Research on human-vehicle interaction based on natural language processing

WEI, hu
Minister of Industry and Information Technology Equipment Industry Development Center, Beijing, 100089, China
huwei@eidec.gov.cn

Abstract. With the rapid development of intelligent networked vehicles and driverless technology, the importance of the dialogue between human and vehicle artificial intelligence has also become prominent. In the case of the driver outputting speech and converting it into text, how to correctly distinguish the driver's intention and make the correct response is the purpose of the research. Google released the first-generation BERT (Bidirectional Encoder Representations from Transformers) training model in 2018, bringing natural language processing methods to a new way of understanding. This article analyzes the current mainstream machine learning algorithms and applies them to natural language processing (NLP) to classify text data sets. The purpose is to study various machine learning models to achieve higher accuracy of text classification. Research results not only lay the foundation for improving vehicle driver interaction, but also serve as a reference for the natural language processing part of the future car networking and driverless technology.

1. Introduction
AI artificial intelligence is a vital part of intelligent networked vehicles and driverless technologies, natural language processing is the core technology of human communication with AI artificial intelligence, especially text Pre-processing method and machine learning algorithms. At present, the most effective technology in the industry for artificial intelligence natural-language processing is the release of BERT (Bidirectional Encoder Representations from Transformers) by Google in 2018. It is expected that natural language processing technologies will be developed on this basis in the next five years. The field of automotive artificial intelligence is a good technological breakthrough opportunity. This article will be developed based on BERT by google and compare its current hot natural language processing technology. This article discusses the model applicability and prediction accuracy according to the difference of feature selection and algorithms model. Natural language processing could be used in the automotive interactive system to perform more precise functional positioning of the driver's voice commands, and can also be applied to provide appropriate feedback on the driver's current emotion judgment. This purpose of this paper is that to improving the algorithms model fitting and predicting accuracy based on experiment outcome and data pre-processing methods, and developing a foundation for improving driver's interactive experience.

2. TESTING method
The experimental dataset contains over 650,000 English texts of a social network user reply and their sentiment classification items (Sentiment Classification). The specific sentiment classification items are divided into 5 levels, and each text corresponds to a sentiment classification level. The experiment will
3 machine learning models (logistic regression, Polynomial NB, linear SVC) and 2 deep learning models (text-CNN, LSTM).

2.1. Programming language and integrated development environment

The programming language used in this article is version 3.6 of Python. Experiments are conducted on the jupyter notebook integrated development environment and integrated packages such as sklearn and scipy which are used to call the algorithm model. The text feature processing divided into two different ways: N-gram word frequency and word embedding.

| Feature Set                          | Model                  |
|--------------------------------------|------------------------|
| Feature set (TF-IDF unigram and bigram) | Logistic Regression    |
|                                      | MultiomialNB           |
|                                      | LinearSVC              |
|                                      | Random Forest Classifier|
|                                      | KNN                    |
| Word embedding                       | Text-CNN (with glove)  |
|                                      | BOW with Keras         |
|                                      | LSTM                   |

2.2. Text pre-processing

We will divide the text into two feature groups according to Table 1 and process them separately to deal with different machine learning models. In the two feature groups that deal with N-gram word frequency statistics (TF-IDF) and word embedding (Word embedding), all text in the training dataset is structured first. However, the volume of the dataset is extremely large and also contain special symbols, abbreviations, and some other languages that will affect the model fitting. Therefore, it is necessary to perform a machine translation of other languages in the pre-processing stage to reduce the "noise" of the data model. Then the text is divided into basic word elements (tokens) and converted into English lowercase. At the same time, there are a large number of pause words, conjunctions, and meaningless words (words less than three English letters) in the text. These words appear frequently and will seriously affect the fit of the model training. These three types must be removed. After completing the above steps, the pure English text can be extracted. In addition, for the changes in the tense of a single English word in the text, this will cause the single word frequency to be counted separately in the word frequency-inverse text frequency statistics. We need to perform the text Stemming and Lemmatization aim to classify tokens with the same meaning. The above is the pre-processing stage.

2.3. Feature extraction

After pre-processing the text content into lexical elements (tokens). Features should be filtered and feature extracted. At present, the "bag-of-words model" is a very effective feature construction method for text mining. Considering the proportion of words in English with three yuan and above Small, so this experiment only considers unigram and bigram to be added to the bag of words. At the same time, considering that there will be high-frequency and low-frequency words in the bag of words, we need to filter the word bag. The filtering method is to filter words with a frequency higher than 95% and 0.001% of low-frequency words in the bag. The number of bag-of-words matrices is (650000, 36480).
2.4. Model training-logistic regression applicability analysis and parameter setting

In the model training stage, a total of 5 models were selected in this experiment to train the above word bag matrix. The training models are logistic regression, naive Bayes, LinearSVC, LSTM and Text-CNN. Among them, in the logistic regression model, the model learns weights and biases through the training set. Each $w_i$ multiplied by a feature $x_i$ represents the importance of this feature; the offset is also the intercept. When testing, enter a new $x$, you can calculate $z$.

$$Z = \sum_{i=1}^{m} w_i x_i + b$$

However, the range of $z$ should be $(1, 5)$, not the range of logistic regression probability $[0, 1]$. So in order to generate probability, we pass $z$ to the sigmoid function $\sigma(z)$, and then sigmoid maps the real number domain to $[0, 1]$, which is convenient for obtaining the probability to complete the discrimination of a single label. In terms of parameter setting, set $set_solver = 'lbfgs'$ to deal with the polynomial loss value (Loss), and then set $max_iter = 1000$. In addition, in order to better fit the model, a quarter of the training data set was split into the test data set, and finally 10 cross-validation was completed on the training model.

