Decision Support Systems 104 (2017) 38–48

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Decision support from financial disclosures with deep neural networks and transfer learning

Mathias Kraus[a,*], Stefan Feuerriegel[a,b]

[a] Chair for Information Systems Research, University of Freiburg, Platz der Alten Synagoge, Freiburg 79098, Germany
[b] ETH Zurich, Weinbergstr. 56/58, Zurich 8092, Switzerland

ARTICLE INFO

Article history:
Received 13 March 2017
Received in revised form 6 September 2017
Accepted 3 October 2017
Available online 9 October 2017

Keywords:
Decision support
Deep learning
Transfer learning
Text mining
Financial news
Machine learning

ABSTRACT

Company disclosures greatly aid in the process of financial decision-making; therefore, they are consulted by financial investors and automated traders before exercising ownership in stocks. While humans are usually able to correctly interpret the content, the same is rarely true of computerized decision support systems, which struggle with the complexity and ambiguity of natural language. A possible remedy is represented by deep learning, which overcomes several shortcomings of traditional methods of text mining. For instance, recurrent neural networks, such as long short-term memories, employ hierarchical structures, together with a large number of hidden layers, to automatically extract features from ordered sequences of words and capture highly non-linear relationships such as context-dependent meanings. However, deep learning has only recently started to receive traction, possibly because its performance is largely untested. Hence, this paper studies the use of deep neural networks for financial decision support. We additionally experiment with transfer learning, in which we pre-train the network on a different corpus with a length of 139.1 million words. Our results reveal a higher directional accuracy as compared to traditional machine learning when predicting stock price movements in response to financial disclosures. Our work thereby helps to highlight the business value of deep learning and provides recommendations to practitioners and executives.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

The semi-strong form of the efficient market hypothesis states that asset prices adapt to new information entering the market [1]. Included among these information signals are the regulatory disclosures of firms, as these financial materials trigger subsequent movements of stock prices [2–5]. Hence, investors must evaluate the content of financial disclosures and then decide upon the new valuation of stocks. Here a financial decision support system can greatly facilitate the decision-making of investors subsequent to the disclosure of financial statements [6–11]. Corresponding decision support systems, such as those used by automated traders, can thereby help identify financially rewarding stocks and exercise ownership.

Decision support systems for news-based trading commonly consist of different components [6,8-11], as schematically illustrated in Fig. 1. On the one hand, they need to assess the information encoded in the narratives of financial disclosures. For this purpose, a decision support system must rate the content of such disclosures in order to identify which stock prices are likely to surge or decrease. In other words, the system must quantify whether a financial disclosure conveys positive or negative content. For example, a prediction engine can forecast the expected price change subsequent to a disclosure. Afterwards, the trading engine decides whether to invest in a stock given the market environment. It also performs risk evaluations, and, if necessary, applies changes to the portfolio. The resulting financial performance of the portfolio largely depends upon the accuracy of the prediction engine, which constitutes the focus of this manuscript. Here even small improvements in prediction performance directly link to better decision-making and thus an increase in monetary profits.

Mathematically, the prediction takes a document $d \in D$ as input and then returns either the expected (excess) return or a class label denoting the direction of the price change (i. e. positive or negative). Here a document d is expressed as an ordered list $[w_1, w_2, \ldots, w_m]$ of words, where the length m of this list differs from document to document. When utilizing a traditional predictive model from machine learning (such as a support vector machine), the ordered list of length m is mapped onto a vector of length N that serves as input to the predictor. Independent of the varying length m, this predictor always

* Corresponding author.
E-mail addresses: mathias.sebastian.kraus@gmail.com (M. Kraus), sfeuerriegel@ethz.ch (S. Feuerriegel).

https://doi.org/10.1016/j.dss.2017.10.001
0167-9236/© 2017 Elsevier B.V. All rights reserved.

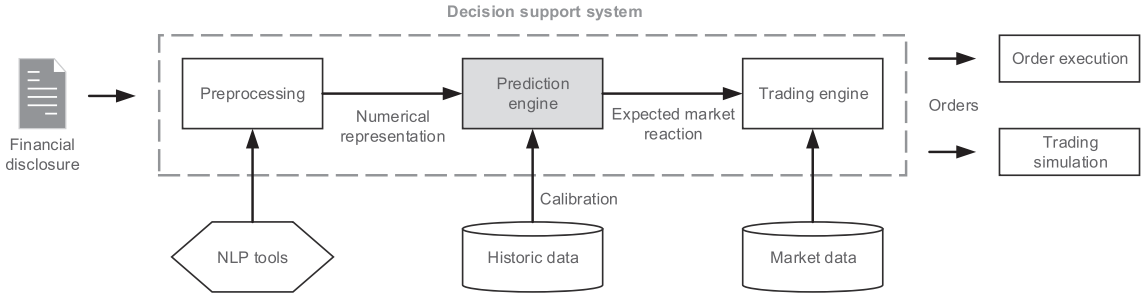

Fig. 1. Schematic illustration of a decision support system for news-based trading. The components are adapted from the systems described in [8–12]. Here NLP refers to natural language processing. This work focuses on improvements to the underlying prediction engine.

entails the same length *N*. For this mapping operation, one primarily follows the bag-of-words approach [13,14], which we detail in the following.

The bag-of-words approach [13] counts the frequency of words (or tuples, so-called *n*-grams), while neglecting the order in which these words (or tuples) are arranged. Hence, this approach does not take into account whether one word (or *n*-gram) appears before or after another.[1] Accordingly, the bag-of-words approach loses information concerning the meaning in a specific context [14–16] and thus struggles with very long context dependencies that might span several sentences or paragraphs. The aforementioned properties underlying the bag-of-words approach are likely to explain why the accuracy of predictive models forecasting stock price movements on the basis of financial narratives is often regarded unsatisfactory [15].

As an alternative to the bag-of-words approach, this paper utilizes recent advances in deep learning or, more precisely, sequence modeling based on deep neural networks. When applied to our case, these models allow us to consider very long context dependencies [17], thereby improving predictive power. The underlying reason is as follows: deep neural networks for sequence modeling iterate over the running text word-by-word while learning a lower-dimensional representation of dimension *n*. By processing the text word-for-word, this approach preserves the word order and incorporates information concerning the context. Moreover, deep learning provides a very powerful framework when using large datasets in a variety of predictive tasks, as it is capable of modeling complex, non-linear relationships between variables or observations [17]. Among the popular variants of deep neural networks for sequence modeling are recurrent neural networks (RNNs) and the long short-term memory (LSTM) model. Sequence modeling has successfully demonstrated the ability to store even long-ranging context information in the weights of network [17]. For instance, in the related field of process mining, LSTMs have proven capable of effectively learning long sequences [18]. Hence, sequence modeling also promises improvements to the predictive power of decision support systems for news-based trading.

Despite recent breakthroughs in deep learning, our literature survey reveals that this approach is seldom employed in financial decision support systems. This gives one the impression that the business value of deep learning in practical applications is generally unknown. Accordingly, this paper sets out to address the following two research questions: (1) Can sequence modeling with deep learning improve the prediction of short-term price movements subsequent to a financial disclosure as compared to bag-of-words approaches? (2) Can we further bolster the predictive performance of deep neural networks by applying transfer learning?

