Intelligent classification of breast cancer based on deep learning

Yuhang Liao¹, Yulan Peng², Dongquan Liu¹,⁎ and JingYan Liu²,⁎

¹School of Computer Science, Sichuan University, Chengdu, Sichuan, 610000, China
²Department of Ultrasonography, West China Hospital, Sichuan University, P.R China

⁎dongcliu@163.com, ⁎jingyan.liu@foxmail.com

Abstract. How to use artificial intelligence to assist physicians in analyzing complex breast cancer medical data has become an urgent problem to be solved. From the perspective of auxiliary diagnosis, we propose a method to classify breast cancer grades by cases. Based on the Chinese electronic medical records of breast cancer provided by the Department of Ultrasound at West China Hospital, we compared neural network methods, including FASTTEXT, TEXTCNN, TEXTRNN. And proposed the N-LM-ATT model. N represents the n-gram feature, LM represents the Long Short-Term Memory (LSTM) network, and ATT represents the Attention Mechanism (AM). This article focuses on the classification of Chinese breast cancer text data based on natural language processing (NLP) method. By comparing the above methods, the N-LM-ATT model achieved the best performance on the breast cancer ultrasound dataset of West China Hospital.

1. Introduction
Breast cancer is a disease that plagues women all over the world and often causes great harm to women. Early diagnosis and treatment of breast cancer can usually reduce this torture. We hope to help doctors and patients make correct diagnosis and treatment at an early stage. The volume of breast medical data is large and complicated, and the description text is difficult to be unified and standardized due to different personal habits of doctors. It takes a lot of time to process it manually, even for experienced doctors. We propose a neural network-based text classification method that can assist doctors in making basic judgments on breast cancer cases with the above problems, thereby reducing processing time and helping medical resource allocation.

As a core step of NLP, text classification is widely used in email analysis and sentiment classification. In this regard, people have proposed strategies based on machine learning or deep learning. In the late 1990s, due to the increase in the number of Internet texts, along with the research of machine learning, a strategy combining statistical learning methods emerged. The main approach in this period was artificial feature engineering combined with shallow classification models. The entire classification model was disassembled into two parts: feature engineering and classifiers. But they all have their own flaws. For example, the traditional rule method requires a lot of time to formulate rules, and the bag-of-words model often causes dimensional disasters. With the development of computers and the improvement of computing power, neural networks have returned to people's field of vision. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have gradually become popular automated text processing strategies.
2. Related Work

Deep learning based on neural networks has achieved great success in NLP, including distributed representation, machine translation, and classification. Recently, the neural network-based NLP method usually divides the sentence into words or characters, and converts each word or character into a corresponding vector sequence, so that the problem is transformed into encoding these sequences. There are three common strategies, RNNs, CNNs, and Attention mechanisms. (Kim et al., 2014[1]) By setting different convolution kernels to capture different receptive fields, CNNs can encode local features better; RNNs can model time series, which makes it a good capture of text features; Attention mechanism (Ashish Vaswani et al., 2017[2]) is different from RNNs that it can directly obtain global information.

RNNs, modeling time series, capture long-distance dependence. It can convert text of any length into a vector, and CBOW can also do it, but CBOW will ignore the time information in the sequence, which makes RNNs very suitable for modeling natural language. At the same time, there are a variety of parameters to ensure that its "memory" mechanism works correctly, allowing us to go deep into the semantic level. For example, (Iñigo Jauregi Unanue et al. 2017[3]) they use bidirectional LSTM to capture advanced features in text. However, because the traditional RNN can only memorize part of the sequence, it performs far inferior to the short sequence on the long sequence, resulting in a decrease in accuracy once the sequence is long. In order to solve these, RNNs have appeared many variants.

Thanks to the ability to capture local dependencies in space, CNNs are widely used in the fields of CV and NLP. For the NLP field, this is a way based on n-gram coding. By setting different sizes of convolution kernel sizes, CNNs can learn the overall features through the pooling layer to improve the generalization ability of the model. For example, (Chen Y, 2015[4]) by learning the features of sentences, each feature is mapped into multiple parts, and the maximum value of each part is retained. Compared with the traditional maximum pooling, dynamic multi-pooling can retain more valuable information without losing the maximum pooling value. And (N. Kalchbrenner et al., 2015[5]) retain edge information through a wide convolution, perform k-max-pooling on the convolution result, take out the largest K values, and retain relative position information, where the k value is dynamically learned of.

Different from traditional RNNs and CNNs, the Attention model. It is inspired by the human visual system. Imagine we are looking at a picture, and a certain area always gets our great attention. In recent years, it has proven to be a successful method in the NLP field. The attention model (AM) was first applied to machine translation (Bahdanau et al., 2014[6]). They proposed a method that allows the model to automatically (soft) search for parts of the source sentence that are related to the predicted target word without having to explicitly form these parts as Hard segment to extend this range. In the process of processing text, pay more attention to certain words that are more important to the topic task. And (S. Chaudhari et al., 2019[7]) AM can also improve the interpretability of neural network methods, and overcome the lack of information and computational overhead caused by RNNs due to too long input.

