RiTUAL-UH at SemEval-2017 Task 5: Sentiment Analysis on Financial Data Using Neural Networks

Sudipta Kar* and **Suraj Maharjan*** and **Thamar Solorio**
Department of Computer Science
University of Houston
Houston, TX 77204-3010
{skar3, smaharjan2}@uh.edu, solorio@cs.uh.edu

Abstract

In this paper, we present our systems for "SemEval-2017 Task-5 on Fine-Grained Sentiment Analysis on Financial Microblogs and News". In our system, we combined hand-engineered lexical, sentiment, and metadata features with the representations learned from Convolutional Neural Networks (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU) having Attention model applied on top. With this architecture we obtained weighted cosine similarity scores of 0.72 and 0.74 for subtask-1 and subtask-2, respectively. Using the official scoring system, our system ranked in the second place for subtask-2 and in the eighth place for the subtask-1. However, it ranked first in both subtasks when evaluated with an alternate scoring metric[1].

1 Introduction

Predicting sentiments of financial data has a wide range of applications. The most important application is being able to predict the ups and downs of the share market as the changes in sentiments and opinions can change the market dynamics (Goonatilake and Herath, 2007; Van de Kauter et al., 2015). Stock market related information is typically found in newspapers (Malo et al., 2013) and people discuss them in social media platforms like Twitter and StockTwits. Positive news has the capability to boost the market by increasing optimism among people (Van de Kauter et al., 2015; Schuster, 2003). SemEval-2017 Task 5 on 'Fine-Grained Sentiment Analysis on Financial Microblogs and News' aims at analyzing the polarity of public sentiments from financial data found in newspapers and social media. In this paper, we describe our systems, which exploit automatically learned representations using deep learning architecture based methods along with hand-engineered features in order to predict the sentiment polarity of financial data.

2 Dataset

Task	Training	Trial	Test
Subtask-1	1,694	10	799
Subtask-2	1,142	14	491

Table 1: Data distribution for subtask-1 and subtask-2.

Table 1 shows the distribution of training, trial, and test data for subtask-1 and subtask-2. For the subtask-1, the financial microblogs and tweets were collected from Twitter[2] and StockTwits[3] whereas for subtask-2, the financial news headlines were collected from different financial news sources such as Yahoo Finance[4]. Each instance was labeled with a floating point value ranging from -1 to +1, indicating the sentiment score. A score of -1 means very negative or bearish whereas, a score of +1 means very positive or bullish. A score of 0 means neutral sentiment. The dataset is noisy and contains URLs, cashtags, digits, usernames, and emoticons. The messages are short with an average number of 13 tokens for the microblog data and 10 tokens for the headlines data.

*Both authors contributed equally.

[1] http://alt.qcri.org/semeval2017/task5/data/uploads/description_second_approach.pdf

[2] https://twitter.com

[3] https://stocktwits.com

[4] https://finance.yahoo.com

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 877–882,
Vancouver, Canada, August 3 - 4, 2017.

3 Methodology

We designed two systems for predicting sentiment polarity scores. The first system exploits the hand-engineered features and uses Support Vector Regression (SVR) to predict the sentiment scores. The next system combines the hand-engineered features with representation learned using CNN and Bi-GRU to predict the sentiment scores. These systems are explained below:

3.1 System 1

With the hand-engineered features explained in Section 3.1.1, we built a support vector regression (SVR) model with linear kernel using the implementation of Scikit-learn (Pedregosa et al., 2011). We only used linear kernel as most of the text classification problems are linearly separable (Joachims, 1998). We tuned the *C* parameter through grid search cross-validation over the values {10, 1, 0.1, 1e-02, 1e-03, 1e-04, 1e-05, 1e-06} during the training phase.

3.1.1 Hand-crafted Features

Before extracting the features, we first lowercased, applied stemming and removed stopwords from the messages. We also replaced named entities (NE), and digits with common identifiers to reduce noise.

Lexical: We extracted word n-grams (n=1,2,3) and character n-grams (n=3,4,5) from the messages as they are strong lexical representations (Cavnar and Trenkle, 1994; Mcnamee and Mayfield, 2004; Sureka and Jalote, 2010).

Sentiment: SenticNet (Cambria et al., 2016) have been used successfully in problems related to sentiment analysis (Bravo-Marquez et al., 2014; Poria et al., 2016; Maharjan et al., 2017) as it provides a collection of concept-level opinion lexicons with scores in five dimensions (*aptitude, attention, pleasantness, polarity, and sensitivity*). We used both of the stemmed and non-stemmed versions of the messages to extract concepts from the knowledge base. We modeled the concepts as bag-of-concepts (BoC) and used them as binary features. We averaged the concept scores of five dimensions for each text and used them as numeric features.

