Sentiment Analysis by Capsules*

Yequan Wang[1] Aixin Sun[2] Jialong Han[3] Ying Liu[4] Xiaoyan Zhu[1]
[1]State Key Laboratory on Intelligent Technology and Systems
[1]Tsinghua National Laboratory for Information Science and Technology
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[3]Tencent AI Lab, Shenzhen, China
[4]School of Engineering, Cardiff University, UK
tshwangyequan@gmail.com;axsun@ntu.edu.sg;jialonghan@gmail.com;liuy81@cardiff.ac.uk;zxy-dcs@tsinghua.edu.cn

ABSTRACT

In this paper, we propose RNN-Capsule, a capsule model based on Recurrent Neural Network (RNN) for sentiment analysis. For a given problem, one capsule is built for each sentiment category *e.g.,* 'positive' and 'negative'. Each capsule has an attribute, a state, and three modules: representation module, probability module, and reconstruction module. The attribute of a capsule is the assigned sentiment category. Given an instance encoded in hidden vectors by a typical RNN, the representation module builds capsule representation by the attention mechanism. Based on capsule representation, the probability module computes the capsule's state probability. A capsule's state is active if its state probability is the largest among all capsules for the given instance, and inactive otherwise. On two benchmark datasets (*i.e.,* Movie Review and Stanford Sentiment Treebank) and one proprietary dataset (*i.e.,* Hospital Feedback), we show that RNN-Capsule achieves state-of-the-art performance on sentiment classification. More importantly, without using any linguistic knowledge, RNN-Capsule is capable of outputting words with sentiment tendencies reflecting capsules' attributes. The words well reflect the domain specificity of the dataset.

ACM Reference Format:
Yequan Wang[1] Aixin Sun[2] Jialong Han[3] Ying Liu[4] Xiaoyan Zhu[1]. 2018. Sentiment Analysis by Capsules. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186015

1 INTRODUCTION

Sentiment analysis, also known as opinion mining, is the field of study that analyzes people's sentiments, opinions, evaluations, attitudes, and emotions from written languages [20, 26]. Many neural network models have achieved good performance, *e.g.,* Recursive Auto Encoder [33, 34], Recurrent Neural Network (RNN) [21, 35], and Convolutional Neural Network (CNN) [13, 14, 18].

*This work was done when Yequan was a visiting Ph.D student at School of Computer Science and Engineering, Nanyang Technological University, Singapore.

Despite the great success of recent neural network models, there are some defects. First, existing models focus on, and heavily rely on, the quality of instance representations. An instance here can be a sentence, paragraph or document. Using a vector to represent sentiment is much limited because opinions are delicate and complex. The capsule structure in our work gives the model more capacity to model sentiments. Second, linguistic knowledge such as sentiment lexicon, negation words (*e.g.,* no, not, never), and intensity words (*e.g.,* very, extremely), need to be carefully incorporated into these models to realize their best potential in terms of prediction accuracy. However, linguistic knowledge requires significant efforts to develop. Further, the developed sentiment lexicon may not be applicable to some domain specific datasets. For example, when patients give feedback to hospital services, words like 'quick' and 'caring' are all considered strong positive words. These words, are unlikely to be considered strong positive in movie reviews. Our capsule model does not need any linguistic knowledge, and is able to output words with sentiment tendencies to explain the sentiments.

In this paper, we make the very first attempt to perform sentiment analysis by capsules. A capsule is a group of neurons which has rich significance [30]. We design each single capsule[1] to contain *an attribute, a state*, and *three modules* (*i.e.,* representation module, probability module, and reconstruction module).

- The attribute of a capsule reflects its dedicated sentiment category, which is pre-assigned when we build the capsule. Depending on the number of sentiment categories in a given problem, the same number of capsules are built. For example, *Positive Capsule* and *Negative Capsule* are built for a problem with two sentiment categories.
- The state of a capsule, *i.e.,* 'active' or 'inactive', is determined by the probability modules of all capsules in the model. A capsule's state is 'active' if the output of its probability module is the largest among all capsules.
- Regarding the three modules, representation module uses the attention mechanism to build capsule representation; Probability module uses the capsule representation to predict the capsule's state probability; Reconstruction module is used to rebuild the representation of the input instance. The input instance of a capsule model is a sequence (*e.g.,* a sentence, or a paragraph). In this work, the input instance representation of a capsule is computed through RNN.

[1]This work was done before the publication of [30]. Capsule in this work is designed differently from that in [30].

In the proposed RNN-Capsule model, each capsule is capable of, not only predicting the probability of its assigned sentiment, but also reconstructing the input instance representation. Both qualities are considered in our training objectives.

Specifically, for each sentiment category, we build a capsule whose attribute is the same as the sentiment category. Given an input instance, we get its instance representation by using the hidden vectors of RNN. Taking the hidden vectors as input, each capsule outputs: (i) the state probability through its probability module, and (ii) the reconstruction representation through its reconstruction module. During training, one objective is to maximize the state probability of the capsule corresponding to the groundtruth sentiment, and to minimize the state probabilities of other capsule(s). The other objective is to minimize the distance between the input instance representation and the reconstruction representation of the capsule corresponding to the ground truth, and to maximize such distances for other capsule(s). In testing, a capsule's state becomes 'active' if its state probability is the largest among all capsules for a given test instance. The states of all other capsule(s) will be 'inactive'. Attribute of the active capsule is selected to be the predicted sentiment category of the test instance.

