from Compustat, CRSP, I/B/E/S, and Thomson Reuters Institutional Holdings (13F) to calculate the variables used in the regressions, our final sample includes 28,873 earnings conference call transcripts from 712 unique companies. Table OA1 in the online Appendix shows the year and 2-digit GICS sector distributions of the sample, which are evenly distributed over the years, with most firms from the consumer discretionary and information technology, and the least from communication services.

We use abnormal returns to capture the market reaction to a conference call and estimate the following regression:

$$CAR = \alpha + \beta_1 Tone_j + \sum \gamma Controls + \varepsilon, \qquad (1)$$

where *CAR* is the cumulative abnormal returns in a three-day window centered on the earnings conference call date, as in prior studies (Price et al. 2012), and $Tone_j$ is the sentiment of the conference call according to NLP algorithm *j*. First, we use the algorithm to label each sentence as positive, negative, or neutral based on the highest predicted likelihood.[30,31] Then, we define the earnings conference call sentiment as the percentage of positive sentences minus the percentage

[30] In a sensitivity test, we use the sentiment likelihood of the first or last 512 tokens of the earnings conference call labeled by FinBERT as the earnings conference call sentiment and find it underperforms our main measure of average sentence-level sentiment from FinBERT in explaining market reactions to the calls (tabulated in the online Appendix, Table OA2, columns (2)–(3)). It also underperforms the average sentence-level sentiments based on BERT, the LM dictionary, NB, SVM, RF, CNN, and LSTM (untabulated).

[31] In another sensitivity test, we assign each sentence proportionally to the three sentiment categories based on predicted likelihoods from the machine learning algorithms. We find that earnings conference call sentiments of FinBERT based on proportional allocation have weaker associations with market reactions than those based on allocating a sentence to its highest likelihood sentiment category (tabulated in the online Appendix, Table OA2, column (4)).

TABLE 6
Descriptive statistics

Variables	N	Mean	SD	Q1	Median	Q3
$Tone_{FinBERT}$	28,873	0.27	0.11	0.19	0.27	0.35
$Tone_{BERT}$	28,873	0.27	0.11	0.19	0.27	0.35
$Tone_{LM}$	28,873	0.09	0.08	0.04	0.09	0.15
$Tone_{NB}$	28,873	0.26	0.11	0.18	0.26	0.34
$Tone_{SVM}$	28,873	0.20	0.08	0.14	0.20	0.26
$Tone_{RF}$	28,873	0.25	0.08	0.19	0.25	0.31
$Tone_{CNN}$	28,873	0.22	0.10	0.15	0.22	0.29
$Tone_{LSTM}$	28,873	0.24	0.10	0.17	0.24	0.31
CAR	28,873	0.08	5.69	−2.96	0.09	3.26
Earn	28,873	0.02	0.02	0.01	0.01	0.03
UE	28,873	0.08	0.37	0.00	0.05	0.16
Accruals	28,873	−0.01	0.03	−0.02	−0.01	0.00
EarnVol	28,873	0.01	0.02	0.00	0.01	0.01
Size	28,873	9.67	1.10	8.92	9.55	10.31
MtoB	28,873	2.07	1.29	1.19	1.64	2.46
Leverage	28,873	0.23	0.16	0.11	0.21	0.32
IO	28,873	0.73	0.22	0.67	0.78	0.87
Analyst	28,873	2.80	0.44	2.56	2.83	3.09
Age	28,873	3.42	0.72	2.94	3.56	3.93
Turnover	28,873	0.55	0.38	0.30	0.44	0.65
PriorAlpha	28,873	0.01	0.18	−0.10	0.01	0.12
Dividend	28,873	0.17	0.37	0.00	0.00	0.00
NASDAQ	28,873	0.21	0.41	0.00	0.00	0.00

Notes: This table presents the descriptive statistics for the sample of earnings conference calls. See Appendix 4 for detailed variable definitions.

of negative sentences in the managers' remarks from both the presentation and Q&A portions.[32] In the regressions, we normalize the tone measures to have a mean of zero and a standard deviation of one to facilitate interpretation of the coefficient magnitudes.

Following prior studies (Price et al. 2012; Blau et al. 2015; Davis et al. 2015; Henry and Leone 2016; Bochkay et al. 2020; Frankel et al. 2022), we control for current-quarter earnings (*Earn*), unexpected earnings (*UE*), accruals (*Accruals*), earnings volatility (*EarnVol*), firm size (*Size*), market-to-book ratio (*MtoB*), leverage (*Leverage*), institutional ownership (*IO*), analyst coverage (*Analyst*), firm age (*Age*), stock turnover and returns before the conference call (*Turnover*, *PriorAlpha*), dividend declaration (*Dividend*), and the trading market (*NASDAQ*).[33] See Appendix 4 for detailed variable definitions. We winsorize all continuous variables at the top and bottom 1%. We also include fiscal year-quarter fixed effects in the regressions and cluster standard errors at the firm level.[34]

Table 6 shows the descriptive statistics for the variables in the market reaction test. The overall sentiments of earnings conference calls are positive, with mean and median tone values all

[32] In sensitivity tests, we aggregate sentence-level sentiments of entire earnings conference calls (i.e., presentation and Q&A portions) or sentence-level sentiment of only the presentation portion and find similar results: FinBERT significantly outperforms NB, SVM, RF, CNN, and LSTM at the 1% level (untabulated).

[33] We do not include negative earnings, return volatility, or the number of segments because they are highly correlated with our existing control variables. In an untabulated sensitivity test, we confirm that including these three variables does not change our main results.

[34] In a sensitivity test, we cluster standard errors by both firm and year-quarter and find similar results (untabulated).

