FinBERT: Financial Sentiment Analysis with Pre-trained Language Models

submitted in partial fulfillment for the degree of master of science

Dogu Araci
12255068

master information studies
data science
faculty of science
university of amsterdam

2019-06-25

	Internal Supervisor	External Supervisor
Title, Name	Dr Pengjie Ren	Dr Zulkuf Genc
Affiliation	UvA, ILPS	Naspers Group
Email	p.ren@uva.nl	zulkuf.genc@naspers.com

FinBERT: Financial Sentiment Analysis with Pre-trained Language Models

Dogu Tan Araci
dogu.araci@student.uva.nl
University of Amsterdam
Amsterdam, The Netherlands

ABSTRACT

Financial sentiment analysis is a challenging task due to the specialized language and lack of labeled data in that domain. General-purpose models are not effective enough because of specialized language used in financial context. We hypothesize that pre-trained language models can help with this problem because they require fewer labeled examples and they can be further trained on domain-specific corpora. We introduce FinBERT, a language model based on BERT, to tackle NLP tasks in financial domain. Our results show improvement in every measured metric on current state-of-the-art results for two financial sentiment analysis datasets. We find that even with a smaller training set and fine-tuning only a part of the model, FinBERT outperforms state-of-the-art machine learning methods.

1 INTRODUCTION

Prices in an open market reflects all of the available information regarding assets exchanged in an economy [16]. When new information becomes available, all actors in the economy update their positions and prices adjust accordingly, which makes beating the markets consistently impossible. However, the definition of "new information" might change as new information retrieval technologies become available and early-adoption of such technologies might provide an advantage in the short-term.

Analysis of financial texts, be it news, analyst reports or official company announcements is a possible source of new information. With unprecedented amount of such text being created every day, manually analyzing these and deriving actionable insights from them is too big of a task for any single entity. Hence, automated sentiment or polarity analysis of texts produced by financial actors using natural language processing (NLP) methods has gained popularity during the last decade [4].

The principal research interest for this thesis is the polarity analysis, which is classifying text as positive, negative or neutral, in a specific domain. It requires to address two challenges: 1) The most sophisticated classification methods that make use of neural nets require vast amounts of labeled data and labeling financial text snippets requires costly expertise. 2) The sentiment analysis models trained on general corpora are not suited to the task, because financial texts have a specialized language with unique vocabulary and have a tendency to use vague expressions instead of easily-identified negative/positive words.

Using carefully crafted financial sentiment lexicons such as Loughran and McDonald (2011) [11] may seem a solution because they incorporate existing financial knowledge into textual analysis. However, they are based on "word counting" methods, which come short in analyzing deeper semantic meaning of a given text.

NLP transfer learning methods look like a promising solution to both of the challenges mentioned above, and are the focus of this thesis. The core idea behind these models is that by training language models on very large corpora and then initializing down-stream models with the weights learned from the language modeling task, a much better performance can be achieved. The initialized layers can range from the single word embedding layer [23] to the whole model [5]. This approach should, in theory, be an answer to the scarcity of labeled data problem. Language models don't require any labels, since the task is predicting the next word. They can learn how to represent the semantic information. That leaves the fine-tuning on labeled data only the task of learning how to use this semantic information to predict the labels.

One particular component of the transfer learning methods is the ability to further pre-train the language models on domain specific unlabeled corpus. Thus, the model can learn the semantic relations in the text of the target domain, which is likely to have a different distribution than a general corpus. This approach is especially promising for a niche domain like finance, since the language and vocabulary used is dramatically different than a general one.

The goal of this thesis is to test these hypothesized advantages of using and fine-tuning pre-trained language models for financial domain. For that, sentiment of a sentence from a financial news article towards the financial actor depicted in the sentence will be tried to be predicted, using the Financial PhraseBank created by Malo et al. (2014) [17] and FiQA Task 1 sentiment scoring dataset [15].

The main contributions of this thesis are the following:

- We introduce FinBERT, which is a language model based on BERT for financial NLP tasks. We evaluate FinBERT on two financial sentiment analysis datasets.
- We achieve the state-of-the-art on FiQA sentiment scoring and Financial PhraseBank.
- We implement two other pre-trained language models, ULMFit and ELMo for financial sentiment analysis and compare these with FinBERT.
- We conduct experiments to investigate several aspects of the model, including: effects of further pre-training on financial corpus, training strategies to prevent catastrophic forgetting and fine-tuning only a small subset of model layers for decreasing training time without a significant drop in performance.

The rest of the thesis is structured as follows: First, relevant literature in both financial polarity analysis and pre-trained language models are discussed (Section 2). Then, the evaluated models are described (Section 3). This is followed by the description of the experimental setup being used (Section 4). In Section 5, we present

the experimental results on the financial sentiment datasets. Then we further analyze FinBERT from different perspectives in Section 6. Finally, we conclude with Section 7.

2 RELATED LITERATURE

This section describes previous research conducted on sentiment analysis in finance (2.1) and text classification using pre-trained language models (2.2).

2.1 Sentiment analysis in finance

Sentiment analysis is the task of extracting sentiments or opinions of people from written language [10]. We can divide the recent efforts into two groups: 1) Machine learning methods with features extracted from text with "word counting" [1, 19, 28, 30], 2) Deep learning methods, where text is represented by a sequence of embeddings [2, 25, 32]. The former suffers from inability to represent the semantic information that results from a particular sequence of words, while the latter is often deemed as too "data-hungry" as it learns a much higher number of parameters [18].

Financial sentiment analysis differs from general sentiment analysis not only in domain, but also the purpose. The purpose behind financial sentiment analysis is usually guessing how the markets will react with the information presented in the text [9]. Loughran and McDonald (2016) presents a thorough survey of recent works on financial text analysis utilizing machine learning with "bag-of-words" approach or lexicon-based methods [12]. For example, in Loughran and McDonald (2011), they create a dictionary of financial terms with assigned values such as "positive" or "uncertain" and measure the tone of a documents by counting words with a specific dictionary value [11]. Another example is Pagolu et al. (2016), where n-grams from tweets with financial information are fed into supervised machine learning algorithms to detect the sentiment regarding the financial entity mentioned.

