Uyghur Sentiment Classification Based on Multiple Features and Deep Neural Networks (Postprint)

Authors: Maimaiti Ayifu, Wushou' er · Silamu, Iskandar Amudulla, Yang Wenzhong, Palidan Muhtar

Date: 2019-04-01T00:00:00+00:00

Abstract

In response to the long-distance dependency issue inherent in traditional machine learning-based sentiment classification methods and the drawback of deep learning approaches neglecting sentiment lexicons, this paper proposes a Uyghur sentiment classification method that integrates attention mechanism with bidirectional long short-term memory network and convolutional neural network model. Multi-feature concatenated vectors are employed as input to the bidirectional long short-term memory network to capture textual context information, while attention mechanism and convolutional network are utilized to extract hidden sentiment feature information from the text, thereby effectively enhancing the capability to capture textual sentiment semantics. Experimental results demonstrate that the proposed method achieves F1-score improvements of 5.59% and 7.73% on binary-classification and five-classification sentiment datasets respectively, compared to machine learning methods.

Full Text

Preamble

Vol. 37 No. 5
Application Research of Computers
ChinaXiv Cooperative Journal

Uyghur Sentiment Classification Based on Multi-Features and Deep Neural Network

Maimaitiayifu, Silamu Wushouer, Aimudoula Aisikaer†, Yang Wenzhong, Muhetaer Palidan

(College of Information Science & Engineering, Xinjiang University, Urumqi 830046, China)

Abstract: Traditional machine learning methods for sentiment classification suffer from long-distance dependency problems, while deep learning approaches ignore sentiment lexicons. To address these limitations, this paper proposes a Uyghur sentiment classification method that combines an attention mechanism with a bidirectional long short-term memory network and convolutional neural network model. The concatenated multi-feature vector serves as input to the bidirectional LSTM to capture textual context information, while the attention mechanism and convolutional network extract hidden emotional feature information from the text, effectively enhancing the ability to capture sentiment semantics. Experimental results demonstrate that the proposed method achieves F1-score improvements of 5.59% and 7.73% on binary and five-category sentiment datasets, respectively, compared to machine learning methods.

Keywords: sentiment classification; bidirectional long short-term memory network; convolutional neural network; attention mechanism; Uyghur

0 Introduction

Sentiment analysis, also known as opinion mining, aims to extract subjective attitudes, tendencies, and opinions from text [?, ?]. People typically express emotions through text, audio, and images, often supplemented with facial expressions [?]. Common sentiment analysis research includes emotion recognition, sentiment polarity analysis, and topic-based sentiment analysis [?].

With the development of information technology, numerous Uyghur-language news websites, forums, and microblog platforms have emerged, promoting education and economic development in Xinjiang while enhancing regional informatization [?]. However, some netizens also publish negative sentiment information that adversely impacts society. Sentiment analysis technology can analyze the orientation of comments and public opinion on social media, helping government and security departments timely understand public sentiment, 舆论 tendencies, and dynamics. Therefore, capturing users' emotional orientation information from text through technical means holds significant theoretical importance and practical value for social stability and long-term peace in Xinjiang.

Domestic and international scholars have conducted extensive research on sentiment analysis, achieving notable results. Current sentiment analysis methods mainly fall into three categories: lexicon-based methods, traditional machine learning methods, and deep learning methods. Lexicon-based approaches construct sentiment dictionaries comprising emotional words and phrases, relying on dictionaries and rules to compute word sentiment orientation using methods like pointwise mutual information (PMI) [?] to determine sentence-level polarity. These methods are limited by dictionary coverage and rule quality, requiring substantial manual effort and prior knowledge for construction. Traditional machine learning methods typically employ Naive Bayes (NB) [?], Maximum Entropy (ME) [?], and Support Vector Machines (SVM) [?] for sentiment classification. While proven simple and effective, these methods heavily depend on background knowledge and feature selection, achieving high accuracy only with sufficient and correctly annotated training corpora. High-quality corpus annotation and feature selection remain labor-intensive and susceptible to human factors. Moreover, these methods tend to lose deep semantic information and struggle to effectively capture sentiment information in text, exhibiting high sensitivity to domain differences.

In recent years, deep neural network technology has rapidly advanced, demonstrating strong performance across various NLP tasks for resource-rich languages like English and Chinese, including machine translation, speech recognition, question answering, text summarization, relation extraction, and sentiment analysis. However, due to the scarcity of Uyghur resources, sentiment analysis research for Uyghur remains limited. Uyghur sentiment analysis has started relatively late and lacks rich sentiment resources comparable to English or Chinese. Most existing work relies on sentiment dictionaries and traditional machine learning methods, with few studies employing deep neural networks.