TABLE 7
Market reaction to earnings conference calls and textual sentiments

	DV = *CAR*							
	(1) FinBERT	(2) BERT	(3) LM	(4) NB	(5) SVM	(6) RF	(7) CNN	(8) LSTM
$Tone_j$	0.907***	0.873***	0.640***	0.647***	0.687***	0.466***	0.692***	0.743***
	(20.35)	(20.20)	(14.89)	(16.17)	(17.19)	(12.59)	(17.00)	(17.93)
Earn	20.009***	19.601***	18.787***	19.840***	19.605***	19.968***	19.996***	19.805***
	(6.64)	(6.49)	(6.27)	(6.55)	(6.54)	(6.53)	(6.69)	(6.60)
UE	3.093***	3.091***	3.161***	3.194***	3.183***	3.255***	3.190***	3.162***
	(18.97)	(18.91)	(19.16)	(19.27)	(19.19)	(19.50)	(19.29)	(19.15)
Accruals	−16.776***	−17.205***	−16.246***	−16.768***	−17.107***	−17.648***	−16.581***	−17.093***
	(−9.41)	(−9.62)	(−9.25)	(−9.42)	(−9.58)	(−9.81)	(−9.36)	(−9.61)
EarnVol	−6.635**	−5.609**	−5.484*	−5.515**	−4.267	−2.684	−5.199*	−4.591*
	(−2.39)	(−2.03)	(−1.96)	(−1.98)	(−1.53)	(−0.98)	(−1.88)	(−1.67)
Size	−0.104**	−0.086*	−0.046	−0.041	−0.075*	−0.021	−0.071	−0.064
	(−2.20)	(−1.86)	(−1.01)	(−0.89)	(−1.67)	(−0.47)	(−1.54)	(−1.40)
MtoB	−0.224***	−0.232***	−0.207***	−0.214***	−0.221***	−0.207***	−0.211***	−0.227***
	(−4.97)	(−5.13)	(−4.67)	(−4.76)	(−4.98)	(−4.52)	(−4.71)	(−5.03)
Leverage	−0.422	−0.420	−0.532**	−0.399	−0.352	−0.372	−0.473*	−0.448*
	(−1.60)	(−1.60)	(−2.13)	(−1.59)	(−1.43)	(−1.53)	(−1.86)	(−1.77)
IO	0.431**	0.465**	0.551**	0.506**	0.471**	0.554***	0.527**	0.470**
	(1.97)	(2.15)	(2.56)	(2.36)	(2.24)	(2.68)	(2.45)	(2.20)
Analyst	−0.305***	−0.314***	−0.285***	−0.340***	−0.281***	−0.240**	−0.307***	−0.325***
	(−2.75)	(−2.86)	(−2.61)	(−3.16)	(−2.64)	(−2.24)	(−2.84)	(−3.00)
Age	−0.008	−0.020	−0.076	−0.054	−0.034	−0.090*	−0.043	−0.031
	(−0.13)	(−0.36)	(−1.39)	(−1.01)	(−0.64)	(−1.73)	(−0.80)	(−0.56)
Turnover	0.210	0.252	0.119	0.188	0.189	0.201	0.195	0.239
	(1.15)	(1.38)	(0.64)	(1.05)	(1.05)	(1.10)	(1.07)	(1.29)
PriorAlpha	−1.238***	−1.234***	−1.095***	−1.070***	−1.104***	−1.023***	−1.110***	−1.142***
	(−5.32)	(−5.29)	(−4.68)	(−4.57)	(−4.72)	(−4.35)	(−4.76)	(−4.88)

(The table is continued on the next page.)

TABLE 7 (continued)

	DV = *CAR*							
	(1) FinBERT	(2) BERT	(3) LM	(4) NB	(5) SVM	(6) RF	(7) CNN	(8) LSTM
Dividend	0.148	0.136	0.126	0.149*	0.097	0.095	0.130	0.121
	(1.60)	(1.48)	(1.37)	(1.69)	(1.09)	(1.07)	(1.44)	(1.33)
NASDAQ	−0.032	−0.019	−0.035	0.027	0.004	0.079	−0.011	0.007
	(−0.26)	(−0.16)	(−0.30)	(0.23)	(0.03)	(0.70)	(−0.10)	(0.06)
Intercept	1.848***	1.656***	1.354**	1.233**	1.393**	0.680	1.387**	1.398**
	(3.05)	(2.76)	(2.26)	(2.08)	(2.38)	(1.16)	(2.32)	(2.34)
Year-Qtr FE	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Observations	28,873	28,873	28,873	28,873	28,873	28,873	28,873	28,873
Adjusted R^2s	0.080	0.079	0.069	0.070	0.071	0.065	0.072	0.073
Vuong's test								
versus $Tone_{FinBERT}$		3.08***	10.97***	12.11***	10.17***	12.82***	11.22***	9.81***
versus $Tone_{BERT}$			9.71***	10.55***	8.81***	12.35***	9.46***	7.76***
Mean R^2s based on bootstrapping	0.083	0.082	0.072	0.073	0.074	0.068	0.075	0.076
versus $Tone_{FinBERT}$		17.87***	136.97***	131.95***	116.51***	194.51***	109.46***	89.08***
versus $Tone_{BERT}$			119.26***	114.23***	98.77***	176.89***	91.72***	71.31***

Notes: This table reports the association between market reaction to earnings conference calls and the textual sentiments of the calls measured with each NLP algorithm. We estimate the OLS regression of model (1). In columns (1) to (8), $Tone_j$ is measured with FinBERT, BERT, LM dictionary, NB, SVM, RF, CNN, and LSTM, respectively. All tone variables are normalized to have a mean of zero and standard deviation of one. All regressions include year-quarter fixed effects. The *t*-statistics based on standard errors clustered by firm are reported in parentheses below the coefficients. Below the regression results, we report the *z*-statistics of Vuong's tests comparing the explanatory power of models with $Tone_{FinBERT}$ and $Tone_{BERT}$, respectively. The last three rows report means of R^2s in 5,000 bootstrapped samples, and the *t*-statistics of whether the means are significantly different from those of $Tone_{FinBERT}$ and $Tone_{BERT}$, respectively. Each sample contains 28,873 randomly selected earnings conference calls from the full sample, with replacements. *, **, and *** indicate two-tailed statistical significance at the 0.10, 0.05, and 0.01 levels, respectively. See Appendix 4 for detailed variable definitions.

measuring greater than zero (i.e., more positive than negative sentences). Untabulated results show that the tone measures of FinBERT and BERT are strongly correlated (Pearson correlation = 0.967), whereas the correlations between the tone of FinBERT and that of the other approaches range from 0.733 (RF) to 0.920 (LSTM). Among the tone measures, those of FinBERT and BERT have higher correlation coefficients with *CAR* (0.150 and 0.147, respectively) than do the other tone measures (ranging from 0.085 to 0.124). This result provides preliminary evidence that NLP algorithms that consider contextual information produce textual sentiments that are more closely associated with investor reaction to the text.

Table 7 tabulates the results of estimating model (1) using the tone measures from the NLP algorithms. As shown, all eight tone measures are positively and significantly related to the three-day *CAR*, consistent with a more positive call sentiment being associated with higher market reactions. Among the measures, $Tone_{FinBERT}$ is the most economically significant, with a one-standard-deviation increase associated with an increase of 0.91% in the three-day *CAR*. Compared with FinBERT, the other algorithms underestimate the economic magnitudes of tone. For example, the LM dictionary underestimates the economic magnitude of tone by 29.4% (0.640 relative to 0.907). Non-BERT machine learning algorithms also underestimate the economic magnitudes (ranging from 18.1% for LSTM to 48.6% for RF).[35]

Next, we compare the explanatory power of the models with the tone measures from the NLP algorithms using the Vuong (1989) test and the bootstrapping technique (Efron and Tibshirani 1994). In the bootstrapping procedure, we generate empirical distributions for R^2s using 5,000 randomly drawn samples (with replacement) from our original sample and then compare differences across tone measures. All samples have the same size as the original sample. In both tests, we find that models with tone measures from FinBERT and BERT model have greater explanatory power than other models (all significant at the 1% level) and that FinBERT further outperforms BERT (significant at the 1% level).[36,37]

In sum, our results for the market reaction tests show that FinBERT's measure of earnings conference call textual sentiment aligns more closely with investor reaction to the calls than measures of other NLP algorithms. These results are consistent with FinBERT better capturing earnings conference calls' textual sentiments and thus corroborate our main analyses based on the researcher-labeled analyst report sample.

