Sentiment Strength Prediction Using Auxiliary Features

Huijun Chen*
School of Data and Computer Science,
Sun Yat-sen University,
Guangzhou, China
chenhj39@gmail.com

Xin Li*
School of Data and Computer Science,
Sun Yat-sen University,
Guangzhou, China
lixin4ever@gmail.com

Yanghui Rao†
School of Data and Computer Science,
Sun Yat-sen University,
Guangzhou, China
raoyangh@mail.sysu.edu.cn

Haoran Xie
Department of Mathematics and Information Technology,
The Education University of Hong Kong,
Tai Po, Hong Kong SAR, China
hrxie2@gmail.com

Fu Lee Wang
Office of the President,
Caritas Institute of Higher Education,
Tseung Kwan O, Hong Kong SAR, China
pwang@cihe.edu.hk

Tak-Lam Wong
Department of Mathematics and Information Technology,
The Education University of Hong Kong,
Tai Po, Hong Kong SAR, China
tlwong@eduhk.hk

ABSTRACT

With an increasingly large amount of sentimental information embedded in online documents, sentiment analysis is quite valuable to product recommendation, opinion summarization, and so forth. Different from most works on identifying documents' qualitative affective information, this research focuses on the measurement of users' intensity over each sentimental category. Affect indicates positive or negative sentiment, while cognition includes certainty and tentative. Thus, our research can help bridge the cognitive and affective gaps between users and documents. The contributions of this study are twofold: (i) we proposed a neural network-based framework to sentiment strength prediction by convolving hybrid vectors, and (ii) we considered words jointly with a set of linguistic features for enhancing model robustness and adaptiveness. By exploiting the auxiliary features of sentiments from the corpus, the proposed model did not rely on well-established lexicons, and showed its robustness over sparse words. Experiments on six corpora validated the effectiveness of our sentiment strength prediction method.

Keywords

Sentiment strength; convolutional neural network; hybrid features

1. INTRODUCTION

With the development of web 2.0 technology, many users express their sentiments and opinions through reviews, blogs, news articles, and tweets. Sentiment analysis, also called "opinion mining", is the field of study that identify users' opinions, sentiments, appraisals, attitudes, and emotions towards subjects [20]. The early studies of sentiment analysis [26] used supervised learning algorithms to classify the polarity of reviews. These algorithms aim to identify the sentiment of the text over positive, negative, and neutral categories primarily, which are applicable to coarse-grained sentiment indicator prediction tasks. However, coarse-grained sentiment classification is limited since it captures only the most dominant sentiment. Besides, it does not consider the sentimental intensity which is an important feature for the cognitive computing and decision-making of potential customers. Affect and cognition are two important psychological conditions, in which positive or negative sentiment is regarded as "affect", while "cognition" includes certainty, tentative, and other dimensions [31]. Considering two product reviews, one with the word "excellent" and the other containing the word "good" as an example, traditional sentiment classification algorithms will assign the same label to these two documents since both of them convey positive meaning, in which cognition indicators (e.g., the confidence of each sentiment) are ignored. In many cases, it is not sentiment polarity but sentiment intensity that reflects the quality of products and desire of users, which indicates that sentiment classification may not be sufficient for real-world applications.

To tackle the aforementioned issue, sentiment strength prediction is introduced to predict the sentimental intensity of documents. Intuitively, since real-valued sentimental intensity is more natural and precise in representing the abstract and fuzzy cognition of users than qualitative labels (e.g., positive or negative category), sentiment strength prediction is more powerful and meaningful than sentiment classification tasks. Early works on sentiment strength prediction have focused mainly on exploiting lexical features. For instance, Thelwall et al. [38, 37] developed an algorithm named SentiStrength, which leveraged machine learning models to explore the sentiment strengths of terms, and then calculated the intensity of sentiments by the constructed lookup table of term strengths. However, such lexicon-based models are heavily dependent on certain key words, which perform limited since sentiments of words are sensitive to the topic domain or even aspect [39]. For example, "large" is negative when used in regards to a battery, but is positive for screen size. Recently, convolutional neural networks (CNNs) have become popular in sentiment analysis [17]. CNNs were initially used in image recognition, but are also valuable for extracting internal and fined-grained information from documents [15]. Input of CNNs in documents is the textual "image", where the height is

*indicates equal contributions.
†The corresponding author.

WWW'17 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054149

the number of words and the width is the dimension of word embeddings. Then, convolutional kernels that are fine-tuned during training perform 1-dimensional or 2-dimensional convolution over the textual "image" and produce feature maps that hold the semantic information of documents. With respect to sentiment analysis, a character-to-sentence CNN (CharSCNN) was proposed to classify sentiments over documents in one context [8]. The model first used two convolutional layers to extract both word and character embeddings. Then, the output of convolutional layers was processed by two successive fully connected layers. CharSCNN is not be affected by the quality of sentiment dictionaries, but it has been reported that the character-level CNNs perform well only when the dataset goes to a scale of several millions, and the selection of alphabet has a great impact on system performance [43]. Besides, since these window-based models heavily rely on the rich context (i.e., word sequence within a window) generated by window sliding, word-level information becomes too limited when input words are sparse.

In light of these considerations, we proposed a hybrid CNN model (HCNN) to predict sentiment strengths based on semantic and syntactic information that hold contextual features. Different from previous studies [8, 17], HCNN aimed to maximize the similarity rather than likelihood between two output vectors. Firstly, HCNN obtained semantic representations of input documents via convolving word vectors within the window. To enhance the representation learning of corpus-specific information, we added the one-hot vector of each word on top of the general real-valued vector (e.g., Word2Vec). To adapt short documents containing few words, HCNN also generated more abstract features by clustering similar word usages and grammatical roles, and leveraged auxiliary information to discover the hidden linguistic relation between words. Specifically, we grouped words by their part-of-speech (POS) tags. We assumed that words in the same "POS-cluster" were similar and shared some important features. For example, "good" and "bad", which lie in the "adjective-cluster", are both potential sentiment indicators of the document. Furthermore, the intensity of sentiments could be implied by adverbs such as "quite" and "extremely". To model the internal knowledge of a cluster, HCNN borrowed the mechanism of convolution and distilled the POS-level features by convolutional filters. Secondly, the concatenated word-level and POS-level features were fed into a fully connected layer to generate the hybrid representation of the text. Finally, a fully connected layer with softmax used the representation as input, and predicted the sentiment strength. It is worth noting that both the one-hot word vector and the POS information were critical to identifying sentiments. As mentioned earlier, the "adjective-cluster" is valuable for separating sentimental words like "good" and "bad" from other features; meanwhile, word vectors are important for detecting the opposite sentiment polarity of these two words.

To the best of our knowledge, this is the first time a CNN-based framework has exploited sentiment-specific features by the "POS-cluster" to overcome the lack of context when applying CNNs on short messages. Our method also took advantage of corpus-specific and domain-independent features (i.e., one-hot vectors and pre-trained word embeddings), rather than well-established lexicons for sentiment strength prediction.

2. RELATED WORK

2.1 Sentiment Analysis

Sentiment analysis, which was proposed primarily to identify the coarse-grained sentimental category (i.e., positive, negative, or neutral) of given documents, has attracted much attention recently due to its significant applications in both industry and academia [20]. Most existing methods of sentiment analysis can be categorized into machine learning algorithms, lexicon-based methods, and labeled topic models [22].