On of the first papers that used deep learning methods for textual financial polarity analysis was Kraus and Feuerriegel (2017) [7]. They apply an LSTM neural network to ad-hoc company announcements to predict stock-market movements and show that method to be more accurate than traditional machine learning approaches. They find pre-training their model on a larger corpus to improve the result, however their pre-training is done on a labeled dataset, which is a more limiting approach then ours, as we pre-train a language model as an unsupervised task.

There are several other works that employ various types of neural architectures for financial sentiment analysis. Sohangir et al. (2018) [26] apply several generic neural network architectures to a StockTwits dataset, finding CNN as the best performing neural network architecture. Lutz et al. 2018 [13] take the approach of using *doc2vec* to generate sentence embeddings in a particular company ad-hoc announcement and utilize multi-instance learning to predict stock market outcomes. Maia et al. (2018) [14] use a combination of text simplification and LSTM network to classify a set of sentences from financial news according to their sentiment and achieve state-of-the-art results for the Financial PhraseBank, which is used in thesis as well.

Due to lack of large labeled financial datasets, it is difficult to utilize neural networks to their full potential for sentiment analysis. Even when their first (word embedding) layers are initialized with pre-trained values, the rest of the model still needs to learn complex relations with relatively small amount of labeled data. A more promising solution could be initializing almost the entire model with pre-trained values and fine-tuning those values with respect to the classification task.

2.2 Text classification using pre-trained language models

Language modeling is the task of predicting the next word in a given piece of text. One of the most important recent developments in natural language processing is the realization that a model trained for language modeling can be successfully fine-tuned for most down-stream NLP tasks with small modifications. These models are usually trained on very large corpora, and then with addition of suitable task-specific layers fine-tuned on the target dataset [6]. Text classification, which is the focus of this thesis, is one of the obvious use-cases for this approach.

ELMo (Embeddings from Language Models) [23] was one of the first successful applications of this approach. With ELMo, a deep bidirectional language model is pre-trained on a large corpus. For each word, hidden states of this model is used to compute a contextualized representation. Using the pre-trained weights of ELMo, contextualized word embeddings can be calculated for any piece of text. Initializing embeddings for down-stream tasks with those were shown to improve performance on most tasks compared to static word embeddings such as word2vec or GloVe. For text classification tasks like SST-5, it achieved state-of-the-art performance when used together with a bi-attentive classification network [20].

Although ELMo makes use of pre-trained language models for contextualizing representations, still the information extracted using a language model is present only in the first layer of any model using it. ULMFit (Universal Language Model Fine-tuning) [5] was the first paper to achieve true transfer learning for NLP, as using novel techniques such as discriminative fine-tuning, slanted triangular learning rates and gradual unfreezing. They were able to efficiently fine-tune a whole pre-trained language model for text classification. They also introduced further pre-training of the language model on a domain-specific corpus, assuming target task data comes from a different distribution than the general corpus the initial model was trained on.

ULMFit's main idea of efficiently fine-tuning a pre-trained a language model for down-stream tasks was brought to another level with Bidirectional Encoder Representations from Transformers (BERT) [3], which is also the main focus of this paper. BERT has two important differences from what came before: 1) It defines the task of language modeling as predicting randomly masked tokens in a sequence rather than the next token, in addition to a task of classifying two sentences as following each other or not. 2) It is a very big network trained on an unprecedentedly large corpus. These two factors enabled in to achieve state-of-the-art results in multiple NLP tasks such as, natural language inference or question answering.

The specifics of fine-tuning BERT for text classification has not been researched thoroughly. One such recent work is Sun et al.

(2019) [27]. They conduct a series of experiments regarding different configurations of BERT for text classification. Some of their results will be referenced throughout the rest of the thesis, for the configuration of our model.

3 METHOD

In this section, we will present our BERT implementation for financial domain named as FinBERT, after giving a brief background on relevant neural architectures.

3.1 Preliminaries

3.1.1 LSTM. Long short-term memory (LSTM) is a type of recurrent neural network that allows long-term dependencies in a sequence to persist in the network by using "forget" and "update" gates. It is one of the primary architectures for modeling any sequential data generation process, from stock prices to natural language. Since a text is a sequence of tokens, the first choice for any LSTM natural language processing model is determining how to initially represent a single token. Using pre-trained weights for initial token representation is the common practice. One such pre-training algorithm is GLoVe (Global Vectors for Word Representation) [22]. GLoVr is a model for calculating word representations with the unsupervised task of training a log-bilinear regression model on a word-word co-occurance matrix from a large corpus. It is an effective model for representing words in a vector space, however it doesn't contextualize these representations with respect to the sequence they are actually used in[1].

3.1.2 ELMo. ELMo embeddings [23] are contextualized word representations in the sense that the surrounding words influence the representation of the word. In the center of ELMo, there is a bidirectional language model with multiple LSTM layers. The goal of a language model is to learn the probability distribution over sequences of tokens in a given vocabulary. ELMo models the probability of a token given the previous (and separately following) tokens in the sequence. Then the model also learns how to weight different representations from different LSTM layers in order to calculate one contextualized vector per token. Once the contextualized representations are extracted, these can be used to initialize any down-stream NLP task[2].

3.1.3 ULMFit. ULMFit is a transfer learning model for down-stream NLP tasks, that make use of language model pre-training [5]. Unlike ELMo, with ULMFit, the whole language model is fine-tuned together with the task-specific layers. The underlying language model used in ULMFit is AWD-LSTM, which uses sophisticated dropout tuning strategies to better regularize its LSTM model [21]. For classification using ULMFit two linear layers are added to the pre-trained AWD-LSTM, first of which takes the pooled last hidden states as input.

ULMFit comes with novel training strategies for further pre-training the language model on domain-specific corpus and fine-tuning on the down-stream task. We implement these strategies with FinBERT as explained in section 3.2.