There still are other ways, FastText (Mikolov et al., 2016[8]) which treats documents as continuous words; RCNN (Siwei Lai et al., 2015[9]) which combines the advantages of CNNs and RNNs; DPCNN (Rie Johnson and Tong Zhang, 2017[10]), multiple convolutional layers are superimposed to obtain high-level semantic features of the text, and Shortcut connections (Kaiming He et al., 2015[11]) are used to alleviate the disappearance of gradients.

On the other hand, learning excellent word vectors is also one of the important tasks. Learning distributed representation word vectors through neural networks requires additional professional domain knowledge and can be adjusted according to different task topics. It is common to use pre-trained distributed representation of word vectors. Word2Vec, (Mikolov et al., 2013[12]) is widely used to generate word vectors. Previously, traditional NLP methods regarded words as atomic units—there was no concept of similarity between words. They proposed a simple model structure that can calculate high-dimensional word vectors from a large number of data sets, and these word vectors can retain the relationship information between words. And researches shows that using a pre-trained corpus will improve many NLP tasks.
3. N-LM-ATT model for character-level classification

The network structure is shown in Figure 1. There are five components: Input layer, Embedding layer, LSTM layer, Attention layer and Output layer. Input layer, input the sentence of this model; embedding layer, map each character and n-gram feature to low-dimensional space, LSTM layer, extract semantic features, Attention layer, learn a weight matrix, integrate the character features of each step into the sentence-level feature vector, output layer, get the final sentence-level feature vector, and use it for classification. The following subsections each describe how we can add n-gram features to the embedding layer, LSTM-encoded sentence-level text features, and AM to capture the weight of different words within a sentence.

![Figure 1. N-LM-ATT Architecture.](image)

3.1. Input layer

The input data is one-hot encoding of the original text.

3.2. Embedding layer

We pre-process the original text at the character level and combine characters into different n-grams. Use a window of size $n$ to slide on the sentence. For each $j$ in the sentence, you will get a token of length $n$, denote as: $v_j = [x_j, x_{j+1}, ..., x_{j+n-1}]$. Since the number of n-grams is much larger than word, we choose a hashing algorithm to map n-gram features into buckets, numbered from 0 to bucket-1. N-grams hashed in the same bucket share an embedding vector. And in order to reduce hash collisions, set buckets to 20000. In this section, select 2-gram features and 3-gram features.

Split the characters on the original sequence, let $x_i \in \mathbb{R}^d$ represent the d-dimensional word vector of the i-th character, $X \in \mathbb{R}^{L \times d}$, where $L$ is the length of the sentence. We randomly initialize the n-gram feature and each character to a d-dimensional real-valued vector. In order to ensure that the n-gram and the original sentence have the same length, we choose the last and last two characters of the sentence as a 2-gram and 3-gram token, respectively. Then, we superimpose the n-gram feature vector and the character vector and send it to the next layer.

3.3. Bidirectional LSTM

RNNs are mainly used to process and predict sequence data. Unlike CNNs, the nodes between the hidden layers of RNNs are connected, which can memorize the previous information, and refer to the previous information to get the output of the subsequent nodes. Given the current input $x_t$ and the output $h_{t-1}$ of the previous hidden layer, the output at the current moment can be obtained. However, due to its chain structure, standard RNNs cannot capture long-distance dependencies. In order to solve this problem, LSTM was proposed and (Ilya et al., 2014) achieved remarkable performance in the field of machine
3.4. Attention mechanism

The AM was first applied in computer vision and later borrowed from the field of machine translation. In recent years, attention-based neural networks have made remarkable achievements in various tasks. This section uses the self-attention mechanism for classification tasks. \( H \) is the high-level feature vector encoded by the LSTM layer, and \( w \) is the weight matrix obtained by learning. Multiply \( w^T \) by \( m \) and then activate it with \( \text{softmax} \) to obtain the distribution of self-attention \( a \). Finally, multiply \( H \) by \( a \) to get the final expression of the sentence. To obtain self-attention, capture the dependence between characters within a sentence, and integrate character-level features into sentence level. Where \( m \) is \( H \) activated by \( \text{tanh} \), the output of \( \text{tanh} \) is \([-1, 1]\), the gradient near zero is large, and the convergence process is faster. Compare to \( \text{relu} \), \( \text{tanh} \)'s output is smoother, and avoiding excessive output values.

\[
m = \text{tanh} (H) \tag{1}
\]
\[
a = \text{softmax} (w^T m) \tag{2}
\]
\[
r = H a^T \tag{3}
\]

