Multimedia Short Text Classification via Deep RNN-CNN Cascade

Authors: Tao Aishan, Tao Aishan

Date: 2019-02-22T00:00:00+00:00

Abstract

Abstract—With the rapid development of mobile technologies, social networking software such as Twitter, Weibo, and WeChat are becoming ubiquitous in our everyday life. These social networks generate a deluge of data that consists of not only plain texts but also images, videos, and audios. Consequently, traditional approaches that classify short text by counting only keywords become inadequate. In this paper, we propose a multimedia short text classification approach using a deep RNN (Recurrent Neural Network) and CNN (Convolutional Neural Network) cascade. We first employ an LSTM (Long Short-Term Memory) network to convert information in images into textual information. Then, a convolutional neural network is used to classify multimedia texts by taking into account both the texts generated from images as well as those contained in the initial message. Experiments using the MSCOCO dataset demonstrate that the proposed method exhibits significant performance improvement over traditional methods.

Full Text

Multimedia Short Text Classification via Deep RNN-CNN Cascade

Tao Aishan[1] (1531844@tongji.edu.cn)
Hu Chaocao[2] (1631580@tongji.edu.cn)

Abstract

With the rapid development of mobile technologies, social networking platforms such as Twitter, Weibo, and WeChat have become ubiquitous in everyday life. These networks generate a deluge of data comprising not only plain text but also images, videos, and audio. Consequently, traditional approaches that classify short texts based solely on keyword counting have become inadequate. In

this paper, we propose a multimedia short text classification approach using a deep cascade of Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN).

We first employ a Long Short-Term Memory (LSTM) network to convert information from images into textual descriptions. Then, a convolutional neural network classifies the multimedia content by considering both the text generated from the image and the text contained in the original message. Experiments on the MSCOCO dataset demonstrate that the proposed method achieves significant performance improvements over traditional approaches.

I. Introduction

In the era of big data, the traditional form of information has fundamentally changed. Most notably, long-form text has been largely replaced by short text. For example, Twitter allows users to publish messages limited to 140 characters. Simultaneously, plain text is gradually being replaced by messages combining images with brief textual content.

Consider a WeChat Moments post: "Today is a special day for every American!" accompanied by a picture depicting Trump and Hillary. Analyzing the picture and text separately yields no definitive conclusion. However, extracting content from the image and combining it with the original text reveals that this message belongs to the politics category, specifically indicating that today is presidential election day.

Traditionally, two common methods existed for classifying multimedia short texts. The first approach classifies raw data using only image information, ignoring textual components [1]. The second method classifies data using only text information, disregarding image content. Text classifiers themselves fall into two categories: traditional models such as mixture modeling text classifiers, probabilistic models, and naive Bayes classifiers [2], [3]; and CNN-based text classifiers built on deep learning [4], [5].

In this paper, we propose combining original text with image content to classify multimedia short texts. We address two key questions: How can we extract useful information from images? What rules should govern this extraction process? We introduce a novel RNN-CNN model that answers these questions and achieves superior results.

The remainder of this paper is organized as follows: Section 2 reviews related work. Section 3 presents the architecture of our proposed model with detailed descriptions. Section 4 demonstrates the effectiveness of our model through experiments. Finally, Section 5 provides a comprehensive summary.

II. Related Works

With the proliferation of social software platforms, messages are organized in diverse ways and feature numerous new characteristics, such as "@username" mentions, slang, percentage signs, and other symbols. Traditional models struggle to summarize these varied data features, and the conventional bag-of-words (BOW) input format suffers from severe data sparsity issues [6], [7], [8]. Deep learning-based text classifiers have gained increasing popularity, with word representation vectors as neural network inputs exerting significant influence on classification results. Effective pre-trained word embeddings can be obtained through neural networks [9], [10], [11]. These embeddings not only preserve word relationships but also satisfy contextual semantic relationships.