3.1.4 Transformer. The Transformer is an attention-based architecture for modeling sequential information, that is an alternative to recurrent neural networks [29]. It was proposed as a sequence-to-sequence model, therefore including encoder and decoder mechanisms. Here, we will focus only on the encoder part (though decoder is quite similar). The encoder consists of multiple identical Transformer layers. Each layer has a multi-headed self-attention layer and a fully connected feed-forward network. For one self-attention layer, three mappings from embeddings (key, query and value) are learned. Using each token's key and all tokens' query vectors, a similarity score is calculated with dot product. These scores are used to weight the value vectors to arrive at the new representation of the token. With the multi-headed self-attention, these layers are concatenated together, so that the sequence can be evaluated from varying "perspectives". Then the resulted vectors go through fully connected networks with shared parameters.

As it was argued by Vaswani 2017 [29], Transformer architecture has several advantages over the RNN-based approaches. Because of RNNs' sequential nature, they are much harder to parallelize on GPUs and too many steps between far away elements in a sequence make it hard for information to persist.

3.1.5 BERT. BERT [3] is in essence a language model that consists of a set of Transformer encoders stacked on top of each other. However it defines the language modeling task differently from ELMo and AWD-LSTM. Instead of predicting the next word given previous ones, BERT "masks" a randomly selected 15% of all tokens. With a softmax layer over vocabulary on top of the last encoder layer the masked tokens are predicted. A second task BERT is trained on is "next sentence prediction". Given two sentences, the model predicts whether or not these two actually follow each other.

The input sequence is represented with token and position embeddings. Two tokens denoted by [CLS] and [SEP] are added to the beginning and end of the sequence respectively. For all classification tasks, including the next sentence prediction, [CLS] token is used.

BERT has two versions: BERT-base, with 12 encoder layers, hidden size of 768, 12 multi-head attention heads and 110M parameters in total and BERT-large, with 24 encoder layers, hidden size of 1024, 16 multi-head attention heads and 340M parameters. Both of these models have been trained on BookCorpus [33] and English Wikipedia, which have in total more than 3,500M words [3].

3.2 BERT for financial domain: FinBERT

In this subsection we will describe our implementation of BERT: 1) how further pre-training on domain corpus is done, 2-3) how we implemented BERT for classification and regression tasks, 4) training strategies we used during fine-tuning to prevent catastrophic forgetting.

3.2.1 Further pre-training. Howard and Ruder (2018) [5] shows that futher pre-training a language model on a target domain corpus improves the eventual classification performance. For BERT, there is not decisive research showing that would be the case as well.

[1]The pre-trained weights for GLoVE can be found here: https://nlp.stanford.edu/projects/glove/
[2]The pre-trained ELMo models can be found here: https://allennlp.org/elmo
[3]The pre-trained weights are made public by creators of BERT. The code and weights can be found here: https://github.com/google-research/bert

Regardless, we implement further pre-training in order to observe if such adaptation is going to be beneficial for financial domain.

For further pre-training, we experiment with two approaches. The first is pre-training the model on a relatively large corpus from the target domain. For that, we further pre-train a BERT language model on a financial corpus (details of the corpus can be found on section 4.2.1). The second approach is pre-training the model only on the sentences from the training classification dataset. Although the second corpus is much smaller, using data from the direct target might provide better target domain adaptation.

3.2.2 FinBERT for text classification. Sentiment classification is conducted by adding a dense layer after the last hidden state of the [CLS] token. This is the recommended practice for using BERT for any classification task [3]. Then, the classifier network is trained on the labeled sentiment dataset. An overview of all the steps involved in the procedure is presented on figure 1.

3.2.3 FinBERT for regression. While the focus of this paper is classification, we also implement regression with almost the same architecture on a different dataset with continuous targets. The only difference is that the loss function being used is mean squared error instead of the cross entropy loss.

3.2.4 Training strategies to prevent catastrophic forgetting. As it was pointed out by Howard and Ruder (2018) [5], catastrophic forgetting is a significant danger with this fine-tuning approach. Because the fine-tuning procedure can quickly cause model to "forget" the information from language modeling task as it tries to adapt to the new task. In order to deal with this phenomenon, we apply three techniques as it was proposed by Howard and Ruder (2018): slanted triangular learning rates, discriminative fine-tuning and gradual unfreezing.

Slanted triangular learning rate applies a learning rate schedule in the shape of a slanted triangular, that is, learning rate first linearly increases up to some point and after that point linearly decreases.

Discriminative fine-tuning is using lower learning rates for lower layers on the network. Assume our learning rate at layer l is α. Then for discrimination rate of θ we calculate the learning rate for layer $l-1$ as $\alpha_{l-1} = \theta\alpha_l$. The assumption behind this method is that the lower layers represent the deep-level language information, while the upper ones include information for actual classification task. Therefore we fine-tune them differently.

With gradual freezing, we start training with all layers but the classifier layer as frozen. During training we gradually unfreeze all of the layers starting from the highest one, so that the lower level features become the least fine-tuned ones. Hence, during the initial stages of training it is prevented for model to "forget" low-level language information that it learned from pre-training.

4 EXPERIMENTAL SETUP

4.1 Research Questions

We aim to answer the following research questions:

(RQ1) What is the performance of FinBERT in short sentence classification compared with the other transfer learning methods like ELMo and ULMFit?

Table 1: Distribtution of sentiment labels and agreement levels in Financial PhraseBank

Agreement level	Positive	Negative	Neutral	Count
100%	%25.2	%13.4	%61.4	2262
75% - 99%	%26.6	%9.8	%63.6	1191
66% - 74%	%36.7	%12.3	%50.9	765
50% - 65%	%31.1	%14.4	%54.5	627
All	%28.1	%12.4	%59.4	4845

(RQ2) How does FinBERT compare to the state-of-the-art in financial sentiment analysis with targets discrete or continuous?

(RQ3) How does futher pre-training BERT on financial domain, or target corpus, affect the classification performance?

(RQ4) What are the effects of training strategies like slanted triangular learning rates, discriminative fine-tuning and gradual unfreezing on classification performance? Do they prevent catastrophic forgetting?

(RQ5) Which encoder layer performs best (or worse) for sentence classification?