The machine learning based algorithms, such as naïve Bayes, support vector machines, and maximum entropy, were aimed to learn classifiers by training data [26]. They tried to extract and combine different features for improving the performance of sentiment analysis, such as n-gram features [18, 17], POS tags [10], and user information [36]. The lexicon-based methods, such as SentiWordNet that extends WordNet, associated each word or phrase with a specific sentiment to obtain sentiment lexicons [2]. The SentiWordNet is a lexical resource developed for supporting sentiment annotation, classification, and other tasks. This resources labels each synset in the WordNet along three sentimental dimensions: positivity, negativity, and neutrality. Another stream of work is focused on exploring sentiments of latent topics. Latent topics represent real-world events, objects, or abstract entities that indicate the subject or context of the sentiment [35]. For instance, Lin and He [19] proposed the joint sentiment-topic model (JSTM) for sentiment analysis of the movie review corpus. JSTM incorporated supervision by constraining the model to use only those topics that correspond to a document's observed label set.

2.2 Sentiment Strength Prediction

Sentiment can be assessed for the strength with which a positive or negative sentiment is expressed [37]. Different from the aforementioned coarse-grained sentiment classification tasks, sentiment strength prediction aims to measure the fine-grained strength of sentiments in given documents.

In a preliminary study, SentiStrength [38] was designed to extract sentiment strengths from comments on MySpace. It firstly established a lexicon in which sentimental words were associated with strength measures and exploited a set of nonstandard sentiment expressing rules. Several methods based on the lexicon and rules were then employed to score the sentiment strength. As an improved version of SentiStrength, Thelwall et al. [37] assessed that the lexicon that mainly contained direct sentimental indications was also effective at predicting the sentiment strength across the social web. However, the limitation of these lexicon-based methods is that they are quite dependent on certain keywords.

2.3 Convolutional Neural Networks

With the emergence of neural language models [3] and distributed vector representations for words [27, 24], CNNs have been widely used in sentiment analysis. Kim [17] designed a CNN that captured n-gram information by convolving neighboring word vectors in a sentence matrix and used multiple kernels to learn meaningful features. Deep CNN [34] is similar to the convolutional architecture proposed by Kim [17], and contains a single convolutional layer followed by a non-linearity, max pooling, and softmax classification layer. Ren et al. [29] proposed a context-based CNN for Twitter sentiment classification, which combined contextual features of current and relevant tweets when producing the semantic representation. Unfortunately, such a method is designed for corpora containing a conversation-based context. Unlike the above works, Johnson et al. [14] proposed a convolutional model that did not use pre-trained word vectors but high-dimensional textual data with word order information as the network input. Johnson et al. [15] also introduced a new method that learned embeddings of small text regions from unlabeled data, and constructed a semi-supervised framework with CNNs. However, both were concerned with sentiment classification, which is not applicable to predicting sentiment strengths.

With respect to CNN models for fine-grained sentiment analysis tasks, dos Santos et al. [8] adopted two levels of convolution mechanisms to incorporate morphological and shape information into feature learning. The proposed CharSCNN used two convolutional layers where the first layer learned character-level embeddings and the second layer generated sentence/document representations by convolving character-level and word-level embeddings. Unfortunately, this method is dependent on the context of datasets and shape of characters. For instance, shape and morphological information may introduce some noise since the gap between datasets from different sources is quite large.

3. SENTIMENT STRENGTH PREDICTION VIA CONVOLVING HYBRID FEATURES

3.1 Problem Definition

Sentiment strength prediction, the problem we investigate in this paper, differs from the traditional sentiment classification in which the document sentiment is encoded as a one-hot polarity vector. Our task treats overall document sentiments as a list of real values ranging from 0 to 1, and each value denotes the intensity of the corresponding sentiment (so-called sentiment strength). The main goal of the sentiment strength prediction system is to predict a strength vector that can reflect multiple sentimental orientations of the document.

Assume that we have a document d with strength vector [0.5, 0.3, 0.2], there are three sentimental classes s_1, s_2, and s_3, and we know that the polarity (i.e., dominant sentimental orientation) of d is s_1 while d also expresses some meaning on s_2. Additionally, d or part of d (words, phrases, or clauses) arouses sentiment s_3. A well-performed sentiment strength prediction system should detect the correct sentiment polarity and also give good predictions on those sentiments not being dominant. In our case, we employed word-level and POS-level convolution.

For convenience of describing the model, we used superscripts w and pos to specify the related notations in the corresponding convolutional layer. Given a vocabulary V and a sequence of input instances $\{I_1^x, ..., I_m^x\}$ ($x = w$ or other POS tags, m denotes length of input), we firstly looked up the corresponding vectors in the feature embeddings $M \in \mathbb{R}^{|V| \times dim_w}$, and constructed the input embedding matrix $E_x \in \mathbb{R}^{m \times dim_w}$ of the input layer. In the above, I_i^x represents the index of the i-th word of input sentences in vocabulary V and dim_w is the dimension of word embeddings. Convolutional kernel K_x with fixed-sized window win_{K_x} performs vector-level convolution by sliding the window over the embedding matrix, where the weight matrix of the kernel is denoted as $W_{K_x} \in \mathbb{R}^{n_{K_x} \times (win_{K_x} \cdot dim_w)}$, that is, the kernel captures internal relations of win_{K_x} consecutive instances by convolving win_{K_x} corresponding low-dimensional vectors. After convolution, we performed max-over-time pooling [7] over the generated feature maps, which produced hybrid feature vectors with a fixed size, i.e., $m - win_{K_x} + 1$, thus solving the problem of variable lengths of input instances. We summarize the frequently used terms and their descriptions in Table 1.

3.2 Network Architecture

In this section, we detail the architecture of the proposed HCNN, which aimed to predict sentiment strength of a given document or sentence with hybrid features. It firstly generated sentence vectors, and then jointly learned representation vectors of POS-tags according to words' POS annotations. The architecture of HCNN is shown in Figure 1.

Table 1: Notations of frequently used terms.

Notation	Description
V	Vocabulary of words occurring in the dataset
V_{inf}	Vocabulary of informative words in the dataset
M	Lookup table of word embeddings
W_{K_x}	Weight matrix of the convolutional kernel K_x
b_{K_x}	Bias vector of the convolutional kernel K_x
win_{K_x}	Window size of the convolutional kernel K_x
n_{cls}	Number of sentiment classes
E_x	Input feature matrix of the x-th convolutional layer
C_x	Feature map from the x-th convolutional layer
n_{K_x}	Dimension of feature map C_x
dim_w	Dimension of word embeddings
dim_{pre}	Dimension of pre-trained word embeddings
dim_{hyb}	Dimension of hybrid feature vectors

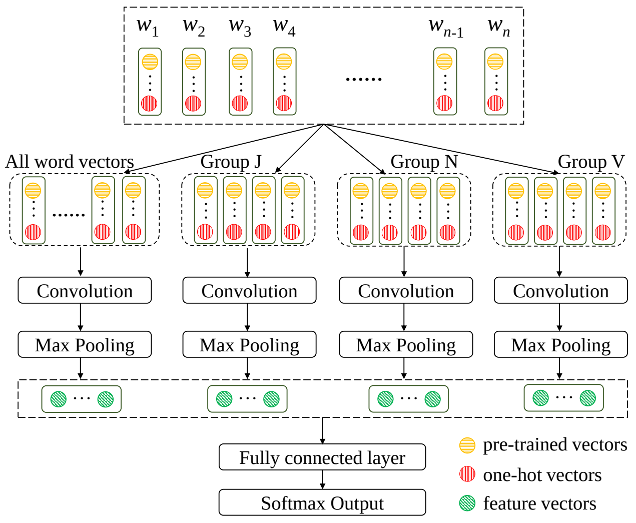


Figure 1: Architecture of the proposed HCNN.

