Measuring the Information Content of Financial News

Ching-Yun Chang[1], Yue Zhang[1], Zhiyang Teng[1], Zahn Bozanic[2], Bin Ke[3]
(1) Singapore University of Technology and Design
(2) Fisher College of Business, The Ohio State University
(3) NUS Business School, National University of Singapore
chang.frannie@gmail.com, yue_zhang@sutd.edu.sg,
zhiyang_teng@mymail.sutd.edu.sg, bozanic.1@fisher.osu.edu,
bizk@nus.edu.sg

Abstract

Measuring the information content of news text is useful for decision makers in their investments since news information can influence the intrinsic values of companies. We propose a model to automatically measure the information content given news text, trained using news and corresponding cumulative abnormal returns of listed companies. Existing methods in finance literature exploit sentiment signal features, which are limited by not considering factors such as events. We address this issue by leveraging deep neural models to extract rich semantic features from news text. In particular, a novel tree-structured LSTM is used to find target-specific representations of news text given syntax structures. Empirical results show that the neural models can outperform sentiment-based models, demonstrating the effectiveness of recent NLP technology advances for computational finance.

1 Introduction

Information has economic value because it allows individuals to make choices that yield higher expected payoffs than they would obtain from choices made in the absence of information. A major source of information is text from the Internet, which embodies news events, analyst reports and public sentiments, and can serve as a basis for investment decisions. Measuring the information content of text is hence a highly important task in computational finance. For investors such as venture capitals, information content should reflect a firm's intrinsic value, or potential of future growth. However, measuring the information content of text can be challenging due to uncertainties and subjectivity. Fortunately, for public companies, the stock price can be used as a metric. Our goal is thus to leverage such data on public companies to train a model for measuring the information content of news on arbitrary companies.

Finance theory suggests that stock prices reflect all available information and expectations about the future prospects of firms (Fama, 1970). Based on this, empirical studies in economics and finance literature have exploited statistical methods to investigate how stock returns react to a particular news or event, which is called *event studies* or *information content effect* (Ball and Brown, 1968). A standard analysis is to measure the *cumulative abnormal return* (CAR) of a firm's price over a period of time centered around the event date (termed the *event window*) (MacKinlay, 1997). Conceptually, a daily *abnormal return* represents the performance of a stock that varies from the expectation, normally triggered by an event, and can be positive or negative depending on whether the stock outperforms or underperforms the expected return. A CAR is the sum of all abnormal returns in an event window, formally described in Section 2.

We study how news affects a public firm's CAR for training a model to measure the information content of arbitrary financial news. It is worth noting that this is different from predicting a firm's stock price movements, which aims at maximizing trading profits. Rather, we investigate whether NLP techniques can assist the understanding of an event's economic value. In finance literature sentiment signals have been used as a standard linguistic feature for representing information content (Tetlock, 2007). We build a baseline using frequency-based features derived from sentiment words to represent news text. However, sentiment polarities are subjective and may not fully represent the message conveyed in news text. For example, event information is also influential in determining a firm's stock price.

To address this, we propose a deep neural model to better present news. A typical way of modeling a sentence is to treat it as a sequence and input the sequence to a long short-term memory (LSTM; (Hochreiter and Schmidhuber, 1997) model, which is capable of learning semantic features automatically. However, information may present different impact to individual firms and therefore, we need a way to represent information conveyed in a news sentence depending on a specific firm. We propose a novel Tree-LSTM which incorporates contextual information with target-dependent grammatical relations to embed a sentence. Because of the target-dependent feature of the proposed embedding model, the representation of a sentence varies from firm to firm.

We train our information content measurement model with news text collected from Reuters Business & Financial News[1]. Results show that the proposed model yields significant improvements over a baseline sentiment-based model. Different from existing event studies which focus on predefined events or firms (Davis et al., 2012), our model is general to various news and firms; and one can measure the effect of information content of news on any companies, including private companies.

The contribution of this paper is two-fold:

- First, we show that the information content of news text can be measured automatically. Our results demonstrate the effectiveness of state-of-the-art NLP techniques in computational finance, and introduce *information content effect* in finance to the ACL community. Given the ubiquitous and instantaneous nature of electronic text, text analytics is an obvious approach for information content analysis.
- Second, we design a novel target-dependent Tree-LSTM-based model for representing news sentences. To our knowledge, this is the first open-domain information content effect prediction system using machine learning and NLP technologies.