6. Conclusion

Despite growing interest in textual analysis of financial documents, most studies rely on NLP algorithms that use a bag-of-words structure and ignore contextual information. In recent years NLP researchers have developed deep-learning-based LLMs that excel at NLP tasks in general texts, but it is unclear whether and how much these LLMs, especially ones customized for financial text, outperform simpler algorithms when conducting tasks most relevant to finance and accounting researchers.

[35] In a sensitivity test, we include both tone from FinBERT and tone from one of the other approaches as independent variables in the market reaction test (tabulated in supporting information in the online Appendix, Table OA3). We find that only FinBERT's coefficient is positive and significant, whereas tones from the other approaches are negative and significant (except BERT, which is insignificant).

[36] In a sensitivity test, we replace tone with the percentages of positive and negative sentences in model (1) and find similar results (untabulated), that the model based on FinBERT has greater explanatory power than those based on the LM dictionary, NB, SVM, RF, CNN, or LSTM (all significant at the 1% level).

[37] In another sensitivity test, we compare the performance of the algorithms using smaller samples using a stratified sampling method to randomly select 15 observations from each of the 73 fiscal quarters, representing 1,095 observations. Untabulated results show that FinBERT's tone measure has higher economic magnitude and higher explanatory power than other algorithms, all significant at the 1% level except BERT's explanatory power, which is significant at the 5% level. In addition, using FinBERT can meaningfully reduce the incidence of failing to reject a true null hypothesis when the sample size is small.

We use Google's BERT algorithm and a large corpus of financial texts to develop FinBERT, an LLM adapted to the finance domain. We evaluate FinBERT's performance in one of the most widely used NLP tasks in financial texts, sentiment classification, using analyst report sentences labeled by researchers as the primary benchmark. We find that FinBERT achieves substantially higher out-of-sample accuracy than other approaches that are popular in finance and accounting research, including the LM dictionary, NB, SVM, RF, CNN, and LSTM. We also document that FinBERT's advantage over other algorithms, including Google's BERT model, is especially large when the training sample is small, likely because FinBERT considers contextual information in financial text and in texts containing financial words not frequently used in general texts. We further demonstrate FinBERT's capacity by fine-tuning it in an additional task, labeling ESG-related discussions. We find similar results as those for sentiment classification. Last, we examine whether the superior performance of FinBERT in sentiment classification extends to other financial texts. Using the association between earnings conference calls' textual sentiments and market reaction to the calls as an alternative benchmark, we find evidence consistent with FinBERT better capturing earnings conference calls' textual sentiments, which corroborates our main results based on the researcher-labeled analyst report sample.

By introducing FinBERT and quantifying its superior performance over algorithms popular in finance and accounting research, this study primarily contributes to the growing literature examining textual analysis in financial economics. Our results also have practical implications for investment professionals and financial market regulators who increasingly use NLP algorithms to extract insights from financial texts.

Appendix 1: Glossary

Algorithm/model	Definition
Bidirectional encoder representation from transformer (BERT) algorithm	A large language model pretraining technique developed by Google to perform bidirectional representation learning—that is, learning that considers both the left and right contexts of words (Devlin et al. 2019)
Bidirectional encoder representation from transformer (BERT) model	A publicly released English-language BERT model by Google that is pretrained using general texts, including Wikipedia and BookCorpus (Devlin et al. 2019)
Convolutional neural network (CNN)	A deep learning algorithm that uses convolutional layers to summarize the local feature of a data instance (e.g., immediately adjacent words)
Deep learning algorithm (deep neural network)	A neural network machine learning algorithm with more than one hidden layer
FinBERT	A large language model based on the BERT algorithm and pretrained using financial texts, including corporate annual and quarterly filings, financial analyst reports, and earnings conference call transcripts
Large language model (LLM)	A deep learning natural language processing algorithm with a large number of parameters that learns the semantic and syntactic relations between words from a large volume of texts and incorporates context in representing texts
Long short-term memory (LSTM)	A deep learning algorithm based on a recurrent neural network that can retain relations between distant sequential data. A recurrent neural network is a deep learning algorithm that can process a sequence of inputs

(The table is continued on the next page.)

(continued)

Algorithm/model	Definition
Naïve Bayes (NB)	A supervised machine learning algorithm that assigns data to the most likely category according to the "naïve" assumption that features are independent of each other in a given category
Neural network	A class of machine learning algorithms that contain an input layer and output layer connected by one or more hidden layers. Usually, a neural network algorithm specifies a nonlinear function and uses training data to identify parameters that minimize prediction errors (i.e., the difference between the prediction generated by the output layer and the actual value)
Neural word embedding (word representation)	An approach to represent words using low-dimensional (usually in hundreds) vectors learned from co-occurrences of words in the training texts
Random forest (RF)	A supervised machine learning algorithm that assigns data to categories by constructing many decision trees and aggregating the outputs of these trees. Each tree is constructed using a random sample of training data and input variables
Support vector machine (SVM)	A supervised machine learning algorithm that maps training data into a vector space and constructs hyperplanes to separate data points into categories
Other key terms	
BaseVocab	The vocabulary of the BERT model (i.e., tokens that appear frequently in the pretraining text of the BERT model)
Fine-tuning	The process of feeding a labeled data set to a pretrained large language model so that the model can accommodate specific NLP tasks, such as sentiment classification. During fine-tuning, the model's parameters are adjusted slightly according to the prediction task
FinVocab	The vocabulary of the FinBERT model (i.e., tokens that appear frequently in the pretraining text of the FinBERT model)
Hyperparameters	Parameters that specify the architecture of a machine learning model (e.g., the number of hidden layers in a deep learning model) or learning procedure (e.g., the learning rate in model optimization). Researchers set hyperparameters prior to training the model and adjust them during hyperparameter tuning
Local interpretable model-agnostic explanations (LIME)	An interpretable machine learning technique that measures the importance of a feature to the classification of a machine learning model according to the change in classification when the feature is absent
Pretraining	The process of training a large language model using a large corpus of text. In pretraining, a model can learn the syntactic and semantic relations and linguistic patterns in a natural language
Token	A unit of input to the large language model; this can be a word, subword, or punctuation symbol. The BERT algorithm splits a word into subwords if it appears less frequently than a pre-defined threshold in the pretraining text

Appendix 2: FinBERT's pretraining and fine-tuning

Overview of the BERT algorithm

Training the BERT algorithm includes two steps: pretraining and fine-tuning. During pretraining, researchers feed a large corpus of texts to the algorithm so that it can learn the syntactic and semantic relations and linguistic patterns in natural language (Goldberg 2019; Tenney et al. 2019). Using sentence pairs as input, the algorithm has two objectives in this step: masked language modeling and next sentence prediction. For masked language modeling, the BERT algorithm randomly masks 15% of the words from an input sentence and then predicts the masked words from the remaining words in the sentence. This objective allows the BERT algorithm to obtain a bidirectional model—that is, consider both the left and right context of words. For the next sentence prediction task, the input includes sentence pair A–B, in which half of the time, B is the actual sentence that follows A in the original text and the other half of the time it is a random sentence. The algorithm predicts whether sentence B is the actual sentence after A so that it can learn the relation between two sentences. The BERT algorithm minimizes the combined loss function of the two prediction tasks during the pretraining process.