Deep neural network-based sentiment analysis primarily uses word embeddings to represent words in text, constructing semantic representations of sentences and documents. Based on these representations, deep neural network models learn sentiment information to analyze text emotions. Commonly used neural network models for sentiment analysis include Recurrent Neural Networks (RNN) [?], Convolutional Neural Networks (CNN) [?, ?, ?], Long Short-Term Memory networks (LSTM) [?, ?, ?], and Gated Recurrent Units (GRU) [?, ?].

Most deep neural network-based sentiment analysis methods treat text as a whole for semantic representation, without emphasizing sentiment words or phrases. Conversely, lexicon-based methods over-rely on sentiment words while ignoring overall semantic relationships. To address these issues, this paper introduces an attention mechanism to encode Uyghur text and sentiment words, proposing a Uyghur sentiment analysis method based on an attention-enhanced BiLSTM-CNN model. The main contributions are:

a) We systematically compiled existing Uyghur sentiment dictionaries, translating and processing the Chinese "HowNet" and "NTUSD" sentiment lexicons to construct a more comprehensive Uyghur sentiment lexicon.

b) We employed syllable feature vectors, part-of-speech feature vectors, and position feature vectors to effectively compensate for the limitations of word vectors.

c) We proposed a deep Uyghur text sentiment classification method (ATT-BiLSTM-CNN) that uses an attention model to obtain deeper emotional feature information, effectively enhancing the capture of textual sentiment semantics and improving classification performance.

d) We compared our model with baseline models on both binary and five-category sentiment datasets, validating the effectiveness of the proposed approach.

1.1 Attention Mechanism

The attention mechanism mimics human brain cognition by allocating more attention to key parts of a target. By computing an attention probability distribution, it highlights the most critical components, thereby optimizing traditional deep learning models. In 2014, Mnih et al. [?] first introduced attention mechanisms to image classification tasks, achieving excellent results and validating their effectiveness. Subsequently, Bahdanau et al. [?] adapted attention mechanisms from image classification to machine translation, making them a hot topic in natural language processing. As research progressed, various improved attention-based models achieved good results in text summarization, text classification, syntactic parsing, sentiment classification, and short-text dialogue.

With the widespread application of deep neural networks and attention models in NLP, various improved attention-based models have emerged. Deep neural networks can effectively learn textual feature representations, solving feature representation problems and improving accuracy for sentiment classification and other NLP tasks. Tang et al. [?] proposed two attention-based models, TD-LSTM and TC-LSTM, with subject information that improved sentiment classification accuracy. Luong et al. [?] proposed a local attention model that adjusts window length to compute alignment probabilities for each word within a specified range. Wu et al. [?] proposed an attention-based CNN-LSTM model that achieved state-of-the-art performance in aspect-based sentiment analysis. In 2015, Yin et al. [?] proposed a CNN network fused with attention mechanisms. These methods validated the effectiveness of combining attention mechanisms with deep neural networks.

Inspired by [?, ?], this paper proposes an attention-based BiLSTM-CNN model that uses BiLSTM and CNN to capture historical/future context information and local information, respectively. The attention mechanism enables the model to focus more on sentiment-bearing parts of the text, thereby improving Uyghur sentiment classification accuracy.

1.2 Neural Networks

Deep neural networks possess excellent self-learning feature extraction capabilities compared to traditional machine learning methods, leading to widespread application across NLP domains. CNN was originally proposed by LeCun et al. [?] for computer vision, where it achieved excellent results. Previous work has demonstrated CNN' s strength in representing latent features for tasks like part-of-speech tagging and sentence classification. In 2014, Kim et al. [?] used CNN for sentiment classification, proving its superior performance over recursive neural networks. Kalchbrenner et al. [?] proposed a novel model combining k-max pooling with multi-layer convolutional neural networks. Another popular deep learning model is the sequential model, Recurrent Neural Network (RNN), which preserves historical text information through hidden states, better capturing semantic relationships between words. RNN variants have been successfully applied to machine translation, text generation, and other tasks. LSTM can capture long-term dependencies in text, enabling holistic understanding of sentiment semantics in reviews.

In recent years, researchers have combined CNN and LSTM due to their complementary modeling capabilities. To our knowledge, no previous work has applied attention mechanism and deep neural network fusion methods to Uyghur sentiment classification.