As a primary contribution to the existing body of research, this work utilizes deep learning to predict stock returns subsequent to the disclosure of financial materials and evaluates the predictive power thereof. Each of the neural networks entails more than 500,000 parameters that empower various variants of non-linearities. We then validate our results using an extensive collection of baseline methods, which are state of the art for bag-of-words. In addition, we tune the performance of our methods by applying concepts from transfer learning. That is, we perform representation learning (i. e. pre-training word embeddings) using a different, but related, corpus with financial language and then transfer the resulting word embeddings to the dataset under study. Based on our findings, we provide managerial guidelines and recommendations for deep learning in financial decision support.

The remainder of this paper is structured as follows. Section 2 provides an overview of related works on both decision support from financial news and deep learning for natural language processing. Section 3 then presents our baseline models, our network architectures, and our approach to transfer learning. These are utilized in Section 4 to evaluate how deep learning can improve decision support in finance. Finally, Section 5 discusses our findings and highlights the implications of our work for research and management. Section 6 concludes.

2. Related work

2.1. Decision support from financial news

Decision support from financial news is either of an explanatory or predictive nature. The former tries to explain the relationship between financial news and stock prices based on historic data. Specifically, this field of research utilizes econometrics in order to quantify the impact of news, establish a causal relationship between news and stock prices or understand which investors respond to news and how. Recent literature surveys [2–5] provide a detailed overview of explanatory works, finding that these commonly count the instances of polarity cues, predefined by manually created dictionaries.

In contrast, predictive approaches forecast the future reception of financial news by the stock market [15,16]. For this purpose, decision support systems are trained on historic data with the specific objective of performing accurately and reliably on unseen news. The resulting directional accuracy is usually only marginally better than 50% (and often merely on a subset of the original dataset), which demonstrates how challenging this task is. The approaches vary in terms of the underlying data source and predictive model, which we discuss in the following paragraphs.

The text source can come in various forms such as, e. g., headlines of news stories [19], the content of newspaper articles [11,20-23]

[1] For instance, let us consider the examples *"The decision was not good; it was actually quite bad"* and *"The decision was not bad; it was actually quite good"*. When counting only the frequency of individual words, one cannot recognize that negation that changes the meaning of *"good"* and *"bad"*. Similar examples can also be constructed for *n*-grams.

or company-specific disclosures [6,24]. In addition, research widely conducts numerical experiments with financial prices in daily resolution [15], and we adhere to this convention. For other resolutions, we refer to earlier works [e. g. [25]] that further analyze the time dimension of news reception, including latency and peak effects in intraday trading.

In order to insert natural language into predictive models, the bag-of-words approach is widely utilized for the purpose of extracting numeric representations from the textual content [15]. Commonly, additional scalings are applied, such as term frequency-inverse document frequency (tf-idf). Accordingly, this study also utilizes the tf-idf approach for bag-of-words models, as our experiments demonstrate superior performance as compared to word frequencies. This numerical representation is then fed into predictive models, which are then trained on historic news and stock prices. Prevalent predictive models for financial news are naïve Bayes, regressions, support vector machines and decision trees [15]. All these methods perform well in situations with few observations and many potential regressors. Even though deep learning has achieved impressive results in natural language processing [26,27], this type of predictive model has been largely neglected in financial text mining. A noteworthy exception is the work in [12]; however, it implements a very early form of deep learning – namely, recursive autoencoders – which can store context-sensitive information only for the course of a few words and is thus limited to learning simple semantics. The SemEval-2017 competition is currently raising awareness in this regard [28].

2.2. Natural language processing with deep learning

For a long time, text mining was dominated by traditional machine learning methods, such as support vector machines, trained with high-dimensional yet very sparse feature vectors. It is only recently that researchers in the field of natural language processing have started to adapt ideas from advances in deep learning [26,27].[2] The utilized deep learning techniques are described in detail in [17,29].

The recurrent neural network processes raw text in sequential order [30]. The connections in the neural network form a direct cycle, which allows for the passing of information from one word to the next. This helps the RNN to implicitly learn context-sensitive features. However, the RNN is subject to drawbacks (vanishing gradient problem and short context dependencies), which often prohibit its application to real-world problems [31].

An improvement to the classical RNN is represented by the long short-term memory model, which is capable of processing sequential inputs with very long dependencies between related input signals [32]. For this purpose, the LSTM utilizes forget gates that prevent exploding gradients during back-propagation and thus numerical instabilities. As a consequence, LSTMs have become state of the art in many fields of research [17] and we thus apply this deep learning architecture in our study on financial decision support.

2.3. Deep learning in financial applications

Despite being a very powerful framework, deep learning has rarely been used in finance research. Among the few such instances, financial time series prediction is one popular application. Here previous research utilizes an autoencoder composed of stacked restricted Boltzmann machines in order to predict future stock prices based on historic time series [33]. Similarly, the LSTM is applied to predict future stock returns and generates a portfolio of stocks that can yield higher returns than the top 10 stocks in the S&P 500 [34].

Recent literature surveys [15,16] do not mention works that utilize deep learning for financial text mining; yet we found a two-stage approach that extracts specific word triples consisting of an actor, an action and an object from more than 400,000 headlines of Bloomberg news and then applies deep learning in order to predict stock price movements [35]. However, this setup diminishes the advantage of deep learning, as it computes word tuples instead of processing the raw text. Closest to our research is the application of a recursive autoencoder to the headlines of approximately 6500 financial disclosures [12]. This early work trains a recursive autoencoder and uses the final code vector to predict stock price movements. However, this network architecture can rarely learn long context dependencies [17] and thus struggles with complex semantic relationships. Furthermore, the dataset is subject to extensive filtering in order to yield a performance that, ultimately, is only marginally better than random guessing.

Based on our review, we are not aware of any works that utilize recent advances in deep learning – namely, recurrent neural networks or LSTMs – in order to improve decision support based on financial news.

3. Methods and materials

This section introduces our methodology, as well as the dataset, to predict stock price movements on the basis of financial disclosures. In brief, we compare naïve machine learning using bag-of-words with novel deep learning techniques (see Fig. 2).

We specifically experiment with (a)~classification, where we assign the direction of the stock price movement – *up* or *down* – to a financial disclosure, and (b)~regression, where we predict the magnitude of the change. In both cases, we study price changes in terms of both nominal returns and abnormal returns. The latter corrects returns for confounding market movements and isolates the effect of the news release itself.

3.1. Dataset

Our corpus comprises 13,135 regulated German ad hoc announcements in English.[3] This type of financial disclosure is an important source of information, since listed companies are obliged by law to publish these disclosures in order to inform investors about relevant company occurrences (cf. German Securities Prospectus Act). They have shown a strong influence on financial markets and have been a popular choice in previous research [24,25].

Consistent with previous research, we reduce noise by omitting the disclosures of penny stock companies, i. e. those with a stock price below €5. In addition, we only select disclosures published on trading days. This yields a sample of 10,895 observations. We compute abnormal returns with daily stock market data using a market model whereby the market is modeled via the CDAX during the 20 trading days prior to the disclosure. In the classification task, we label each disclosure as *positive* (encoded as 1) or *negative* (encoded as 0) based on the sign of the corresponding return.

Table 1 shows descriptive statistics. Ad hoc releases are composed of 168.80 words on average, which is significantly longer than the documents used in most applications of deep learning in the field of natural language processing. In total, 12,444 unique words appear in our dataset. The distributions of both abnormal and nominal return

[2] A comprehensive overview on deep learning for natural language processing is given in the tutorial of Richard Socher and Christopher Manning held at NAACL HLT, 2013. URL: http://nlp.stanford.edu/courses/NAACL2013/, visited on July 6, 2017.