With respect to learning the semantic representations, we extended each word to a vector with a fixed length. As a traditional method, one-hot representation is simple and understandable, but suffers from the curse of dimensionality [23]. One common solution is to select a small subset of features in the dataset. We adopted the document frequency thresholding method, which assumes that rare terms are either non-informative for prediction or not influential for global performance [41], to perform dimension reduction. One-hot representation is also limited since it cannot record the sentimental meaning and relevance between different words [5]. Thus, we introduced real-valued word embeddings trained from large-scale corpora [27, 24] to tackle this issue. As mentioned in previous document classification systems [17, 16], convolution is a powerful technique in learning semantic representation due to its property of encoding contextual information automatically. Thus, we fed the concatenation of corpus-specific information (i.e., one-hot vectors) and domain-independent word embeddings (i.e., pre-trained vectors) into a convolutional architecture to obtain feature vectors that were more meaningful at the semantic level. First, we constructed the embedding vector of each word instance I^w. For the corpus-specific part, we selected the most informative words by filtering words with low document frequency [41] and removing stop words, and produced one-hot vectors based on these informative words. For other parts that were independent to the corpus, we directly employed pre-trained word vectors. Then, we concatenated these two parts as follows:

$$M[I^w] = R_{oh}[I^w] \oplus R_{pre}[I^w], \tag{1}$$

where R_{oh} and R_{pre} are the one-hot and pre-trained representation vectors of each word, respectively; $\oplus$ denotes the vector concatenation operator; and the dimension of word embeddings $dim_w = dim_{pre} + |V_{inf}|$. Given that input document d contains n word instances $\{I_1^w, ..., I_n^w\}$, we look up the vectors for these words in the document as follows:

$$r_i^w = M[I_i^w] \quad (i = 1, ..., n), \tag{2}$$

where r_i^w is the representation of the i-th input word. Then, a document matrix $E_w \in \mathbb{R}^{n \times dim_w}$ is constructed vector by vector and fed into the word-level convolutional layer, as follows:

$$\begin{aligned} C_w[j] = W_{K_w} \odot (E_w[j] \oplus ... \\ \oplus E_w[j + win_{K_w} - 1]) + b_{K_w}, \end{aligned} \tag{3}$$

where a word-level kernel weight matrix W_{K_w} is used to extract contextual features from win_{K_w} neighboring word vectors with length dim_w, and feature maps with the same dimensionality, i.e., n_{K_w}, are produced. In this paper, $\odot$ represents the matrix multiplication. We also note that the number of semantic features maps for different documents are not equal, and therefore we performed max-over-time pooling, which preserved only the most important features. The process of obtaining semantic representation f_{sem} is given below:

$$f_{sem} = h_r(max(C_w[:, 1]) \oplus ... \oplus max(C_w[:, n_{K_w}])), \tag{4}$$

where $h_r(*)$ is the ReLU activation function at this layer, and $C_w[:, i]$ can be regarded as a product of the i-th 2-dimensional spatial convolution over the sentence matrix E_w, i.e., the convolution process here is equivalent to employing n_{K_w} kernels to encode semantic information separately. Thus, the pooling is actually a process that preserves the most active features from the generated n_{K_w} feature vectors (i.e., $C_w[:, 1], ..., C_w[:, n_{K_w}]$), and drops the others.

As with syntactic-level features, previous works [10, 30] observed that adjectives, verbs, and nouns have high correlation with sentiment polarity. However, it is challenging to associate POS information with sentiment strengths, that is, how to use POS information to bridge words with both sentiments and intensities. Our solution is generating sentimental features from the "POS-cluster". A "POS-cluster" is actually a word cluster that consists of words with the same POS annotation, and is similar to a topic that contains semantically related words via topic modeling [4]. However, the "POS-cluster" is different from topics in that a "POS-cluster" is not corpus-specific or domain-dependent. For example, "large" in different corpora may belong to different topics, but it belongs to the same "POS-cluster" (i.e., "ADJ-cluster") for every corpus. From this view, exploiting those clusters to enrich context-independent sentiment information is useful in sentiment strength prediction.

In our work, we employed the existing POS parser to annotate documents and categorize the generated tags into four groups according to [40]. Group information of POS tags is illustrated in Table 2. Since cluster "O" contains many background words that characterize non-discriminative information and may bring some noise, we preserved clusters "J", "N", and "V" for each document. Note that non-informative nouns and verbs were filtered by the document frequency thresholding method [41], and all adjectives

Table 2: Group information of POS tags.

Group	POS tags
J	Adjectives, Adverbs
N	Nouns
V	Verbs
O	Other POS tags

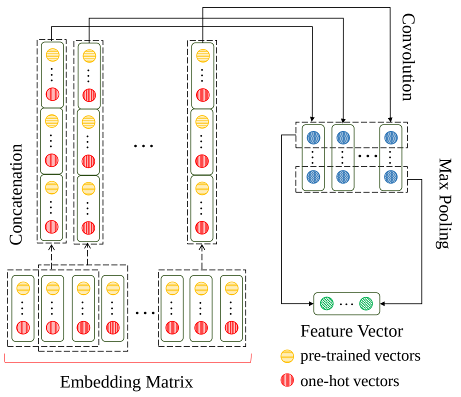


Figure 2: Convolution and pooling operation.

and adverbs were used for sentiment strength prediction. After constructing "POS clusters", we associated their words with sentiments. Since vector space models suffer from the curse of dimensionality and fail to capture synonyms and polysemous words [12], traditional methods, like the naïve Bayes [21] method, are limited. This motivated us to explore neural models based on low-dimensional dense representations. In this article, we extended a single word in the cluster to a vector that combined dense representation and "thin" one-hot representation. We also leveraged the convolutional kernel to distill sentiment information from word-level semantic representations for the "POS cluster". Similar to Eq. 2, we firstly looked up word vectors from M and built an input embedding matrix for each cluster. For instance, given an input text "I like my new book" and the corresponding POS tag sequence {O, V, O, J, N}, we can construct three embedding matrixes: E_J=[r_4^w], E_N=[r_5^w], and E_V=[r_2^w]. Then, we learned POS-level representations of the document via the following convolution (ref., Figure 2):

$$\begin{aligned} C_{pos}[j] = W_{K_{pos}} \odot (E_{pos}[j] \oplus ... \oplus E_{pos}[j+ \\ win_{K_{pos}} - 1]) + b_{K_{pos}}, (pos = J, N, V), \end{aligned} \tag{5}$$

$$f_{pos} = max(C_{pos}[:, 1]) \oplus ... \oplus max(C_{pos}[:, n_{K_{pos}}]), \tag{6}$$

where $W_{K_{pos}}$ is the parameter matrix of the convolutional layer used to extract the sentiment features of "POS cluster" ($pos = J, N, V$), and $n_{K_{pos}}$ is the number of kernels used in this convolutional layer. Since we aimed to encode cluster vectors as sentiment information, the produced three feature vectors f_J, f_N, and f_V for the corresponding "POS cluster" had the same size of $n_{K_{pos}}$, i.e., $n_{cls} = n_{K_{pos}}$.