(RQ6) How much fine-tuning is enough? That is, after pre-training, how many layers should be fine-tuned to achieve comparable performance to fine-tuning the whole model?

4.2 Datasets

4.2.1 TRC2-financial. In order to further pre-train BERT, we use a financial corpus we call TRC2-financial. It is a subset of Reuters' TRC2[4], which consists of 1.8M news articles that were published by Reuters between 2008 and 2010. We filter for some financial keywords in order to make corpus more relevant and in limits with the compute power available. The resulting corpus, TRC2-financial, includes 46,143 documents with more than 29M words and nearly 400K sentences.

4.2.2 Financial PhraseBank. The main sentiment analysis dataset used in this paper is Financial PhraseBank[5] from Malo et al. 2014 [17]. Financial Phrasebank consists of 4845 english sentences selected randomly from financial news found on LexisNexis database. These sentences then were annotated by 16 people with background in finance and business. The annotators were asked to give labels according to how they think the information in the sentence might affect the mentioned company stock price. The dataset also includes information regarding the agreement levels on sentences among annotators. The distribution of agreement levels and sentiment labels can be seen on table 1. We set aside 20% of all sentences as test and 20% of the remaining as validation set. In the end, our train set includes 3101 examples. For some of the experiments, we also make use of 10-fold cross validation.

[4]The corpus can be obtained for research purposes by applying here: https://trec.nist.gov/data/reuters/reuters.html

[5]The dataset can be found here: https://www.researchgate.net/publication/251231364_FinancialPhraseBank-v10

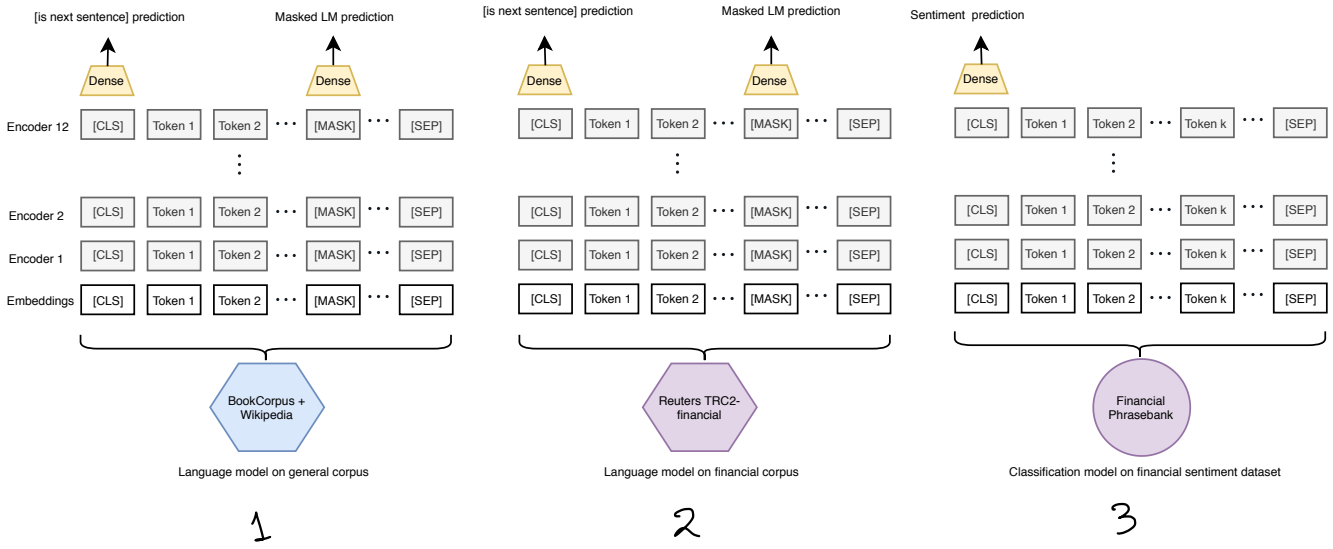

Figure 1: Overview of pre-training, further pre-training and classification fine-tuning

4.2.3 FiQA Sentiment. FiQA [15] is a dataset that was created for WWW '18 conference financial opinion mining and question answering challenge[6]. We use the data for Task 1, which includes 1,174 financial news headlines and tweets with their corresponding sentiment score. Unlike Financial Phrasebank, the targets for this datasets are continuous ranging between $[-1, 1]$ with 1 being the most positive. Each example also has information regarding which financial entity is targeted in the sentence. We do 10-fold cross validation for evaluation of the model for this dataset.

4.3 Baseline Methods

For contrastive experiments, we consider baselines with three different methods: LSTM classifier with GLoVe embeddings, LSTM classifier with ELMo embeddings and ULMFit classifier. It should be noted that these baseline methods are not experimented with as thoroughly as we did with BERT. Therefore the results should not be interpreted as definitive conclusions of one method being better.

4.3.1 LSTM classifiers. We implement two classifiers using bidirectional LSTM models. In both of them, a hidden size of 128 is used, with the last hidden state size being 256 due to bidirectionality. A fully connected feed-forward layer maps the last hidden state to a vector of three, representing likelihood of three labels. The difference between two models is that one uses GLoVe embeddings, while the other uses ELMo embeddings. A dropout probability of 0.3 and a learning rate of 3e-5 is used in both models. We train them until there is no improvement in validation loss for 10 epochs.

4.3.2 ULMFit. As it was explained in section 3.1.3, classification with ULMFit consists of three steps. The first step of pre-training a language model is already done and the pre-trained weights are released by Howard and Ruder (2018). We first further pre-train AWD-LSTM language model on TRC2-financial corpus for 3 epochs. After that, we fine-tune the model for classification on Financial PhraseBank dataset, by adding a fully-connected layer to the output of pre-trained language model.