2 Cumulative Abnormal Return

Formally, the *abnormal return* of a firm j on date t is the difference between the *actual return* R_{jt} and the *expected return* $\hat{R_{jt}}$. $\hat{R_{jt}}$ can be an estimated return based on an asset pricing model, using a long run historical average, or it can be the return on an index, such as the Dow Jones or the S&P 500 during the same period. For example, suppose that a firm's stock price rose by 3%, and the market index increased by 5% over the same period. If the stock is expected to perform equally to what the market does, namely 5%, the stock yielded an abnormal return of -2% even though the firm's actual return is positive.

The *cumulative abnormal return* (CAR) of a firm j in an n-day event window is defined as the sum of abnormal returns of each day:

$$CAR_{jn} = \sum_{t=1}^{n} (R_{jt} - \hat{R_{jt}}) \tag{1}$$

The event window is normally centered at the event date and only trading days are considered in a window. The CARs before and after the event date mimic possible information leaks and delayed response to the information, respectively. Depending on the span of an event window, CARs provide analysts with short- and long-term information about the impact of an event on a stock's price. The most common event window found in research is a three-day window (-1, 1) where an event is centered at day 0 (Tetlock, 2007; Davis et al., 2012), and the corresponding CAR is denoted as CAR_3.

In this paper we model the effect of a news release on a firm's CAR_3. If a news release occurs during trading hours, day 0 is the news release date; otherwise, day 0 is the next trading period. As prior research demonstrates that there is no significant difference between a modeled expected return and the market return for a short-term event window (Kothari and Warner, 2004), we compute the expected return $\hat{R_{jt}}$ by the return of the equally-weighted market index including all the stocks on NYSE, Amex and NASDAQ.

3 Information Content Prediction

Our goal is to estimate the polarity of information content $y \in$ {0: negative effect, 1: positive effect} given a sentence s in which a target firm is mentioned. Assume that there is an embedding model that

[1]http://www.reuters.com/finance

maps a sentence to a feature vector $g \in \mathbb{R}^{d_g}$. The probability of positive information content effect is defined as:

$$p(y=1|s) = softmax(W_p g + b_p) \tag{2}$$

W_p and b_p are parameters. We build two methods to obtain g from text, one being a traditional method in the finance literature (Section 4) and the other being a novel neural network (Section 5).

It is possible that there is more than one sentence, $\langle s_1, s_2, \ldots, s_m \rangle$, mentioning a firm of interest in the same event window. In this case, a neural attention mechanism adapted from Bahdanau et al. (2014) is utilized to synthesize the corresponding information embeddings $\langle g_1, g_2, \ldots, g_m \rangle$. To compute the attention vector, we define:

$$u^{(i)} = v^T tanh(W_u g_i) \qquad a^{(i)} = softmax(u^{(i)}) \qquad g' = \sum_{i=1}^{m} a^{(i)} g_i \tag{3}$$

where v and W_u are learnable parameters. $u^{(i)}$ is the score of how much attention should be put on g_i, and $a^{(i)}$ is the normalized score. The final synthetic feature vector g' substitutes for g in Equation 2 to predict the effect.

4 Sentiment-based Representation

A growing body of finance research literature examines the correlation between financial variables, such as stock returns and earning surprises, and the sentiment of corporate reports, press releases, and investor message boards (Li, 2010; Davis et al., 2012), most of which are based on purpose-built sentiment lexicon. A commonly used source is the list compiled by Loughran and McDonald (2011),[2] which consists of 353 positive words and 2,337 negative words. As a baseline we follow prior literature (Mayew and Venkatachalam, 2012) and represent the information content of a sentence based on counts derived from the lexicon of Loughran and McDonald. The feature vector consists of raw frequency counts and sentence length-normalized values of positive words, negative words, and the difference of positives and negatives. In addition, the sentence length is also considered.

5 Deep Neural Representation

As mentioned in the introduction, it is useful to model news information content beyond their sentiment signals. Deep learning has shown being effective in automatically inducing features that capture semantic information over nature language sentences. To verify the relative effectiveness of deep neural models, we first build a baseline neural network model using a popular LSTM structure (Section 5.1) and then develop a novel syntactic tree-structured LSTM that is sensitive to specific target entities (Section 5.2).