Word Embeddings: Word embeddings have been shown to capture semantic information. Hence, in order to capture the semantic representation of the messages, we used publicly available word vectors[5] trained on Google News. It was trained by the method proposed by (Mikolov et al., 2013) and has 3M vocabulary entries. We averaged the word vectors of every word in the messages and represented them with a 300-dimensional vector. If any word is not available in the pre-trained vectors vocabulary, we skipped that word. The coverage of the Google word embedding is 73% and 82% for the microblog and headlines data, respectively.

Metadata: We used the message sources, cashtags and company names as metadata features.

3.2 System 2

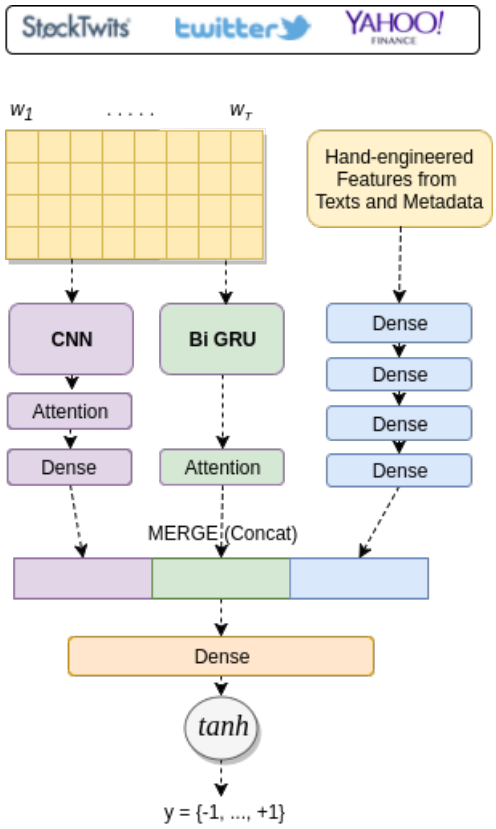

Figure 1: Architecture of System 2

Figure 1 shows the overall system architecture of our neural network model. The main motivation to use deep learning methods is the wide success these methods have achieved in various NLP tasks (Bahdanau et al., 2014; Attia et al., 2016; Samih et al., 2016; Collobert and Weston, 2008). It is a combination of two deep learning architecture based models and a multilayer perceptrons (MLP) model operating on hand-engineered fea-

[5] https://code.google.com/archive/p/word2vec/

tures discussed in Section 3.1.1.

$$S_i \rightarrow x_{1:T} = [x_1, x_2, ..., x_T] \quad (1)$$

We tokenized each messages and represented them as sequences of word vectors as in Equation 1. The maximum length (T) of the sequences was set to 18 for headlines and 33 for microblogs. These lengths were determined from the training data. The embeddings for the words were initialized using pre-trained word embeddings. We used zero vectors to pad the shorter sequences and represent the missing words in the pre-trained vectors. We used Keras (Chollet, 2015) to build the model with Theano (Theano Development Team, 2016) as the back end.

3.2.1 Convolutional Neural Network (CNN)

We used two parallel deep learning architecture based models on the embeddings. As the first model, we used a Convolutional Neural Network (CNN) (LeCun et al., 1989). In this model, we stacked 4 sets of convolution modules with 512 filters each for filter sizes 1, 2, 3, and 4 to capture the n-grams (n = *1 to 4*). The t-th convolution output using filter size c is defined by:

$$h_{c,t} = ReLU(W_c x_{t:t+c-1} + b_c) \quad (2)$$

The filter is applied from window t to window $t + c - 1$ of size c. Each convolution unit calculates a convolution weight W_c and a bias b_c. Each filter of size c produces a high-level feature map h_c.

$$h_c = [h_{c,1}, h_{c,2}, ..., h_{c,T-c+1},] \quad (3)$$

On those filters, we apply pooling operation using an attention layer. Attention models have been used effectively in many problems related to computer vision (Mnih et al., 2014; Ba et al., 2014) and adopted successfully in natural language related problems (Bahdanau et al., 2014; Seo et al., 2016). An attention layer applied on top of a feature map h_i computes the weighted sum c_i.

$$c_i = \sum_j \alpha_{ij} h_{ij} \quad (4)$$

The weight α_{ij} is defined by

$$\alpha_{ij} = \frac{exp(u_{ij}^\top u_w)}{\sum_j exp(u_{ij}^\top u_w)}, \quad (5)$$

where

$$u_{ij} = tanh(W_w h_{ij} + b_w) \quad (6)$$

Here, W_w, b_w and u_w are model parameters. A dense layer containing 128 neurons were applied on the attention layer to get the final representation for the high-level features produced by the CNN model.