Compared with most existing neural network models for sentiment analysis, RNN-Capsule model does not heavily rely on the quality of input instance representation. In particular, the RNN layer in our model can be realized through the widely used Long Short-Term Memory (LSTM) model, Gated Recurrent Unit (GRU) model or their variants. RNN-Capsule does not require any linguistic knowledge. Instead, each capsule is capable of outputting words with sentiment tendencies reflecting its assigned sentiment category. Recall that the representation module of a capsule uses attention mechanism to build the capsule representation. We observe through experiments that the attended words by each capsule well reflect the capsule's sentiment category. These words reflect the domain specificity of the dataset, although not included in sentiment lexicon. For instance, our model is able to identify 'professional', 'quick', and 'caring' as strong positive words in patient feedback to hospitals. We also observe that the attended words include not only high frequency words, but also medium and low frequency words, and even typos which are common in social media. These domain dependent sentiment words could be extremely useful for decision makers to identify the positive and negative aspects of their services or products. The main contributions are as follows:

- To the best of our knowledge, RNN-Capsule is the first attempt to use capsule model for sentiment analysis. A capsule is easy to build with input instance representations taken from RNN. Each capsule contains an attribute, a state, and three simple modules (representation, probability, and reconstruction).
- We demonstrate that RNN-Capsule does not require any linguistic knowledge to achieve state-of-the-art performance. Further, capsule model is able to attend opinion words that reflect domain knowledge of the dataset.
- We conduct experiments on two benchmark datasets and one proprietary dataset, to compare our capsule model with strong baselines. Our experimental results show that capsule model is competitive and robust.

2 RELATED WORK

Early methods for sentiment analysis are mostly based on manually defined rules. With the recent development of deep learning techniques, neural network based approaches become the mainstream. On this basis, many researchers apply linguistic knowledge for better performance in sentiment analysis.

Traditional Sentiment Analysis. Many methods for sentiment analysis focus on feature engineering. The carefully designed features are then fed to machine learning methods in a supervised learning setting. Performance of sentiment classification therefore heavily depends on the choice of feature representation of text. The system in [24] implements a number of hand-crafted features, and is the top performer in SemEval 2013 Twitter Sentiment Classification Track. Other than supervised learning, Turney [38] introduces an unsupervised approach by using sentiment words/phrases extracted from syntactic patterns to determine document polarity. Goldberg and Zhu [6] propose a semi-supervised approach where the unlabeled reviews are utilized in a graph-based method.

In terms of features, different kinds of representations have been used in sentiment analysis, including bag-of-words representation, word co-occurrences, and syntactic contexts [26]. Despite its effectiveness, feature engineering is labor intensive, and is unable to extract and organize the discriminative information from data [7].

Sentiment Analysis by Neural Networks. Since the proposal of a simple and effective approach to learn distributed representations of words and phrases [23], neural network based models have shown their great success in many natural language processing (NLP) tasks. Many models have been applied to sentiment analysis, including Recursive Auto Encoder [4, 29, 33], Recursive Neural Tensor Network [34], Recurrent Neural Network [22, 36], LSTM [9], Tree-LSTMs [35], and GRU[3].

Recursive autoencoder neural network builds the representation of a sentence from subphrases recursively [4, 29, 33]. Such recursive models usually depend on a tree structure of input text. In order to obtain competitive results, all subphrases need to be annotated. By utilizing syntax structures of sentences, tree-based LSTMs have proved effective for many NLP tasks, including sentiment analysis [35]. However, such models may suffer from syntax parsing errors which are common in resource-lacking languages. Sequence models like CNN, do not require tree-structured data, which are widely adopted for sentiment classification [13, 14]. LSTM is also common for learning sentence-level representation due to its capability of modeling the prefix or suffix context as well as tree-structured data [9, 35]. Despite the effectiveness of those methods, it is still challenging to discriminate different sentiment polarities at a fine-grained level.

In [8], the proposed neural model improves coherence by exploiting the distribution of word co-occurrences through the use of neural word embeddings. The list of top representative words for each inferred aspect reflects the aspect, leading to more meaningful results. The approach in [19] combines two modular components, generator and encoder, to extract pieces of input text as justifications. The extracted short and coherent pieces of text alone is sufficient for the prediction, and can be used to explain the prediction.

Linguistic Knowledge. Linguistic knowledge has been carefully incorporated into models to realize the best potential in terms of prediction accuracy. Classical linguistic knowledge or sentiment resources include sentiment lexicons, negators, and intensifiers.

Sentiment lexicons are valuable for rule-based or lexicon-based models [10]. There are also studies for automatic construction of sentiment lexicons from social data [39] or from multiple languages [2]. Recently, a context-sensitive lexicon-based method was proposed based on a simple weighted-sum model [37]. It uses an RNN to learn the sentiments strength, intensification, and negation of lexicon sentiments in composing the sentiment value of sentences. Aspect information, negation words, sentiment intensities of phrases, parsing tree and combination of them were applied into models to improve their performance. Attention-based LSTMs for aspect-level sentiment classification were proposed in [40]. The key idea is to add aspect information to the attention mechanism. A linear regression model was proposed to predict the valence value for content words in [41]. The valence degree of the text can be changed because of the effect of intensity words. In [28], sentiment lexicons, negation words, and intensity words are all considered into one model for sentence-level sentiment analysis.

However, linguistic knowledge requires significant human effort to develop. The developed sentiment lexicon may not be applicable to some domain specific dataset. All of those limit the application of models based on linguistic knowledge.

3 RNN-CAPSULE MODEL

The architecture of the proposed RNN-based capsule model is shown in Figure 1. The number of capsules N is the same as the number of sentiment categories to be modeled, each corresponding to one sentiment category. For example, five capsules are used to model five fine-grained sentiment categories: 'very positive', 'positive', 'neutral', 'negative', and 'very negative'. Each sentiment category is also known as the capsule's attribute.

All capsules take the same instance representation as their input, which is computed by an RNN network, as shown in the figure. The RNN can be materialized by Long Short-Term Memory (LSTM) model, Gated Recurrent Unit (GRU) or their variants, *e.g.,* bi-directional and two-layer LSTM. Given an instance (*e.g.,* a sentence, or a paragraph), represented in dense vector, RNN encodes the instance and outputs the hidden vectors. The instance is then represented by the hidden vectors. That is, the input to all capsules are the hidden vectors of RNN encoding.