5.1 Bidirectional LSTM

A popular way of modeling a sentence s is to represent each word by a vector $x \in \mathbb{R}^{d_x}$ (Mikolov et al., 2013), and sequentially input its word vectors $\langle x_1, x_2, \ldots, x_{|s|} \rangle$ to a long short-term memory (LSTM; Hochreiter and Schmidhuber (1997)) model, which is a form of recurrent neural network (RNN; Pearlmutter (1989)). We take a variation of LSTM with *peephole connections* (Gers and Schmidhuber, 2000), which uses a *input gate* i_t, a *forget gate* f_t and a *output gate* o_t in the same memory block to learn from the current cell state. In addition, to simplify model complexity, we adopt coupled i_t and f_t (Cho et al., 2014). The following equations show how the LSTM *cell state* c_t and the *output* of the memory block h_t are updated given *input* x_t at time step t:

$$\begin{aligned} i_t &= \sigma(W_1 x_t + W_2 h_{t-1} + W_3 c_{t-1} + b_1) & c_t &= f_t \otimes c_{t-1} + i_t \otimes tanh(W_7 x_t + W_8 h_{t-1} + b_3) \\ f_t &= 1 - i_t & h_t &= o_t \otimes tanh(c_t) \\ o_t &= \sigma(W_4 x_t + W_5 h_{t-1} + W_6 c_t + b_2) \end{aligned} \tag{4}$$

The W terms are the weight matrices (W_3 and W_6 are diagonal weight matrices for peephole connections); the b terms denote bias vectors; σ is the logistic sigmoid function; and $\otimes$ computes element-wise multiplication of two vectors.

[2] http://www.nd.edu/~mcdonald/Word_Lists.html

A deep LSTM is built by stacking multiple LSTM layers, with the output memory block sequence of one layer forming the input sequence for the next. At each time step the input goes through multiple non-linear layers, which progressively build up higher level representations from the current level. In our information embedding models, we embody a deep LSTM architecture with 2 layers.

One of our baseline information embedding models is a bidirectional LSTM model (Graves et al., 2013), called ***BI-LSTM***, consisting of two 2-layer LSTMs running on the input sequence in both forward and backward directions yielding vectors $\langle \overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{|s|}} \rangle$ and $\langle \overleftarrow{h_{|s|}}, \overleftarrow{h_{|s|-1}}, \ldots, \overleftarrow{h_1} \rangle$, respectively. We exclude stopwords and punctuations from each sentence. The final model outputs the information embedding g by concatenating the final outputs of the two LSTMs, namely $\overrightarrow{h_{|s|}}$ and $\overleftarrow{h_{|s|}}$.

5.2 Dependency Tree-LSTM

A syntactic approach for modeling a sentence s is to use a tree-structured LSTM (Tree-LSTM), embedding the parse tree of a sentence (Le and Zuidema, 2015; Tai et al., 2015; Zhu et al., 2015; Miwa and Bansal, 2016). Our hypothesis is that dependency relations between words convey a certain level of information content. For example, a dependency parser can tell *Facebook* is the subject which did the action *acquired* to the object *Whatsapp* in the sentence *Facebook acquired Whatsapp*. We parse sentences with ZPar (Zhang and Clark, 2011)[3] and adopt the N-ary Tree-LSTM of Tai et al. (2015) with peephole connections to run on a binarized dependency-based parse tree.

For the specific task, we leverage the structure of a binary Tree-LSTM, and develop a novel way to represent a dependency relation between two words using this structure (Section 5.2.1). We propose an algorithm to transform a dependency parse tree to a binary tree, where leaf nodes are words and internal nodes are dependency relations, so that the transformed tree can be embedded using binary Tree-LSTM (Section 5.2.2). Finally we explain how the task of information content effect measurement can benefit from the target-dependent feature of the proposed binarization algorithm (Section 5.2.3).