3.2.2 Bidirectional GRU (Bi-GRU)

The second model was based on a bidirectional GRU (Bahdanau et al., 2014). It summarized the contextual information from both directions of a sequence and provided annotation for the words. The bidirectional GRU contains a forward GRU of 200 units and another backward GRU of 200 units. The forward GRU $\overrightarrow{f}$ reads a sequence s_i of size n from w_1 to w_n to calculate a sequence of *forward hidden states* $(\overrightarrow{h_1}, ..., \overrightarrow{h_n})$ and the backward GRU $\overleftarrow{f}$ reads the same sequence from w_n to w_1 to calculate a sequence of *backward hidden states* $(\overleftarrow{h_1}, ..., \overleftarrow{h_n})$. For each word w_j, we get an annotation h_j by concatenating the forward hidden state $\overrightarrow{h_j}$ and backward hidden state $\overleftarrow{h_j}$, i.e. $h_j = [\overrightarrow{h_j}; \overleftarrow{h_j}]$.

We applied an attention layer similar to CNN on the word annotations to find out the important features and got a vector of 200 dimensions.

3.2.3 Multilayer Perceptrons (MLP)

To use the hand-engineered features we employed a multilayer perceptron in parallel with the deep learning architecture based models. We fed the extracted features in the input layer and used four hidden dense layers having 200, 100, 50, and 10 neurons respectively. For the feature vector representation $\overrightarrow{x_i} = [x_{i,1}, x_{i,2}, ..., x_{i,T}]$ of message m_i, each neuron of a hidden layer j calculates a vector $\overrightarrow{h_j}$ defined by the following equation.

$$\overrightarrow{h_{ij}} = ReLU(W_{ij} x_i + b_j) \quad (7)$$

Here, W_{ij} is the weight matrix and b_j is the bias vector of the layer j. This model produced a high-level feature representation in the output layer of size 10.

By concatenating the outputs of these three models we created a merged layer of size 338. It contained the three types of high-level features computed by three different types of models. CNN captured the local information, Bi-GRU captured the sequence information and MLP represented the hand-engineered features. We apply a dense layer of 128 neurons on top of this merged layer. It was similar to the layers used in the MLP model but we used *tanh* as the activation function instead of *ReLU* here. The outputs of this layer were passed to the activation layer containing only one neuron having *tanh* as the activation function. We used *tanh* in the final two layers as it produces val-

Model	ST-1 : Microblogs	ST-2 : Headlines
Baseline	0.29	0.68
System 1	**0.70**	0.67
MLP	0.66	0.64
CNN	0.58	0.65
Bi-GRU	0.68	0.68
System 2	0.69	**0.71**

Table 2: Results of 10-fold cross-validation on training and trail data with different models using the official scorer.

ues between -1 and +1 and it is also the range of sentiment scores.

4 Experiments and Results

As the trial dataset too small compared to the training data, we merged it with the training data and ran 10-fold cross-validation to evaluate different models. We tuned the hyper-parameters during the training phase through grid-search cross validation method for System 1. We also experimented with different architectures to build System 1 in this phase.

We evaluated our models using the official scoring system that measures the weighted cosine similarity, similar to the scorer used in Ghosh et al. (2015). As the predictions are continuous values between -1 and +1, cosine similarity measures the degree of alignment between the true values and the predictions. The final weighted score is computed by multiplying the cosine similarity with the ratio of predicted values against the number of test cases. As no official baseline scores were provided, we did a baseline experiment using a simple linear regression model with the hand-engineered features to compare our models.

Table 2 presents the weighted cosine scores achieved by the models we experimented with. System-1 achieved weighted cosine scores of 0.70 and 0.67 for subtask-1 and subtask-2, respectively. Among the neural network models, Bi-GRU performed better than the others. It achieved 0.68 for both of the subtasks. The combination of the three neural network based model (System-2) performed better than the individual models. The neural network models were trained for 10 epochs. We observed issue of overfitting when we increased the number of epochs beyond this.