---

2 Method

This chapter details the architecture of the attention-based BiLSTM-CNN neural network (AT-BiLSTM-CNN) for Uyghur sentence sentiment classification. The proposed model consists of a word vector representation layer, BiLSTM layer, attention layer, CNN layer, and sentiment computation layer. Each layer' s output serves as input to the next network layer. In the word vector representation layer, concatenated vectors of word, POS, syllable, and position features serve as input to the BiLSTM network to capture textual context information. The BiLSTM model' s output is encoded by the attention layer, whose output is then fed to the CNN network for further feature extraction training. Finally, the sentiment computation layer outputs the classification result. The framework of the attention-based BiLSTM-CNN network model is shown in Figure 1 [Figure 1: see original paper].

2.1 Word Vector Representation Layer

The bottommost layer of our model framework is the word vector representation layer, which serves as the entire framework' s input. Assuming a sentence S consists of n words, S = {s , s , ···, s }, each Uyghur sentence is represented by a sentence vector matrix R formed by concatenating the word vector matrix R_w, POS vector matrix R_POS, syllable vector matrix R_SYL, and position vector matrix R_LOC, where denotes vector concatenation.

2.1.1 Word Vectors Word embedding [?, ?] fundamentally converts words into low-dimensional, dense real-valued vectors that computers can process, effectively mapping semantic relationships between words to positional relationships in vector space. Word vectors exhibit excellent computational properties —simple mathematical operations on them can preserve textual feature information, largely mitigating the curse of dimensionality.

To generate a word vector lookup table, we train on large-scale unlabeled corpora. In our experiments, we use the skip-gram model from the Gensim toolkit in Python to train Uyghur word vector models on large-scale sentence corpora. For a given corpus, we store word vectors in a matrix M ^{|V|×d_w}, where |V| is the vocabulary size of the unlabeled Uyghur corpus and d_w is the word vector dimension. For sentence S = {s , s , ⋯, s } with length n, the sentence matrix is represented as X = {x , x , ⋯, x }, where x is the word vector for word s obtained from the lookup matrix M.

Assuming the corpus consists of words w , w , ⋯, w , the Skip-Gram model aims to maximize the following function:

$$\frac{1}{m}\sum_{j=1}^{m}\sum_{-n\leq i\leq n, i\neq 0}\log p(w_{j+i}|w_j)$$

where n is the training window size parameter. In our word vector model training, the context window size is set to 5 and iteration count to 8, generating word vector models with dimensions of 100, 200, 300, and 400. For out-of-vocabulary words, we randomly initialize word vectors using a uniform distribution U(-0.01, 0.01).

2.1.2 POS Vectors POS features contain rich semantic information about words. Converting POS features into vectors and using them as neural network input can further discover structural relationships between words and word sentiment information. For example, adjectives like “ ” or “ ” can express sentiment intensity, which neural network models can identify through POS features. Therefore, we incorporate POS features to further improve sentiment classification accuracy.

Currently, no unified POS tagging set exists for Uyghur. Various research institutions, including the Xinjiang Multilingual Information Technology Laboratory, have established their own POS tagging sets. The Xinjiang Multilingual Information Technology Laboratory manually built a POS tagging set containing 1.2 million words, including a first-level POS tagging set (15 tags) (Table 1), a second-level set (71 tags), and a third-level set (51 tags). To better learn sentiment feature information of emotional words in sentences, we added two POS tags to the first-level set: positive sentiment words are relabeled with the “POS” tag, and negative sentiment words with the “NEG” tag, making emotional words more prominent. We represent POS vectors using 17-dimensional one-hot vectors (v_pos denotes the POS feature vector, R_pos denotes the POS vector matrix), which are concatenated with word, syllable, and position vectors to generate hybrid vectors as BiLSTM input, improving sentiment classification accuracy.

2.1.3 Syllable Vectors Negation and degree words in Uyghur significantly impact the polarity and intensity of sentiment words. Negation words change sentiment polarity, while degree words enhance or weaken sentence sentiment intensity. Uyghur negative sentences are expressed through verb negation forms, where the verb as predicate appears at the sentence end, and negation meaning is conveyed by adding negative suffixes (syllables like][) to the final verb, e.g., " + = ". Similarly, Uyghur degree adverbs express adverbial degree by adding suffixes to nouns and adjectives, constructing them by adding syllables like][to noun/adjective endings, e.g., " + = ". In Uyghur, negation and degree adverb expression is accomplished by adding negation/degree syllables to verb, noun, and adjective endings. Therefore, for given Uyghur sentiment sentences, we first segment words then syllables to conduct finer-grained research on the most critical sentiment-determining units. Finally, concatenating syllable vectors with word and POS vectors as model input improves Uyghur sentiment classification accuracy.