In the top row of Figure 1, each capsule outputs a state probability and a reconstruction representation, through its probability module and its reconstruction module, respectively. Among all capsules, the one with the highest state probability will become 'active' and the rest will be 'inactive'. During training, one objective is to maximize the state probability of the capsule corresponding to the ground truth sentiment, and to minimize the state probability of the rest capsule(s). The other objective is to minimize the distance between the reconstruction representation of the capsule selected by ground truth and the instance representation, and to maximize such distances for other capsule(s). In the testing process, a capsule's state will be 'active' if its state probability is the largest among all capsules. All other capsule(s) will then be 'inactive' because only

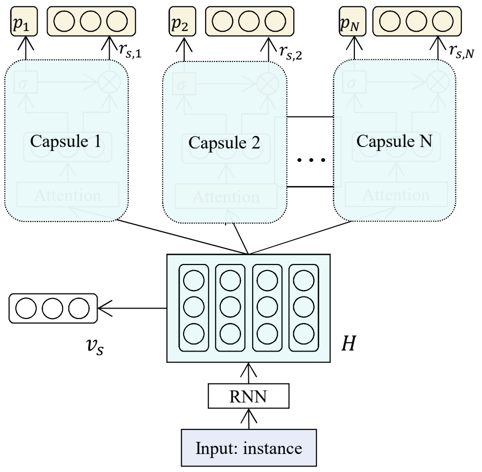

Figure 1: Architecture of RNN-Capsule. The number of capsules equals the number of sentiment categories. $H = [h_1, h_2, \ldots, h_{N_s}]$ is the hidden vectors of an input instance encoded by RNN, where N_s is the number of words. The instance representation $v_s = \frac{1}{N_s}\sum_{i=1}^{N_s} h_i$ is the average of the hidden vectors. All capsules take the hidden vectors as input, and each capsule outputs a state probability p_i and a reconstruction representation $r_{s,i}$.

one capsule can be in active state. The active capsule's attribute is selected as the test instance's sentiment category.

Because the capsule model is based on RNN, next we give preliminaries of RNN before detailing the capsule structure and the training objective.

3.1 Recurrent Neural Network

A Recurrent Neural Network (RNN) is a class of artificial neural network where connections between units form a directed cycle. This allows the network to exhibit dynamic temporal behavior. Unlike feedforward neural networks, RNNs can use their internal memory to process arbitrary sequences of inputs. However, it is known that standard RNNs have the problem of gradient vanishing or exploding. To overcome these issues, Long Short-term Memory network (LSTM) was developed and has shown superior performance in many tasks [9].

Briefly speaking, in LSTM, the hidden states h_t and memory cell c_t are function of the previous h_{t-1} and c_{t-1}, and input vector x_t, or formally:

$$c_t, h_t = \mathrm{LSTM}(c_{t-1}, h_{t-1}, x_t) \tag{1}$$

The hidden state h_t denotes the representation of position t while encoding the preceding contexts of the position. For more details about LSTM, we refer readers to [9].

A variation of LSTM is the Gated Recurrent Unit (GRU), introduced in [3]. It combines the forget gate and input gate into a single *update gate.* It also merges the cell state and hidden state, among other changes. The resulting model is simpler than standard LSTM models, and has become a popular model in many tasks. Similarly, the hidden state h_t in GRU denotes the representation of position t

while encoding the preceding contexts of the position (see [3] for more details) .

$$h_t = \text{GRU}(h_{t-1}, x_t) \tag{2}$$

RNN can be bi-directional, by using a finite sequence to predict or label each element of the sequence based on the element's past and future contexts. This is achieved by concatenating the outputs of two RNNs, one processes the sequence from left to right, and the other from right to left.

Instance Representation. As shown in Figure 1, the instance representation to all capsules is encoded by RNN. Formally, the instance representation v_s, is the average of the hidden vectors obtained from RNN.

$$v_s = \frac{1}{N_s} \sum_{i=1}^{N_s} h_i, \tag{3}$$

where N_s is the length of instance, *e.g.,* number of words in a given sentence. Here, each word is represented by a dense vector obtained through word2vec or similar techniques.

3.2 Capsule Structure

The structure of a single capsule is shown in Figure 2. A capsule contains three modules: *representation module*, *probability module* and *reconstruction module*. Representation module uses attention mechanism to build the capsule representation $v_{c,i}$. Probability module uses sigmoid function to predict the capsule's active state probability p_i. Reconstruction module computes the reconstruction representation of an instance by multiplying p_i and $v_{c,i}$.

Representation Module. Given the hidden vectors encoded by RNN, we use the attention mechanism to construct capsule representation inside a capsule. The attention mechanism enables the representation module to decide the importance of words based on the prediction task. For example, word 'clean' is likely to be informative and important in patient feedback to hospital. However, this word is less important if it appears in movie review.

We use an attention mechanism inspired by [1, 5, 40, 44] with a single parameter in capsule:

$$e_{t,i} = h_t w_{a,i} \tag{4}$$

$$\alpha_{t,i} = \frac{exp(e_{t,i})}{\sum_{j=1}^{N_s} exp(e_{j,i})} \tag{5}$$

$$v_{c,i} = \sum_{j=1}^{N_s} a_{t,i} h_t \tag{6}$$

In the above formulation, h_t is the representation of word at position t (*i.e.,* the hidden vector from RNN) and $w_{a,i}$ is the parameter of capsule i for the attention layer. The attention importance score for each position, $\alpha_{t,i}$, is obtained by multiplying the representations with the weight matrix, and then normalizing to a probability distribution over the words. $\alpha_i = [\alpha_{1,i}, \alpha_{2,i}, \ldots, \alpha_{N_s,i}]$. Lastly, the capsule representation vector, $v_{c,i}$, is a weighted summation over all the positions using the attention importance scores as weights.

Note that, this capsule representation vector obtained from the attention layer is a high-level encoding of the entire input text. This capsule representation vector will be used to reconstruct the presentation of the input instance. We observe that adding the attention mechanism improves the model's capability and robustness.

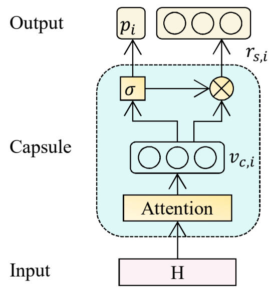

Figure 2: The architecture of a single capsule. The input to a capsule is the hidden vectors $H = [h_1, h_2, \ldots, h_{N_s}]$ from RNN.

Probability Module. After getting the capsule representation vector $v_{c,i}$, we calculate the active state probability p_i through

$$p_i = \sigma(W_{p,i} v_{c,i} + b_{p,i}), \tag{7}$$

where $W_{p,i}$ and $b_{p,i}$ are the parameters for the active probability of the current capsule i.