[6]Data can be found here: https://sites.google.com/view/fiqa/home

4.4 Evaluation Metrics

For evaluation of classification models, we use three metrics: Accuracy, cross entropy loss and macro F1 average. We weight cross entropy loss with square root of inverse frequency rate. For example if a label constitutes 25% of the all examples, we weight the loss attributed to that label by 2. Macro F1 average calculates F1 scores for each of the classes and then takes the average of them. Since our data, Financial PhraseBank suffers from label imbalance (almost 60% of all sentences are neutral), this gives another good measure of the classification performance. For evaluation of regression model, we report mean squared error and R^2, as these are both standard and also reported by the state-of-the-art papers for FiQA dataset.

4.5 Implementation Details

For our implementation BERT, we use a dropout probability of $p = 0.1$, warm-up proportion of 0.2, maximum sequence length of 64 tokens, a learning rate of $2e - 5$ and a mini-batch size of 64. We train the model for 6 epochs, evaluate on the validation set and choose the best one. For discriminative fine-tuning we set the discrimination rate as 0.85. We start training with only the classification layer unfrozen, after each third of a training epoch we unfreeze the next layer. An Amazon p2.xlarge EC2 instance with one NVIDIA K80 GPU, 4 vCPUs and 64 GiB of host memory is used to train the models.

5 EXPERIMENTAL RESULTS (RQ1 & RQ2)

The results of FinBERT, the baseline methods and state-of-the-art on Financial PhraseBank dataset classification task can be seen on table 2. We present the result on both the whole dataset and subset with 100% annotator agreement.

Table 2: Experimental Results on the Financial PhraseBank dataset

	All data			Data with 100% agreement		
Model	Loss	Accuracy	F1 Score	Loss	Accuracy	F1 Score
LSTM	0.81	0.71	0.64	0.57	0.81	0.74
LSTM with ELMo	0.72	0.75	0.7	0.50	0.84	0.77
ULMFit	0.41	0.83	0.79	0.20	0.93	0.91
LPS	-	0.71	0.71	-	0.79	0.80
HSC	-	0.71	0.76	-	0.83	0.86
FinSSLX	-	-	-	-	0.91	0.88
FinBERT	**0.37**	**0.86**	**0.84**	**0.13**	**0.97**	**0.95**

Bold face indicates best result in the corresponding metric. LPS [17], HSC [8] and FinSSLX [15] results are taken from their respective papers. For LPS and HSC, overall accuracy is not reported on the papers. We calculated them using recall scores reported for different classes. For the models implemented by us, we report 10-fold cross validation results.

For all of the measured metrics, FinBERT performs clearly the best among both the methods we implemented ourselves (LSTM and ULMFit) and the models reported by other papers (LPS [17], HSC [8], FinSSLX [14]). LSTM classifier with no language model information performs the worst. In terms of accuracy, it is close to LPS and HSC, (even better than LPS for examples with full agreement), however it produces a low F1-score. That is due to it performing much better in neutral class. LSTM classifier with ELMo embeddings improves upon LSTM with static embeddings in all of the measured metrics. It still suffers from low average F1-score due to poor performance in less represented labels. But it's performance is comparable with LPS and HSC, besting them in accuracy. So contextualized word embeddings produce close performance to machine learning based methods for dataset of this size.

ULMFit significantly improves on all of the metrics and it doesn't suffer from model performing much better in some classes than the others. It also handily beats the machine learning based models LPS and HSC. This shows the effectiveness of language model pre-training. AWD-LSTM is a very large model and it would be expected to suffer from over-fitting with this small of a dataset. But due to language model pre-training and effective training strategies, it is able to overcome small data problem. ULMFit also outperforms FinSSLX, which has a text simplification step as well as pre-training of word embeddings on a large financial corpus with sentiment labels.

FinBERT outperforms ULMFit, and consequently all of the other methods in all metrics. In order to measure the performance of the models on different sizes of labeled training datasets, we ran LSTM classifiers, ULMFit and FinBERT on 5 different configurations. The result can be seen on figure 2, where the cross entropy losses on test set for each model are drawn. 100 training examples is too low for all of the models. However, once the training size becomes 250, ULMFit and FinBERT starts to successfully differentiate between labels, with an accuracy as high as 80% for FinBERT. All of the methods consistently get better with more data, but ULMFit and FinBERT does better with 250 examples than LSTM classifiers do with the whole dataset. This shows the effectiveness of language model pre-training.

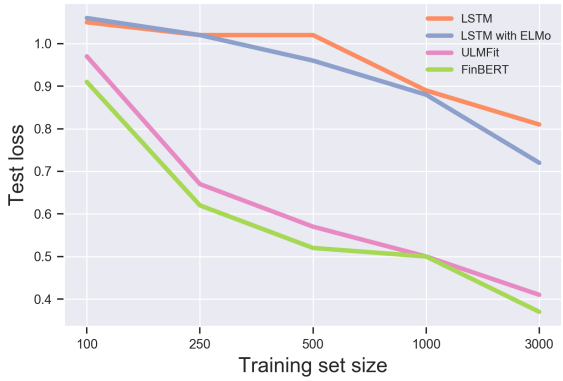

Figure 2: Test loss different training set sizes

The results for FiQA sentiment dataset, are presented on table 3. Our model outperforms state-of-the-art models for both MSE and R^2. It should be noted that the test set these two papers [31] [24] use is the official FiQA Task 1 test set. Since we don't have access to that we report the results on 10-Fold cross validation. There is no indication on [15] that the train and test sets they publish come from different distributions and our model can be interpreted to be at disadvantage since we need to set aside a subset of training set as test set, while state-of-the-art papers can use the complete training set.

6 EXPERIMENTAL ANALYSIS

6.1 Effects of further pre-training (RQ3)

We first measure the effect of further pre-training on the performance of the classifier. We compare three models: 1) No further pre-training (denoted by Vanilla BERT), 2) Further pre-training on classification training set (denoted by FinBERT-task), 3) Further pre-training on domain corpus, TRC2-financial (denoted by FinBERT-domain). Models are evaluated with loss, accuracy and

Table 3: Experimental Results on FiQA Sentiment Dataset

Model	MSE	R^2
Yang et. al. (2018)	0.08	0.40
Piao and Breslin (2018)	0.09	0.41
FinBERT	**0.07**	**0.55**

Bold face indicated best result in corresponding metric. Yang et. al. (2018) [31] and Piao and Breslin (2018) [24] report results on the official test set. Since we don't have access to that set our MSE, and R^2 are calculated with 10-Fold cross validation.