Uyghur words can consist of multiple syllables or a single syllable. We represent them as: Word = {syllable_n, syllable_{n-1}, ···, syllable_2, syllable_1}

For syllable segmentation, we use the syllable segmentation tool developed by the Xinjiang Multilingual Laboratory, employing a backward segmentation approach—extracting syllables sequentially from a word' s end. For example, given the word " " (students'), its syllable representation is , + + + + so F_syl = [F_syl4; F_syl3; F_syl2; F_syl1], where F_syl1 =][, F_syl2 =][, F_syl3 =][, F_syl4 =][.

In this experiment, we randomly generate 30-dimensional syllable vectors using a uniform distribution U(-0.01, 0.01).

2.1.4 Sentiment Word Position Vectors Through detailed statistical analysis of the Uyghur sentiment corpus provided by our laboratory, we found that emotional interjections and mimetic words generally appear at sentence beginnings, where they indicate strong sentiment and are followed by commas (or exclamation marks for intense emotions). Sentiment words appearing at sentence ends typically express affirmation or negation, generally as verbs. Sentiment words in middle or penultimate positions are usually adjectives or degree adverbs. Table 2 shows examples of Uyghur sentiment word positions in sentences.

To improve Uyghur sentiment classification accuracy, we explored additional sentiment information as features. Statistics on sentiment word positions in over 60,000 Uyghur sentiment sentences show that most sentiment words appear at sentence beginnings and ends (Table 3).

Before training, we vectorize each word in sentences using one-hot representation, setting the position to 1 for sentiment words and 0 otherwise. We use the longest sentence length in the corpus (23) as the position vector dimension (v_loc denotes position vector, R_loc denotes the position vector matrix). Finally, the hybrid vector generated by concatenating word, POS, syllable, and position vectors serves as model input.

2.2 Bidirectional Long Short-Term Memory Network

2.2.1 Long Short-Term Memory Unit Recurrent Neural Networks (RNN) [**?**] are commonly used sequential learning methods in deep learning. Their recurrent structure provides certain memory capabilities, theoretically enabling capture of arbitrarily long historical information. However, practical applications often suffer from vanishing or exploding gradients, preventing learning of long-distance logical relationships in input sequences.

To address RNN limitations, Hochreiter et al. [**?**] proposed the LSTM model, replacing RNN hidden layers to avoid gradient vanishing. An LSTM unit comprises an input gate i_t, output gate o_t, forget gate f_t, and memory cell c_t, with the memory cell as the core component, as shown in Figure 2 [Figure 2: see original paper]. For a Uyghur sentence' s word vector sequence S = {x , x , ···, x }, x_t is the LSTM unit' s input at step t, representing the hybrid vector for the corresponding word.

The three gates and memory cell in an LSTM unit are computed as follows:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$

where W_* ^{m×d} represent weight matrices; b_* ^m represent biases; is the nonlinear activation function; m = 100 is the number of LSTM network units; denotes element-wise multiplication; x_t is the input vector to the LSTM unit; h_t ^m is the hidden state vector. For a given sentence S = {x , x , ···, x }, each word x_i maps to its word vector v_i^w ^{d_w}, each word' s POS maps to POS vector v_i^{pos} ^{d_p}, each word' s syllable vector maps to v_i^{syl} ^{d_s} (extracted using the syllable segmentation method described in Section 2.1.3), and each word' s position vector maps to v_i^{loc} ^{d_l} (where d_w = 300, d_p = 17, d_s = 30, d_l = 23 are the dimensions of word, POS, syllable, and position vectors, respectively). Therefore, the hybrid word vector dimension at the neural network input layer is d = d_w + d_p + d_s × n + d_l, where n is the number of syllable features extracted from the word (manually set before training). These vectors are concatenated to generate the sentence vector matrix:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \Rightarrow \begin{bmatrix} v_1^w \oplus v_1^{pos} \oplus v_1^{syl} \oplus v_1^{loc} \\ v_2^w \oplus v_2^{pos} \oplus v_2^{syl} \oplus v_2^{loc} \\ \vdots \\ v_n^w \oplus v_n^{pos} \oplus v_n^{syl} \oplus v_n^{loc} \end{bmatrix}$$

2.2.2 Bidirectional Long Short-Term Memory Network While LSTM can capture long-term historical information from input sequences, it cannot capture future information. Bidirectional LSTM (BiLSTM) consists of forward and backward LSTM networks that can capture historical and future information, respectively, obtaining more contextual dependencies. The two hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ represent past and future information. Their concatenated vector provides complete contextual history and future information, with the final output being a fused result h_t. Each word in a sentence is embedded as a word vector into the network. Using BiLSTM to encode Uyghur text ensures forward and backward semantic information are considered equally, thereby capturing sentiment information in the text.