5.2.1 Binary Tree-LSTM for Dependency Arcs

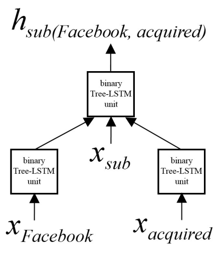
Similar to the LSTM memory block described in Section 5.1, a binary Tree-LSTM unit (Tai et al., 2015) takes *input* x_t at time step t and updates its *cell state* c_t and the *output* of the memory block h_t controlled by *input gate* i_t, *forget gate* f_t and *output gate* o_t. However, instead of depending on only one previous memory block as in a sequential LSTM model, a binary Tree-LSTM unit takes two children units, namely left (l) and right (r), into consideration. In this case, there are two forget gates f_t^l and f_t^r for the left and right children, respectively, so that information from each child can be selectively incorporated. The unit activations are defined by the following set of equations:

$$
\begin{aligned}
i_t &= \sigma(W_9 x_t + \sum_{D\in\{l,r\}} (W_{10}^D h_{t-1}^D + W_{11}^D c_{t-1}^D) + b_4) & c_t &= \sum_{D\in\{l,r\}} f_t^D \otimes c_{t-1}^D \\
f_t^l &= \sigma(W_{12} x_t + \sum_{D\in\{l,r\}} (W_{13}^D h_{t-1}^D + W_{14}^D c_{t-1}^D) + b_5) & &+ i_t \otimes tanh(W_{21} x_t + \sum_{D\in\{l,r\}} W_{22}^D h_{t-1}^D + b_8) \\
f_t^r &= \sigma(W_{15} x_t + \sum_{D\in\{l,r\}} (W_{16}^D h_{t-1}^D + W_{17}^D c_{t-1}^D) + b_6) & h_t &= o_t \otimes tanh(c_t) \\
o_t &= \sigma(W_{18} x_t + \sum_{D\in\{l,r\}} W_{19}^D h_{t-1}^D + W_{20} c_t + b_7)
\end{aligned}
\tag{5}
$$

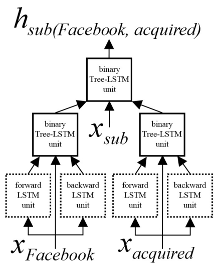
The above binary Tree-LSTM model was proposed to represent binary-branching constituents (Tai et al., 2015). In this paper, however, we show that it can be used to represent dependency arcs. For example, given the subject dependency arc *sub* between *acquired* (head) and *Facebook* (dependent), Figure 1a illustrates the bottom-up information propagation in a binary Tree-LSTM model, where x_{sub}, $x_{acquired}$ and $x_{Facebook}$ are vector representations of *sub*, *acquired* and *Facebook*, respectively. The output of the last unit $h_{sub(Facebook,acquired)}$ is the dependency embedding of *sub(acquired, Facebook)*. We call this model *LEX-TLSTM*.

Miwa and Bansal (2016) incorporate bidirectional LSTMs into the input of a Tree-LSTM unit by concatenating x_i, $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ so that contextual information of individual words can be considered. Instead

[3]We use the default English dependency parser model available at https://github.com/frcchang/zpar/releases/tag/v0.7.5



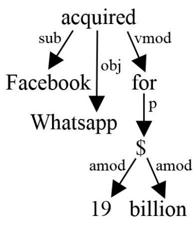
(a) Lexical input.



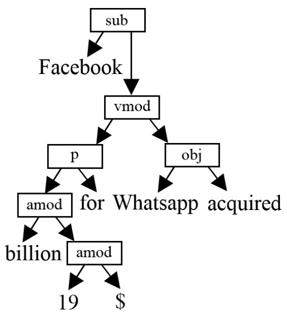
(b) Contextual input.

Figure 1: Binary Tree-LSTM models.

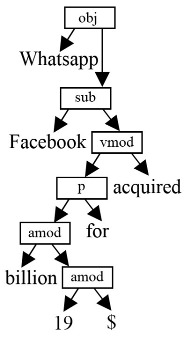
```
Input: target node $n$, dependency parse tree $T_d$
Output: binarized dependency tree $T_b$
$T_b \leftarrow NewBinaryTree$
$T_b.root \leftarrow Binarize(n, T_d)$
Function Binarize(n, $T_d$)
    if $T_d$.HasNoDep(n) then
        return $n$
    end
    if $T_d$.HasParent(n) then
        $p \leftarrow T_d.GetParent(n)$
        $btn \leftarrow NewBinaryTreeNode$
        $btn.dep \leftarrow T_d.RemoveDep(p, n)$
        $btn.LChild \leftarrow Binarize(n, T_d)$
        $btn.RChild \leftarrow Binarize(p, T_d)$
        return $btn$
    end
    if $T_d$.HasChild(n) then
        $c \leftarrow T_d.GetOneChild(n)$
        $btn \leftarrow NewBinaryTreeNode$
        $btn.dep \leftarrow T_d.RemoveDep(n, c)$
        $btn.LChild \leftarrow Binarize(c, T_d)$
        $btn.RChild \leftarrow Binarize(n, T_d)$
        return $btn$
    end
```
Algorithm 1: Dependency tree Binarization



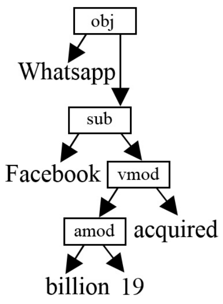
(a) Dependency parse tree.