As illustrated in Figure 4, during pretraining, the BERT algorithm first divides an input sentence pair into tokens based on a vocabulary, adds a special token [CLS] to the beginning and another special token [SEP] to separate the sentence pair. Next, it maps the tokens (e.g., Tok N in Figure 4) to token embeddings, which are 768-dimensional vectors representing tokens in the model (e.g., E_N in Figure 4).[38]

The BERT algorithm then uses the token embeddings (vectors) as inputs and processes them using a multilayer (12 layers in the $\text{BERT}_{\text{Base}}$ model) bidirectional Transformer encoder.[39] In short, each layer of the encoder takes the token embeddings (vectors) from the previous layer as input, measures the correlation between words in different positions from left to right and right to left, and generates output embeddings for each token. For more details about the Transformer architecture, see Vaswani et al. (2017). The bidirectional design allows the output embedding of a token to incorporate information from other tokens in the sentence, that is, reflect its contextual information. The multilayer Transformer encoder can summarize an entire sentence's semantics from different facets. For example, Clark et al. (2019) show that some layers encode the syntactic information of a sentence, whereas other layers encode its semantic information. The multilayer bidirectional Transformer encoder's output is a list of 768-dimensional vectors (one for each token including the special token [CLS], for example T_N in Figure 4), which can be used as inputs to downstream tasks such as sentiment classification.

FinBERT pretraining details

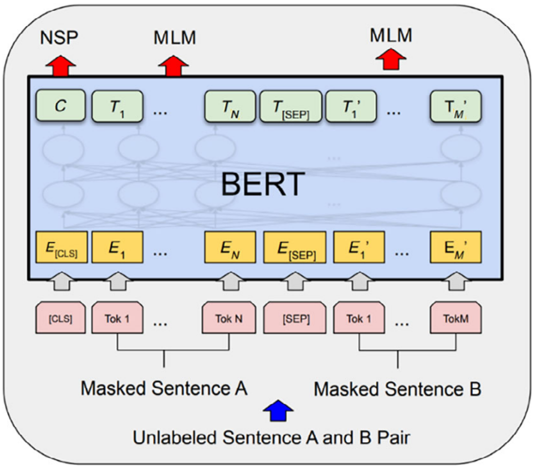
As discussed in section 2, we pretrain FinBERT using a large corpus of financial texts containing 4.9 billion tokens: 2.5 billion tokens for corporate annual and quarterly filings—the business description, risk factor, and MD&A sections of forms 10-K and 10-Q for Russell 3,000 firms between 1994 and 2019; 1.1 billion tokens for analyst reports of S&P 500 firms between 2003 and 2012; and 1.3 billion tokens for earnings conference call transcripts of 7,740 public firms between 2004 and 2019.

Pretraining a BERT-based algorithm requires a vocabulary containing words that frequently appear in texts so that the algorithm can learn their meanings. We pretrain FinBERT from scratch

[38] At the beginning of pretraining, the token embeddings are random numbers. During pretraining, the BERT algorithm updates the embedding for each token (and other parameters in the model) to achieve its two objectives. In the $\text{BERT}_{\text{LARGE}}$ model, token embeddings are 1,024-dimensional vectors.

[39] The $\text{BERT}_{\text{LARGE}}$ model uses a 24-layer bidirectional Transformer encoder and has a substantially higher training cost than the $\text{BERT}_{\text{BASE}}$ model. We follow prior domain adaptations of BERT and use the $\text{BERT}_{\text{BASE}}$ architecture to train FinBERT (Alsentzer et al. 2019; Beltagy et al. 2019; Lee et al. 2019).

Figure 4 Pretraining process for the BERT algorithm (Devlin et al. 2019)



by constructing a financial vocabulary (hereafter, FinVocab) from the corpus of financial texts. We use Google's WordPiece algorithm (also used by the BERT model), which retains words that appear frequently but decomposes words that appear less frequently into meaningful subwords. For example, if the word "liquidity" appears less frequently in the corpus, WordPiece decomposes it into "liquid" and "ity," two subwords that appear more frequently. This decomposition preserves the meaning of less frequent words maintaining a reasonable vocabulary size (Wu et al. 2016).

In constructing FinVocab with WordPiece, we define tokens that appear fewer than 8,500 times in our corpus as less frequent tokens that should be decomposed. We choose 8,500 as the cutoff so that the constructed FinVocab has 30,873 tokens (case-insensitive), similar to the BERT model's vocabulary (BaseVocab, 30,522 tokens).[40] Despite the negligible difference in size, only 12,498 tokens (around 41%) appear in both vocabularies, consistent with FinVocab including a substantial number of finance domain-specific terms not frequently used in general texts.[41] Examples of Unique-FinVocab (i.e., tokens that are in FinVocab but not in BaseVocab) include "liquidity," "liabilities," "depreciation," "amortization," "volatility," "outperform," "backlog," and "headwind."[42]

In addition to pretraining from scratch, there are two alternatives to adapt BERT to a domain text. The first is to take Google's BERT model's vocabulary and parameters as given and *further* pretrain (i.e., update the parameters) with domain text (Araci 2019). The second is to add words and subwords from domain text to the BERT model's vocabulary (Jain et al. 2020; Nayak et al. 2020). Pretraining from scratch requires more text and computational resources during pretraining but should offer better performance in NLP tasks in the domain (Alsentzer et al. 2019;

[40] Prior studies that pretrain BERT-algorithm-based models using domain-specific texts also set the vocabulary size to match that of the BERT model (Beltagy et al. 2019; Chalkidis et al. 2020).

[41] Beltagy et al. (2019) report a token overlap of similar magnitude (42%) between the BERT and SciBERT vocabularies, suggesting that the difference in frequently used words between financial and general domain texts is as large as that between scientific and general domain texts.

[42] These tokens are broken up into smaller tokens in BaseVocab: liquidity = liquid + ity; liabilities = lia + bilities; depreciation = de + pre + ciation; amortization = amor + ti + zation; volatility = vol + ati + lity; outperform = out + per + form; backlog = back + log; and headwind = head + wind.

Beltagy et al. 2019). We find that, compared to FinBERT, both alternatives result in smaller improvement over the BERT model.