We then concatenated semantic representation f_{sem} and POS-level sentiment features, and fed the vector into a fully-connected layer. Output was the hybrid feature representation of document d, i.e.,

$$f_{all} = f_{sem} \oplus f_J \oplus f_N \oplus f_V, \tag{7}$$

$$f_d = h_r(W_{fully} \odot f_{all} + b_{fully}), \tag{8}$$

where $W_{fully} \in \mathbb{R}^{dim_{hyb} \times (n_{K_w} + 3 \cdot n_{K_{pos}})}$ and $b_{fully} \in \mathbb{R}^{dim_{hyb}}$ are weight parameters of the fully-connected layer, and dim_{hyb} is the number of neural units in the fully-connected layer, i.e., the dimension of hybrid feature vectors. The ReLU activation function $h_r(*)$ was added on the output of convolution. To reduce the impact of overfitting, we also added dropout in the network, that is, we randomly left some nodes unused in the training process, as follows:

$$\hat{f}_d = f_d \otimes mask, \tag{9}$$

where $\hat{f}_d$ is the feature vector of document d after dropout, $\otimes$ is the element-wise multiplication operator, and $mask$ is a vector with binary values. Particularly, each element in $mask$ follows binomial distribution with probability p, which is a hyper parameter in our model. Note that we used dropout during training to alleviate the issue of overfitting, and chose to remove dropout, i.e., $\hat{f}_d = f_d$, for each testing document to forward more information to the next layer.

The last component of our network was the prediction module, which was a softmax layer. The sentiment strength of document d was calculated as follows:

$$SV[d] = h_s(W_{soft} \odot \hat{f}_d + b_{soft}), \tag{10}$$

where $h_s(*)$ is the softmax activation function, and W_{soft} and b_{soft} denote the parameter matrix and bias vector, respectively. System output $SV[d]$ is the predicted strength vector of document d, and its value on each dimension is the predicted intensity of the corresponding sentiment.

3.3 Parameter Estimation

The parameter set of our model was $\theta = [W_{K_w}; W_{K_{pos}}; W_{fully}; W_{soft}; b_{K_w}; b_{K_{pos}}; b_{fully}; b_{soft}]$. As a frequently used loss function in classification-oriented works [17, 16, 32], negative log-likelihood was leveraged to maximize the likelihood between the true label and predict label, which is a little bit different from the objective of sentiment strength prediction. The output strengths of a good sentiment strength prediction system should be as close as possible to golden ones over dominant sentiment and non-dominant sentiments. Therefore, we designed a regression-oriented objective function to minimize the distance between predicted and actual strengths. In this work, we used Kullback-Leibler divergence (KL divergence) [25], which measures the dissimilarity of two probability distributions as a training objective. The training loss is defined in the following equations:

$$\begin{aligned} Loss(d) = &\sum_{i=1}^{n_{cls}} SV_{gold}[d,i] \cdot log(SV_{gold}[d,i]) \\ &- \sum_{i=1}^{n_{cls}} SV_{gold}[d,i] \cdot log(SV[d,i]), \end{aligned} \tag{11}$$

$$Loss(\theta) = \sum_{d \in D} Loss(d), \tag{12}$$

where SV_{gold} and SV_d are golden and predicted strength vectors, respectively, D denotes a collection of training documents, $Loss(d)$ is the prediction loss of training document d, $Loss(\theta)$ is the total loss of D, and n_{cls} is the number of pre-defined sentiments. Model training was done via back-propagation and stochastic gradient descent. To avoid manually tuning learning rate for different datasets, we adopted $Adadelta$ [42] as an optimizer.

4. EXPERIMENT

In this section, we evaluate the performance of our framework for sentiment strength prediction over textual data.

4.1 Dataset

To verify the adaptiveness, effectiveness, and robustness of our model on sentiment strength prediction over sparse words, a real-world short corpus was employed in our experiments[1]. The corpus was also used to evaluate the performance of existing sentiment strength detection methods [37]. This dataset included BBC Forum posts (BBC), Digg.com posts (Digg), MySpace comments (MySpace), Runners World forum posts (Runners World), Twitter posts (Twitter), and YouTube comments (YouTube). The above data sources represented different types of social environments, i.e, news-related discussion, comments on new stories, messages between friends, common-interest group messages, microblog broadcasts, and comments on videos, respectively. Each document was manually labelled by users with positive and negative sentiment strength. The positive sentiment strength ranged from 1 (not positive) to 5 (extremely positive), and the negative sentiment strength ranged from -1 (not negative) to -5 (extremely negative). To simplify evaluation [36], we concatenated the positive strength and absolute value of negative strength and treated the normalized strength vector as the actual strength vector of documents. For instance, the strength vector of a document with 1 and -3 as the positive and negative strengths was [0.25, 0.75].

Table 3: Dataset statistics.

Dataset	# of documents	Mean words
BBC	1000	64.76
Digg	1077	33.63
MySpace	1041	19.76
Runners World	1046	64.25
Twitter	4242	16.81
YouTube	3407	17.38

The dataset statistics are summarized in Table 3, where the second column presents the number of documents in each subset, and the third one shows the mean number of words in every document for each subset. Note that punctuation marks and stop words were removed to produce informative features.

4.2 Experimental Design

To evaluate the performance of the proposed HCNN, we implemented the following baselines for comparison:

- Character to Sentence Convolutional Neural Network (CharSCNN) [8]. Two convolutional layers are employed to extract features from character to sentence. The result of the second convolutional layer is then passed to two fully-connected layers to compute the sentiment score for each label. Due to the sparseness of features, we set the context windows of word and character to 1. The dimension of convolution units was 20 for the character-level and 150 for the word-level. The dimension of the character embeddings was 5, while the number of hidden units was 200. We used 0.005 as the learning rate for fine-grained training.

- Convolutional Neural Network (CNN) [17, 34]. A straightforward convolutional architecture that employs one convolutional layer with multiple kernels to learn sentence representation and add dropout to prevent over-fitting. Hyper parameters of the model were specified according to the literature [17]. Sentiment strength was the output distribution of the softmax layer.

[1]http://sentistrength.wlv.ac.uk/documentation/.

Table 4: Parameter setting of HCNN.

Parameter	Value
dim_{pre}	200
$(win_{K_w}, win_{K_{pos}})$	(1, 1)
$(dim_{K_w}, dim_{K_{pos}})$	(80, 2)
dim_{hyb}	100

- Long Short-Term Memory (LSTM) [11]. LSTM takes the whole corpus as a single sequence, and the mean of the whole hidden states of all words is used as the feature for prediction. Similar to the previous baseline, we treated the output of the softmax layer as the sentiment strength. Conventionally, the hidden representation dimension was set to 128.

- Convolutional Gated Recurrent Neural Network (ConvGRNN) [36]. The ConvGRNN model learns semantic representation hierarchically. Firstly, the model obtains sentence vectors by convolving pre-trained word embeddings. Secondly, the generated sentence vectors are fed into Gated RNN to produce the representation vector of each document. Hyper parameters of the model were specified according to the literature [36].