(b) Binarized dependency tree using the root word *acquired* as the target.



(c) Target dependent tree using *Whatsapp* as the target.



(d) Remove stopwords and punctuations from tree (c).

Figure 2: A dependency parse tree and its binarized versions given different targets.

of inputting the three vectors as a whole into a Tree-LSTM unit, we treat forward and backward 2-layer LSTM units as the left and right children, respectively, while inputting a word vector as shown in Figure 1b, which we refer as *CTX-TLSTM*.

5.2.2 Binarized Dependency Tree

Figure 2a shows the dependency parse tree of the tokenized sentence *Facebook acquired Whatsapp for $ 19 billion*, where the root word is at *acquired* and head words are at the upper ends of dependency branches. To adapt a dependency parse tree to a binary Tree-LSTM model, Algorithm 1 presents a recursive algorithm to binarize a dependency parse tree given a target word.

The algorithm starts from a given target word and creates a binary tree node to represent a dependency relation between the target word and another word, with the dependent and head placed at the left and right children, respectively. The rule for selecting the dependency relation is that the dependency with the target's head is considered first followed by that of the target's dependents. In addition, we sort the dependents of a target word in a way that left context words are always in front of right context words, and both of the left and rigth context words are ordered by the distance to the target word in descending order. For instance, the sorted dependent list of *acquired* in the example is $\langle$*Facebook*, *for*, *Whatsapp*$\rangle$. After deciding a dependency relation to binarize, the head and dependent words become targets of the algorithm to expand the binary tree recursively until all the dependency relations are transformed. After the transformation, a binarized tree has words at leaf nodes, and each internal node represents a dependency relation as shown in Figure 2b.

5.2.3 Target Dependent Tree-LSTM

Recall that our objective is to model the information content effect on a target firm mentioned in a sentence. In the previous example, a bidirectional model would output only one representation for the sentence no matter what the target firm is. In contrast, the proposed tree transformation algorithm outputs different binarized trees when different targets are given. Figure 2b and Figure 2c show the binarized trees using *Facebook*[4] and *Whatsapp* as targets, respectively. As in BI-LSTM, we remove stopwords and punctuations from a tree. Figure 2d demonstrates the result after removing *for* and *\$* from Figure 2c. When one child is ignored, the current node is replaced by the other child.

When applying a binary Tree-LSTM model on a binarized dependency tree, information is propagated from the bottommost leaf nodes to the topmost dependency node (e.g. $\boxed{\text{sub}}$ in Figure 2b), and the final output h is treated as the information embedding g. As the proposed binarization algorithm tends to leave the target at the top of the binary tree, information effectively flows from context to the target firm. For a binary Tree-LSTM model running on a target-dependent tree, we add the prefix *TGT-* to the model identification; otherwise the target is the root word defined by the parser, and *RT-* is prefixed to the model identification.

5.3 Training

We pre-train skip-gram embeddings (Mikolov et al., 2013) of size 100 on a collection of Bloomberg financial news from October 2006 to November 2013, and the size of the trained vocabulary is 320,618. In addition, firm names and an UNK token for representing any words out of the vocabulary are added to the vocabulary, having an initial embedding as the average of the pre-trained word vectors. The word embeddings are fine-tuned during model training, with dropout (Srivastava et al., 2014) using a probability of 0.5 to avoid overfitting. The other hyperparameters for our models along with dependency type representations are initialized according to the method of Glorot et al. (2010)

For sequential LSTMs we use a 2-layer structure with inputs of size 100 and outputs of size 200. For Tree-LSTM models, only one layer is exploited, and the input and output dimensions are the same as that of sequential LSTMs.

Training is done by maximizing the conditional log-likelihood of the target effect category for 15 iterations. The parameters are optimized by stochastic gradient descent with momentum (Rumelhart et al., 1988) using an initial learning rate of 0.005, with L2 regularization at strength 10^{-6}. Every 1,000 training examples the parameters are evaluated on the development set by the macro-averaged F-score, and those achieving the highest value are kept.