2.3 Attention Layer

As is well known, sentence sentiment often varies across different parts of a sentence. Certain sentiment words or phrases play decisive roles in determining sentiment, while other words are irrelevant. Therefore, we introduce an attention mechanism to focus on these important words and transform their representations into sentence vectors. Essentially, the attention mechanism computes a context vector for the sentence.

In our model, the hybrid vector composed of word, POS, syllable, and position vectors serves as BiLSTM network input for encoding. The hidden state vector h_t produced by BiLSTM at each time step is fed into the attention model to encode Uyghur sentences. The attention model first uses a single-layer perceptron (MLP) to weight the input hidden state h_t, constructing a context-relevant sentence representation vector r, as shown in Equation (9):

$$r = \sum_{t=1}^{n} \alpha_t h_t$$

where α_t is the attention weight for hidden state vector h_t output by BiLSTM, related to each time step' s input state and a randomly initialized context vector m_w, and can be calculated through Equations (10) and (11):

$$\alpha_t = \frac{\exp(m_t^T m_w)}{\sum_{i=1}^{n} \exp(m_i^T m_w)}$$

$$m_t = \tanh(W_w h_t + b_w)$$

In Equation (11), W_w is the model weight; b_w is the bias, which together with m_w serve as model parameters learned through continuous training. Through this attention model, we construct a fixed-length context-relevant text representation vector r containing information about the importance of each input state, enabling the model to focus more on sentiment-bearing words or phrases in the text, thereby improving Uyghur sentiment classification accuracy. This text representation allows assigning different attention weights to feature sequences, making it easier to identify important information such as sentiment words.

2.4 CNN Network Layer

Convolutional Neural Networks (CNN) [**?**, **?**, **?**] can capture local feature information in text and have demonstrated excellent performance in sentence-level sentiment classification tasks [**?**]. Therefore, we use CNN as a complement to the BiLSTM model, implementing an attention-based BiLSTM-CNN hybrid model (abbreviated as AT-BiLSTM-CNN).

CNN consists of an input layer, convolutional layer, pooling layer, and fully connected layer. In this experiment, we concatenate the attention model' s output r_t with sentiment word vector v_e as the input vector matrix r* for training, i.e., r* = [r_t v_e]. The convolutional layer uses different filters to perform convolution operations on the input vector, extracting local features. The calculation formula is:

$$u(i) = f(W \cdot r *_{i:i+h-1} + b)$$

where r*_{i:i+h-1} represents the local feature matrix extracted from rows i to i+h-1 of the input vector; u(i) is the convolution output of a filter at position i; W is the filter; b is the convolution bias term; f(·) is the nonlinear activation function, for which we use ReLU.

Subsequently, the max-pooling layer computes the maximum value across all u(i) in vector u to obtain the most salient feature value:

$$\hat{c}_j = \max\{u(i)\}$$

where j denotes the j-th convolutional filter. In our experiments, we used convolutional filters of different sizes: h = 3, h = 4, h = 5. The feature information sampled from windows with T convolutional filters is shown in Equation (15). The feature vector output by the subsampling layer serves as input to the fully connected layer.

2.5 Sentiment Computation Layer

We use the softmax function [**?**, **?**] as the sentiment classifier, taking the feature representation $\hat{c}$ output by the fully connected layer as input to predict sentiment polarity categories. The calculation formula is:

$$\hat{y} = \text{softmax}(W_f \hat{c} + b_f)$$

where $\hat{y}$ is the model' s predicted text sentiment category; W_f and b_f are the weight matrix and bias of the fully connected layer, respectively.

2.6 Model Training

We use backpropagation to train and update the model, optimizing it by minimizing cross-entropy loss [**?**, **?**]. The cross-entropy loss function is calculated as:

$$\text{loss} = -\frac{1}{|T|}\sum_{i=1}^{|T|}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) + \lambda\|\theta\|^2$$

where T is the training dataset; C is the number of sentiment categories; y is the actual sentiment category; $\lambda\|\theta\|^2$ is the regularization term as a penalty; λ is a hyperparameter.

3.1 Sentiment Lexicon Construction

A sentiment lexicon comprises sentiment words, sentiment phrases, negation words, adverbs, adjectives, interjections, modal particles, and idiomatic expressions with sentiment orientation.

Currently, no publicly available Uyghur sentiment lexicon exists. For resource-scarce languages, research has attempted to translate sentiment lexicons from resource-rich languages, achieving good classification results. Therefore, we first use the Chinese-Uyghur translation interface developed by the Xinjiang Information Technology Laboratory to translate the widely used "HowNet" Chinese sentiment analysis word lists (positive/negative sentiment words, positive/negative evaluation words) and the "NTUSD" Chinese sentiment dictionary created by National Taiwan University. After manual alignment and cleaning to remove words that lost sentiment orientation after translation, we additionally manually collected sentiment words from emotional sentences. The final lexicon, UySentiDict, contains 5,643 words, including 2,411 positive and 3,232 negative sentiment words.