The parameters are learned based on the aforementioned objectives, *i.e.,* maximizing the state probability of capsule selected by ground truth sentiment, and minimizing the state probability of other capsule(s). In testing, a capsule's state will be active if p_i is the largest among all capsules.

Reconstruction Module. The reconstruction representation of an input instance is obtained by multiplying $v_{c,i}$ and probability p_i

$$r_{s,i} = p_i v_{c,i}, \tag{8}$$

where p_i is the active state probability of the current capsule and $v_{c,i}$ is the capsule vector representation.

The three modules complement each other. The capsule representation matches its attribute, and the state of one capsule is corresponding to the input instance. Therefore, the probability module, which is based on the capsule representation, will be the largest if the capsule's sentiment fit the input instance. Reconstruction module is developed from the capsule representation and its state probability, so the reconstruction representation is able to stand for the input instance representation if its state is 'active'.

3.3 Training Objective

The training objective of the proposed capsule model considers two aspects. One is to minimize the reconstruction error and maximize the active state probability of the capsule matching ground truth sentiment. The other is to maximize the reconstruction error and minimize the active state probability of other capsule(s). To achieve the objective, we adopt the contrastive max-margin objective function that has been used in many studies [8, 11, 32, 42].

Probability Objective. Because only one capsule is active for each given training instance, we have both positive sample (*i.e.,* the active capsule) and negative samples (*i.e.,* the remaining inactive capsules). Recall that our objective is to maximize the active state probability of the active capsule and to minimize the probabilities of inactive capsules. The unregularized objective J can be formulated as a hinge loss:

$$J(\theta) = \sum \max(0, 1 + \sum_{i=1}^{N} y_i p_i) \tag{9}$$

For a given training instance, $y_i = -1$ for the active capsule (*i.e.,* the one that matches the training instance's ground truth sentiment). All remaining y's are set to 1. We use a mask vector to indicate which capsule is active for each training instance.

Reconstruction Objective. The other objective is to ensure that the reconstruction representation $r_{s,i}$ of the active capsule is similar to the instance representation v_s, meanwhile v_s is different from the reconstruction representations of inactive capsules. Similarly, the unregularized objective U can be formulated as another hinge loss that maximizes the inner product between $r_{s,i}$ and v_s and simultaneously minimizes the inner product between $r_{s,i}$ from the inactive capsules and v_s:

$$U(\theta) = \sum \max(0, 1 + \sum_{i=1}^{N} y_i v_s r_{s,i}) \tag{10}$$

Again, $y_i = -1$ if the capsule is active and $y_i = 1$ if the capsule is inactive.

Considering both objectives, our final objective function L is obtained by adding J and U:

$$L(\theta) = J(\theta) + U(\theta) \tag{11}$$

4 EXPERIMENT

4.1 Dataset

We conduct experiments on two benchmark datasets, namely Movie Review (MR) [25] and Stanford Sentiment Treebank (SST) [31], and one proprietary dataset. Both MR and SST have been widely used in sentiment classification evaluation which enables us to benchmark our result against the published results.

Movie Review. Movie Review (MR)[2] is a collection of movie reviews in English [25], collected from www.rottentomatoes.com. Each instance, typically a sentence, is annotated with its source review's sentiment categories, either 'positive' or 'negative'. There are 5331 positive and 5331 negative processed sentences.

Stanford Sentiment Treebank. SST[3] is the first corpus with fully labeled parse trees, which allows for a comprehensive analysis of the compositional effects of sentiment in language [31]. This corpus is based on the dataset introduced by Pang and Lee [25]. It includes fine-grained sentiment labels for 215,154 phrases parsed by the Stanford parser [16] in the parse trees of 11,855 sentences. The sentiment label set is {0,1,2,3,4}, where the numbers correspond to 'very negative', 'negative', 'neutral', 'positive', and 'very positive', respectively. Note that, because SST provides phrase-level annotations on the parse trees, some of the reported results are obtained based on the phrase-level annotations. In our experiments, we only utilize the sentence-level annotations because our capsule model does not need the expensive phrase-level annotation.

[2]Sentence polarity dataset v1.0. http://www.cs.cornell.edu/people/pabo/movie-review-data/
[3]https://nlp.stanford.edu/sentiment/index.html

Table 1: Number of instances in hospital feedback dataset

Question	Sentiment	Number of answers
What I liked?	Positive	25,042
What could be improved?	Negative	21,240

Hospital Feedback. We use a proprietary patient opinion dataset that was generated by a non-profit feedback platform for health services in the UK.

We use the text content from the feedback forms filled by patients. Specifically, we make sentiment analysis on the answers of two questions: *"What I liked?"*, and *"What could be improved?"*. There is another question in the feedback form: *Anything else?* whose answers are not used in our experiments because the sentiment is uncertain. The number of answers (or instances) to the two questions are reported in Table 1.

Given the large number of instances, manually annotating all sentences in hospital feedback is time consuming. In this study, we simply consider an answer to the question *"What I liked?"* processes 'positive' sentiment, and an answer to the question *"What could be improved?"* processes 'negative' sentiment. The average length of the answers is about 120 words, and we consider each answer as one instance without further splitting an answer into sentences.

We note that the simple labeling scheme (*i.e.,* assigning answers to *"What I liked?"* positive and answers to *"What could be improved?"* negative) introduces some noise in the dataset. A patient may write "perfect, nothing to improve" to answer "What could be improved", and will be labeled as 'negative'. Such noise cannot be avoided without manual annotation. However, their number is negligible by observation.

4.2 Implementation Details

In our experiments, all word vectors are initialized by Glove[4]. The word embedding vectors are pre-trained on an unlabeled corpus whose size is about 840 billion and the dimension of word vectors we used is 300 [27]. The dimension of hidden vectors encoded by RNN is 256 if the RNN is single-directional, and 512 if the RNN is bi-directional. More specifically, on MR and SST datasets, we use bi-directional and two-layer LSTM, and on Hospital Feedback dataset, we use two-layer GRU. The models are trained with a batch size of 32 examples on SST, 64 examples on MR and Hospital Feedback datasets. There is a checkpoint every 32 mini-batch on SST, and 64 on MR and Hospital Feedback dataset. The embedding dropout is 0.3 on MR and Hospital Feedback dataset, and 0.5 on SST. The same RNN cell dropout of 0.5 is applied on all the three datasets. The dropout on capsule representation in probability modules of capsules is also set to 0.5 on all datasets. The length of attention weights is the same as the length of sentence.