6 Experiments Settings

Data: We collect publicly available financial news from Reuters from October 2006 to December 2015. Instead of taking a whole news article or simply a news title into consideration, we target the section of text which appears in the HTML 'class' attribute of 'focus paragraph' of Reuters news articles. This is invariably the first paragraph of such articles, which provide additional detail to the information contained in the article's title. For example, the focus paragraph of the news titled *Exclusive: Target gets tough with vendors to speed up supply chain* (4 May 2016, 12:22pm EDT) is:

Discount retailer Target Corp (TGT.N) is cracking down on suppliers as part of a multi-billion dollar overhaul to speed up its supply chain and better compete with rivals including Wal-Mart Stores Inc (WMT.N) and Amazon.com Inc (AMZN.O).[5]

As *Target Corp*, *Wal-Mart Stores Inc* and *Amazon.com Inc* are mentioned in the paragraph, we assume that the information content of the paragraph affects CAR_3 of these three firms. We ignore focus paragraphs that do not contain any names of U.S.-based, publicly listed firms. Finally, text are grouped per firm per event date, and for each group a CAR_3 is computed accordingly. This yields 22,317 instances, 5,848 of which have information gathered from more than one news. A total of 1,330 firms are covered

[4]Targeting at *acquired* and *Facebook* happen to have the same binarized tree.

[5]http://www.reuters.com/article/us-target-suppliers-exclusive-idUSKCN0XV096

	$+CAR_3$	$-CAR_3$
Train	9,674	9,643
Dev	493	507
Test	995	1,005

Table 1: Numbers of CAR_3 in the datasets.

	$+CAR_3$	$-CAR_3$
Sentiment-based	0.53	0.55
BI-LSTM	0.61	0.62
RT-LEX-TLSTM	0.62	0.63
RT-CTX-TLSTM	0.62	0.63
TGT-LEX-TLSTM	0.63	0.63
TGT-CTX-TLSTM	**0.64**	**0.66**

Table 2: AUCs of different embedding methods.

	$+CAR_3$	$-CAR_3$	$+CAR_3$	$-CAR_3$
			$>+2\%$	$<-2\%$
Sentiment-based	0.53	0.55	0.59	0.58
BI-LSTM	0.58	0.58	0.66	0.63
TGT-CTX-TLSTM	**0.63**	**0.62**	**0.70**	**0.68**

Table 3: Final AUCs.

in our data. We randomly select 1,000 and 2,000 instances as development and test sets, respectively, and the rest are used for training. The numbers of positive and negative CAR_3 examples in the training, development and test sets are fairly balanced, as shown in Table 1.

Evaluation Metric: Although the task of information content effect prediction is a binary classification problem, we do not evaluate our models using the accuracy metric because the data are automatically aligned and some CAR_3 values may not reflect the information correctly. Instead, we evaluate the performance of the models by the area under the precision-recall curve (AUC), where *precision* is the fraction of retrieved positive/negative effect instances that really have positive/negative impact, and *recall* is the fraction of positive/negative effect instances that are retrieved.

6.1 Development Experiments

Table 2 summarizes AUCs of both positive and negative effect predictions on the development set for models using different embedding methods. First, the deep neural embedding approaches outperform the conventional sentiment-based representation widely exploited in finance research. This shows that deep neural models are stronger in capturing news information on and beyond sentiment signals. Second, compared with the sequential embedding strategy, namely BI-LSTM, those Tree-LSTM based (TLSTM) dependency embeddings perform better, demonstrating the benefit of syntactic information. Finally, the information content prediction can benefit from the target-dependent tree transformation (TGT) compared with that using the root word (RT). In addition, the performance of target-dependent models can be improved by incorporating an input word embedding (LEX) with its contextual information (CTX). It is worth noting that the main improvement comes from using the neural feature representation instead of the sentiment word statistics.

6.2 Final Results

Table 3 gives the AUCs for the baseline sentiment-based representation, the sequential embedding BI-LSTM, and the targeted dependency tree method TGT-CTX-TLSTM evaluated on the test set. TGT-CTX-TLSTM outperforms the other baselines, and the improvements between models are statistically significant ($p \leq 0.05$).