[3] These disclosures are publicly available via the website of the "DGAP" (www.dgap.de). Moreover, the specific dataset of this study can be downloaded from https://github.com/MathiasKraus/FinancialDeepLearning.

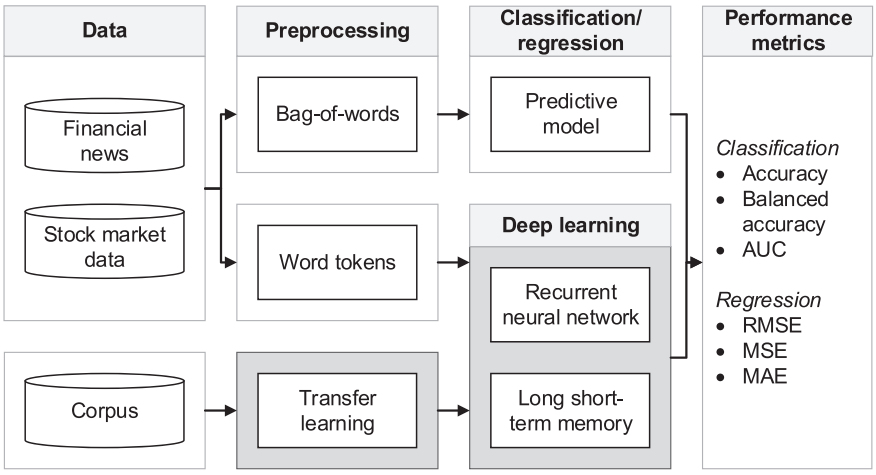

Fig. 2. Research framework evaluating the performance gains from deep learning architectures and transfer learning.

Table 1
Summary statistics of the stock market data, as well as the length of disclosures.

Variable	Obs.	Mean	Std. dev.	Coef. of var.	Percentiles				
					5 %	25 %	50 %	75 %	95 %
Abnormal return	10,895	0.699	6.923	9.908	−7.639	−1.712	0.223	2.749	10.010
Nominal return	10,895	0.778	6.876	8.835	−7.410	1.530	0.220	2.757	9.961
Length (in words)	10,895	168.801	117.397	0.695	61.000	95.000	137.000	203.000	381.000

are substantially right-skewed, as indicated by the 25% and 75% percentiles. Our dataset contains a few market movements of larger magnitude due to a number of factors, including mergers and acquisitions, as well as bankruptcy. We deliberately include these values in our data sample as a decision support system in live application can neither recognize nor filter them. We further observe an unbalanced set of labels, a fact which needs to be adjusted for in the performance measurements. For the nominal return, a positive label appears 9% more frequently than a negative label.

To measure the predictive performance, we split the dataset into a training and a test set. The first 80% of the time frame gives our training data, while the last 20% defines our test set. This differs from earlier studies which appear to draw hold-out samples from the same time period as the training set [36]. As a drawback, the latter process ignores the chronological order of disclosures and, hence, training would erroneously benefit from data samples that otherwise are only ex post available.[4] We thus follow [37,38] and split the training and test sets in chronological order. A similar approach is applied in cross-validation, as detailed later. After splitting, the dataset contains 8716 disclosures for training and 2179 for testing.

3.2. Baselines with bag-of-words

This section briefly describes baseline models based on bag-of-words. First, we tokenize each document, convert all characters to lower case and remove punctuation as well as numbers. Moreover, we perform stemming [13], which maps inflected words onto a base form; e. g. *"increased"* and *"increasing"* are both mapped onto *"increas"* [39]. Since we a priori cannot know whether stemming actually improves the predictive performance, we later incorporate this decision as a tuning parameter. Thereby, stemming is only utilized when it actually improves the predictive performance on the validation set. We then transform the preprocessed content into numerical feature vectors by utilizing the tf-idf approach, which puts stronger weights on characteristic terms [13]. Furthermore, we report the results from using unigrams as part of our evaluation. In addition, we later perform a sensitivity analysis and incorporate short context dependencies by employing sequences of adjacent words up to length n to form n-grams.

The selection of baseline predictors includes linear models, such as ridge regression, lasso and elastic nets, as well as non-linear models, such as random forest, AdaBoost and gradient boosting. We also employ support vector machines with both linear and non-linear kernels [40]. These models have been shown to perform well on machine learning problems with many features and few observations [41]; hence, they are especially suited to our task, where the number of predictors exceeds the number of documents.

3.3. Deep learning architectures

We first introduce the RNN, followed by its extension, the LSTM, which can better memorize information [17]. Both network architectures iterate over sequential data $x_1, x_2, \ldots$ of arbitrary length. Here the input vector x_i consists of the words (or stems) in one-hot encoding. Mathematically this specifies a vector consisting of zeros, except for a single element with a 1 that refers to the i-th word in the sequence [17]. This yields high-dimensional but sparse vectors as input. In addition, we experiment with word embeddings in the case of the LSTM, as detailed below.

3.3.1. Recurrent neural networks

The recurrent neural network [30] allows the connections between neurons to form cycles, based on which the network can

[4] As an example, let us consider a case where the training/test data is not split in chronological order. For instance, data from 2010 (i. e. after the financial crisis) is used for training, while data from 2008/09 (i. e. during the financial crisis) is used for testing. The algorithm would then learn that the term "Lehman Brothers" has a negative connotation and, as a result, it would accurately predict the bankruptcy of Lehman Brothers, since it had ex post knowledge that a decision support system would not have had in a real-world setting.

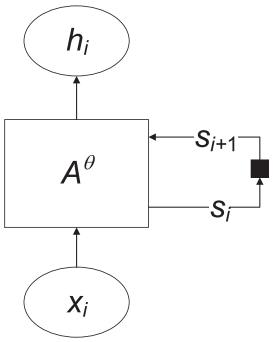

Fig. 3. Schematic structure of a recurrent neural network with input x_i, state s_i, output h_i and one feedforward neural network A^θ parameterized by θ. When moving from word i to $i+1$, the recurrent neural network can pass information related to the current and previous words on by sending information from state s_i to the next state s_{i+1}. It thereby draws upon previous terms and encodes context dependencies between words.

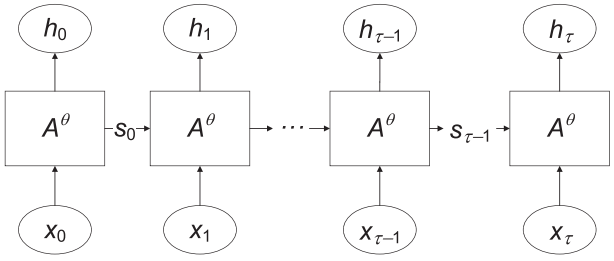

Fig. 4. Recurrent neural network unrolled over inputs $x_0,\ldots,x_\tau$, states $s_0,\ldots,s_{\tau-1}$, outputs $h_0,\ldots,h_\tau$ and a feedforward neural network A^θ.

memorize information that persists when moving from word x_i to x_{i+1}. The architecture of an RNN is illustrated in Fig. 3.

Let x_i be the input in iteration i. Furthermore, A^θ denotes the feedforward neural network parameterized by θ, while s_i is the hidden state and h_i is the output in iteration i. When moving from iteration i to $i+1$, the RNN calculates the output h_{i+1} from the neural network A^θ, the previous state s_i and the current input x_{i+1}, i. e.

$$h_{i+1} = A^\theta(s_i, x_{i+1}). \tag{1}$$

By modeling a recurrent relationship between states, the RNN can pass information onwards from the current state s_i to the next s_{i+1}. To illustrate this, Fig. 4 presents the processing of sequential data by unrolling the recurrent structure.