WTF?

Table 4: Performance with different pre-training strategies

Model	Loss	Accuracy	F1 Score
Vanilla BERT	0.38	0.85	0.84
FinBERT-task	0.39	0.86	**0.85**
FinBERT-domain	**0.37**	**0.86**	0.84

Bold face indicates best result in the corresponding metric. Results are reported on 10-fold cross validation.

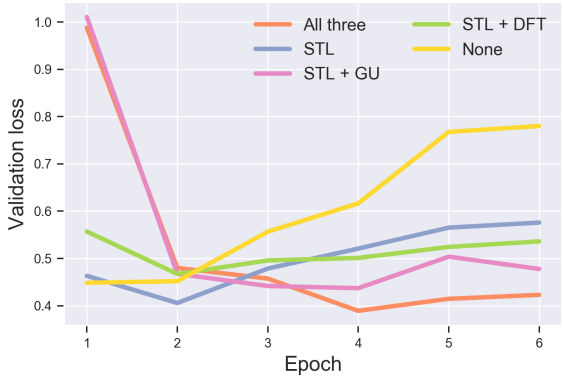

Figure 3: Validation loss trajectories with different training strategies

Table 5: Performance with different fine-tuning strategies

Strategy	Loss	Accuracy	F1 Score
None	0.48	0.83	0.83
STL	0.40	0.81	0.82
STL + GU	0.40	0.86	**0.86**
STL + DFT	0.42	0.79	0.79
All three	**0.37**	**0.86**	0.84

Bold face indicates best result in the corresponding metric. Results are reported on 10-fold cross validation. STL: slanted triangular learning rates, GU: gradual unfreezing, DFT: discriminative fine-tuning.

macro average F1 scores on the test dataset. The results can be seen on table 4.

The classifier that were further pre-trained on financial domain corpus performs best among the three, though the difference is not very high. There might be four reasons behind this result: 1) The corpus might have a different distribution than the task set, 2) BERT classifiers might not improve significantly with further pre-training, 3) Short sentence classification might not benefit significantly from further pre-training, 4) Performance is already so good, that there is not much room for improvement. We think that the last explanation is the likeliest, because for the subset of Financial Phrasebank that all of the annotators agree on the result, accuracy of Vanilla BERT is already 0.96. The performance on the other agreement levels should be lower, as even the humans can't agree fully on them. More experiments with another financial labeled dataset is necessary to conclude that effect of further pre-training on domain corpus is not significant.

6.2 Catastrophic forgetting (RQ4)

For measuring the performance of the techniques against catastrophic forgetting, we try four different settings: No adjustment (NA), only with slanted triangular learning rate (STL), slanted triangular learning rate and gradual unfreezing (STL+GU) and the techniques in the previous one, together with discriminative fine-tuning. We report the performance of these four settings with loss on test function and trajectory of validation loss over training epochs. The results can be seen on table 5 and figure 3.

Applying all three of the strategies produce the best performance in terms of test loss and accuracy. Gradual unfreezing and discriminative fine-tuning have the same reasoning behind them: higher level features should be fine-tuned more than the lower level ones, since information learned from language modeling are mostly present in the lower levels. We see from table 5 that using only discriminative fine-tuning with slanted triangular learning rates performs worse than using the slanted triangular learning rates alone. This shows that gradual unfreezing is the most important technique for our case.

One way that catastrophic forgetting can show itself is the sudden increase in validation loss after several epochs. As model is trained, it quickly starts to overfit when no measure is taken accordingly. As it can be seen on the figure 3, that is the case when none of the aforementioned techniques are applied. The model achieves the best performance on validation set after the first epoch and then starts to overfit. While with all three techniques applied, model is much more stable. The other combinations lie between these two cases.

6.3 Choosing the best layer for classification (RQ5)

BERT has 12 Transformer encoder layers. It is not necessarily a given that the last layer captures the most relevant information regarding classification task during language model training. For

Table 6: Performance on different encoder layers used for classification

Layer for classification	Loss	Accuracy	F1 Score
Layer-1	0.65	0.76	0.77
Layer-2	0.54	0.78	0.78
Layer-3	0.52	0.76	0.77
Layer-4	0.48	0.80	0.77
Layer-5	0.52	0.80	0.80
Layer-6	0.45	0.82	0.82
Layer-7	0.43	0.82	0.83
Layer-8	0.44	0.83	0.81
Layer-9	0.41	0.84	0.82
Layer-10	0.42	0.83	0.82
Layer-11	0.38	0.84	0.83
Layer-12	0.37	0.86	0.84
All layers - mean	0.41	0.84	0.84

Table 7: Performance on starting training from different layers

First layer unfreezed	Loss	Accuracy	Training time
Embeddings layer	0.37	0.86	332s
Layer-1	0.39	0.83	302s
Layer-2	0.39	0.83	291s
Layer-3	0.38	0.83	272s
Layer-4	0.38	0.82	250s
Layer-5	0.40	0.83	240s
Layer-6	0.40	0.81	220s
Layer-7	0.39	0.82	205s
Layer-8	0.39	0.84	188s
Layer-9	0.39	0.84	172s
Layer-10	0.41	0.84	158s
Layer-11	0.45	0.82	144s
Layer-12	0.47	0.81	133s
Classification layer	1.04	0.52	119s

this experiment, we investigate which layer out of 12 Transformer encoder layers give the best result for classification. We put the classification layer after the CLS] tokens of respective representations. We also try taking the average of all layers.

As shown in table 6the last layer contributes the most to the model performance in terms of all the metrics measured. This might be indicative of two factors: 1) When the higher layers are used the model that is being trained is larger, hence possibly more powerful, 2) The lower layers capture deeper semantic information, hence they struggle to fine-tune that information for classification.

6.4 Training only a subset of the layers (RQ6)

BERT is a very large model. Even on small datasets, fine-tuning the whole model requires significant time and computing power. Therefore if a slightly lower performance can be achieved with fine-tuning only a subset of all parameters, it might be preferable in some contexts. Especially if training set is very large, this change might make BERT more convenient to use. Here we experiment with fine-tuning only the last k many encoder layers.