We use Adam [15] as our optimization method. The learning rate for model parameters except word vectors are $1e-3$, and $1e-4$ for word vectors. The two parameters β_1 and β_2 in Adam are 0.9 and 0.999, respectively. The capsule models are implemented on Pytorch[5] (version 0.2.0_3) and the model parameters are randomly initialized.

[4]http://nlp.stanford.edu/projects/glove/
[5]https://github.com/pytorch

Table 2: The accuracy of methods on Movie Review (MR) and Stanford Sentiment Treebank (SST) datasets. Note that the models only use sentence-level annotation and not the phrase-level annotation in SST. The accuracy marked with * are reported in [12, 14, 18, 33]; and the accuracy marked with # are reported in [28].

Model	Movie Review (MR)	SST (Sentence-level)
RAE	77.7*	43.2*
RNTN	75.9#	43.4#
LSTM	77.4#	45.6#
Bi-LSTM	79.3#	46.5#
LR-LSTM	81.5#	48.2#
LR-Bi-LSTM	82.1#	48.6#
Tree-LSTM	80.7#	48.1#
CNN	81.5*	46.9#
CNN-Tensor	-	**50.6***
DAN	-	47.7*
NCSL	82.9#	47.1#
RNN-Capsule	**83.8**	49.3

4.3 Evaluation on Benchmark Datasets

Both MR and SST datasets have been widely used in evaluating sentiment classification. This gives us the convenience of directly comparing the result of our proposed capsule model against the reported results using the same experimental setting. Table 2 lists the accuracy of sentiment classification of baseline methods on the two datasets reported in a recent ACL 2017 paper [28]. Our capsule model, named RNN-Capsule, is listed in the last row.

Baseline Methods. We now briefly introduce the baseline methods, all based on neural networks. Recursive Auto Encoder (RAE, also known as RecursiveNN) [33] and Recursive Tensor Neural Network (RNTN) [31] are based on parsing trees. RNTN uses tensors to model correlations between different dimensions of child nodes' vectors. Bidirectional LSTM (Bi-LSTM) is a variant of LSTM which is introduced in Section 3.1. Both LSTM and Bi-LSTM are based on sequence structure of the sentences. LR-LSTM and LR-Bi-LSTM are linguistically regularized variants of LSTM and Bi-LSTM, respectively. Tree-Structured LSTM (Tree-LSTM) [35] is a generalization of LSTMs to tree-structured network topologies. Convolutional Neural Network (CNN) [14] uses convolution and pooling operations, which is popular in image captioning. CNN-Tensor [18] is different from CNN where the convolution operation is replaced by tensor product. Dynamic programming is applied in CNN-Tensor to enumerate all skippable trigrams in a sentence. Deep Average Network (DAN) [12] has three layers: one layer to average all word vectors in a sentence, an MLP layer, and the last layer is the output layer. Neural Context-Sensitive Lexicon (NCSL) [37] uses a Recurrent Neural Network to learn the sentiments values, based on a simple weighted-sum model, but requires linguistic knowledge.

Observations. On the Movie Review dataset, our proposed RNN-Capsule model achieves the best accuracy of 83.8. Among the baseline methods, LR-Bi-LSTM and NCSL outperform the other baselines. However, both LR-Bi-LSTM and NCSL requires linguistic knowledge like sentiment lexicon and intensity regularizer. It is worth noting that lots of human efforts are required to build such linguistic knowledge. Our capsule model does not use any linguistic knowledge. On the SST dataset, our model is the second best performer after CNN-Tensor. However, CNN-Tensor is much more computationally intensive due to the tensor product operation. Our model only requires simple linear operations on top of the hidden vectors obtained through RNN. Our model also outperforms other strong baselines like LR-Bi-LSTM which requires dedicated linguistic knowledge.

Table 3: Accuracy on Hospital Feedback Dataset

Method	Accuracy
Navie Bayes	84.7
Navie Bayes (+Bigram)	81.9
Linear SVM	87.6
Linear SVM (+Bigram)	88.9
Word2vec-SVM (CBOW)	85.5
Doc2vec-SVM (PV-DM)	77.7
Doc2vec-SVM (PV-DBOW)	81.8
Doc2vec-SVM (PV-DM+PV-DBOW)	83.2
LSTM	89.8
Attention-LSTM	90.2
RNN-Capsule	**91.6**

4.4 Evaluation on Hospital Feedback

Baseline Methods. We now evaluate RNN-Capsule on the hospital feedback dataset. Although neural network models have shown their effectiveness on many other datasets, it is better to provide a complete performance overview for a new dataset. To this end, we evaluate three kinds of baseline methods listed in Table 3: (i) The traditional machine learning models based on Naive Bayes and Support Vector Machines (SVMs) using unigram and bigram representations; (ii) SVMs with dense vector representations obtained through Word2vec and Doc2vec; and (iii) LSTM based baselines, due to the promising accuracy obtained by LSTM based models among neural network models reported earlier.

Specifically, for the model named Word2vec-SVM, word vectors learned through CBOW are used to learn the SVM classifiers on patient feedback. Each feedback is represented by the averaged vector of its words. For Doc2vec-SVM, Doc2vec is used to learn vectors for all feedbacks where PV-DBOW, PV-DM, or their concatenation (*i.e.,* PV-DBOW + PV-DM) are used [17]. Because attention mechanism is utilized in our RNN-Capsule model, we also evaluated Attention-LSTM. This model is the same as LSTM, except that an additional attention weight vector is trained. The weight vector is applied to the LSTM outputs at every position to produce weights for different time stamps. The weighted average of LSTM outputs is used for sentiment classification[6]. Naive Bayes, Linear SVM, word2vec/doc2vec, and LSTM/Attention-LSTM are implemented by using NLTK, Scikit-learn, Gensim, and Keras, respectively.

[6]From each instance, up to the first 300 words are used in LSTM models for computational efficiency. More than 90% of the instances are shorter than 300 words.