As previously noted, images often exhibit strong associations with their accompanying text in multimedia messages. Recent deep learning-based image classification methods have achieved remarkable success [12], [13], enabling category prediction based solely on image content. For instance, a picture containing a soldier typically belongs to the military category. In this paper, we incorporate image information when classifying multimedia short texts using deep learning methods. To combine original text and image content as a unified input for CNN text classifiers, both modalities must be projected into the same vector space. Furthermore, when extracting useful sentences from pictures, these sentences should be grammatically correct and consistent with the image content. Numerous image captioning models exist [14], [15], but they face data sparsity challenges in big data applications.

Recently, specialized RNN models applied to machine translation have achieved significant breakthroughs [16], [17], [18]. With advances in computer vision [19], image classification accuracy has reached impressive levels. However, these image recognition methods cannot effectively describe picture content when images are blurry. There is a high probability that [Figure 1: see original paper]. This is an inserted RNN-CNN graphic.

In this paper, we apply an LSTM model [20] to generate sentences describing images. Even for blurry images, the generated sentences can provide relevant verbs and adjectives that aid classification. The core of the LSTM model is to predict each word in the sentence correctly. The Kiros team [21] used a convolutional neural network to predict the next word based on given sentences and images. Other works have applied recurrent neural networks to prediction tasks [22], [23].

Where $S = (s_1, \cdots, s_T)$ represents the sentence generated by the RNN model. $s_t, t = 1, \cdots, T$, represents a word in the sentence S, where T is the number of words in the sentence and is an unbounded value. I is an input image. In the RNN process, the model generates a single word s_t at each iteration. We employ the GoogLeNet pre-trained model on the ILSVRC 2014 dataset for object recognition and detection [25], which is currently the largest image classification dataset. The iteration continues until it generates the `<EOS>` character.

III. RNN-CNN Cascade Model

A. Overall Architecture

Our model consists of an image captioning model and a CNN text classifier. First, we use a pre-trained CNN model to encode an image I as a fixed-length vector [24]. The "show and tell" model [20] is then adopted to generate sentences subsequently. The objective function $p(S|I)$ is used to train the LSTM model, where S represents the generated sentence. The correct sentence representation of an image is obtained when $p(S|I)$ reaches its maximum value. We then connect this generated sentence with the original short text as input to the CNN text classifier. This end-to-end model is called RNN-CNN (Recurrent Neural Network cascade with Convolutional Neural Network) classifier, as shown in Figure 1.

During description generation, the model must satisfy two requirements described in the Introduction. We select the RNN model for generating descriptions because it can preserve contextual relationships. For a given image in a multimedia short text, the best image description can be computed by conditional probability as follows.

Each RNN iteration has three inputs and one output, where t represents the iteration number. One input is the memory h_t, a hidden state of fixed length. The memory h_t saves information generated from the beginning to the end and is updated at each time step with another input x_t. The final input is the memory cell state c_{t-1}. The output of each iteration is a probability distribution over all words: $h_{t+1} = f(h_t, x_t)$. We choose LSTM as $f(\cdot)$. The LSTM is designed to avoid the long-term dependency problem through its carefully designed gate structure. The gate controls information flow from h_{t-1}.

The formulae to calculate the memory cell output and update the memory cell state through gates are as follows:

$$
\begin{aligned}
f_t &= \sigma(W_{fx}x_t + W_{fh}h_t) \\
i_t &= \sigma(W_{ix}x_t + W_{ih}h_t) \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}x_t + W_{ch}h_t) \\
o_t &= \sigma(W_{ox}x_t + W_{oh}h_t) \\
h_{t+1} &= o_t \odot \tanh(c_t)
\end{aligned}
$$

σ is a sigmoid function. $\odot$ is an element-wise multiplication operation. The various W matrices are trained parameters. The forget gate f_t determines how much information should be discarded, with values between 0 (drop completely) and 1 (keep completely). The input gate i_t controls the updating process, deciding which parts of the input x_t and previous hidden state h_t should be updated to the memory cell c_t. The output gate o_t determines which parts of the memory cell c_t should be output. tanh is the hyperbolic tangent function. p_{t+1}

represents the probability distribution over all words. The initialized input of the LSTM is an image processed by CNN.