Untabulated results show that, the *further*-pretrained version achieves *Accuracy* of 86.3% in sentiment classification, which represents a 1.3% improvement over the BERT model (85% as reported in Table 1, panel A).[43] In the vocabulary augmentation version, we replace all 994 unused tokens in BERT's vocabulary with 994 tokens that most frequently appear in financial text but are not in BERT's vocabulary. The augmented-BERT-vocabulary version has *Accuracy* of 85.6% in sentiment classification (untabulated), almost identical to the BERT model, which suggests that adding finance tokens to the vocabulary without allowing the model to learn their semantic and syntactic relation in financial texts does not improve LLMs' performance.

Next, we separate the texts into sentences using the standard sentence tokenizer in the Python spaCy library (https://spacy.io/). We use entire raw sentences as inputs to pretrain FinBERT. The hyperparameters for pretraining are 1,000,000 iterations with a learning rate of $2e^{-5}$ and a batch size of 128.[44] The pretraining stage, done on an NVIDIA DGX-1 server with four Tesla P100 GPUs and 128 gigabytes of GPU memory, takes approximately two days. We use the Horovod framework for multi-GPU training.[45] The pretrained FinBERT can be further fine-tuned for downstream financial NLP tasks. We have made the pretrained FinBERT model available to the research community to facilitate adoption (available at https://github.com/yya518/FinBERT).

Fine-tuning FinBERT for downstream tasks

Although pretraining a BERT-algorithm-based model such as FinBERT requires a large amount of text data and substantial computing resources, researchers can leverage the pretrained model in NLP tasks, such as sentiment classification, for a much smaller labeled data set, a process known as fine-tuning. In this process, the model slightly adjusts its parameters according to the customized prediction task. Recent studies show that fine-tuning a BERT-algorithm-based model for subsequent NLP tasks is efficient (i.e., takes little time compared with the pretraining process) and usually results in state-of-the-art performance (Devlin et al. 2019).

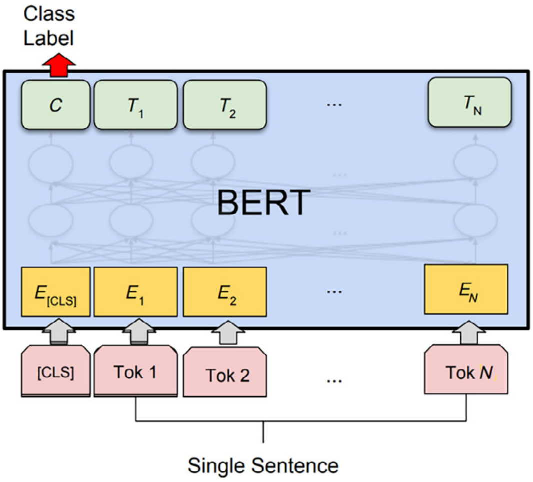
We use sentiment classification as an example to demonstrate FinBERT's fine-tuning process in a single sentence classification task. In this task, FinBERT takes a single sentence as the input and outputs the probability that the sentence belongs to a given category. Another example of such tasks includes identifying sentences that discuss ESG issues. As illustrated in Figure 5, in these tasks, the inputs to the BERT algorithm includes a special token, [CLS], and all word tokens of the sentence to be classified. The output is the vector representations (all of which have 768 dimensions) of every token, including [CLS]. In the BERT algorithm, the output vector of the special token [CLS] (***h***) summarizes the entire sentence's information, which is then used as

[43] We also examine the performance of the model by Araci (2019), which further pretrains BERT with finance text, and find that it underperforms FinBERT (untabulated). First, we directly use Araci's (2019) model fine-tuned for sentiment classification on our research-labeled sample and find that it achieves 74.25% *Accuracy*, lower than FinBERT's *Accuracy* of 88.2%. Second, we find that FinBERT fine-tuned on the Financial PhraseBank achieves a higher *Accuracy* (87.2%) and F_1 *score* (87.7%), compared to the 86% and 84% reported in Araci's (2019) table 2. Liu et al. (2021) also discuss the performance of a BERT-algorithm-based model pretrained on Wikipedia, BookCorpus, financial news, and social media discussion. However, it does not publish the model so we are unable to compare it with FinBERT.

[44] Batch size dictates how often the algorithm updates its parameters. A batch size of 128 means the algorithm will update its parameters after working through 128 samples, each containing a pair of sentences (as shown in Figure 4). The learning rate affects the magnitude at which the model updates parameters after each batch; 1,000,000 iterations mean the algorithm undergoes 1,000,000 parameter updates. That is, the total training sample is 1,000,000 × 128 samples, randomly drawn from the pretraining corpus.

[45] The Horovod framework is one of the most popular distributed deep learning frameworks developed by Uber (Sergeev and Del Balso 2018). Other popular frameworks include Ray, PyTorch, DeepSpeed and Distributed TensorFlow.

Figure 5 Fine-tuning the BERT algorithm on single sentence classification (Devlin et al. 2019)



input for the sentence classification task. That is, we feed $\boldsymbol{h}$ into a multinomial logistic classifier (i.e., it has a softmax function) to predict the probability of category label C:

$$p(C|\boldsymbol{h}) = softmax(\boldsymbol{W_C} \times \boldsymbol{h}) = \frac{\boldsymbol{e}^{\boldsymbol{W_C} \times \boldsymbol{h}}}{\sum \boldsymbol{e}^{\boldsymbol{W_C} \times \boldsymbol{h}}}, \tag{2}$$

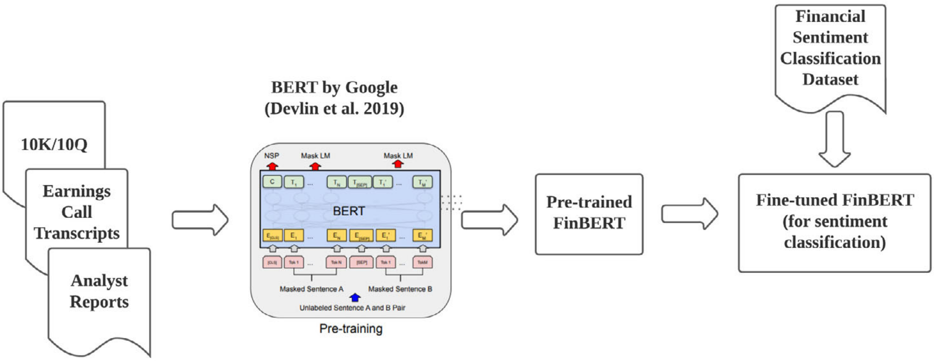
where C = {neutral, positive, negative} in sentiment classification. During fine-tuning, we allow all parameters of the FinBERT model and $\boldsymbol{W}$ to change to maximize the probability of correctly labeling the sentence's category.