One possible application of the proposed model is the use as a security recommender in the financial domain. Thus we apply the model to instances with $|CAR_3| > 2\%$, namely information with high impact. A total of 1,021 instances in the test set pass this threshold. As shown in Table 3 the proposed model achieves AUCs of 0.7 and 0.68 on $+CAR_3$ and $-CAR_3$, respectively. The results not only show the robustness of our model compared to the baselines but also demonstrate its applicability.

To demonstrate the attention mechanism for weighing individual news, Table 4 shows three sets of news, each of which consists news from the same event window and mentioning a specific firm. Both the CAR_3 and the predicted effect probability for each event window are given, and the modeled weight is shown in front of each news, which meets human expectations. For example, one would expect that the news of Wal-Mart Stores Inc missing its profit expectation is more influential than that of being capable of paying shopping using smartphones at Wal-Mart, as shown in the first news group.

- Target: Wal-Mart Stores Inc; CAR$_3$: -4.7%; TGT-CTX-TLSTM: 0.21
  - 0.95 Wal-Mart Stores Inc's (WMT.N) full-year profit may miss analysts' expectations as growth slows in its international markets, pressuring the company even as its U.S. discount stores continue to prosper.
  - 0.05 A group of big retailers that includes Wal-Mart Stores Inc, Target Corp and Japan's 7-Eleven is developing a mobile payment network, adding to the proliferation of options that let consumers pay with smartphones.
- Target: Oshkosh Corp; CAR$_3$: 9.8%; TGT-CTX-TLSTM: 0.81
  - 0.66 Activist investor Carl Icahn offered to buy all the outstanding shares of Oshkosh Corp (OSK.N) Thursday for a 21-percent premium to the U.S. truckmaker's closing price on Wednesday, sending its shares to their highest in more than a year.
  - 0.34 Truck maker Oshkosh Corp (OSK.N) advised its shareholders on Thursday to take no action related to activist investor Carl Icahn's offer to buy all outstanding shares in the company for $32.50 each.
- Target: Sony Corp; CAR$_3$: -2.3%; TGT-CTX-TLSTM: 0.27
  - 0.86 Sony Corp stuck with its full-year profit forecast after slashing its outlook for TV sales, confident that other units will perform better than earlier anticipated to offset additional losses in the unit.
  - 0.14 Japan's biggest technology conglomerates reported quarterly results, with weak TV demand a common theme at both Sony Corp and Sharp Corp.

Table 4: Learned weights for different news.

7 Related Work

Our work is related to research that applies NLP techniques on financial text to predict stock prices and market activities. In terms of corpora, financial news (Leinweber and Sisk, 2011; Xie et al., 2013; Luss and d'Aspremont, 2015; Ding et al., 2015), firm reports (Kogan et al., 2009; Li, 2010; Lee et al., 2014; Qiu et al., 2014) and web content, such as tweets (Bollen et al., 2011; Vu et al., 2012) and forum posts (Das and Chen, 2007; Gilbert and Karahalios, 2010) have been studied. In terms of linguistic features, existing work can be classified into tree major categories: bag-of-words (Kogan et al., 2009; Lee et al., 2014; Qiu et al., 2014), sentiment-based (Das and Chen, 2007; Li, 2010; Bollen et al., 2011; Vu et al., 2012; Luss and d'Aspremont, 2015), and information-retrieval-based (Schumaker and Chen, 2009; Xie et al., 2013; Ding et al., 2015) methods. Our work falls into the category of information retrieval-based features by exploiting syntax information derived from a dependency parser. However, different from the aforementioned work, our goal is not to predict stock prices but to measure the economic value of news information content.

The proposed Tree-LSTM based model for automatically representing syntactic dependencies is in line with recent research that extends the standard sequential LSTM in order to support more complex structures, such as Grid LSTM (Kalchbrenner et al., 2015), Spatial LSTM (Dyer et al., 2015), and Tree-LSTM (Le and Zuidema, 2015; Tai et al., 2015; Zhu et al., 2015; Miwa and Bansal, 2016). We consider the information content prediction as a semantic-heavy task and demonstrate that it can benefit significantly from a novel target-specific dependency Tree-LSTM model.

8 Conclusion

We showed that the impact of information in news release can be predicted using a firm's CAR, and that a proposed target-depend Tree-LSTM model, incorporating contextual information with syntax dependencies, is more effective in representing information content in news text compared to the classic bidirectional LSTM model and a baseline sentiment-based representation. The proposed model can serve as a security assessment tool for financial analysts, and benefit more comprehensive financial models and studies.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ray Ball and Philip Brown. 1968. An empirical evaluation of accounting income numbers. *Journal of Accounting Research*, pages 159–178.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8.