The image I is input only once: $x_{-1} = \mathrm{CNN}(I)$. The output sequence appears as $(s_1 \cdots s_N)$. Our loss is the sum of the negative log likelihood of the correct word at each step:

$$L = -\sum \log p_t(s_t)$$

where p_t represents the probability of generating the correct word s_t. The loss function reaches its minimum value when the sum of these probabilities is maximized. After obtaining sentences generated by the LSTM model, we must use metrics to evaluate them [26]. We use CIDEr [27] as the primary metric and BLEU [28] as an auxiliary metric. Higher scores indicate more effective sentences, according to both CIDEr and BLEU.

B. CNN Model

The following describes the complete RNN-CNN model process (Figure 1). The first layer is an input layer. The sentence $S = \{s_1, s_2, \cdots, s_N\}$ is generated by the LSTM model, where N represents the number of words in the longest sentence. $Y = \{y_1, y_2, \cdots, y_{T'}\}$ is the original caption in a multimedia short text. We now connect them as $P = \{r_1, r_2, \cdots, r_n\}$, where $P = S \oplus Y$. The value n equals the sum of N and T'. We apply word embedding from scratch to represent each word in the sentence. For the sentence $P = \{r_1, r_2, \cdots, r_n\}$, it is projected to a matrix $M \in \mathbb{R}^{n \times k}$ obtained through a lookup table, where k is the dimension of word embedding. Every word in the sentence becomes a k-dimensional word vector.

In the convolution layer, we use $r_{i:i+j-1}$ to represent the concatenation of words $r_i, r_{i+1}, \cdots, r_{i+j-1}$, where j is the window kernel size of convolution. The weight $W \in \mathbb{R}^{n \times k}$ is applied to the convolution operation to produce a new feature. The process of generating new feature c_i is as follows:

$$c_i' = f(W' \cdot r_{i:i+j-1} + b)$$

where f is a non-linear function and $b \in \mathbb{R}$ is a bias term. We apply the kernel with window size j to slide across sentence $P = \{r_1, r_2, \cdots, r_n\}$ to obtain a feature map:

$$c = [c_1, c_2, \cdots, c_{n-j+1}]$$

where $c \in \mathbb{R}^{n-j+1}$ is generated with a window kernel size of j. The main purpose of the convolutional layer is to strengthen local features. Different convolutional kernels with different sizes can extract multiple features.

The next layer is the max pooling layer. The max pooling operation selects the maximum value $\hat{c} = \max\{c\}$ from c as the feature. The maximum value captures the most salient feature of each feature map. The max pooling layer reduces the number of parameters and decreases the time complexity of the entire model. It also solves the problem of variable sentence lengths.

The subsequent layer is a fully connected softmax layer with ℓ_2-norm dropout operation. Some neural units are ignored to prevent co-adaptation of hidden units. After processing by the max pooling layer, the output is $z = [\hat{c}_1, \cdots, \hat{c}_m]$, where m represents the number of filters. The dropout process is as follows:

$$h'' = W'' \cdot (q \odot z) + b''$$

where h'' is an output unit in forward propagation, W'' is the weight matrix, $\odot$ is the element-wise multiplication operator, and $q \in \mathbb{R}^m$ is a "masking" vector whose elements are Bernoulli random variables. This dropout operation primarily prevents overfitting. Meanwhile, the Softmax operation is applied as a classifier to generate the probability distribution over categories.

Our experiment employs a multichannel architecture, as shown in Figure 1. One channel is static, where parameters remain fixed during training. The other is non-static, where parameters can be fine-tuned via backpropagation. Multichannel architectures demonstrate better performance on many datasets because fine-tuning brings vectors closer to the specific task while the static channel limits the extent of vector changes. Figure 2 shows an example of multimedia short text application, demonstrating that our model achieves better classification than traditional CNN models.

IV. Experiment

MSCOCO [29] is a suitable dataset for multimedia short text classification because it contains 91 categories, each consisting of many multimedia short texts. To test the robustness of our model, we conduct three groups of experiments on two-category, three-category, and five-category datasets respectively (Figure 3 [Figure 3: see original paper]).