Observations. Among traditional machine learning models based on Naive Bayes and Support Vector Machines, Linear SVM learned by using both unigram and bigram (*i.e.,* Linear SVM (+Bigram)) is a clear winner with accuracy of 88.9. This accuracy is much higher than all SVM models learn on dense representation from either Word2vec or Doc2vec.

LSTM-based methods outperform Linear SVM with bigram. When enhanced with the attention mechanism, attention-LSTM slightly outperforms the vanilla LSTM by achieving accuracy of 90.2. Our proposed model, RNN-Capsule, being the top-performer, further improves the accuracy to 91.6.

5 EXPLAINABILITY ANALYSIS

In Section 4, we show that RNN-Capsule achieves comparable or better accuracy than state-of-the-art models, without using any linguistic knowledge. Now, we show that RNN-Capsule is capable of outputting words with sentiment tendencies reflecting domain knowledge. In other words, we try to explain for a given dataset, based on which words, our RNN-Capsule model predicts the sentiment categories. These domain dependent sentiment words could be extremely useful for decision makers to identify the positive and negative aspects of their services or products.

Attended Words by Capsule. Because of the attention mechanism in our capsule model, each word is assigned an attention weight. The attention weight of a word is computed as follows:

$$w_{c,i} = p_i \alpha_i, \tag{12}$$

where p_i is the active state probability of capsule i, and α_i is the attention weight in the representation module of capsule i.

Because each capsule corresponds to one sentiment category, we collect the attended words by individual capsules. More specifically, for each capsule, we build a dictionary, where the key is a word and the value is the sum of attention weights for this word in the capsule, as the word may appear in multiple test instances. The sum of attention weights is updated for the word only if the capsule is 'active' for the input instance. After evaluating all test instances, we get the list of attended words for each capsule with their attention weights.

A straightforward way of ranking the attended words is to compute the *averaged attention weight* for each word (recall a word may appear multiple times). We observe that many top-ranked words are of low frequency. That is, the words have very high attention weight (or strong sentiment tendencies) but do not appear often. To get a ranking of medium and high frequency words that are attended by each capsule, we multiple averaged attention weight of a word and *the logarithm of word frequency.* In the following, we discuss both rankings: the attended words with medium/high word frequency, and the attended words with low frequency.

5.1 Attended Words with Medium/High Word Frequency

Tables 4a, 4b and 4c list the top 20 ranking words attended by the different capsules on the three datasets. The words are ranked by the product of averaged attention weight and the logarithm of word frequency. Most of the words have medium to high word frequency in the corresponding dataset. All the words are self-explanatory for the assigned sentiment category. To further verify the sentiment tendencies of the words, we match the words with a sentiment lexicon [43]. In this sentiment lexicon, there are six sentiment tendencies, {'strong-positive', 'weak-positive', 'weak-neutral', 'strong-neutral', 'weak-negative', 'strong-negative'}. We indicate the matching words in the tables using {++, +, 0^-, 0^+, −, − −} for the six sentiment tendencies. The words that are not included in the sentiment lexicon are marked with 'N'. There are also words, which do not match any words in sentiment lexicon but are possible to match with morphological changes. We indicate these underlined words like 'fails' and 'lacks'. Note that the punctuation marks are processed as tokens and it is not surprising that many of them are attended by the neutral capsule.

Observe from the three tables, the attended words not only well reflect the sentiment tendencies, but also reflect the domain difference. We use the hospital feedback as an example (see Table 4c). Word 'leave' or 'leaving' in most contexts are considered not having any sentiment tendencies. The word is not included in the sentiment lexicon as expected. However, it is ranked at the second position in the positive capsule on hospital feedback. A closer look at the dataset shows that many patients express their happiness for being able to 'leave' hospital or being able to 'leave' earlier than expected. The words like 'quickly', 'attentative', 'professional', 'cared', and 'caring' clearly make sense for carrying strong positive sentiments in the context of the dataset. For the negative capsules, because the sentences are for answering the question *'What could be improved'*, many of them contain various forms of 'improve'. From answers like 'perfect, nothing to improve', the words 'perfect' and 'nothing' are attended. There are also patients requesting to improve 'everything', particularly, 'parking'.

5.2 Attended Words with Low Word Frequency

Tables 5a, 5b and 5c list the top 20 words by average attention weights. Most of them are low frequency words with no more than three appearances. Again, the words are self-explainable for the corresponding sentiment category. On movie review dataset, our negative capsule identifies 'dopey', 'execrable', 'self-satisfied', and 'cloying' as strong negative words which are very meaningful for comments on movies. Interestingly, the capsule model is not sensitive to typos, which are common in social media. The word 'noneconsideratedoctors' is attended to be negative with the correct spelling of 'none considerate doctors'.

From these tables, we demonstrate that our capsule model is capable of outputting words with sentiment tendencies reflecting domain knowledge, even if the words only appear one or two times.

6 CONCLUSION

The key idea of RNN-Capsule model is to design a simple capsule structure and use each capsule to focus on one sentiment category. Each capsule outputs its active probability and the reconstruction representation. The objective of learning is to maximize the active probability of the capsule matching the ground truth and to minimize its reconstruction representation with the given instance representation. At the same time, the other capsules' active probability

Table 4: Medium/high frequency words attended by different capsules on the three datasets. {++, +, 0^-, 0^+, −, − −} indicate {'strong-positive', 'weak-positive', 'weak-neutral', 'strong-neutral', 'weak-negative', 'strong-negative'} respectively, based on the sentiment lexicon [43]. 'N' denotes that the word is not included in the sentiment lexicon. A word is underlined if the word does not match any word in sentiment lexicon but matches a morphological variant of a word in sentiment lexicon.