Theoretically, RNNs are very powerful, and yet two issues limit their practical application [17]. First, vanishing and exploding gradients during training result in numerical instabilities and, second, information usually only persists for a few iterations in the memory [31].

3.3.2. Long short-term memory networks

Long short-term memory networks advance RNNs to capture very long dependencies among input signals [17]. For this purpose, LSTMs still process information sequentially, but introduce a cell state c_i, which remembers and forgets information, similar to a memory [32]. This cell state is passed onwards, similar to the state of an RNN. However, the information in the cell state is manipulated by additional structures called gates. The LSTM has three of them – namely the forget gate, the input gate and the output gate – each of which is a neural network layer with its own sigmoid activation function. This is schematically visualized in Fig. 5.

The forget gate takes the output h_{i-1} from the previous word and the numerical representation x_i of the current word as input. It then returns a vector f_i with elements in the range $[0,1]$. The values correspond to the strength with which each element in cell state c_i should be passed on to the next cell state. Here a zero refers to discarding, a one to remembering.

Next, we compute what information finds its way into the cell state. On the one hand, an input gate takes h_{i-1} and x_i as input and returns a vector u_i denoting which elements in c_{i-1} are updated. On the other hand, an additional neural network layer computes a vector of candidate values $\tilde{c}_i$ that might find its way into the cell state. Both are combined by element-wise multiplication, as indicated by the operator $\odot$.

Lastly, we need to define how a new cell state c_i translates into an output h_i. This is accomplished with an output gate that computes a numeric vector o_i with elements in the range $[0,1]$. These values refer to the elements in c_i which are passed on to the output. The new output is obtained through element-wise multiplication, i. e. $h_i = o_i \odot c_i$. The new cell state stems from the updating rule

$$c_i = f_i \odot c_{i-1} + u_i \odot \tilde{c}_i. \tag{2}$$

In order to make predictions, the LSTM utilizes the final output h_τ and inserts this into an additional feedforward layer [17]. Therefore, one simultaneously optimizes both the activation functions of the gates and this last feedforward layer with a combined target function. During training, we tune several parameters inside the LSTM (see Section 3.4).

In practice, we not only insert binary vectors with words as one-hot encoding into the LSTM, but also utilize word embeddings [27,42]. Word embeddings construct a lower-dimensional, dense vector representation of word tokens from the originally sparse and high-dimensional vectors, while preserving contextual similarity. We generate word embeddings by the GloVe

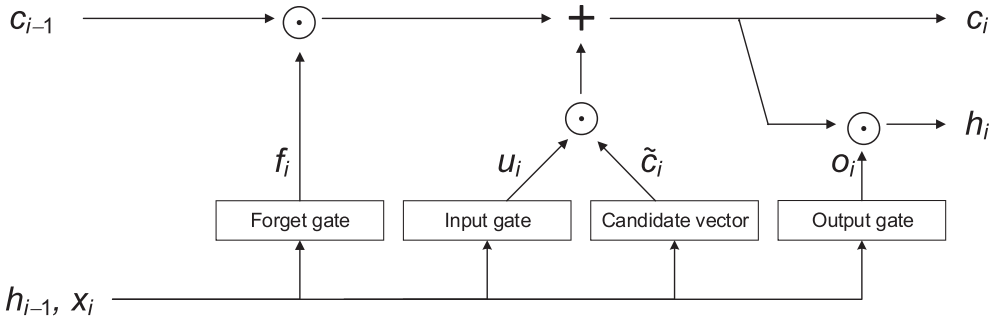

Fig. 5. Long short-term memory with input x_i, output h_i, cell state c_i and four gates to filter information.

algorithm [27] and, subsequently, fine-tune them together with the weights of the neurons during the actual training phase.

3.4. Model tuning

Algorithm 1 describes the tuning in order to find the best-performing parameters based on time-series cross-validation that employs a rolling forecasting origin [43]. We first split the training data $\mathcal{T}$ into k disjoint subsets $\mathcal{T}_1, \ldots, \mathcal{T}_k$ that are chronologically ordered. Afterwards, we set the tuning range for each parameter, iterate over all possible combinations and over all subsets of $\mathcal{T}$. In each iteration i, we measure the performance $perf_i$ of the method on the validation set $\mathcal{T}_i$ while using the subsets $\mathcal{T}_1, \ldots, \mathcal{T}_{i-1}$ from the previous points in time as training. Finally, we return the best-performing parameter setting.

Algorithm 1. Parameter tuning with time-series cross-validation

$\mathcal{T} \leftarrow$ Training data which is chronologically ordered
Split $\mathcal{T}$ into k disjoint subsets $\mathcal{T}_1, \ldots, \mathcal{T}_k$ by maintaining the chronological order
$\mathcal{R}_i \leftarrow$ Ranges of tuning parameter i with $i = 1, \ldots, l$
for $(p_1, \ldots, p_l)$ **in** $\mathcal{R}_1 \times \ldots \times \mathcal{R}_l$ **do**
  for i **in** $2, \ldots, k$ **do**
    Train model m with data from subsets $\mathcal{T}_1, \ldots, \mathcal{T}_{i-1}$
    $perf_i \leftarrow$ Measure performance of model m on $\mathcal{T}_i$
  end for
  $perf_{(p_1,\ldots,p_l)} \leftarrow \frac{1}{k} \sum_{i=1}^{k} perf_i$
end for
return $\arg\max_{(p_1,\ldots,p_l)} perf_{(p_1,\ldots,p_l)}$

The tuning parameters are detailed in the online appendix. To tune all parameters of the baseline methods, we perform a grid search with 10-fold time-series cross-validation on the training set. In the case of deep learning, we tune the parameters of architectures by using the last 10% of the training set for validation due to high computational demand. We optimize the deep neural networks by utilizing the Adam algorithm with learning rates tuned on the interval $[0.0001, 0.01]$ with a step size of 0.0005, while weights are initialized by the Xavier algorithm. The size of the word embeddings is tuned on the set $\{30, 40, \ldots, 100\}$. We initialize the word embeddings based on a continuous uniform distribution $\mathcal{U}(-0.1, 0.1)$ and set the size of each neural network layer within the RNN and LSTM to the dimension of the word embeddings.

3.5. Transfer learning

Transfer learning performs representation learning on a different, but related, dataset and then applies the gained knowledge to the actual training phase [44]. In the case without transfer learning, the weights in the neural network are initialized randomly and then optimized for the training set. Given the sheer number of weights in deep neural networks, this approach requires a large number of training samples in order for the weights to converge. The idea behind transfer learning is to initialize the weights not randomly, but rather with values that might be close to the optimized ones. For this purpose, we utilize an additional dataset with 8-K filings and train the neural network (including word embeddings, if applicable) to predict stock price movements from this dataset. The resulting weights then serve as initial values when performing the actual training process for optimizing the weights with the ad hoc announcements.

More explicitly, we draw upon 34,782 Form 8-K filings, spanning the years 2010 to 2013, with a total length of 139.1 million words.[5] This type of disclosure is mandated by the U. S. Securities and Exchange Commission to inform investors about stock-relevant events. Form 8-K filings contain considerably more words than ad hoc announcements: they comprise an average of 4000.15 words, compared to a mean of 168.80 words in the case of ad hoc announcements. The vocabulary of the 8-K filings includes 57,732 unique terms, out of which 7239 entries also appear in ad-hoc releases. Word embeddings for all terms not part of 8-K filings are drawn from a uniform distribution $\mathcal{U}(-0.1,\ 0.1)$.