In our experiments, sentiment words in sentences are converted to word vectors and concatenated with hidden state vectors output by the attention layer as CNN network input, thereby improving sentiment classification accuracy.

3.2 Sentiment Analysis Dataset

To construct a Uyghur sentence-level sentiment corpus, we selected publicly available articles and comment information from Uyghur websites such as Tianshan Net. We used a web crawler tool developed by our laboratory to download webpages, filtering out sentiment-bearing comment information after deduplication and denoising. The collected corpus was sentence-segmented and manually classified to construct a binary classification dataset (UySenti2Data), as shown in Table 4 .

In addition to the Uyghur binary sentiment (positive/negative) corpus, we also constructed a five-category sentiment corpus (neutral, happy, angry, surprised, sad) (UySenti5Data), as shown in Table 5 .

To verify the effectiveness of our hybrid model, we constructed the five-category dataset based on the binary sentiment dataset. Table 6 provides examples for each category with corresponding translations.

3.3 Evaluation Metrics

As a text classification task, sentiment classification performance is evaluated using precision, recall, and F1-score. The calculation formulas are:

$$\text{Precision} = \frac{TP}{TP + FP}$$
$$\text{Recall} = \frac{TP}{TP + FN}$$
$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where TP (true positive) is the number of positive instances correctly predicted as positive; FN (false negative) is the number of positive instances incorrectly predicted as negative; FP is the number of negative instances incorrectly predicted as positive; TN is the number of negative instances correctly predicted as negative.

3.4 Data Preprocessing

Text corpora downloaded from the internet are often informal, typically containing URLs, email addresses, various punctuation marks, numbers, and usernames, which pose significant challenges to sentiment classification. To remove noise interference, we preprocess Uyghur sentiment text corpora with the following rules:

a) All numbers are replaced with "0" .

b) URLs and email address strings are deleted.

c) Punctuation marks except periods, commas, exclamation marks, and question marks are removed to eliminate potential noise affecting classification performance.

d) Chinese words in sentences are translated using our laboratory' s Chinese-Uyghur translation interface, with translation results replacing the original Chinese words.

e) Consecutive exclamation marks, question marks, periods, or other characters are reduced to a single instance.

3.5 Experimental Parameter Settings

To achieve optimal sentiment classification performance, we conducted extensive experiments to tune model hyperparameters through cross-validation minimizing cross-entropy loss. Word vector dimension is set to 300, LSTM unit size to 100. We train the model using mini-batch size of 128 and the Adam optimization algorithm. The model achieves best accuracy at 19 training epochs, with further details in Table 7 .

3.6 Experimental Comparison Models

We compare different sentiment classification methods with our proposed multi-feature attention-based BiLSTM-CNN hybrid model to validate its effectiveness.

We also demonstrate that word, POS, syllable, and position vector features improve classification accuracy. The comparison models include:

a) **Sentiment Dictionary (SentiDict)**: Classifies Uyghur text based on our laboratory' s sentiment dictionary—sentences containing positive words are classified as positive, negative words as negative.

b) **Multinomial Naive Bayes (MNB)**: A typical traditional machine learning method widely used in text and sentiment classification.

c) **CNN Model**: Uses single-channel and multi-channel convolutional neural networks to learn Uyghur sentence representations, capturing local features through convolution and max-pooling for sentiment classification.

d) **Support Vector Machine (SVM)**: The most common traditional machine learning method for sentiment classification, typically using n-gram features. We use unigram, bigram, and trigram features.

e) **LSTM**: Standard unidirectional LSTM network encoding Uyghur sentences into variable-length target sequences.

f) **BiLSTM**: Bidirectional LSTM without attention mechanism for Uyghur sentence sentiment classification.

g) **BiLSTM-CNN**: BiLSTM-CNN hybrid model without attention mechanism for Uyghur sentence sentiment classification.

h) **Attention-based BiLSTM-CNN (ATT-BiRNN-CNN)**: Our proposed attention-enhanced BiLSTM-CNN hybrid model.

---

4 Experimental Results and Analysis

To validate the effectiveness of our ATT-BiRNN-CNN model for Uyghur sentiment classification, we conducted comparative experiments on our constructed binary and five-category sentiment datasets. The specific data quantities for training, development, and test sets in both datasets are shown in Tables 4 and 5.