We follow this procedure to fine-tune FinBERT for sentiment classification on a sample of sentences from financial analyst reports with sentiments labeled by researchers (A. H. Huang et al. 2014). The sample includes 10,000 sentences, of which 3,577 are positive, 4,586 are neutral, and 1,837 are negative. We randomly split the sample and use 81% for training, 9% for validation, and the remaining 10% for testing (i.e., 8,100, 900, and 1,000 sentences, respectively). Note that unlike many other NLP algorithms, such as NB, SVM, and RF, which require preprocessing inputs (e.g., removing stop words and punctuation, stemming, and lemmatizing), the BERT algorithm takes the entire raw sentences as inputs in fine-tuning (as well as in pre-training).[46] Moreover, researchers who use non-BERT algorithms often need to decide the model structure, such as the number of layers and filters and size of word embedding. FinBERT is simpler in comparison because its model structure is fixed.

We use the training sample to fit the parameter of FinBERT and BERT, the validation sample to set the hyperparameters and monitor training loss, and the testing sample to evaluate the final model. For the hyperparameters, we find that a learning rate of $2e^{-5}$ with a batch size of 32

[46] Note that in fine-tuning FinBERT, the BERT algorithm converts sentences into tokens based on FinVocab, adding special tokens ([CLS] and [SEP], as shown in Figure 4) and padding or truncating sentences to the maximum length allowed (512 tokens). However, these are standard procedures included in BERT algorithm's code as released by Google. Therefore, researchers who use FinBERT do not need to go through these manual steps. As a comparison, Frankel et al. (2022) include a five-step procedure to clean texts, including converting text to lower case, removing stop words, removing numbers and punctuations, stemming words and creating n-grams, before they use these texts in the machine learning algorithms including SVM, RF, and supervised latent Dirichlet allocation.

Figure 6 FinBERT pretraining and fine-tuning (for sentiment classification)


Financial Sentiment Classification Dataset
BERT by Google (Devlin et al. 2019)
10K/10Q
Earnings Call Transcripts
Analyst Reports
NSP
Mask LM
Mask LM
C
T1
TN
T[SEP]
T1'
TM'
BERT
E[CLS]
E1
EN
E[SEP]
E1'
EM'
[CLS]
Tok 1
Tok N
[SEP]
Tok 1
TokM
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
Pre-trained FinBERT
Fine-tuned FinBERT (for sentiment classification)

is optimal for both FinBERT and BERT.[47] We fine-tune the model for five epochs, each taking around 10 seconds with one NVIDIA RTX 3090 (24-GB of GPU memory).[48] Figure 6 illustrates the FinBERT pretraining and fine-tuning procedures.

In the hope that researchers and practitioners can benefit from the FinBERT model without incurring substantial pretraining and sample labeling costs, we release the FinBERT model fine-tuned for classifying sentiment and ESG issues (available at https://github.com/yya518/FinBERT). We share the model via the transformers Python library, a state-of-the-art and commonly used deep learning NLP framework and provide a four-step tutorial on how to use our fine-tuned FinBERT in Python to classify sentiments in supporting information in the online Appendix A.

Appendix 3: Implementation of naïve Bayes, SVM, RF, CNN, and LSTM models

In this Appendix, we discuss how we implement non-BERT-algorithm based machine learning models, including NB, SVM, RF, CNN, and LSTM, in the sentiment classification tasks.[49] For each machine learning algorithm, we use a grid search to find hyperparameters that achieve the best performance measured with accuracy in the validation sample.

For NB, we use the multinomial NB classifier to accommodate multi-class (e.g., positive, neutral, and negative) sentiment classification. NB has one hyperparameter, the additive smoothing parameter, which allows words that do not appear in the sample to have a non-zero probability. We search the additive smoothing parameter within [0.01, 0.1, 1.0, 10] and find that the parameter of 1.0 achieves the best performance.

SVM has two hyperparameters, a kernel function that can separate nonlinear regions and a regularization parameter that controls how much the model avoids misclassification.[50] We grid search the kernel in [rbf, linear] and regularization parameter in [0.01, 0.1, 1.0, 10] and find that a linear kernel and regularization parameter of 1.0 achieve the best performance.

[47] The batch size is set to 32 to fit the GPU's memory.

[48] In each epoch, all training examples are used once in training the model.

[49] For classification of ESG-related discussion, we tune the model's hyperparameters using its optimal value in the sentiment classification as the starting point. We find that these hyperparameters perform well for the ESG classifications.

[50] A kernel is a function that computes the inner product of two vectors—that is, similarity between input data pairs (Hofmann et al. 2008).

For RF, we grid search the number [100, 200, 300, 400, 500] and maximum depth [20, 50, 100, 200] of the trees and find that 500 trees with the maximum depth of 100 achieve the best performance.

Before we train the NB, SVM, and RF models, we remove stop words from the training sample and use a lower-case vocabulary size of 5,000. Thus, we represent each sentence with a 5,000 × 1 vector (one-hot encoding). For NB, we use unigrams; that is, the vocabulary comprises 5,000 unigrams with the highest frequencies in the training sample. For SVM and RF, we use unigram and bigram terms; that is, the vocabulary comprises 5,000 unigram and bigram terms with the highest frequencies in the training sample. We use the Python package scikit-learn (https://scikit-learn.org/stable/index.html) to implement these three machine learning algorithms.

We use a CNN model with a convolution (i.e., MaxPooling) layer, an intermediate dense layer with three hidden units (for the three sentiment labels), and a softmax activation function.[51] Textual data is one-dimensional, so we use a 1D convolution filter with a kernel size of 3 and the linear rectification unit function (ReLU) as the nonlinear activation function (Kim 2014). We tune two hyperparameters for the CNN model, the number of filters for the 1D convolution layer and the number of hidden units in the intermediate dense layer. We grid search the two hyperparameters in [8, 16, 32, 64] and [100, 200, 250] and find that 32 filters and 250 hidden units produce the best performance.

For LSTM, we use a two-layer bidirectional architecture. The bidirectional layer allows the model to summarize text into an embedding from left to right and from right to left (Schuster and Paliwal 1997). The most important hyperparameters for the LSTM model are the dimension of the output (the last layer) and the Dropout rate.[52] We grid search the output dimension in [25, 50, 100] and Dropout rate in [0.1, 0.2, 0.5] and find that output dimension of 50 and Dropout rate of 0.2 produces the best performance.

For the CNN and the LSTM models, we use the Adam optimizer (Kingma and Ba 2014) and cross-entropy as the loss function. We train each model 10 epochs with a batch size of 128. For both models, we lower-case the text and then use the word2vec pretrained by Google (Mikolov et al. 2013) as input representations for each word (Azimi and Agrawal 2021). We use the Python package keras (https://keras.io/) to implement these two models.

In terms of training time, on a server with one NVIDIA RTX 3090 (24 GB GPU memory), NB and SVM take 2–3 seconds, RF takes around 10 seconds, and CNN and LSTM take around 1.5 and 3 seconds per epoch, respectively.

Appendix 4: Definitions of variables in the market reaction test

Variable	Definition
Tone$_j$	Tone of the earnings conference call, measured by the number of positive sentences minus the number of negative sentences, scaled by the total number of sentences said by managers during both the presentation and Q&A portions. For FinBERT, BERT, NB, SVM, RF, CNN, and LSTM, we classify sentences into the category with the highest probability assigned by the algorithm. For the LM dictionary, we classify a sentence as negative if it contains at least one word from LM's Fin-Neg list, as positive if it does not contain any word from LM's Fin-Neg list but contains at least one word from LM's Fin-Pos list, and neutral otherwise

(The table is continued on the next page.)