(a) Stanford Sentiment Treebank

No.	Very-Pos-Cap	Attr	Pos-Cap	Attr	Neutral-Cap	Attr	Neg-Cap	Attr	Very-Neg-Cap	Attr
1	best	++	enjoy	+	?	N	n't	N	bad	− −
2	hilarious	++	good	+	.	N	no	N	worst	− −
3	excellent	++	worthwhile	++	!	N	too	−	ugly	− −
4	astonishing	++	worth	++	but	N	fails	N	mess	− −
5	wonderful	++	funny	++	,	N	nothing	N	incoherent	− −
6	brilliant	++	refreshing	++	not	N	not	N	unfunny	N
7	stunning	++	delivers	N	hopkins	N	lacks	N	waste	−
8	rare	0^-	fun	++	there	N	problem	−	unpleasant	− −
9	spectacular	++	intelligent	++	point	0^-	never	N	junk	−
10	perfect	++	and	N	like	++	boring	− −	disjointed	N
11	performances	N	enjoyable	++	again	N	bad	− −	substandard	−
12	finest	N	compelling	++	than	N	neither	N	overproduced	N
13	most	0^-	effective	+	though	0^+	lack	− −	stupid	− −
14	exquisitely	++	well	+	times	N	feels	0^+	dumb	− −
15	ingenious	++	provides	N	down	−	loses	N	poorly	− −
16	beautifully	++	works	N	'	N	instead	N	excuse	−
17	great	++	appealing	++	it	N	gets	N	completely	0^-
18	greatest	++	clever	++	little	− −	ridiculous	− −	movie	N
19	performance	N	haunting	− −	to	N	gone	N	.	N
20	impeccable	++	surprisingly	0^+	line	N	less	−	no	N

(b) Movie Review

No.	Pos-Cap	Attr	Neg-Cap	Attr
1	funny	++	bad	− −
2	absorbing	N	plodding	N
3	terrific	++	falls	N
4	enjoyable	++	worst	− −
5	mesmerizing	++	terrible	− −
6	effective	+	awful	− −
7	fun	++	bland	− −
8	effectively	0^-	dull	−
9	compelling	++	predictable	0^-
10	romantic	++	isn't	N
11	exhilarating	++	suffers	N
12	enjoy	+	lousy	− −
13	delivers	N	problem	−
14	entertaining	++	off	N
15	good	+	mess	− −
16	intelligent	++	fails	N
17	rare	0^-	never	N
18	genuine	+	pretentious	− −
19	manages	N	boring	− −
20	wonderful	++	unfortunately	− −

(c) Hospital Feedback

No.	Pos-Cap	Attr	Neg-Cap	Attr
1	friendly	++	nothing	N
2	leaving	N	improved	+
3	polite	+	?	N
4	helpful	+	n	N
5	nice	++	none	N
6	liked	N	improve	+
7	quick	0^-	500	N
8	courteous	++	nil	N
9	quickly	N	parking	N
10	good	+	improving	+
11	attentative	N	improvement	+
12	professional	N	keep	N
13	clean	+	perfect	++
14	helpfull	+	above	+
15	easy	+	nothink	N
16	thank	++	everthing	N
17	pleasant	+	applicable	N
18	efficient	+	improvements	+
19	cared	N	absolutely	0^+
20	caring	N	signposts	N

needs to be minimized, and the distance between their reconstruction representations with the instance representation needs to be maximized. We show that this simple capsule model achieves state-of-the-art sentiment classification accuracy without any carefully designed instance representations or linguistic knowledge. We also show that the capsule is able to output the words best reflecting the sentiment category. The words well reflect the domain specificity of the dataset, and many words carry sentiment tendencies within

Table 5: Low frequency words attended by different capsules on the three datasets, following the same notations as in Table 4

(a) Stanford Sentiment Treebank

No.	Very-Pos-Cap	Attr	Pos-Cap	Attr	Neutral-Cap	Attr	Neg-Cap	Attr	Very-Neg-Cap	Attr
1	sweetest	N	brilliant	++	marred	N	insomnia	N	travesty	– –
2	masterpiece	++	solidly	N	snoozer	N	settles	N	dreadful	– –
3	flawless	++	illuminating	++	foul	– –	confusing	– –	unwatchable	N
4	smartest	++	saves	N	immediately	0^+	sorry	– –	hate	– –
5	finest	N	worthwhile	++	gags	N	pill	N	insulting	– –
6	remarkably	++	succeeds	N	ecks	N	dicey	N	clunker	N
7	tremendous	++	sudsy	N	melodrama	N	suffer	–	misfire	N
8	breathtakingly	++	soothing	N	brimful	N	bounces	N	mire	– –
9	incredible	++	bet	N	hmm	0^+	valid	+	unpleasant	– –
10	priceless	++	deliciously	N	no.	N	steal	–	laziest	N
11	gloriously	++	refined	+	downer	– –	painful	–	disgusting	– –
12	fabulous	++	enduring	++	wan	N	drowns	N	pathetic	– –
13	astonishing	++	oodles	N	?	N	misses	N	worst	– –
14	amazingly	++	laughing	N	disappointingly	– –	heck	– –	dull	–
15	loved	N	warmed	N	dulled	N	awkward	– –	sink	N
16	stellar	++	haunting	– –	slap	– –	ridiculous	– –	hated	N
17	wonderful	++	wildly	–	branagh	N	pandering	N	ugly	– –
18	brilliant	++	engages	N	sad	– –	failed	N	incoherent	– –
19	superlative	++	simmering	N	entries	N	regret	– –	dud	–
20	soulful	N	breathtaking	++	forgivable	N	fails	N	junk	–

(b) Movie Review

No.	Pos-Cap	Attr	Neg-Cap	Attr
1	rewarding	++	by-the-numbers	N
2	bracing	N	swill	N
3	skillful	++	heavy-handed	N
4	amaze	++	dopey	N
5	oozes	N	execrable	N
6	thrilling	++	resembles	N
7	wonderful	++	meandering	N
8	gloriously	++	snoozer	N
9	therapeutic	N	sucks	N
10	brilliant	++	generic	N
11	guaranteed	N	useless	–
12	refreshing	++	self-satisfied	N
13	savvy	++	mediocre	– –
14	breathtaking	++	boring	– –
15	beaut	N	loses	N
16	fantastic	++	undistinguished	N
17	indeed	0^+	cloying	N
18	balanced	+	unpleasant	– –
19	danang	N	cable	N
20	chops	N	pretension	N