We selected lexicon-based methods, typical traditional machine learning methods, and simple neural network models as baselines for comparison with our proposed model.

4.1 Comparison of Neural Network and Baseline Methods

The first group of experiments compared neural network methods with lexicon and traditional machine learning methods on UySenti2Data and UySenti5Data. For fair comparison, except for the lexicon method, all other machine learning and neural network methods used our pre-trained 300-dimensional word vectors as input, as shown in Table 8 .

The results show that our ATT-BiLSTM-CNN model achieves the highest F1-scores for both binary and five-category classification among all baseline methods. For the lexicon method, sentences cannot be correctly classified if sentiment words are not in the dictionary. MNB requires large corpora for good performance. SVM shows strong generalization ability, with SVM-bigram performing best among SVM variants. Among single deep neural network models, CNN performs well by capturing local features of sentiment words. Among hybrid models, our proposed model achieves the best classification performance.

Table 9 shows the five-category classification results of each model for the same sentence “ ”(Don’t take others’things without permission). SVM-bigram, CNN, BiLSTM-CNN, and ATT-BiLSTM-CNN correctly classify this instance for both binary and five-category tasks.

4.2 Impact of Multi-Features on Sentiment Classification

To validate the effectiveness of POS, syllable, and position features, we used different feature combinations as model input. We sequentially added POS vector (F_pos), syllable vector (F_syl), and position vector (F_loc) to the word vector (F_w) to create hybrid vectors (F = F_pos F_w F_syl F_loc) for training.

First, using the vector concatenating POS and word vectors (F = F_pos F_w) as input improved binary and five-category F1-scores by 0.26% and 0.96%, respectively. Second, adding syllable vectors to word and POS vectors, where F_syl1 is the word’ s last syllable, F_syl2 the last two syllables, F_syl3 the last three syllables, etc. (padding with zeros for insufficient syllables), we tested using up to the last 4 syllables as features. Results show that using F_syl3 as feature yields significant improvements: 1.57% and 2.19% for binary and five-category accuracy, respectively. Finally, adding position vectors as word vector supplements improved binary and five-category F1-scores by 3.05% and 4.0%, respectively. Table 10 shows these results.

Uyghur has many polysemous words and out-of-vocabulary words that appear frequently, which traditional statistical and rule-based methods cannot fully resolve. For the sentence “ ” (Don’ t take others’ things without permission), “ ” is a polysemous word (verb: don’ t take; noun: apple). In this sentence it’ s a verb, and syllable segmentation yields “ + ” , where “ ” is the negative suffix expressing negation at the sentence end.

Traditional statistical models classify this sentence as neutral, while our hybrid model correctly classifies it as "angry" using POS, syllable, and position vectors.

4.3 Impact of Network Parameters on Sentiment Classification

This group of experiments observes the impact of major network parameters including optimization function, dropout, iteration count, and word vector dimension. First, to mitigate overfitting, we use dropout regularization. While dropout is typically set to 0.5, we experimented with values in [0.2, 0.3, 0.4, 0.5, 0.6, 0.7], with results shown in Figure 3 [Figure 3: see original paper].

Second, we tested RMSprop, SGD, Adadelta, and Adam as optimization functions, with Adam achieving the best classification performance, as shown in Figure 4 [Figure 4: see original paper]. Third, we experimented with training iteration counts, achieving optimal performance at 19 epochs, as shown in Figure 5 [Figure 5: see original paper]. Finally, we tested word vector dimensions of 100, 200, 300, and 400, with dimension 300 yielding the best results, as shown in Figure 6 [Figure 6: see original paper].

5 Conclusion

This paper proposes a hybrid model combining multi-features with attention-based deep learning for Uyghur sentiment classification. First, POS, syllable, and position feature vectors supplement word vectors, effectively mapping Uyghur sentences to low-dimensional abstract feature matrices. The concatenated hybrid vectors mine inherent word characteristics. The BiLSTM network then encodes text to capture historical contextual dependencies. The attention mechanism focuses on sentiment-bearing words, and the concatenated vector of sentiment word vectors and attention layer hidden states serves as CNN input to capture sentiment-related local information. Finally, the softmax function produces sentiment classification results. Experimental results demonstrate that our hybrid model significantly outperforms traditional machine learning and simple deep learning methods in accuracy, recall, and F1-score on both Uyghur binary and five-category sentiment tasks, validating its effectiveness.

Future work will apply our model to other NLP tasks to evaluate its effectiveness and further optimize and improve its network parameters.

References

[1] Shin B, Timothy L, Jinho C. Lexicon integrated CNN models with attention for sentiment analysis[J]. arXiv preprint arXiv: 1610.06272, 2016.