3.5. Predictive classification

After passing AM, we obtain the final hidden layer state, and after sending it to the two fully connected layers, use \( \text{softmax} \) to normalize, and output the one with the largest number of labels as the predicted value. The loss function is cross entropy, \( r \) is the output, and \( \text{class} \) is the label.

\[
\text{Loss}(r, \text{class}) = - \log \left( \frac{\exp(r_{\text{class}})}{\sum \exp(r_{ij})} \right) \tag{4}
\]

4. Experiment

4.1. Data set and experimental parameters

Our data set comes from the Ultrasound Department of West China Hospital, including case data for the whole year of 2016. These data are based on standard diagnosis from pathology. Contains 10072 labeled samples. Among them, 8000 were used for training and 2072 were used for testing. We use prediction accuracy to evaluate the model. And we use 100-dimensional character vector and n-gram vector to randomly initialize the embedding layer, learning rate is 0.001, batch is 128, and dropout is 0.5.

4.2. Experimental results

In Table 1, we compare the performance of several advanced text classification methods on our data set.

| Model            | Accuracy  |
|------------------|-----------|
| FastText         | 96.01%    |
| TextCNN          | 96.63%    |
| TextRNN          | 96.47%    |
| TextRNN-ATT      | 97.02%    |
| P-N-LM-ATT       | 97.46%    |
| R-N-LM-ATT       | 97.79%    |

FastText: It uses superimposed vectors to construct a vector representing sentence features. The idea behind it is the traditional bag-of-words model, which treats sentences as a collection of words.

TextCNN: By setting different convolution kernel sizes to capture n-gram features, the width of the convolution kernel is embedding-size, so that each convolution is a word or character, ensuring the rationality of the minimum granularity.
TextRNN: Use bidirectional LSTM to capture long-distance advanced features, and output prediction results through softmax. The accuracy rate on the data set is 96.47%.

TextRNN-ATT: AM is introduced on the advanced features obtained by the LSTM layer to capture the internal dependence of the sentence and calculate the contribution of different words to the text.

In our proposed model, n-gram is added to the Embedding layer as a candidate set of text features, and optimizes the contribution of unregistered words and low-frequency words. We also compared the difference between pre-training and random initialization. Since the samples in the pre-training corpus did not cover breast cancer cases, the accuracy of the latter would be better. The accuracy rate of 97.79% reached the best performance.

5. Conclusion
In general, the n-gram feature is introduced into the input of our model. The high-level features obtained on this basis optimize the unknown words and low-frequency words of low-frequency cases in breast cancer data, and obtain the best performance compared with other classification methods. It can help the doctors to make early diagnosis, improve and optimize the utilization of medical resources, and help patients to diagnose and treat early.

Acknowledgments
This work was supported by Department of Science and Technology of Sichuan Province (Grant No. 2019YFS0126)

References
[1] Kim, Y. (2014) Convolutional neural networks for sentence classification. In: Empirical Methods in Natural Language Processing (EMNLP). Doha. pp. 1746–1751.
[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I. (2017) Attention is all you need. In: Neural Information Processing Systems. Long Beach. pp. 6000–6010.
[3] Jauregi Unanue, I., Borzeshi, E.Z., and Piccardi, M. (2017) Recurrent neural networks with specialized word embeddings for healthdomain named-entity recognition. Journal of biomedical informatics., 76:102–109.
[4] Chen, Y., Xu, L., Liu, K., Zeng, D., Zhao, J. et al. (2015) Event extraction via dynamic multi-pooling convolutional neural networks. In: ACL. Beijing. pp. 167–176.
[5] Kalchbrenner, N., Grefenstette, E., Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. In: 52nd Annual Meeting of the Association for Computational Linguistics. Baltimore. Pp: volume 1, 655–665.
[6] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. https://arxiv.org/abs/1409.0473.
[7] S. Chaudhari, G. Polatkan, R. Ramanath, and V. Mithal. (2019) An attentive survey of attention models. https://arxiv.org/abs/1904.02874?context=cs.
[8] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. (2016) Bag of tricks for efficient text classification. https://arxiv.org/abs/1607.01759.
[9] Lai, S. Xu, L. Liu, K. and Zhao, J. (2015) Recurrent convolutional neural networks for text classification. In: Twenty-Ninth AAAI Conference on Artificial Intelligence. Austin. pp. 2267–2273
[10] Johnson, R and Zhang, T. (2017) Deep pyramid convolutional neural networks for text categorization. In: ACL. Vancouver. pp: volume 1, 562–570.
[11] He, K., Zhang, X., Ren, S., Sun, J. (2015) Deep residual learning for image recognition. https://arxiv.org/abs/1512.03385.
[12] Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. https://arxiv.org/abs/1301.3781.