- Supervised SentiStrength (Ssth) [38, 37]. Ssth is a method specifically designed for sentiment strength detection over our employed corpus. It is a lexicon-based classifier that uses linguistic information and rules to predict sentiment strengths in short informal English text, and the supervised version tends to be more accurate than unsupervised SentiStrength and many other machine learning methods [37].

To obtain rich semantic features, we used pre-trained *GloVe* [27] word embeddings for HCNN and baseline models of CharSCNN, CNN, LSTM, and ConvGRNN. Particularly, we initialized word embeddings with *GloVe* vectors trained on 2 billion tweets with 27 billion tokens and 1.2 million words[2]. Words that do not appear in *GloVe* were initialized randomly. Although there are more recent models on sentiment or emotion detection tasks, they were used for either conversation corpora [29] or news articles [28] that should provide conversation context and sufficient words in an individual text. With respect to the POS annotation of documents, we utilized *openNLP* POS tagger[3] in the experiment. Following the method proposed by Glorot and Bengio [9], we initialized all parameters by sampling from uniform distribution ($-\sqrt{6/(r+c)}$, $\sqrt{6/(r+c)}$), where r and c are numbers of rows and columns in the parameter matrix, respectively. The document frequency threshold was fine-tuned on the validation set, and hyper parameters of the model were also set accordingly. We summarize the parameter settings of the proposed HCNN in Table 4. Parameters used in $Adadelta$ were kept the same as in the literature [42].

We employed three evaluation metrics as indicators of model performance: root mean square error (*RMSE*) [36, 13], Pearson's correlation coefficient (*Corr*) [38, 37], and accuracy (*Acc*) [8, 17, 28]. *RMSE* and *Corr* are fine-grained metrics that measure the difference between the predicted and actual strength values over all sentiments. For each sentiment strength s_i ($i = 1, ..., n_{cls}$), *RMSE* and *Corr* are respectively calculated as follows:

$$RMSE(s_i) = \sqrt{\frac{\sum_{d=1}^{|D_{test}|}(SV[d,i] - SV_{gold}[d,i])^2}{|D_{test}|}}, \quad (13)$$

[2]http://nlp.stanford.edu/software/GloVe-1.2.zip.

[3]https://opennlp.apache.org/.

$$Corr(s_i) = \frac{Cov(SV[:,i], SV_{gold}[:,i])}{\sqrt{Var(SV[:,i]) \cdot Var(SV_{gold}[:,i])}}, \quad (14)$$

where D_{test} is the collection of testing documents; $SV[:,i] \in \mathbb{R}^{|D_{test}|}$ is a concatenation vector of $SV[d,i]$ ($d = 1, ..., |D_{test}|$) as in Eq. 10, i.e., the predicted strength values over sentiment s_i for all testing documents; and *Cov* and *Var* denote the covariance and variance, respectively. The lower the model's *RMSE* value, the better its performance. With respect to *Corr*, its value ranges from -1 to 1, where 1 indicates that actual and system-produced sentiment strengths are correlated perfectly. To make an appropriate comparison with other neural network-based methods [36], we concatenated the positive strength and absolute value of negative strength to a normalized strength vector for each document. Thus, a single *RMSE* or *Corr* value was output for each case. We also used *Acc* as a coarse-grained metric to evaluate the overall classification performance. Accordingly, a predicted ranked list of sentiment labels was correct if the list's first item was identical to the actual ranked list's first item. If two sentiment labels in the actual ranked list had the same number of votes, then their positions were interchangeable. Particularly, an evaluation index $Pred_{\hat{d}}$ was applied to measure prediction quality as a binary variable, where 0 and 1 represent incorrect and correct prediction, respectively. $Pred_{\hat{d}}$ is defined as follows:

$$Pred_{\hat{d}} = \begin{cases} 1, & y_{\hat{d}} \in Y_{\hat{d}}^{gold}, \\ 0, & otherwise, \end{cases} \quad (15)$$

where $y_{\hat{d}}$ is the top-ranked predicted sentiment label, and $Y_{\hat{d}}^{gold}$ is the set of top-ranked labels derived from the actual strength vector of $\hat{d}$. Then, we computed the *Acc* value based on $Pred_{\hat{d}}$ as follows:

$$Acc = \frac{\sum_{\hat{d} \in D_{test}} Pred_{\hat{d}}}{|D_{test}|}. \quad (16)$$

A larger value of *Acc* indicates that the model is more effective in predicting the top-ranked sentiment.

The ablation experiment was conducted to respectively test the improvement of POS information, corpus-specific vectors, and pre-trained embeddings for HCNN. By discarding particular features, we obtained three models named HCNN - POS, HCNN - one-hot, and HCNN - *GloVe* for comparison, where "-" means that the specific feature was eliminated. Both single-source and cross-sources testing were performed to evaluate the effectiveness of our HCNN and baseline models. For the single-source testing, models were compared on each aforementioned subset (ref., Table 3), as follows. Given instances from a same source such as BBC, we conducted single-source testing by randomly selecting 60% as training samples, 20% as validation samples, and the remaining 20% for testing. To evaluate the adaptiveness and robustness of different models, we also conducted cross-sources testing by using one subset for training and others for testing. For instance, given a Twitter subset as the training set, we randomly selected 20% samples in the other subset *A* (i.e., BBC, Digg, MySpace, Runners World, or YouTube) as the validation set, and the rest of *A* as the testing set.

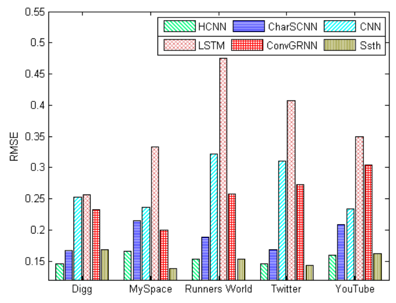
4.3 Results and Analysis

In this section, we compare the performance of different models in terms of *RMSE*, *Corr*, and *Acc*. Table 5 presents model performance of single-source testing, with the best result boldfaced.

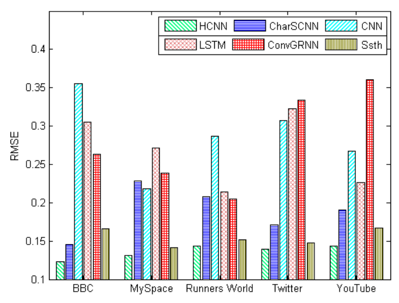
The first four rows of Table 5 present results of the ablation experiment for each subset, from which we can observe that pre-trained embeddings via *GloVe* had a positive impact on HCNN for sentiment strength prediction. Compared to HCNN - *GloVe*, the averaged performance of HCNN over the 6 subsets improved by

Table 5: Model performance of single-source testing.