[2] Zhao Yanyan, Qin Bing, Liu Ting. Sentiment analysis[J]. Journal of Software. 2010, 21 (8): 1834-1848.

[3] Pang B, Lee L, Vaithyanathan S. Thumbs up?: Sentiment classification using machine learning techniques[C]// Proc of ACL-02 Conference on Empirical Methods in Natural Language Processing. [S.l.]: Association for Computational Linguistics, 2002: 79-86.

[4] Wang Yequan, Huang Minlie, Zhu Xiaoyan, et al. Attention-based LSTM for aspect-level sentiment classification[C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2016: 606-615.

[5] Tong Yixuan, Zhang Yangsen, Jiang Yuru. Study of sentiment classification for Chinese microblog based on recurrent neural network[J]. Chinese Journal of Electronics. 2016, 25 (4): 601-607.

[6] Liu Jishuo, Li Zhe, Ye Xin, et al. Sentiment orientation of text: bfsmPMI-SVM[J]. Journal of Wuhan University: Natural Science Edition. 2017, 63 (3): 259-264.

[7] Zhang Dongwen, Xu Hua, Su Zengcai, et al. Chinese comments sentiment classification based on word2vec and SVM perf[J]. Expert Systems with Applications. 2015, 42 (2): 1857–1863.

[8] Wang Jin, Yu Liangchih, Robert Lai K, et al. Dimensional sentiment analysis using a regional CNN-LSTM model[C]// Proc of the 54th Annual Meeting of the Association for Computational Linguistics. 2016.

[9] Yoon Kim. Convolutional neural networks for sentence classification[C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2014: 1746-1751.

[10] Gichag L, Jeong J, Seungwan S, et al. Sentiment classification with localization based on weakly supervised learning with a convolutional neural network[J]. Knowledge-Based Systems, 2018, 152(7): 70-82.

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

[12] Miyazaki R, Komachi M. Japanese sentiment classification using a tree-structured long short-term memory with attention[Z]. 2017.

[13] Kuta M, Morawiec M, Kitowski J. Sentiment analysis with tree-structured gated recurrent units[Z]. 2017.

[14] Mnih V., Heess N., Graves A., et al. Recurrent models of visual attention[C]// Advances in Neural Information Processing Systems, 2014: 2204-2212.

[15] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[J]. Computer Science, 2014.

[16] Tang Duyu, Qin Bing, Liu Ting. Document modeling with gated recurrent neural network for sentiment classification[C]// Proc of Conference on Empirical

Methods in Natural Language Processing. 2015: 1422-1432.

[17] Luong M T, Pham H, Manning C D. Effective approaches to attention-based neural machine translation[J]. Computer Science, 2015.

[18] Wu C, Wu F, Liu J, et al. THU_NGN at SemEval-2018 task 1: fine-grained tweet sentiment intensity analysis with attention CNN-LSTM[C]// Proc of the 12th International Workshop on Semantic Evaluation. 2018: 186-192.

[19] Yin Wenpeng, Schütze H, Xiang Bing, et al. ABCNN: attention-based convolutional neural network for modeling sentence Pairs[J]. Transactions of the Association for Computational Linguistics, arXiv preprint arXiv: 1512.05193, 2015.

[20] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Computation, 2012, 9 (8): 1735-1780.

[21] Lecun Y, Bengio Y, Hinton G. Deep learning[J]. Nature. 2015, 521 (7553): 436-444.

[22] Mikolov T, Chen Kai, Corrado G, et al. Efficient estimation of word representations in vector space[C/OL]// Proc of Workshop at ICLR. 2013. arXiv: 1301.3781.

[23] Mikolov T, Sutskever I, Chen Kai, et al. Distributed representations of words and phrases and their compositionality[C/OL]// Advances in Neural Information Processing Systems, 2013, arXiv: 1310.4546.

[24] Rush A M, Chopra S, Weston J. A neural attention model for sentence summarization[C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2015: 379-389.

[25] Feng Shi, Wang Yang, Liu Liran, et al. Attention based hierarchical LSTM network for context-aware microblog sentiment classification[J]. World Wide Web, 2018 (2): 1-23.

[26] Rush A M, Chopra S, Weston J. A neural attention model for sentence summarization[C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2015: 379-389.

[27] Wang Yequan, Huang Minlie, Zhu Xiaoyan, et al. Attention-based LSTM for aspect-level sentiment classification[C]// Proc of Conference on Empirical Methods in Natural Language Processing. 2016: 606-615.

[28] Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences[C]// Proc of the 52nd Annual Meeting of the Association for Computational Linguistics. 2014: 1749-1751.

Note: Figure translations are in progress. See original paper for figures.