4. Results

This section compares the performance of bag-of-words and deep learning architectures on the basis of financial disclosures. The evaluation provides evidence that deep learning is superior to traditional bag-of-words approaches in predicting the direction and magnitude of stock price movements. In addition, our results clearly demonstrate additional benefits from using transfer learning in order to further bolster the performance of deep neural networks.

Undertaking transfer learning is often computationally intensive, especially in cases with an extensive corpus such as ours. We thus utilized a computing system consisting of an Intel Xeon E5-2673 V3 with 8 cores running at 3.2 GHz and 16 GB RAM. The training of LSTMs ranges below 5 h, while the overall runtime, including transfer learning, amounted to approximately 22 h.

We implemented all baselines in Python using scikit-learn, while we used TensorFlow and Theano for all experiments with deep learning. The resulting neural networks from the deep learning process are available for download via https://github.com/MathiasKraus/FinancialDeepLearning. This is intended to facilitate future comparisons and direct implementations in practical settings. All networks are shipped in the HDF5 file format as used by the Keras library.

We report the following metrics for comparing the performance of both regression and classification tasks. In case of the former, we compute the root mean squared error (RMSE), the mean squared error (MSE) and the mean absolute error (MAE) in order to measure the deviation from the true return. Our classifications specifically compare the balanced accuracy, which is defined as the arithmetic mean of sensitivity and specificity, in order to account for unbalanced classes in our dataset. For the same reason, we also provide the area under the curve (AUC).

4.1. Classification: direction of nominal returns

We now proceed to evaluate the classifiers for predicting the direction of nominal returns. The corresponding results are detailed in Table 2. The first row reflects the performance of our naïve benchmark when using no predictor (i. e. voting the majority class). It results in an accuracy above average due to the severe class imbalance. This also explains why we compare merely the balanced accuracy in the following analysis. Among the baseline models from traditional machine learning, we find the highest balanced accuracy on the test set when using the random forest, which yields an improvement of 4.7 percentage points compared to the naïve benchmark. Its results stem from a random forest with 500 trees, where 3 variables are sampled at each split. This highlights once again the strength of the random forest as an out-of-the-box classifier.

Deep learning outperforms traditional machine learning. For instance, the LSTM with word embeddings yields an improvement of

[5] These disclosures are publicly available via the website of the U. S. Securities and Exchange Commission (www.sec.gov/edgar.shtml).

Table 2
Out-of-sample results from classifying the direction of the nominal return. Values in bold indicate approaches that outperform the naïve baseline and all models from traditional machine learning.

Method	Training set	Test set			Absolute improvement on test set over baseline		
	Accuracy	Accuracy	Balanced accuracy	AUC	Accuracy	Balanced accuracy	AUC
NAÏVE BASELINE							
Majority class	0.549	0.540	0.500	0.500	–	–	–
TRADITIONAL MACHINE LEARNING							
Ridge regression	0.534	0.534	0.528	0.539	-0.006	0.028	0.039
Lasso	0.549	0.540	0.500	0.500	0.000	0.000	0.000
Elastic net	0.549	0.540	0.500	0.500	0.000	0.000	0.000
Random forest	0.557	0.562	0.547	0.552	0.022	0.047	0.052
SVM	0.552	0.545	0.522	0.556	0.005	0.022	0.056
AdaBoost	0.555	0.552	0.538	0.555	0.012	0.038	0.055
Gradient boosting	0.553	0.554	0.532	0.556	0.014	0.032	0.056
DEEP LEARNING							
RNN	0.588	0.545	0.530	0.529	0.005	0.030	0.029
LSTM	**0.601**	**0.577**	**0.562**	**0.563**	**0.037**	**0.062**	**0.063**
LSTM with word embeddings	**0.597**	**0.576**	**0.563**	**0.568**	**0.036**	**0.063**	**0.068**
TRANSFER LEARNING							
RNN with pre-training	**0.596**	0.548	0.533	0.530	0.008	0.033	0.033
LSTM with pre-training	**0.576**	**0.578**	**0.564**	**0.577**	**0.038**	**0.064**	**0.077**
LSTM with pre-training and word embeddings	**0.581**	**0.580**	**0.571**	**0.568**	**0.040**	**0.071**	**0.068**

6.8 percentage points over the naïve baseline. The word embeddings contribute to this increase by a mere 0.1 percentage points; however, they elevate the AUC score by 0.5 percentage points.

Transfer learning yields consistent improvements for deep learning variants. As a result, the LSTM model with word embeddings performs best among all approaches, amounting to a total improvement of 7.1 percentage points. In other words, transfer learning enhances the balanced accuracy by an additional 0.8 percentage points. Compared to the strongest traditional model (AUC of 0.556), transfer learning increases the AUC score by 0.021 (significant at the 0.05 level), thereby reaching an AUC of 0.577.

4.2. Classification: direction of abnormal returns

Table 3 reports the results for predicting the direction of abnormal returns, depicting a picture similar to that of the classification of nominal returns. The random forest again scores well with a balanced accuracy of 0.542, but is beaten by 0.545 from the ridge regression. The latter achieves a balanced accuracy that is 4.5 percentage points higher than the naïve benchmark. This performance is obtained when α is set to 0.99.

With regard to deep learning, both RNNs (with and without transfer learning) fail to improve performance beyond traditional machine learning models. However, the LSTM still succeeds in this task, exceeding the balanced accuracy of the naïve benchmark by 5.6 percentage points. Further performance gains come from the use of word embeddings, representing an improvement of 6.6 percentage points compared to the naïve approach.

When applying transfer learning, LSTMs show further, considerable improvements, since they outperform the balanced accuracies of the LSTMs without transfer learning by 0.7 and 1.7 percentage points, respectively. The LSTM with both pre-training and word embeddings further enhances this value to 8.3 percentage points.

4.3. Regression: nominal returns

While the previous section studied the accuracy in terms of classifying the direction of stock price changes, we now incorporate the nominal magnitude of the price adjustment. The corresponding results from the regression task are given in Table 4. With regard to the baseline models, only support vector regression yields favorable results in comparison to the naïve approach (which represents the mean return of the training set). Its performance stems from choosing a radial basis function kernel and setting the cost to 0.05. While the previous section studied the accuracy in terms of classifying the direction of stock price changes, we now incorporate the nominal magnitude of the price adjustment. The corresponding results from the regression task are given in Table 4. With regard to the baseline models, only support vector regression yields favorable results in comparison to the naïve approach. Its performance stems from choosing a radial basis function kernel and setting the cost to 0.05.

The performance of the RNN is consistently inferior to both the naïve approach and traditional machine learning. However, the LSTM outperforms the baselines on all metrics. It reduces the mean squared error of the random guess by 1.950 or 5.08%. Here word embeddings diminish the predictive performance, since they increase the mean squared error of the LSTM by 0.105.

Again, favorable results originate from transfer learning across all deep learning models, highlighting the benefits of the additional pre-training. Overall, the LSTM with pre-training and word embeddings performs best, decreasing the mean squared error by 2.053 (i. e. 5.34%) compared to the naïve approach.

4.4. Regression: abnormal returns

Table 5 evaluates the regression task with abnormal returns, which corrects for confounding market movements. Here the elastic net achieves the lowest mean squared error among the traditional machine learning models, amounting to 37.614, which is −0.930 below the naïve benchmark.