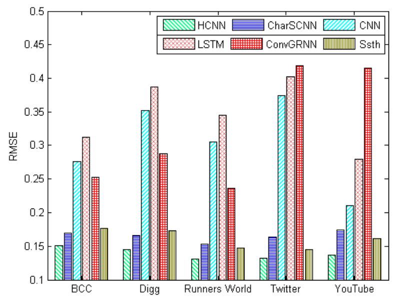
Model	BBC			Digg			MySpace			Runners World			Twitter			YouTube		
	RMSE	*Corr*	*Acc*	*RMSE*	*Corr*	*Acc*	*RMSE*	*Corr*	*Acc*	*RMSE*	*Corr*	*Acc*	*RMSE*	*Corr*	*Acc*	*RMSE*	*Corr*	*Acc*
HCNN	**0.1260**	**0.4462**	**0.9150**	**0.1297**	**0.6000**	0.8519	**0.1024**	**0.6585**	**0.9278**	**0.1133**	**0.4942**	**0.8857**	**0.1177**	**0.5625**	**0.9058**	**0.1360**	**0.7264**	**0.8942**
HCNN - POS	0.1287	0.3925	**0.9150**	0.1410	0.4928	0.8472	0.1141	0.5532	0.9183	0.1210	0.4331	0.8667	0.1189	0.5523	**0.9058**	0.1433	0.7105	0.8750
HCNN - one-hot	0.1340	0.3121	**0.9150**	0.1370	0.5200	**0.8560**	0.1141	0.5185	0.8942	0.1195	0.3824	0.8476	0.1198	0.5348	0.8939	0.1370	0.7235	**0.8942**
HCNN - *GloVe*	0.1342	0.2834	**0.9150**	0.1572	0.2798	0.7963	0.1203	0.4443	0.8942	0.1285	0.2630	0.8286	0.1310	0.4039	0.8610	0.1370	0.5562	0.8061
CharSCNN	0.1335	0.2860	0.9100	0.1505	0.3329	0.8279	0.1233	0.4291	0.8990	0.1263	0.1433	0.8278	0.1309	0.4037	0.8632	0.1586	0.6280	0.8649
CNN	0.2061	0.3172	0.8900	0.2139	0.4533	0.8380	0.3098	0.5375	0.9038	0.2967	0.4325	**0.8857**	0.3438	0.3948	0.8763	0.2432	0.6704	0.8899
LSTM	0.2943	0.2989	0.8800	0.2743	0.4501	0.8519	0.2993	0.5015	0.9038	0.3458	0.1955	0.8000	0.3112	0.3887	0.8716	0.2273	0.6866	0.8767
ConvGRNN	0.2002	0.1788	0.9100	0.1548	0.4260	0.8287	0.2745	0.5514	0.9187	0.1955	0.0178	0.8333	0.2385	0.5202	0.8928	0.2582	0.6204	0.8856
Ssth	0.1538	0.4430	0.7600	0.1641	0.4174	0.7070	0.1319	0.4762	0.8990	0.1543	0.2956	0.8373	0.1454	0.4422	0.8868	0.1613	0.6024	0.8473



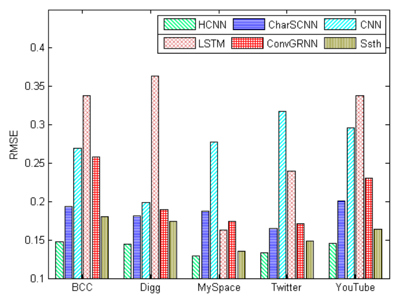
(a) BBC vs Others



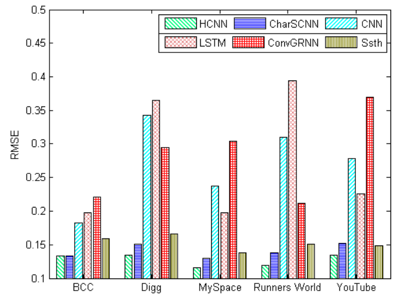
(b) Digg vs Others



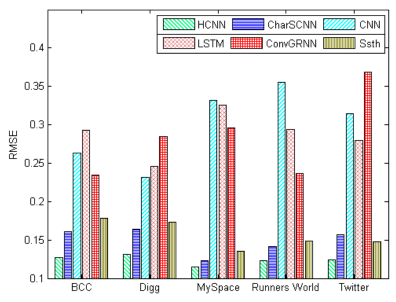
(c) MySpace vs Others



(d) Runners World vs Others



(e) Twitter vs Others

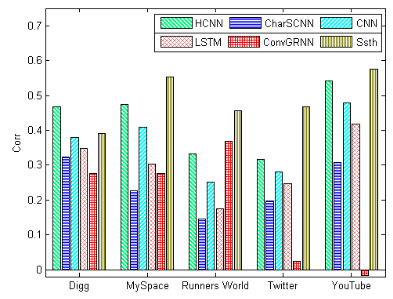


(f) YouTube vs Others

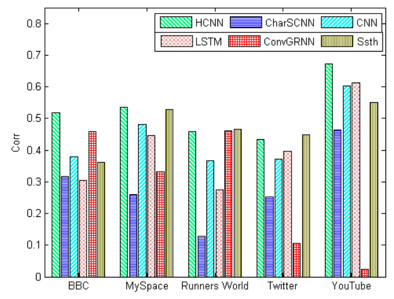
Figure 3: Model performance of cross-sources testing on *RMSE*.

10.3%, 56.4%, and 5.5% on *RMSE*, *Corr*, and *Acc*, respectively. This indicated that word embeddings trained on a large-scale external corpus were quite valuable for alleviating the sparseness of words in our employed datasets. Corpus-specific embeddings (i.e., one-hot word vectors) and POS-level features are also both useful to boost performance, especially for the fine-grained metrics *Corr* and *RMSE*. However, POS-level features have limited effect when each document's words are extremely sparse (e.g., Twitter). Compared to HCNN - POS, the performance of HCNN over Twitter only improved by 1.0% and 1.8% on *RMSE* and *Corr*, respectively. Furthermore, there was no difference between HCNN - POS and HCNN over Twitter in terms of *Acc*. A possible reason for this is that very short text cannot provide enough POS information. We also observed that POS-features, one-hot vectors, and *GloVe* did not affect *Acc* much. This was because the basic convolutional architecture was effective to capture enough features for coarse-grained polarity detection.

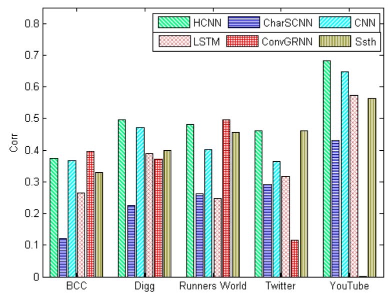
Compared to baseline methods of CharSCNN, CNN, LSTM, ConvGRNN, and Ssth, the proposed HCNN outperformed by a large margin on fine-grained metrics (i.e., *RMSE* and *Corr*), and also performed better on the coarse-grained metric (i.e., *Acc*). Besides, both HCNN and CNN achieved competitive performance over other methods on the classification evaluation metric (i.e., *Acc*), which indicated that encoding contextual information via convolution is quite useful for classification tasks. However, the difference between model performance was generally indistinctive since *Acc* did not take sentimental distributions into account. With respect to the distance-based metric *RMSE*, the proposed HCNN could generate sentiment strength distributions that were most similar to the golden distributions. Compared to the best-performing baseline of CharSCNN, the prediction error (i.e., *RMSE*) of our HCNN reduced from 6% to 17% over these 6 subsets. This indicated that the regression-oriented objective function (i.e., KL divergence) was better than classification-oriented objective functions in sentiment strength prediction. The proposed HCNN also outperformed baselines in terms of the other fine-grained metric *Corr*. For instance, our HCNN performed an average of 27% better than CNN over these 6 subsets. These results indicated that hybrid features are a benefit to sentiment strength prediction, and it is insufficient to use only pre-trained word embeddings. We also observed that CNN performed worse than CharSCNN on Twitter. A possible reason for this is that the lexical variation caused by typos, use of slang, and abbreviations, leads to a great number of singletons and out-of-vocabulary words [1]. Thus, it is difficult to estimate and update pre-trained word embeddings for these words. In this case, character-level embeddings provide some features that word-level