The two-category experiment uses the vehicle category (including five subclasses: car, boat, truck, airplane, train) and the animal category (including five subclasses: giraffe, zebra, bear, sheep, elephant). The three-category dataset extends the two-category dataset with the furniture category (including five subclasses: bed, couch, toilet, table, chair). The five-category dataset further adds the sports category (including five subclasses: skis, snowboard, kite, baseball, skateboard) and the food category (including five subclasses: sandwich, cake, orange, broccoli, carrot) to the three-category dataset.

To ensure fairness and accuracy, we extract 1,000 multimedia short texts for each class during training (200 per subclass) and 300 multimedia short texts for

each class during testing (60 per subclass).

We train and test a text CNN using caption texts and their labels. Then we train and test an image ResNet using images and their labels. Finally, the RNN-CNN model (with CIDEr value of 85) performs a new classification experiment using multimedia short texts. We carefully compare the results of these three approaches (Table 1).

The classification accuracy of the image ResNet34 model is four percent higher than the average accuracy of the text CNN model. However, the image-text RNN-CNN model achieves even higher accuracy than the image ResNet34 model. Meanwhile, the accuracy of both the image ResNet34 and text CNN models decreases as the number of categories increases, while the image-text RNN-CNN model maintains nearly the same accuracy for both two-category and multi-category classification. The RNN-CNN model demonstrates higher stability and robustness.

V. Conclusion

This paper addresses the new data pattern of multimedia content and proposes a novel end-to-end classifier model that can automatically classify new media data. Traditional image classifiers can capture image features, but images cannot be recognized correctly when they are blurry. By converting images to sentences, we obtain helpful semantic relationships such as verbs and adjectives that are very useful for classification. When the generated sentence is combined with the original description as input, the accuracy and stability of the RNN-CNN model show significant improvement, as demonstrated by our experiments. The RNN-CNN model has extensive applicability and can be widely applied to new media data classification tasks.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] C. C. Aggarwal and C. Zhai, "A survey of text classification algorithms," in *Mining text data*. Springer, 2012, pp. 163–

[3] C. C. Aggarwal, S. C. Gates, and P. S. Yu, "On using partial supervision for text categorization," *IEEE Transactions on Knowledge and data Engineering*, vol. 16, no. 2, pp. 245–255, 2004.

[4] Y. Kim, "Convolutional neural networks for sentence classification," arXiv preprint arXiv:1408.5882, 2014.

[5] P. Wang, J. Xu, B. Xu, C.-L. Liu, H. Zhang, F. Wang, and H. Hao, "Semantic clustering and convolutional neural network for short text categorization." in *ACL (2)*, 2015, pp. 352–357.

[6] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, "Short text classification in twitter to improve information filtering," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2010, pp.

[7] M. Chen, X. Jin, and D. Shen, "Short text classification improved by learning multi-granularity topics," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[8] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi, "Learning to classify short and sparse text & web with hidden topics from large-scale data collections," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 91–100.

[9] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions,"in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[14] A. Farhadi, M. Hejrati, M. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth, "Every picture tells a story: Generating sentences from images," *Computer vision–ECCV 2010*, pp. 15–29, 2010.

[15] G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, "Babytalk: Understanding and generating simple image descriptions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2891–2903, 2013.

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," arXiv preprint arXiv:1409.0473, 2014.

[17] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[20] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3156–3164.

[21] R. Kiros, R. Salakhutdinov, and R. Zemel, "Multimodal neural language models," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 595–603.

[22] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille, "Explain images with multimodal recurrent neural networks," arXiv preprint arXiv:1410.1090, 2014.

[23] R. Kiros, R. Salakhutdinov, and R. S. Zemel, "Unifying visual-semantic embeddings with multimodal neural language models," arXiv preprint arXiv:1411.2539, 2014.

[24] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," arXiv preprint arXiv:1312.6229, 2013.

[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv preprint arXiv:1502.03167, 2015.

[26] M. Hodosh, P. Young, and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics," *Journal of Artificial Intelligence Research*, vol. 47, pp. 853–899, 2013.

[27] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4566–4575.

[28] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

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