[51] The MaxPooling layer is an operation that calculates the maximum value in a feature map so that the intermediate representations can be down-sampled. A dense layer is a neural network layer in which each neuron connects to every neuron of its preceding layer (i.e., a fully connected layer).

[52] Dropout is a technique that prevents a deep recurrent network from overfitting (Srivastava et al. 2014).

(continued)

Variable	Definition
CAR	Cumulative abnormal returns in the three-day window around the earnings conference call date, multiplied by 100. The abnormal return equals raw return minus value-weighted market return
Earn	Income before extraordinary items in a quarter scaled by the book value of assets at the beginning of the quarter
UE	Actual quarterly earnings per share (EPS) minus analyst consensus EPS forecast issued immediately before the earnings announcement date, scaled by the share price at the end of the fiscal quarter and multiplied by 100. Actual EPS and analyst consensus EPS forecasts are from I/B/E/S
Accruals	Net income minus cash flow from operations during the quarter scaled by the book value of assets at the beginning of the quarter
EarnVol	Standard deviation of the return on assets during the prior 16 quarters, with return on assets calculated as quarterly earnings scaled by total assets at the beginning of that quarter
Size	Log of book value of total assets at the end of the quarter
MtoB	Market value of equity plus book value of total liabilities scaled by book value of total assets at the end of the quarter
Leverage	Ratio of long-term debt to total assets at the end of the quarter
IO	Proportion of outstanding shares held by institutions. Institutional holding data are from Thomson 13f filings database
Analyst	Logarithm of one plus the number of analysts following the firm
Age	Logarithm of one plus the number of years since a firm first appeared in CRSP monthly file
Turnover	Cumulative turnover ratio in the window [−65, −6] relative to the conference call date. Turnover ratio equals the number of shares traded scaled by the number of outstanding shares
PriorAlpha	The intercept from a firm-specific regression of the Fama–French 3 factor model using daily data in the window [−65, −6], relative to the conference call date
Dividend	An indicator variable that equals one if a firm declares dividends within the three-day window surrounding the conference call date, and zero otherwise
NASDAQ	An indicator variable that equals one if a firm is traded on NASDAQ, and zero otherwise

References

Adadi, A., and M. Berrada. 2018. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access* 6: 52138–60.

Alsentzer, E., J. R. Murphy, W. Boag, W. Weng, D. Jin, T. Naumann, and M. McDermott. 2019. Publicly available clinical BERT embeddings, arXiv:1904.03323.

Alvarez-Melis, D., and T. S. Jaakkola. 2018. On the robustness of interpretability methods, arXiv: 1806.08049.

Araci, D. 2019. FinBERT: Financial sentiment analysis with pre-trained language models, arXiv:1908.10063.

Azimi, M., and A. Agrawal. 2021. Is positive sentiment in corporate annual reports informative? Evidence from deep learning. *Review of Asset Pricing Studies* 11 (4): 762–805.

Beltagy, I., A. Cohan, and K. Lo. 2019. SciBERT: Pretrained contextualized embeddings for scientific text, arXiv:1903.10676.

Bender, E. M., T. Gebru, A. McMillan-Major, and S. Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*.

Blau, B. M., J. R. DeLisle, and S. M. Price. 2015. Do sophisticated investors interpret earnings conference call tone differently than investors at large? Evidence from short sales. *Journal of Corporate Finance* 31: 203–19.

Bochkay, K., S. V. Brown, A. J. Leone, and J. W. Tucker. 2023. Textual analysis in accounting: What's next? *Contemporary Accounting Research* 40 (2): 765–805.

Bochkay, K., J. Hales, and S. Chava. 2020. Hyperbole or reality? Investor response to extreme language in earnings conference calls. *The Accounting Review* 95 (2): 31–60.

Bonsall, S. B., A. J. Leone, B. P. Miller, and K. Rennekamp. 2017. A plain English measure of financial reporting readability. *Journal of Accounting and Economics* 63 (2–3): 329–57.

Brown, S. V., L. A. Hinson, and J. W. Tucker. 2021. Financial statement adequacy and firms' MD&A disclosures. Working paper.

Brown, S. V., X. S. Tian, and J. W. Tucker. 2018. The spillover effect of SEC comment letters on qualitative corporate disclosure: Evidence from the risk factor disclosure. *Contemporary Accounting Research* 35 (2): 622–56.

Brown, S. V., and J. W. Tucker. 2011. Large-sample evidence on firms' year-over-year MD&A modifications. *Journal of Accounting Research* 49 (2): 309–46.

Buehlmaier, M. M., and T. M. Whited. 2018. Are financial constraints priced? Evidence from textual analysis. *Review of Financial Studies* 31 (7): 2693–728.

Castelvecchi, D. 2016. Can we open the black box of AI? *Nature News* 538: 20–3.

Chalkidis, I., M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. 2020. Legal-BERT: The Muppets straight out of law school, arXiv:2010.02559.

Clark, K., U. Khandelwal, O. Levy, and C. D. Manning. 2019. What does BERT look at? An analysis of BERT's attention, arXiv:1906.04341.

Das, S. R. 2014. Text and context: Language analytics in finance. *Foundations and Trends in Finance* 8 (3): 145–261.

Davis, A. K., W. Ge, D. Matsumoto, and J. L. Zhang. 2015. The effect of manager-specific optimism on the tone of earnings conference calls. *Review of Accounting Studies* 20 (2): 639–73.

De la Parra, D. 2021. Disclosure softness of corporate language. Ph.D. dissertation, Rice University.

Devlin, J., M. W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Donovan, J., J. Jennings, K. Koharki, and J. Lee. 2021. Measuring credit risk using qualitative disclosure. *Review of Accounting Studies* 26 (2): 815–63.

Efron, B., and R. J. Tibshirani. 1994. *An Introduction to the Bootstrap*. London: CRC Press.

Elliott, W. B., K. M. Rennekamp, and B. J. White. 2015. Does concrete language in disclosures increase willingness to invest? *Review of Accounting Studies* 20 (2): 839–65.

Frankel, R., J. Jennings, and J. Lee. 2016. Using unstructured and qualitative disclosures to explain accruals. *Journal of Accounting and Economics* 62 (2–3): 209–27.

Frankel, R., J. Jennings, and J. Lee. 2022. Disclosure sentiment: Machine learning vs. dictionary methods. *Management Science* 68 (7): 5514–32.

Friedman, J., T. Hastie, and R. Tibshirani. 2001. *The Elements of Statistical Learning*, Vol. 1. New York: Springer.

Gentzkow, M., B. Kelly, and M. Taddy. 2019. Text as data. *Journal of Economic Literature* 57 (3): 535–74.