From the results for subtask-1 we can see that all the other models performed better than CNN. It indicates that other features captured by Bi-GRU and hand-engineering process were more informative than the local information captured by

	ST-1 : Microblogs		ST-2 : Headlines	
Official scorer	sub-2	0.72 (8)	sub-2	0.74 (2)
	sub-1	0.70 (11)	sub-1	0.74 (2)
Alternate scorer	sub-2	0.73 (1)	sub-2	0.71 (1)
	sub-1	0.70 (11)	sub-1	0.70 (3)

Table 3: Weighted cosine scores with ranks achieved by the submissions using the official scorer and the alternate scorer.

	Text	True	Pred.
MB1	Worst performers today: $RIG -13% $EK -10% $MGM $IO -6% $CAR -5,5% / best stock: $WTS +15%	0.857	-0.365
MB2	$GDX $GDXJ $JNUG - strong move today for the Junior Gold Miners - keep an EYE out for a gap fill	0.750	0.750
MB3	Weird day $GPRO up $amba down	-0.649	0.588
HL1	MarketsWolseley shares wilt 8.8% after full year results	-0.787	0.192
HL2	Kingfisher share price: FY statutory pre-tax profit falls 20.5%	-0.426	-0.425
HL3	Shell eyes $700 million exit from Gabon - sources	0.562	-0.123

Table 4: Sample texts from both subtasks with annotated scores and predicted scores by the systems. (MB: Microblogs, HL: Headlines)

CNN. We can understand the strength of the hand-crafted features also by observing the performance of SVR. Although it did not perform as expected on the test data of subtask-1, it showed good performance on the validation set.

We submitted predictions by System-1 (sub-1) and System-2 (sub-2) for subtask-1 as they were the best models. Due to comparatively better performance of System-2 in subtask-2, we submitted predictions from two different models with this system but varied the number of epochs from 10 (sub-1) to 20 (sub-2). For subtask-1, sub-1 and sub-2 was ranked eleventh and eighth, respectively. For subtask-2, both of the submissions achieved almost similar scores and ranked second.

Submitted systems were evaluated simultaneously with an alternate scoring system that measures cosine similarity by grouping instances based on the related company. Our systems ranked the first for both subtasks when evaluated with this scorer.

Table 4 shows that our system worked well when there are more plain texts (*MB2, HL2*). But

it struggles when the text contains more statistics (*e.g. $RIG -13% $EK -10%*) than plain texts or mix of words with strong positive and negative sentiments (*worst, strong, weird, wilt, falls, exit*). If we look at *MB1*, it is very difficult to determine the sentiment polarity from the message. The message starts with *'Worst performers today'*, which indicates a negative polarity. Rest of the message contains statistics for different companies. Among them four indicate a drop in prices and only one indicates a rise in the stock price. It is noticeable that, although the message started with negative impression, it ended with a positive impression by saying *'best stock: $WTS +15%'*. As this is the only possible reason for the highly positive true sentiment polarity score of 0.857 for this message, we get a hint that our systems might need to put more attention on how a message ends.

MB3 starts with the phrase *'Weird day'* followed by a positive and a negative news about stock prices of two companies and our model predicted 0.588 as the polarity score where gold score is -0.649. We tried to find out the possible reason behind our prediction by simply looking at the distribution of the words in this message. In the training data, the word *'weird'* appeared only once. 68% of the 241 messages that contain *'day'* are positive in the training data. Out of 270 messages that contain *'up'*, 201 messages (75%) are positives. We found 118 messages that contain *'down'* and 80 (68%) of them are negative. So, we can see that if a message contains *'up'* and *'down'*, chance of predicting it as positive is higher. Related cashtag $GPRO was found in three messages and $amba appeared only in one message.

Our model predicted a positive sentiment for *HL1* although it contains a clear indication of a negative polarity by the word *'wilt'*. To find out the reason we observed that, *'wilt'* did not appear in the headlines training data at all. Its polarity is -0.087 in the scale of -1 to +1 according to the SenticNet database we used. So we can say that, our model needs to handle this type of trigger word that can control the polarity itself.

5 Conclusions

In this paper, we presented our system for analyzing sentiments from microblogs and news headlines. We used deep learning architecture based models to automatically identify important local and sequential features from the texts and concatenated them with multilayer perceptron-based representation of hand-engineered features extracted from the data. Future works include analyzing the statistics of ups and downs in stock prices of companies from the messages to incorporate them as features of the model.