Among deep learning approaches, the LSTM with word embeddings achieves the lowest mean squared error, outperforming the naïve approach by an absolute reduction of 2.281. This model corresponds to choosing a learning rate of 0.0005 and 50-dimensional vectors. In contrast to the previous regression task with nominal returns, we observe mixed results when incorporating word embeddings: doing so decreases the mean squared error, but increases the mean absolute error.

Using transfer learning further improves the prediction performance of LSTMs. It shrinks the mean squared error by an additional 0.110 and 0.083 for the classical LSTM model and the LSTM model using word embeddings, respectively. Overall, the LSTM with pre-training and word embeddings outperforms the naïve approach by an absolute value of 2.364 (i. e. 6.1%) in terms of mean squared error.

Table 3
Out-of-sample results from classifying the direction of the abnormal return. Values in bold indicate models that outperform both the naïve baseline and traditional machine learning.

Method	Training set	Test set			Absolute improvement on test set over baseline		
	Accuracy	Accuracy	Balanced accuracy	AUC	Accuracy	Balanced accuracy	AUC
NAÏVE BASELINE							
Majority class	0.542	0.528	0.500	0.500	–	–	–
TRADITIONAL MACHINE LEARNING							
Ridge regression	0.539	0.549	0.545	0.562	0.021	0.045	0.062
Lasso	0.542	0.528	0.500	0.500	0.000	0.000	0.000
Elastic net	0.542	0.528	0.500	0.500	0.000	0.000	0.000
Random forest	0.559	0.552	0.542	0.559	0.024	0.042	0.059
SVM	0.536	0.557	0.527	0.558	0.029	0.027	0.058
AdaBoost	0.537	0.539	0.538	0.561	0.011	0.038	0.061
Gradient boosting	0.541	0.550	0.526	0.557	0.022	0.026	0.057
DEEP LEARNING							
RNN	0.583	0.548	0.534	0.536	0.020	0.034	0.036
LSTM	**0.597**	**0.573**	**0.556**	0.558	**0.045**	**0.056**	0.058
LSTM with word embeddings	**0.593**	**0.579**	**0.566**	0.551	**0.051**	**0.066**	0.051
TRANSFER LEARNING							
RNN with pre-training	**0.594**	0.552	0.538	0.538	0.024	0.038	0.038
LSTM with pre-training	**0.601**	**0.576**	**0.563**	0.552	**0.048**	**0.063**	0.052
LSTM with pre-training and word embeddings	**0.578**	**0.578**	**0.583**	**0.568**	**0.050**	**0.083**	**0.068**

4.5. Sensitivity analysis

We now investigate the sensitivity of our models to modifications in their parameters (see online appendix for details). We first explore the effect of introducing *n*-grams within traditional machine learning models and varying the size of *n*. In short, in the classification task with abnormal returns, bag-of-words models indicate mixed results when changing the length of *n*-grams (i. e. bigrams and trigrams). For instance, the test accuracy of ridge regression improves by 0.3 percentage points when utilizing bigrams compared to unigrams, but decreases by 0.4 percentage points when applying trigrams. On the contrary, the test accuracy of AdaBoost decreases by 0.3 percentage points for bigrams, but increases by 0.3 percentage points for trigrams. Altogether, we observe no clear pattern that can guide our choice of *n* and, more importantly, none of our experiments resulted in a performance that is superior to LSTMs.

We empirically investigate whether the deep neural network can store long sequences that span a complete sentence or even more extensive text passages. For this purpose, we shorten each document and merely extract the first words. We then train a deep neural network with this shortened text fragment in order to determine whether the deep neural network with the complete documents yields a superior predictive performance. Due to space constraints, we only detail the regression task with abnormal returns for an LSTM with word embeddings. As a result, the RMSE on the test set attains a value of 6.162 when considering merely the first sentence, 6.184 when restricting the document to the first 50 words and 6.114 for the first 100 words. Here we observe that all experiments result in a worse predictive performance than the network with the complete documents (RMSE of 6.022). This implies that the neural network can learn to store even long sequences in its weights.

Deep learning usually works as a black-box approach and, as a remedy, we contribute to explanatory insights as follows: we draw upon the finance-specific dictionary from Loughran-McDonald that comprises terms labeled as either positive or negative, where the underlying categorization stems from subjective human ratings. We then treat each word as a single document and insert them as input into our deep neural network. The resulting predictions allow us to infer whether a word links to a positive or negative market reaction. In other words, the prediction scores the polarity of the words and specifies how markets perceive them. We show an excerpt in Table 6, while the supplementary materials provide the complete list.

Table 4
Out-of-sample results from regressing the nominal return. Bold values indicate models that outperform all baselines (i. e. naïve and traditional machine learning).

Method	Training set	Test set			Absolute error reduction on test set over baseline		
	RMSE	RMSE	MSE	MAE	RMSE	MSE	MAE
NAÏVE BASELINE							
Mean return	7.060	6.197	38.402	3.069	–	–	–
TRADITIONAL MACHINE LEARNING							
Ridge regression	6.765	6.127	37.541	3.114	−0.070	−0.861	0.045
Lasso	6.918	6.122	37.486	3.089	−0.075	−0.916	0.020
Elastic net	6.892	6.108	37.308	3.091	−0.089	−1.094	0.022
Random forest	6.873	6.145	37.761	3.111	−0.052	−0.641	0.042
SVR	6.890	6.171	38.081	3.058	−0.026	−0.321	−0.011
AdaBoost	7.994	7.282	53.028	4.837	1.085	14.626	1.768
Gradient boosting	6.872	6.146	37.773	3.111	−0.051	−0.629	0.042
DEEP LEARNING							
RNN	6.859	6.139	37.685	3.102	−0.058	−0.717	0.033
LSTM	6.892	**6.038**	**36.452**	**3.024**	**−0.159**	**−1.950**	**−0.045**
LSTM with word embeddings	6.954	**6.046**	**36.557**	**3.043**	**−0.151**	**−1.845**	**−0.026**
TRANSFER LEARNING							
RNN with pre-training	6.875	**6.101**	**37.222**	3.099	**−0.096**	**−1.180**	0.030
LSTM with pre-training	6.887	**6.036**	**36.439**	**3.020**	**−0.161**	**−1.963**	**−0.049**
LSTM with pre-training and word embeddings	6.876	**6.029**	**36.349**	**3.011**	**−0.168**	**−2.053**	**−0.058**

Table 5
Out-of-sample results from regressing the abnormal return. Values in bold indicate an improvement compared to all baselines (i. e. naïve and traditional machine learning).

Method	Training set	Test set			Absolute error reduction on test set over baseline		
	RMSE	RMSE	MSE	MAE	RMSE	MSE	MAE
NAÏVE BASELINE							
Mean abnormal return	7.102	6.208	38.544	3.126	–	–	–
TRADITIONAL MACHINE LEARNING							
Ridge regression	6.908	6.153	37.860	3.168	−0.055	−0.684	0.042
Lasso	6.992	6.153	37.860	3.166	−0.055	−0.684	0.040
Elastic net	6.956	6.133	37.614	3.144	−0.075	−0.930	0.018
Random forest	6.927	6.173	38.106	3.176	−0.035	−0.438	0.050
SVR	6.906	6.183	38.235	3.122	−0.025	−0.309	−0.004
AdaBoost	8.046	7.159	51.251	4.704	0.951	12.707	1.578
Gradient boosting	6.923	6.173	38.106	3.176	−0.035	−0.438	0.050
DEEP LEARNING							
RNN	6.966	6.169	38.062	3.162	−0.039	−0.482	0.036
LSTM	**6.873**	**6.030**	**36.364**	**3.118**	**−0.178**	**−2.180**	**−0.008**
LSTM with word embeddings	**6.904**	**6.022**	**36.263**	**3.127**	**−0.186**	**−2.281**	0.001
TRANSFER LEARNING							
RNN with pre-training	6.934	6.133	37.614	3.158	−0.064	−0.788	0.089
LSTM with pre-training	**6.845**	**6.021**	**36.254**	**3.109**	**−0.187**	**−2.290**	**−0.017**
LSTM with pre-training and word embeddings	**6.687**	**6.015**	**36.180**	**3.104**	**−0.193**	**−2.364**	**−0.022**

5. Discussion

5.1. Comparison

We additionally compare our predictive performance of stock market movements to earlier publications studying the same financial disclosures. However, the results are often not comparable, as different papers utilize additional (subjective) filter rules, report accuracies instead of balanced accuracies, incorporate other splits into training and test sets, or neglect to perform time-series cross-validation. We refer to previous literature overviews [15,45] for a comparison of predictive accuracy across different news sources.