Kyunghyun Cho, Bart Van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Conference on EMNLP*, pages 1724–1734, Doha, Qatar.

Sanjiv R Das and Mike Y Chen. 2007. Yahoo! for Amazon: Sentiment extraction from small talk on the web. *Management Science*, 53(9):1375–1388.

Angela Kay Davis, Jeremy Max Piger, and Lisa Marie Sedor. 2012. Beyond the numbers: Measuring the information content of earnings press release language. *Contemporary Accounting Research*, 29(3):845–868.

Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *Proceedings of the Twenty-Fourth ICJAI*, pages 2327–2333.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A Smith. 2015. Transition-based dependency parsing with stack long short-term memory. *arXiv preprint arXiv:1505.08075*.

Eugene F Fama. 1970. Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25(2):383–417.

Felix Gers and Jürgen Schmidhuber. 2000. Recurrent nets that time and count. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, pages 189–194.

Eric Gilbert and Karrie Karahalios. 2010. Widespread worry and the stock market. In *ICWSM*, pages 59–65.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *International conference on artificial intelligence and statistics*, pages 249–256.

Alan Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. 2013. Hybrid speech recognition with deep bidirectional LSTM. In *ASRU, 2013 IEEE Workshop on*, pages 273–278.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. 2015. Grid long short-term memory. *arXiv preprint arXiv:1507.01526*.

Shimon Kogan, Dimitry Levin, Bryan R Routledge, Jacob S Sagi, and Noah A Smith. 2009. Predicting risk from financial reports with regression. In *Proceedings of the Conference on NAACL*, pages 272–280. Association for Computational Linguistics.

SP Kothari and Jerold B Warner. 2004. The econometrics of event studies. *Handbook of Empirical Corporate Finance*.

Phong Le and Willem Zuidema. 2015. Compositional distributional semantics with long short term memory. In *Joint Conference on Lexical and Computational Semantics*.

Heeyoung Lee, Mihai Surdeanu, Bill MacCartney, and Dan Jurafsky. 2014. On the importance of text analysis for stock price prediction. In *LREC*, pages 1170–1175.

David Leinweber and Jacob Sisk. 2011. Event driven trading and the'new news'. *Journal of Portfolio Management*, 38(1).

Feng Li. 2010. The information content of forward-looking statements in corporate filings — a naïve Bayesian machine learning approach. *Journal of Accounting Research*, 48(5):1049–1102.

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

Ronny Luss and Alexandre d'Aspremont. 2015. Predicting abnormal returns from news using text classification. *Quantitative Finance*, 15(6):999–1012.

A Craig MacKinlay. 1997. Event studies in economics and finance. *Journal of Economic Literature*, 35(1):13–39.

William J Mayew and Mohan Venkatachalam. 2012. The power of voice: Managerial affective states and future firm performance. *The Journal of Finance*, 67(1):1–43.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using LSTMs on sequences and tree structures. *Preprint arXiv: 1601.00770.*

Barak A Pearlmutter. 1989. Learning state space trajectories in recurrent neural networks. *Neural Computation*, 1(2):263–269.

Xin Ying Qiu, Padmini Srinivasan, and Yong Hu. 2014. Supervised learning models to predict firm performance with annual reports: An empirical study. *Journal of the Association for Information Science and Technology*, 65(2):400–413.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3.

Robert P Schumaker and Hsinchun Chen. 2009. Textual analysis of stock market prediction using breaking financial news: The AZFin text system. *ACM Transactions on Information Systems (TOIS)*, 27(2):12.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the Annual Meeting of the ACL.*

Paul C Tetlock. 2007. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168.

Tien-Thanh Vu, Shu Chang, Quang Thuy Ha, and Nigel Collier. 2012. An experiment in integrating sentiment features for tech stock prediction in twitter. In *Proceedings of the Conference on COLING*.

Boyi Xie, Rebecca J Passonneau, Leon Wu, and Germán G Creamer. 2013. Semantic frames to predict stock price movement. In *Proceedings of the Annual Meeting of the ACL*, pages 873–883.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational linguistics*, 37(1):105–151.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *Proceedings of International Conference on Machine Learning*, July.