References

Mohammed Attia, Suraj Maharjan, Younes Samih, Laura Kallmeyer, and Thamar Solorio. 2016. Cogalex-v shared task: Ghhh - detecting semantic relations via word embeddings. In *Proceedings of the 5th Workshop on Cognitive Aspects of the Lexicon (CogALex - V)*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 86–91. http://aclweb.org/anthology/W16-5311.

Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. 2014. Multiple object recognition with visual attention. *CoRR* abs/1412.7755. http://arxiv.org/abs/1412.7755.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473. http://arxiv.org/abs/1409.0473.

Felipe Bravo-Marquez, Marcelo Mendoza, and Barbara Poblete. 2014. Meta-level sentiment models for big social data analysis. *Knowledge-Based Systems* 69:86 – 99.

Erik Cambria, Soujanya Poria, Rajiv Bajpai, and Bjoern Schuller. 2016. Senticnet 4: A semantic resource for sentiment analysis based on conceptual primitives. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 2666–2677. http://aclweb.org/anthology/C16-1251.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*. pages 161–175.

François Chollet. 2015. Keras. `https://github.com/fchollet/keras`.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, pages 160–167.

Aniruddha Ghosh, Guofu Li, Tony Veale, Paolo Rosso, Ekaterina Shutova, John Barnden, and Antonio Reyes. 2015. Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Association for Computational Linguistics, Denver, Colorado, pages

470–478. http://www.aclweb.org/anthology/S15-2080.

Rohitha Goonatilake and Susantha Herath. 2007. The volatility of the stock market and news. *International Research Journal of Finance and Economics* 3(11):53–65.

Thorsten Joachims. 1998. Text categorization with suport vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*. Springer-Verlag, London, UK, UK, ECML '98, pages 137–142. http://dl.acm.org/citation.cfm?id=645326.649721.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* 1(4):541–551. https://doi.org/10.1162/neco.1989.1.4.541.

Suraj Maharjan, John Arevalo, Manuel Montes, Fabio A. González, and Thamar Solorio. 2017. A multi-task approach to predict likability of books. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 1217–1227. http://www.aclweb.org/anthology/E17-1114.

Pekka Malo, Ankur Sinha, Pyry Takala, Pekka J. Korhonen, and Jyrki Wallenius. 2013. Good debt or bad debt: Detecting semantic orientations in economic texts. *CoRR* abs/1307.5336. http://arxiv.org/abs/1307.5336.

Paul Mcnamee and James Mayfield. 2004. Character n-gram tokenization for european language text retrieval. *Inf. Retr.* 7(1-2):73–97.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*. Curran Associates Inc., USA, NIPS'13, pages 3111–3119. http://dl.acm.org/citation.cfm?id=2999792.2999959.

Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. 2014. Recurrent models of visual attention. *CoRR* abs/1406.6247. http://arxiv.org/abs/1406.6247.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Soujanya Poria, Erik Cambria, Newton Howard, Guang-Bin Huang, and Amir Hussain. 2016. Fusing audio, visual and textual clues for sentiment analysis from multimodal content. *Neurocomput.* 174(PA):50–59. https://doi.org/10.1016/j.neucom.2015.01.095.

Younes Samih, Suraj Maharjan, Mohammed Attia, Laura Kallmeyer, and Thamar Solorio. 2016. Multilingual code-switching identification via lstm recurrent neural networks. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*. Association for Computational Linguistics, Austin, Texas, pages 50–59. http://aclweb.org/anthology/W16-5806.

Thomas Schuster. 2003. Meta-communication and market dynamics. reflexive interactions of financial markets and the mass media. Finance, EconWPA.

Paul Hongsuck Seo, Zhe Lin, Scott Cohen, Xiaohui Shen, and Bohyung Han. 2016. Hierarchical attention networks. *CoRR* abs/1606.02393. http://arxiv.org/abs/1606.02393.

Ashish Sureka and Pankaj Jalote. 2010. Detecting duplicate bug report using character n-gram-based features. In *Proceedings of the 2010 Asia Pacific Software Engineering Conference*. IEEE Computer Society, Washington, DC, USA, APSEC '10, pages 366–374. https://doi.org/10.1109/APSEC.2010.49.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* abs/1605.02688. http://arxiv.org/abs/1605.02688.

Marjan Van de Kauter, Diane Breesch, and Véronique Hoste. 2015. Fine-grained analysis of explicit and implicit sentiment in financial news articles. *Expert Syst. Appl.* 42(11):4999–5010. https://doi.org/10.1016/j.eswa.2015.02.007.