Goldberg, Y. 2019. Assessing BERT's syntactic abilities, arXiv:1901.05287.

Henry, E., and A. J. Leone. 2016. Measuring qualitative information in capital markets research: Comparison of alternative methodologies to measure disclosure tone. *The Accounting Review* 91 (1): 153–78.

Hoberg, G., and G. Phillips. 2016. Text-based network industries and endogenous product differentiation. *Journal of Political Economy* 124 (5): 1423–65.

Hofmann, T., B. Schölkopf, and A. J. Smola. 2008. Kernel methods in machine learning. *Annals of Statistics* 36 (3): 1171–220.

Huang, A. H., R. Lehavy, A. Y. Zang, and R. Zheng. 2018. Analyst information discovery and interpretation roles: A topic modeling approach. *Management Science* 64 (6): 2833–55.

Huang, A. H., J. Shen, and A. Y. Zang. 2022. The unintended benefit of the risk factor mandate of 2005. *Review of Accounting Studies* 27 (4): 1319–55.

Huang, A. H., A. Y. Zang, and R. Zheng. 2014. Evidence on the information content of text in analyst reports. *The Accounting Review* 89 (6): 2151–80.

Huang, X., A. Nekrasov, and S. H. Teoh. 2018. Headline salience, managerial opportunism, and over- and underreactions to earnings. *The Accounting Review* 93 (6): 231–55.

Ivers, M. 1991. *The Random House Guide to Good Writing*. New York: Random House.

Jain, A., D. N. Meenachi, and D. B. Venkatraman. 2020. NukeBERT: A pre-trained language model for low resource nuclear domain, arXiv:2003.13821.

Keskek, S., S. Tse, and J. W. Tucker. 2014. Analyst information production and the timing of annual earnings forecasts. *Review of Accounting Studies* 19 (4): 1504–31.

Kim, Y. 2014. Convolutional neural networks for sentence classification, arXiv:1408.5882.

Kingma, D. P., and J. Ba. 2014. Adam: A method for stochastic optimization, arXiv:1412.6980.

Lang, M., and L. Stice-Lawrence. 2015. Textual analysis and international financial reporting: Large sample evidence. *Journal of Accounting and Economics* 60 (2–3): 110–35.

Lansford, B., J. Lee, and J. W. Tucker. 2009. Disclosure of management guidance in conference calls: Materiality, determinants, and consequences. Working paper.

Lauriola, I., A. Lavelli, and F. Aiolli. 2022. An introduction to deep learning in natural language processing: Models, techniques, and tools. *Neurocomputing* 470: 443–56.

LeCun, Y., Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* 521: 436–44.

Lee, J., W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang. 2019. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36 (4): 1234–40.

Li, F. 2008. Annual report readability, current earnings, and earnings persistence. *Journal of Accounting and Economics* 45 (2–3): 221–47.

Li, F. 2010a. The information content of forward-looking statements in corporate filings—A naïve Bayesian machine learning approach. *Journal of Accounting Research* 48 (5): 1049–102.

Li, F. 2010b. Textual analysis of corporate disclosures: A survey of the literature. *Journal of Accounting Literature* 29: 143–65.

Li, F., R. Lundholm, and M. Minnis. 2013. A measure of competition based on 10-K filings. *Journal of Accounting Research* 51 (2): 399–436.

Liu, Z., D. Huang, K. Huang, Z. Li, and J. Zhao. 2021. FinBERT: A pre-trained financial language representation model for financial text mining. In *Proceedings of the 29th International Conference on International Joint Conferences on Artificial Intelligence*.

Loughran, T., and B. McDonald. 2011. When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks. *Journal of Finance* 66 (1): 35–65.

Loughran, T., and B. McDonald. 2014. Measuring readability in financial disclosures. *Journal of Finance* 69 (4): 1643–71.

Loughran, T., and B. McDonald. 2016. Textual analysis in accounting and finance: A survey. *Journal of Accounting Research* 54 (4): 1187–230.

Lundholm, R. J., R. Rogo, and J. L. Zhang. 2014. Restoring the Tower of Babel: How foreign firms communicate with US investors. *The Accounting Review* 89 (4): 1453–85.

Malo, P., A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology* 65 (4): 782–96.

Manela, A., and A. Moreira. 2017. News implied volatility and disaster concerns. *Journal of Financial Economics* 123 (1): 137–62.

Manning, C. D., and H. Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, 26.

MSCI. 2022. MSCI ESG ratings methodology.

Nayak, A., H. Timmapathini, K. Ponnalagu, and V. G. Venkoparao. 2020. Domain adaptation challenges of BERT in tokenization and sub-word representations of out-of-vocabulary words. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*.

Peters, M. E., M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. 2018. Deep contextualized word representations, arXiv:1802.05365.

Price, S. M., J. S. Doran, D. R. Peterson, and B. A. Bliss. 2012. Earnings conference calls and stock returns: The incremental informativeness of textual tone. *Journal of Banking & Finance* 36 (4): 992–1011.

Purda, L., and D. Skillicorn. 2015. Accounting variables, deception, and a bag of words: Assessing the tools of fraud detection. *Contemporary Accounting Research* 32 (3): 1193–223.

Radford, A., K. Narasimhan, T. Salimans, and I. Sutskever. 2018. Improving language understanding by generative pre-training, https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

Ribeiro, M. T., S. Singh, and C. Guestrin. 2016. “Why should I trust you?” Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Rogers, A., O. Kovaleva, and A. Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics* 8: 842–66.

Schuster, M., and K. K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45 (11): 2673–81.

Schwartz, R., J. Dodge, N. A. Smith, and O. Etzioni. 2020. Green AI. *Communications of the ACM* 63 (12): 54–63.

Sergeev, A., and M. Del Balso. 2018. Horovod: Fast and easy distributed deep learning in TensorFlow, arXiv:1802.05799.

Siano, F., and P. Wysocki. 2021. Transfer learning and textual analysis of accounting disclosures: Applying big data methods to small(er) datasets. *Accounting Horizons* 35 (3): 217–44.

Sinha, K., R. Jia, D. Hupkes, J. Pineau, A. Williams, and D. Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little, arXiv:2104.06644.

Sinha, K., P. Parthasarathi, J. Pineau, and A. Williams. 2021. Unnatural language inference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15 (1): 1929–58.

Strubell, E., A. Ganesh, and A. McCallum. 2019. Energy and policy considerations for deep learning in NLP, arXiv:1906.02243.

Tenney, I., D. Das, and E. Pavlick. 2019. BERT rediscovers the classical NLP pipeline, arXiv:1905.05950.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 30.

Vuong, Q. H. 1989. Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica* 57 (2): 307–33.

Wallach, H. M. 2006. Topic modeling: Beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*.

Wu, Y., M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, and K. Macherey. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation, arXiv:1609.08144.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of this article:
Appendix A. Tutorial on using fine-tuned FinBERT for NLP tasks in Python
Appendix B. Examples of sentences mislabeled by the LM dictionary