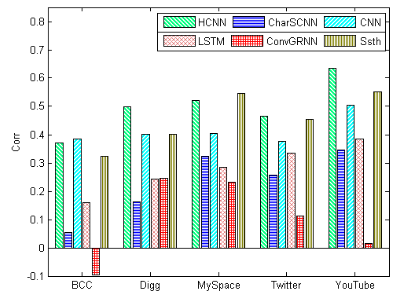
(a) BBC vs Others



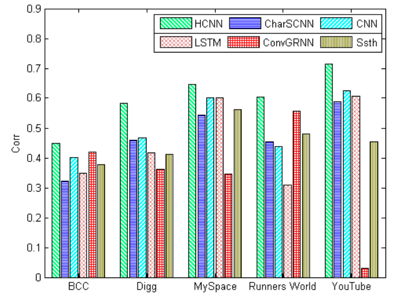
(b) Digg vs Others



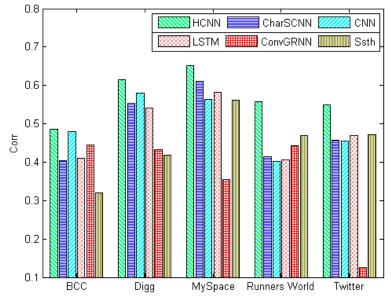
(c) MySpace vs Others



(d) Runners World vs Others

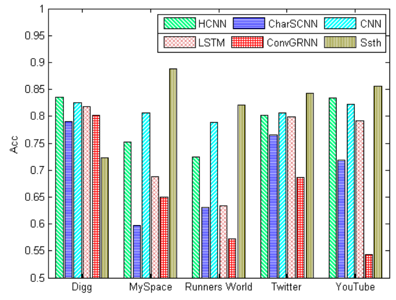


(e) Twitter vs Others

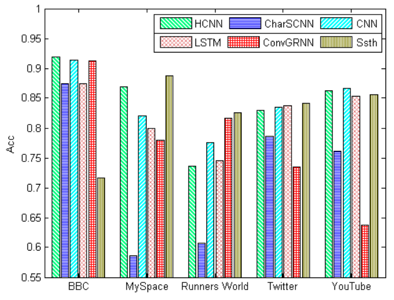


(f) YouTube vs Others

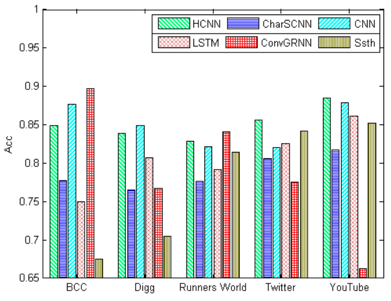
Figure 4: Model performance of cross-sources testing on *Corr*.



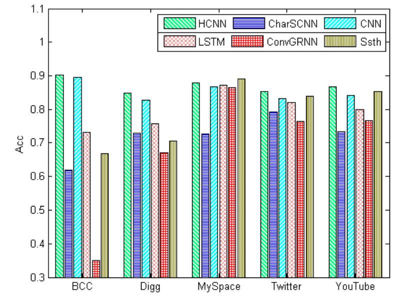
(a) BBC vs Others



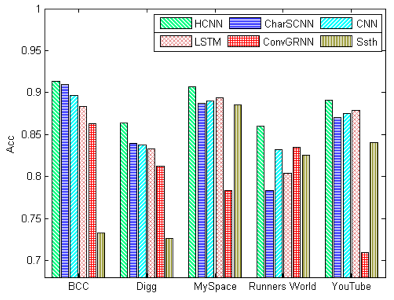
(b) Digg vs Others



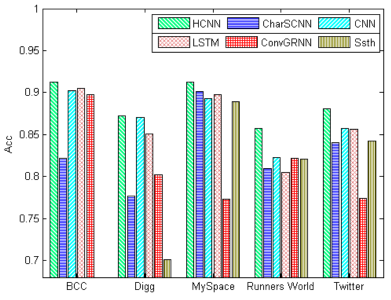
(c) MySpace vs Others



(d) Runners World vs Others



(e) Twitter vs Others



(f) YouTube vs Others

Figure 5: Model performance of cross-sources testing on *Acc*.

Table 6: The p values of statistical tests on the HCNN and baselines over *RMSE*.

Models	single-source		cross-sources	
	F-test	t-test	F-test	t-test
CharSCNN	0.376	0.029	0.000	0.000
CNN	0.002	0.001	0.000	0.000
LSTM	0.011	0.000	0.000	0.000
ConvGRNN	0.006	0.001	0.000	0.000
Ssth	0.469	0.001	0.303	0.000

embeddings do not have. However, CharSCNN performed worse than CNN on other subsets, which may have been caused by the overfitting of character-level embeddings. For the proposed HCNN model, we concatenated pre-trained word embeddings and corpus-specific one-hot vectors to consolidate both global and local semantics of words. These results of single-source testing validated the effectiveness of our method.

For cross-sources testing, we present the performance over *RMSE* in Figure 3. Compared to the best-performing baseline of Ssth, the proposed HCNN performed competitively when using BBC for training and other subsets for testing. For all other cases, our HCNN outperformed Ssth on the fine-grained metric. Ssth performed well on cross-sources testing because it was manually developed to cope with a wide variety of types of text, especially for short informal English messages [38]. The following processes were also employed by Ssth to enhance the sentiment strength detection effectiveness [37]: First, the sentiment word list was extended with negative terms with human-coded sentiment weights. Second, a supervised learning algorithm was used to optimize sentiment word strengths and potentially change polarity. Third, spelling mistakes and slang words were corrected using a dictionary. With respect to *Corr* (ref., Figure 4) and *Acc* (ref., Figure 5), the proposed HCNN performed better than CharSCNN, CNN, LSTM, ConvGRNN, and Ssth by (on average) 104.2%, 25.0%, 54.6%, 147.5%, and 12.0%, and 16.6%, 1.9%, 7.8%, 17.9%, and 6.1%, respectively. We also conducted the ablation experiment on cross-sources testing, and results indicated that POS-features and pre-trained embeddings had a larger positive impact on HCNN than one-hot vectors. For example, the prediction error (i.e., *RMSE*) of HCNN reduced by 15.7%, 2.9%, and 2.7% on average when compared with HCNN - *GloVe*, HCNN - POS, and HCNN - one-hot, respectively.