The results are presented on table 7. Fine-tuning only the classification layer does not achieve close performance to fine-tuning other layers. However fine-tuning only the last layer handily outperforms the state-of-the-art machine learning methods like HSC. After Layer-9, the performance becomes virtually the same, only to be outperformed by fine-tuning the whole model. This result shows that in order to utilize BERT, an expensive training of the whole model is not mandatory. A fair trade-off can be made for much less training time with a small decrease in model performance.

6.5 Where does the model fail?

With 97% accuracy on the subset of Financial PhraseBank with 100% annotator agreement, we think it might be an interesting exercise to examine cases where the model failed to predict the true label. Therefore in this section we will present several examples where model makes the wrong prediction. Also in Malo et al. (2014)[17], it is indicated that most of the inter-annotator disagreements are between positive and neutral labels (agreement for separating positive-negative, negative-neutral and positive-neutral are 98.7%, 94.2% and 75.2% respectively). Authors attribute that the difficulty of distinguishing "commonly used company glitter and actual positive statements". We will present the confusion matrix in order to observe whether this is the case for FinBERT as well.

Example 1: `Pre-tax loss totaled euro 0.3 million , compared to a loss of euro 2.2 million in the first quarter of 2005 .`
True value: Positive **Predicted:** Negative

Example 2: `This implementation is very important to the operator , since it is about to launch its Fixed to Mobile convergence service in Brazil`
True value: Neutral **Predicted:** Positive

Example 3: `The situation of coated magazine printing paper will continue to be weak .`
True value: Negative **Predicted:** Neutral

The first example is actually the most common type of failure. The model fails to do the math in which figure is higher, and in the absence of words indicative of direction like "increased", might make the prediction of neutral. However, there are many similar cases where it does make the true prediction too. Examples 2 and 3 are different versions of the same type of failure. The model fails to distinguish a neutral statement about a given situation from a statement that indicated polarity about the company. In the third example, information about the company's business would probably help.

The confusion matrix is presented on figure 4. 73% of the failures happen between labels positive and negative, while same number is 5% for negative and positive. That is consistent with both the inter-annotator agreement numbers and common sense. It is easier

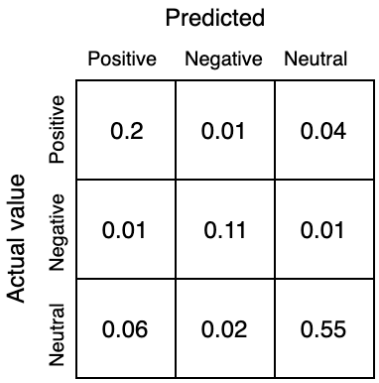

Figure 4: Confusion matrix

to differentiate between positive and negative. But it might be more challenging to decide whether a statement indicates a positive outlook or merely an objective observation.

7 CONCLUSION AND FUTURE WORK

In this paper, we implemented BERT for the financial domain by further pre-training it on a financial corpus and fine-tuning it for sentiment analysis (FinBERT). This work is the first application of BERT for finance to the best of our knowledge and one of the few that experimented with further pre-training on a domain-specific corpus. On both of the datasets we used, we achieved state-of-the-art results by a significant margin. For the classification task, we increased the state-of-the art by 15% in accuracy.

In addition to BERT, we also implemented other pre-training language models like ELMo and ULMFit for comparison purposes. ULMFit, further pre-trained on a financial corpus, beat the previous state-of-the art for the classification task, only to a smaller degree than BERT. These results show the effectiveness of pre-trained language models for a down-stream task such as sentiment analysis especially with a small labeled dataset. The complete dataset included more than 3000 examples, but FinBERT was able to surpass the previous state-of-the art even with a training set as small as 500 examples. This is an important result, since deep learning techniques for NLP have been traditionally labeled as too "data-hungry", which is apparently no longer the case.

We conducted extensive experiments with BERT, investigating the effects of further pre-training and several training strategies. We couldn't conclude that further pre-training on a domain-specific corpus was significantly better than not doing so for our case. Our theory is that BERT already performs good enough with our dataset that there is not much room for improvement that further pre-training can provide. We also found that learning rate regimes that fine-tune the higher layers more aggressively than the lower ones perform better and are more effective in preventing catastrophic forgetting. Another conclusion from our experiments was that, comparable performance can be achieved with much less training time by fine-tuning only the last 2 layers of BERT.

Financial sentiment analysis is not a goal on its own, it is as useful as it can support financial decisions. One way that our work might be extended, could be using FinBERT directly with stock market return data (both in terms of directionality and volatility) on financial news. FinBERT is good enough for extracting explicit sentiments, but modeling implicit information that is not necessarily apparent even to those who are writing the text should be a challenging task. Another possible extension can be using FinBERT for other natural language processing tasks such as named entity recognition or question answering in financial domain.

8 ACKNOWLEDGEMENTS

I would like to show my gratitude to Pengjie Ren and Zulkuf Genc for their excellent supervision. They provided me with both independence in setting my own course for the research and valuable suggestions when I need them. I would also like to thank Naspers AI team, for entrusting me with this project and always encouraging me to share my work. I am grateful to NIST, for sharing Reuters TRC-2 corpus with me and to Malo et al. for making the excellent Financial PhraseBank publicly available.