(c) Hospital Feedback

No.	Pos-Cap	Attr	Neg-Cap	Attr
1	cleaniness	N	communicationcommunication	N
2	helpfullness	N	cleanlinessnursesreceptionist	N
3	clever	++	errrr	N
4	dischaged	N	noneconsideratedoctors	N
5	staffanything	N	noshing	N
6	decorated	N	nothingcleanlinessprofessionalism	N
7	surviving	N	hampshire	N
8	treatedanything	N	fare	N
9	promising	++	significance	+
10	comunity	N	anythging	N
11	sufficiantly	N	improoved	N
12	informal	N	nothingradiologyx	N
13	congratulation	N	workable	++
14	opointments	N	postin	N
15	brillaint	N	gutted	N
16	attentative	N	5/5	N
17	cmfortable	N	signposts	N
18	attentiveas	N	qs	N
19	fullanything	N	nothingexcept	N
20	natured	N	noithing	N

the context defined by the data. Such words are not included in any sentiment lexicon, but these words become extremely useful in real applications for decision makers to further understand the quality of their products and services.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation of China under grant No. 61272227/61332007, China Scholarship Council, and Singapore Ministry of Education Research Fund MOE2014-T2-2-066.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR* abs/1409.0473 (2014).

[2] Yanqing Chen and Steven Skiena. 2014. Building Sentiment Lexicons for All Major Languages. In *Proc. ACL, Volume 2: Short Papers*. 383–389.

[3] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proc. EMNLP*. 1724–1734.

[4] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification. In *Proc. ACL, Volume 2: Short Papers*. 49–54.

[5] Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proc. EMNLP*. 1615–1625.

[6] Andrew B Goldberg and Xiaojin Zhu. 2006. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In *Proc. Workshop on Graph Based Methods for Natural Language Processing*. Association for Computational Linguistics, 45–52.

[7] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. 2016. *Deep Learning*. MIT Press.

[8] Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An Unsupervised Neural Attention Model for Aspect Extraction. In *Proc. ACL, Volume 1: Long Papers*. 388–397.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[10] Minqing Hu and Bing Liu. 2004. Mining Opinion Features in Customer Reviews. In *Proc. National Conference on Artificial Intelligence, Conference on Innovative Applications of Artificial Intelligence*. 755–760.

[11] Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan L. Boyd-Graber, and Hal Daumé III. 2016. Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships. In *Proc. NAACL HLT*. 1534–1544.

[12] Mohit Iyyer, Varun Manjunatha, Jordan L. Boyd-Graber, and Hal Daumé III. 2015. Deep Unordered Composition Rivals Syntactic Methods for Text Classification. In *Proc. ACL, Volume 1: Long Papers*. 1681–1691.

[13] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proc. ACL, Volume 1: Long Papers*. 655–665.

[14] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. EMNLP*. 1746–1751.

[15] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[16] Dan Klein and Christopher D. Manning. 2003. Accurate Unlexicalized Parsing. In *Proc. ACL*. 423–430.

[17] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proc. ICML*. II–1188–II–1196.

[18] Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2015. Molding CNNs for text: non-linear, non-consecutive convolutions. In *Proc. EMNLP*. 1565–1575.

[19] Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2016. Rationalizing Neural Predictions. In *Proc. EMNLP*. 107–117.

[20] Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

[21] Tomáš Mikolov. 2012. Statistical language models based on neural networks. *Presentation at Google, Mountain View, 2nd April* (2012).

[22] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proc. INTERSPEECH*. 1045–1048.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Proc. NIPS*. 3111–3119.

[24] Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *Proc. Workshop on Semantic Evaluation, SemEval@NAACL-HLT*. 321–327.

[25] Bo Pang and Lillian Lee. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proc. ACL*. 115–124.

[26] Bo Pang and Lillian Lee. 2007. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval* 2, 1-2 (2007), 1–135.

[27] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proc. EMNLP*. 1532–1543.

[28] Qiao Qian, Minlie Huang, JinHao Lei, and Xiaoyan Zhu. 2017. Linguistically Regularized LSTMs for Sentiment Classification. In *Proc. ACL*, Vol. 1. 1679–1689.

[29] Qiao Qian, Bo Tian, Minlie Huang, Yang Liu, Xuan Zhu, and Xiaoyan Zhu. 2015. Learning Tag Embeddings and Tag-specific Composition Functions in Recursive Neural Network. In *Proc. ACL, Volume 1: Long Papers*. 1365–1374.

[30] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic Routing Between Capsules. In *Proc. NIPS*. 3859–3869.

[31] Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013. Parsing with Compositional Vector Grammars. In *Proc. ACL, Volume 1: Long Papers*. 455–465.

[32] Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. 2014. Grounded Compositional Semantics for Finding and Describing Images with Sentences. *TACL* 2 (2014), 207–218.

[33] Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *Proc. EMNLP*. 151–161.

[34] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. EMNLP*, Vol. 1631. Citeseer, 1642.

[35] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proc. ACL, Volume 1: Long Papers*. 1556–1566.

[36] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document Modeling with Gated Recurrent Neural Network for Sentiment Classification. In *Proc. EMNLP*. 1422–1432.

[37] Zhiyang Teng, Duy-Tin Vo, and Yue Zhang. 2016. Context-Sensitive Lexicon Features for Neural Sentiment Analysis. In *Proc. EMNLP*. 1629–1638.

[38] Peter D. Turney. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proc. ACL*. 417–424.

[39] Duy-Tin Vo and Yue Zhang. 2016. Don't Count, Predict! An Automatic Approach to Learning Sentiment Lexicons for Short Text. In *Proc. ACL, Volume 2: Short Papers*. 219–224.

[40] Yequan Wang, Minlie Huang, Li Zhao, and Xiaoyan Zhu. 2016. Attention-based LSTM for Aspect-level Sentiment Classification. In *Proc. EMNLP*. 606–615.

[41] Wen-Li Wei, Chung-Hsien Wu, and Jen-Chun Lin. 2011. A Regression Approach to Affective Rating of Chinese Words from ANEW. In *Proc. Conference on Affective Computing and Intelligent Interaction, Part II*. 121–131.

[42] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. WSABIE: Scaling Up to Large Vocabulary Image Annotation. In *Proc. IJCAI*. 2764–2770.

[43] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In *Proc. HLT EMNLP*. 347–354.

[44] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. 2016. Hierarchical Attention Networks for Document Classification. In *Proc. NAACL HLT*. 1480–1489.