To evaluate the differences in model performance, we performed two statistical tests on the HCNN and each baseline model in terms of *RMSE*, *Corr*, and *Acc*, and the corresponding p values are shown in Table 6, Table 7, and Table 8, respectively. The first evaluated the stability of performance in terms of variances, and the second compared performance in terms of means. We used the conventional significance level (i.e., p-value) of 0.05. First, we employed the analysis of variance named F-test to test the underlying assumption of homoscedasticity (i.e., the homogeneity of variance). The result showed that the performance of HCNN was significantly more stable than most baselines on cross-sources testing, and the difference in variances between the HCNN and each baseline model was more significant on cross-sources testing than that of single-source testing. Second, we conducted t-tests to test the underlying assumption that the difference in performance between paired models had a mean value of zero. The results indicated that the proposed HCNN outperformed the baseline methods on cross-sources testing significantly, except for CNN over the coarse-grained metric (i.e., *Acc*). For single-source testing, the difference in performance between our HCNN and each baseline model was more significant

Table 7: The p values of statistical tests on the HCNN and baselines over *Corr*.

Models	single-source		cross-sources	
	F-test	t-test	F-test	t-test
CharSCNN	0.174	0.011	0.034	0.000
CNN	0.358	0.057	0.399	0.003
LSTM	0.150	0.038	0.120	0.000
ConvGRNN	0.047	0.053	0.001	0.000
Ssth	0.461	0.022	0.058	0.007

Table 8: The p values of statistical tests on the HCNN and baselines over *Acc*.

Models	single-source		cross-sources	
	F-test	t-test	F-test	t-test
CharSCNN	0.290	0.055	0.001	0.000
CNN	0.369	0.143	0.044	0.279
LSTM	0.270	0.050	0.099	0.005
ConvGRNN	0.219	0.177	0.000	0.000
Ssth	0.020	0.032	0.013	0.001

on fine-grained metrics than that of the coarse-grained metric. The above results validated the effectiveness of our method on sentiment strength prediction tasks, especially when the training set and the testing set are from different sources.

5. CONCLUSIONS

In this article, we proposed a neural network-based framework for sentiment strength prediction. Our model introduced one-hot vectors to capture corpus-specific information and jointly learned hybrid features at semantic and syntactic levels for enhancing model robustness and adaptiveness. Experimental results validated the effectiveness of the proposed sentiment strength prediction method. The main conclusions of our paper are the following:

- Corpus-specific embeddings and POS-level features are both useful to boost sentiment strength prediction performance.
- The differences in variances and performance between our proposed method and each baseline model are more significant on cross-sources testing than that of single-source testing.

In the future, the following directions will be studied: Firstly, we plan to explore effective methods to obtain and exploit topic level features, which is valuable to enrich the semantic information. Secondly, other neural network-based architectures, such as gated recurrent unit [6] and bidirectional recurrent neural networks [33], will be leveraged to learn semantic representation of documents, which could then be incorporated into our method. Finally, we plan to extend our model on cross-domain sentiment analysis.

6. ACKNOWLEDGMENTS

The authors are thankful to the reviewers for their constructive comments and suggestions on this paper. The work described in this paper was supported by the National Natural Science Foundation of China (61502545), two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (UGC/FDS11/E03/16 and UGC/FDS11/E06/14), the Start-Up Research Grant (RG 37/2016-2017R), the Internal Research Grant (RG 66/2016-2017) of The Education University of Hong Kong, and the Special Program for Applied Research on Super Computation of the NSFC-Guangdong Joint Fund (the second phase).

7. REFERENCES

[1] R. Astudillo, S. Amir, W. Lin, M. Silva, and I. Trancoso. Learning word representations from scarce and noisy data with embedding sub-spaces. In *ACL*, pages 1074–1084, 2015.

[2] S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, pages 2200–2204, 2010.

[3] Y. Bengio, H. Schwenk, J. Senécal, F. Morin, and J. Gauvain. Neural probabilistic language models. *J. Mach. Learn. Res.*, 3(6):1137–1155, 2003.

[4] D. M. Blei. Probabilistic topic models. *ACM Commun.*, 55(4):77–84, 2012.

[5] X. Chen, L. Xu, Z. Liu, M. Sun, and H. Luan. Joint learning of character and word embeddings. In *IJCAI*, pages 1236–1242, 2015.

[6] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[7] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, pages 160–167, 2008.

[8] C. dos Santos and M. Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78, 2014.

[9] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, pages 249–256, 2010.

[10] V. Hatzivassiloglou and J. Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. In *COLING*, pages 299–305, 2000.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997.

[12] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. Improving word representations via global context and multiple word prototypes. In *ACL*, pages 873–882, 2012.

[13] A. Jain and D. S. Bais. Performance analysis of image transmission over dstbc with convolutional code. In *IOTA*, pages 379–382, 2016.

[14] R. Johnson and T. Zhang. Effective use of word order for text categorization with convolutional neural networks. In *NAACL-HLT*, pages 103–112, 2015.

[15] R. Johnson and T. Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. In *NIPS*, pages 919–927, 2015.

[16] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. In *ACL*, pages 655–665, 2014.

[17] Y. Kim. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751, 2014.

[18] E. Kouloumpis, T. Wilson, and J. Moore. Twitter sentiment analysis: The good the bad and the omg. In *AAAI*, pages 538–541, 2011.

[19] C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *CIKM*, pages 375–384, 2009.

[20] B. Liu. Sentiment analysis and opinion mining. *Synth. Lect. Hum. Lang. Technol.*, 5(1):1–167, 2012.

[21] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*. Cambridge University Press, 2008.

[22] W. Medhat, A. Hassan, and H. Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams Eng. J.*, 5(4):1093–1113, 2014.

[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[24] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[25] K. P. Murphy. *Machine learning: A probabilistic perspective*. The MIT Press, 2012.

[26] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *EMNLP*, pages 79–86, 2002.

[27] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

[28] Y. Rao. Contextual sentiment topic model for adaptive social emotion classification. *IEEE Intell. Syst.*, 31(1):41–47, 2016.

[29] Y. Ren, Y. Zhang, M. Zhang, and D. Ji. Context-sensitive twitter sentiment classification using neural network. In *AAAI*, pages 215–221, 2016.

[30] E. Riloff, J. Wiebe, and T. Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *CONLL*, pages 25–32, 2003.

[31] D. M. Romero, B. Uzzi, and J. Kleinberg. Social networks under stress. In *WWW*, pages 9–20, 2016.

[32] C. Santos and B. Zadrozny. Learning character-level representations for part-of-speech tagging. In *ICML*, pages 1818–1826, 2014.

[33] M. Schuster and K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45(11):2673–2681, 1997.

[34] A. Severyn and A. Moschitti. Twitter sentiment analysis with deep convolutional neural networks. In *SIGIR*, pages 959–962, 2015.

[35] V. Stoyanov and C. Cardie. Annotating topics of opinions. In *LREC*, pages 3213–3217, 2008.

[36] D. Tang, B. Qin, and T. Liu. Learning semantic representations of users and products for document level sentiment classification. In *ACL*, pages 1014–1023, 2015.

[37] M. Thelwall, K. Buckley, and G. Paltoglou. Sentiment strength detection for the social web. *J. Assoc. Inf. Sci. Technol.*, 63(1):163–173, 2012.

[38] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas. Sentiment strength detection in short informal text. *J. Assoc. Inf. Sci. Technol.*, 61(12):2544–2558, 2010.

[39] P. D. Turney and M. L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 21(4):315–346, 2003.

[40] R. Xia, C. Zong, X. Hu, and E. Cambria. Feature ensemble plus sample selection: Domain adaptation for sentiment classification. *IEEE Intell. Syst.*, 28(3):10–18, 2013.

[41] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, pages 412–420, 1997.

[42] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[43] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *NIPS*, pages 649–657, 2015.