REFERENCES

[1] Basant Agarwal and Namita Mittal. 2016. *Machine Learning Approach for Sentiment Analysis.* Springer International Publishing, Cham, 21–45. https://doi.org/10.1007/978-3-319-25343-5_3

[2] Oscar Araque, Ignacio Corcuera-Platas, J. Fernando Sánchez-Rada, and Carlos A. Iglesias. 2017. Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications* 77 (jul 2017), 236–246. https://doi.org/10.1016/j.eswa.2017.02.002

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2018). https://doi.org/arXiv:1811.03600v2 arXiv:1810.04805

[4] Li Guo, Feng Shi, and Jun Tu. 2016. Textual analysis and machine leaning: Crack unstructured data in finance and accounting. *The Journal of Finance and Data Science* 2, 3 (sep 2016), 153–170. https://doi.org/10.1016/J.JFDS.2017.02.001

[5] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. (jan 2018). arXiv:1801.06146 http://arxiv.org/abs/1801.06146

[6] Neel Kant, Raul Puri, Nikolai Yakovenko, and Bryan Catanzaro. 2018. Practical Text Classification With Large Pre-Trained Language Models. (2018). arXiv:1812.01207 http://arxiv.org/abs/1812.01207

[7] Mathias Kraus and Stefan Feuerriegel. 2017. Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems* 104 (2017), 38–48. https://doi.org/10.1016/j.dss.2017.10.001 arXiv:1710.03954

[8] Srikumar Krishnamoorthy. 2018. Sentiment analysis of financial news articles using performance indicators. *Knowledge and Information Systems* 56, 2 (aug 2018), 373–394. https://doi.org/10.1007/s10115-017-1134-1

[9] Xiaodong Li, Haoran Xie, Li Chen, Jianping Wang, and Xiaotie Deng. 2014. News impact on stock price return via sentiment analysis. *Knowledge-Based Systems* 69 (oct 2014), 14–23. https://doi.org/10.1016/j.knosys.2014.04.022

[10] Bing Liu. 2012. Sentiment Analysis and Opinion Mining. *Synthesis Lectures on Human Language Technologies* 5, 1 (may 2012), 1–167. https://doi.org/10.2200/s00416ed1v01y201204hlt016

[11] Tim Loughran and Bill Mcdonald. 2011. When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks. *Journal of Finance* 66, 1 (feb 2011), 35–65. https://doi.org/10.1111/j.1540-6261.2010.01625.x

[12] Tim Loughran and Bill Mcdonald. 2016. Textual Analysis in Accounting and Finance: A Survey. *Journal of Accounting Research* 54, 4 (2016), 1187–1230. https://doi.org/10.1111/1475-679X.12123

[13] Bernhard Lutz, Nicolas Pröllochs, and Dirk Neumann. 2018. *Sentence-Level Sentiment Analysis of Financial News Using Distributed Text Representations and Multi-Instance Learning.* Technical Report. arXiv:1901.00400 http://arxiv.org/abs/1901.00400

[14] Macedo Maia, Andrïŧ¡ Freitas, and Siegfried Handschuh. 2018. FinSSLx: A Sentiment Analysis Model for the Financial Domain Using Text Simplification. In *2018 IEEE 12th International Conference on Semantic Computing (ICSC).* IEEE, 318–319. https://doi.org/10.1109/ICSC.2018.00065

[15] Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross Mcdermott, Manel Zarrouk, Alexandra Balahur, and Ross Mc-Dermott. 2018. Companion of the The Web Conference 2018 on The Web Conference 2018, {WWW} 2018, Lyon , France, April 23-27, 2018. ACM. https://doi.org/10.1145/3184558

[16] Burton G Malkiel. 2003. The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives* 17, 1 (feb 2003), 59–82. https://doi.org/10.1257/

089533003321164958

[17] Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology* 65, 4 (2014), 782–796. https://doi.org/10.1002/asi.23062 arXiv:arXiv:1307.5336v2

[18] G. Marcus. 2018. Deep Learning: A Critical Appraisal. *arXiv e-prints* (Jan. 2018). arXiv:cs.AI/1801.00631

[19] Justin Martineau and Tim Finin. 2009. Delta TFIDF: An Improved Feature Space for Sentiment Analysis.. In *ICWSM*, Eytan Adar, Matthew Hurst, Tim Finin, Natalie S. Glance, Nicolas Nicolov, and Belle L. Tseng (Eds.). The AAAI Press. http://dblp.uni-trier.de/db/conf/icwsm/icwsm2009.html#MartineauF09

[20] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in Translation: Contextualized Word Vectors. Nips (2017), 1–12. arXiv:1708.00107 http://arxiv.org/abs/1708.00107

[21] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and Optimizing LSTM Language Models. *CoRR* abs/1708.02182 (2017). arXiv:1708.02182 http://arxiv.org/abs/1708.02182

[22] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[23] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. (2018). https://doi.org/10.18653/v1/N18-1202 arXiv:1802.05365

[24] Guangyuan Piao and John G Breslin. 2018. Financial Aspect and Sentiment Predictions with Deep Neural Networks. 1973–1977. https://doi.org/10.1145/3184558.3191829

[25] Aliaksei Severyn and Alessandro Moschitti. 2015. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR '15*. ACM Press. https://doi.org/10.1145/2766462.2767830

[26] Sahar Sohangir, Dingding Wang, Anna Pomeranets, and Taghi M Khoshgoftaar. 2018. Big Data: Deep Learning for financial sentiment analysis. *Journal of Big Data* 5, 1 (2018). https://doi.org/10.1186/s40537-017-0111-6

[27] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification? (2019). arXiv:1905.05583 https://arxiv.org/pdf/1905.05583v1.pdfhttp://arxiv.org/abs/1905.05583

[28] Abinash Tripathy, Ankit Agrawal, and Santanu Kumar Rath. 2016. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications* 57 (sep 2016), 117–126. https://doi.org/10.1016/j.eswa.2016.03.028

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. Nips (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[30] Casey Whitelaw, Navendu Garg, and Shlomo Argamon. 2005. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM '05*. ACM Press. https://doi.org/10.1145/1099554.1099714

[31] Steve Yang, Jason Rosenfeld, and Jacques Makutonin. 2018. Financial Aspect-Based Sentiment Analysis using Deep Representations. (2018). arXiv:1808.07931 https://arxiv.org/pdf/1808.07931v1.pdfhttp://arxiv.org/abs/1808.07931

[32] Lei Zhang, Shuai Wang, and Bing Liu. 2018. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8, 4 (mar 2018), e1253. https://doi.org/10.1002/widm.1253

[33] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. (jun 2015). arXiv:1506.06724 http://arxiv.org/abs/1506.06724