In short, the SVM-based approach in [45] is fed with a different time frame (starting in 1997), which results in a skewer distribution of positive and negative labels, with 58.3% of them being positive. Since the authors do not report a balanced accuracy, we cannot make a fair comparison. Furthermore, they train their classifiers without temporal distinction, resulting in an overestimated performance. Hence, we replicated their experiments with 2-gram and 3-gram SVMs using our dataset, training processes and evaluation metrics. However, the performance of the SVM-based approach is substantially inferior to that of deep learning. The recursive autoencoder in [12] splits the announcements into three classes (up, down or steady) according to the abnormal return and then discards the steady samples a priori. Moreover, their approach relies merely upon headlines and reports the accuracy (56%) instead of the balanced accuracy. Furthermore, their accuracy is lower than ours due to the advanced deep learning architectures. By additionally incorporating the discourse structure, the authors of [46] achieve a balanced accuracy amounting to 54.32% for the same dataset. However, this performance is yet again inferior to that of deep and transfer learning.

Table 6
Standardized predictions for individual terms from the Loughran-McDonald finance-specific word list. The results stem from an LSTM with word embeddings for the regression task with abnormal returns. The complete list is reported in the supplements.

Entry	Label	Predicted score
Absence	Negative	−0.176
Abuse	Negative	−0.034
Achieve	Positive	0.338
Adequately	Positive	0.284
Advantage	Positive	0.256
…	…	…

5.2. Generalizability and limitations

The aforementioned models based on deep learning are not limited to sentiment analysis or natural language processing, but can be beneficial in any task of advanced complexity, such as time series prediction, voice control or information retrieval. In this respect, deep learning can help to encode context information that spans multiple words or even sentences.

The majority of deep learning architectures are trained in a supervised fashion and thus need a sufficiently large labeled dataset. This requirement can be partially relaxed by transfer learning, which first performs representation learning on a different, but related, dataset and then solves problems regarding the actual data. For instance, decision support from financial disclosures can benefit from systems trained on a different type of news. To do so, one first tunes the parameters on the basis of general finance-related narratives in order to acquire a basic understanding of language in a finance-specific context and, in a next step, tailors the weights in the output layer to the particular problem.

In comparison to classical machine learning tasks, accurate predictions based on financial news are still difficult to obtain due to the complexity of natural language and the efficiency of markets where historic prices contribute only marginally to explaining future returns [47]. Both difficulties underline the necessity for more complex models in the field of deep learning. Nevertheless, even minor improvements in the predictive performance can result in a considerable economic impact. For this reason, we assume a portfolio of $1000, one year with 200 trading days and each with a single disclosure that triggers a log-return of 5%. By utilizing a random guess in the prediction engine, this would result in an expected final portfolio of $1000 . A 51 % accuracy increases the monetary value of the portfolio considerably, since the portfolio now attains a log-return of 10% over the course of the year.

5.3. Implications for management

Deep learning is applicable to the improvement of decision support in many core areas of organizations and businesses, such as recommender systems, question-answering mechanisms and customer support. To further augment potential use cases, the long short-term memory model enables the processing of sequential data, often with unprecedented performance. This model thus allows one to bolster existing tool chains, where traditional predictive models will soon be replaced by deep architectures.

This substitution, however, can be challenging in practical application due to rapid advances in the underlying software libraries. As an example, the above results required considerable adjustments in TensorFlow and Theano, such as regularization to avoid overfitting of the networks. While many pre-trained networks are available for image-related tasks, this is often not the case for natural language processing, which is why we published our trained networks as open-source.

5.4. Implications for research

This work demonstrates the achievements of deep learning in relation to decision support systems and, at the same time, presents opportunities for research aimed at the enhancement of transfer learning in natural language processing. Transfer learning has been predominantly applied to image-related tasks; however, empirical results are scarce when it comes to natural language processing. Common obstacles derive from the fact that large, pre-assembled datasets are often not readily available. The incorporation of these large-scale corpora, however, is essential to building powerful models. Therefore, future research might adapt the idea behind the ImageSet dataset and publish extremely large unlabeled and labeled datasets for text classifications.

6. Conclusion and outlook

Financial disclosures greatly aid investors and automated traders in deciding whether to exercise ownership in stocks. While humans are usually able to interpret textual content correctly, computerized decision support systems struggle with the complexity and ambiguity of natural language.

This paper analyzes the switch from traditional bag-of-words models to deep, non-linear neural networks. Each of the neural networks comprises more than 500,000 parameters that help in making accurate predictions. Thereby, we contribute to the existing literature by showing how deep learning can enhance financial decision support by explicitly incorporating word order, context-related information and semantics. For this purpose, we engage in the task of predicting stock market movements subsequent to the disclosure of financial materials. Our results show that long short-term memory models can outperform all traditional machine learning models based on the bag-of-words approach, especially when we further pre-train word embeddings with transfer learning. We thus identify two critical ingredients for superior predictive performance, namely being able to infer context-dependent information from ordered sequences of words and capturing highly non-linear relationships. Yet the configuration of deep neural networks represents a challenging task, as it still requires extensive parameter tuning to achieve favorable results. With regard to news-based predictions, it is an interesting question for future research to further detail the gains in predictive performance from deep learning for intraday data (including potential latency effects) and in the long run.

We expect that deep learning will soon expand beyond the realm of academic research and the rather limited number of firms that specialize in predictive analytics, especially as decision support systems can benefit from deep learning in multiple ways. First of all, deep learning can learn to incorporate context information from sequential data. Second, competition will drive firms and organizations towards using more powerful architectures in predictive tasks and, in this regard, deep neural networks with transfer learning often represent the status quo.

Acknowledgments

We thank Ryan Grabowski for his proof-reading.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.dss.2017.10.001.

References

[1] E.F. Fama, The behavior of stock-market prices, J. Bus. 38 (1965) 34–105.
[2] I.E. Fisher, M.R. Garnsey, M.E. Hughes, Natural language processing in accounting, auditing and finance: a synthesis of the literature with a roadmap for future research, Intell. Syst. Account. Financ. Manag. 23 (2016) 157–214.
[3] C. Kearney, S. Liu, Textual sentiment in finance: a survey of methods and models, Int. Rev. Financ. Anal. 33 (2014) 171–185.
[4] F. Li, Textual analysis of corporate disclosures: a survey of the literature, J. Account. Lit. 29 (2010) 143–165.
[5] T.I. Loughran, B. McDonald, Textual analysis in accounting and finance: a survey, J. Account. Res. 54 (2016) 1187–1230.
[6] S. Feuerriegel, H. Prendinger, News-based trading strategies, Decis. Support. Syst. 90 (2016) 65–74.
[7] E.J. de Fortuny, T. de Smedt, D. Martens, W. Daelemans, Evaluating and understanding text-based stock price prediction models, Inf. Process. Manag. 50 (2014) 426–441.
[8] T. Geva, J. Zahavi, Empirical evaluation of an automated intraday stock recommendation system incorporating both market data and textual news, Decis. Support. Syst. 57 (2014) 212–223.
[9] R.P. Schumaker, H. Chen, Textual analysis of stock market prediction using breaking financial news: the AZFin text system, ACM Trans. Inf. Syst. 27 (2009) 12.
[10] R.P. Schumaker, H. Chen, A quantitative stock prediction system based on financial news, Inf. Process. Manag. 45 (2009) 571–583.
[11] R.P. Schumaker, Y. Zhang, C.-N. Huang, H. Chen, Evaluating sentiment in financial news articles, Decis. Support. Syst. 53 (2012) 458–464.
[12] S. Feuerriegel, R. Fehrer, Improving Decision Analytics With Deep Learning: The Case Of Financial Disclosures, 24th European Conference on Information Systems, 2015.
[13] C.D. Manning, H. Schütze, Foundations Of Statistical Natural Language Processing, MIT Press, Cambridge, MA, 1999.
[14] B. Pang, L. Lee, Opinion mining and sentiment analysis, Found. Trends Inf. Retr. 2 (2008) 1–135.
[15] A.K. Nassirtoussi, S. Aghabozorgi, T.Y. Wah, D.C.L. Ngo, Text mining for market prediction: a systematic review, Expert Syst. Appl. 41 (2014) 7653–7670.
[16] K. Ravi, V. Ravi, A survey on opinion mining and sentiment analysis: tasks, approaches and applications, Knowl.-Based Syst. 89 (2015) 14–46.
[17] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, Cambridge, MA, 2017.
[18] J. Evermann, J.-R. Rehse, P. Fettke, Predicting process behaviour using deep learning, Decis. Support. Syst. (2017)
[19] C.-J. Huang, J.-J. Liao, D.-X. Yang, T.-Y. Chang, Y.-C. Luo, Realization of a news dissemination agent based on weighted association rules and text mining techniques, Expert Syst. Appl. 37 (2010) 6409–6413.
[20] B. Wüthrich, D. Permunetilleke, S. Leung, W. Lam, V. Cho, J. Zhang, Daily prediction of major stock indices from textual WWW data, HKIE Trans. 5 (1998) 151–156.
[21] X. Li, H. Xie, L. Chen, J. Wang, X. Deng, News impact on stock price return via sentiment analysis, Knowl.-Based Syst. 69 (2014) 14–23.
[22] X. Li, X. Huang, X. Deng, S. Zhu, Enhancing quantitative intra-day stock return prediction by integrating both market news and stock prices information, Neurocomputing 142 (2014) 228–238.
[23] S.W.K. Chan, M.W.C. Chong, Sentiment analysis in financial texts, Decis. Support. Syst. 94 (2016) 53–64.
[24] N. Pröllochs, S. Feuerriegel, D. Neumann, Negation scope detection in sentiment analysis: decision support for news-driven trading, Decis. Support. Syst. 88 (2016) 67–75.
[25] S.S. Groth, J. Muntermann, An intraday market risk management approach based on textual analysis, Decis. Support. Syst. 50 (2011) 680–691.
[26] R. Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, C. Potts, Recursive deep models for semantic compositionality over a sentiment treebank, Conference on Empirical Methods in Natural Language Processing 1631 (2013) 1631–1642.
[27] J. Pennington, R. Socher, C.D. Manning, Glove: Global Vectors For Word Representation, Conference on Empirical Methods in Natural Language Processing, 2014, 1532–1543.
[28] K. Cortis, A. Freitas, T. Daudert, M. Hürlimann, M. Zarrouk, S. Handschuh, B. Davis, SemEval-2017 Task 5: Fine-grained sentiment analysis on financial microblogs and news, 11th International Workshop on Semantic Evaluations (SemEval-2017), 2017. pp. 519–535.
[29] Y. Goldberg, A primer on neural network models for natural language processing, J. Artif. Intell. Res. 57 (2016) 345–420.
[30] D. Williams, G.E. Hinton, Learning representations by back-propagating errors, Nature 323 (1986) 533–536.
[31] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Trans. Neural Netw. 5 (1994) 157–166.
[32] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780.

[33] L. Takeuchi, Y.-Y.A. Lee, Applying Deep Learning To Enhance Momentum Trading Strategies In Stocks: Working Paper, 2013.
[34] J.B. Heaton, N.G. Polson, J.H. Witte, Deep Learning In Finance, 2016.arXiv preprint arXiv:1602.06561.
[35] X. Ding, Y. Zhang, T. Liu, J. Duan, Deep Learning For Event-Driven Stock Prediction, Int. Joint Conf. Artif. Intell. (2015) 2327–2333.
[36] A.H.-L. Lau, A five-state financial distress prediction model, J. Account. Res. (1987) 127–138.
[37] J.J. Faraway, Does data splitting improve prediction? Stat. Comput. 26 (2016) 49–60.
[38] C. Serrano-Cinca, B. Gutierrez-Nieto, L. Lopez-Palacios, Determinants of default in P2P lending, PloS one 10 (2015)
[39] M.F. Porter, An algorithm for suffix stripping, Program 14 (1980) 130–137.
[40] S. Wang, C.D. Manning, Baselines and bigrams: simple, good sentiment and topic classification, Proceedings of the 50th Annual Meeting on Association for Computational Linguistics (ACL '12), 2012. pp. 90–94.
[41] T. Hastie, R. Tibshirani, J.H. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed. ed., Springer, New York, NY, 2009.
[42] T. Mikolov, J. Dean, Distributed representations of words and phrases and their compositionality, Adv. Neural Inf. Proces. Syst. (2013) 3111–3119.
[43] R.J. Hyndman, G. Athanasopoulos, Forecasting: Principles and Practice, OTexts, Heathmont, Australia, 2014.
[44] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (2010) 1345–1359.
[45] M. Hagenau, M. Liebmann, D. Neumann, Automated news reading: stock price prediction based on financial news using context-capturing features, Decis. Support. Syst. 55 (2013) 685–697.
[46] J. Märkle-Huß, S. Feuerriegel, H. Prendinger, Improving Sentiment Analysis with Document-Level Semantic Relationships from Rhetoric Discourse Structures, 50th Hawaii International Conference on System Sciences, 2017.
[47] P.C. Tetlock, Giving content to investor sentiment: the role of media in the stock market, J. Financ. 62 (2007) 1139–1168.

Mathias Kraus is a Ph.D. student at the Chair of Information Systems Research of the University of Freiburg with focus on machine learning and computer science. Previously, he has completed his Bachelor's and Master's studies in computer science and mathematics at the Karlsruhe Institute of Technology. His research focuses on innovative methods for natural language processing that explicitly cater for semantic information.

Stefan Feuerriegel is an assistant professor for management information systems at ETH Zurich. His research focuses on cognitive information systems and business intelligence, including text mining and sentiment analysis of financial news. Previously, he obtained his Ph.D. from the University of Freiburg where he also worked as a research group leader at the Chair for Information Systems Research. He has co-authored research publications in the European Journal of Operational Research, the European Journal of Information Systems, the Journal of Information Technology and Decision Support Systems.