2.5. Naive Bayes Applicability Analysis

The polynomial naive Bayes classifier is suitable for classification with discrete features. It considers the conditional probabilities of the classified features in each dimension, and then combines the probabilities of the features to classify the predictions, thereby reducing the dimensions of the parameters, thereby Save time and storage space. In the parameter setting, set the additive (Laplace/Lidstone) smoothing parameter $alpha = 2$, then split a quarter of the training data set into the test data set, and finally complete 10 cross-validation.

2.6. LinearSVC applicability analysis

LinearSVC implements "one-to-one" multiple classification strategies to train multiple types of models. In the training process, samples of a certain category are classified into one category, and the remaining samples are classified into another category. $K$ SVMs are constructed from the samples of $k$ categories, and the unknown samples are classified as the samples with the maximum value of the classification function.
2.7. **CNN applicability analysis**

The neural network model uses pretrained word embeddings as input, and each word in each pretrained document (Pretrain Embedding Document) will be represented by 300 vectors. In addition, the BiLSTM model is used in this experiment to enhance the fit of the model. The BiLSTM model can predict extended words based on the context to promote the model to better understand the training text. Next, we use dropout and pooling methods to capture the important features in this article to avoid model overfitting, and then connect the tf-idfas layer with the input layer to make the training model fully activate this layer.

1. The first layer is the input layer, where we input word embeddings. If we have N words, the dimension of the vector is K, and the size of this matrix is N*K.
2. The next layer performs convolution on the word vector after multiple filtering. For example, the area size exceeds (2,3,4) words at a time. It will be aggregated into different feature groups (listed as 1).
3. Then the results of these feature maps (from the convolutional layer) are merged into a one-dimensional long feature vector. Finally, all layers will be fully connected by adding dropout and L2 normalization to the softmax layer.
4. Modify the basic text-CNN model:
   - Merge the embedded vocabulary completed with training and the original training set.
   - Add two more convolutional layers, apply batch normalization, and then connect them to the SoftMax layer.
In terms of parameter settings, two convolutional layers are added to the CNN model. In the Convolution1D() function, the first parameter is the filter size, and the second parameter is the kernel size. The third is the filling state. In the first parameter, the dimensionality of the output filter in the convolution, this value is set to 256, which means that due to the huge amount of network embedding, more filters can be provided. After batch normalization and activation function "relu", we compress it into 128 dimensions.

The second parameter is the length of the one-dimensional convolution window, which is set to \{3,4,5\} as the length of the feature mapping. Set the Dense parameter to 6 on the output parameter, which means (number of classes)+1, and calculate "5+1=6".

2.8. **LSTM applicability analysis**

The long short-term memory network (LSTM) improved from the neural convolutional network architecture is a deep learning method. It is suitable for the task of distinguishing sentences that contain the same words but in different sequences. In the LSTM model, we put the sequence string converted into a one-hot vector into the embedding layer, and then the embedding part learns weights from the order of words in each sentence. After the model is aggregated, the output layer will extract three-length numerical vectors through softmax. In the model tested in this article, we convert the text and label values into one-hot vectors. In terms of parameter settings, setting the first parameter of the LSTM() function to 100 and the second parameter to dropout=0.2 have met the text classification requirements.

![LSTM implementation principle](image)

3. **TESTING method**

The experimental procedures and parameter settings used in this article are running on the same computer. The data volume and running time should be taken into consideration to reduce the number of cross-validation. If the computer performance is satisfied, the number of verifications should be increased to ensure the model fit.

3.1. **Validation results**

The model fit is 72.4%, and the specific loss function and accuracy are as follows in the training set and test set.
The accuracy rate of LSTM in the training set reaches 70%, but the actual accuracy rate in the test set is 62.3%. The reason for the difference between the accuracy of the model and the actual classification is the excessive noise in the data set.

3.2. CNN verification results
CBB + Pertrain developed based on the BERT model achieved the highest accuracy rate in this experiment. Its accuracy rate fluctuates according to its pretrain embedding number (Pretrain Embedding Number), and the floating interval is (66.06%-70.33). Among them, when the number of pre-training embeddings is 4, its prediction accuracy is the highest (70.033%), but its training time is the longest of all models. If you need to consider other training with larger text volume in the future, you need to propose a higher level of hardware. Requirements. As shown below.

| Model | TD-IDF | Train time  | Number of pretrain embedding | Test accuracy |
|-------|--------|-------------|-----------------------------|---------------|
| CNN   | Yes    | Around 2 hours | 4                           | 70.033%       |
|       | Yes    | Around 2 hours | 3                           | 69.906%       |
|       | No     | Around 1.5 hours | 3                           | 69.367%       |
|       | No     | Around 1.5 hours | 2                           | 66.616%       |

3.3. TF-IDF feature group
The actual accuracy rates of the three models (logistic regression, naive Bayes and LinearSVG) using the TF-IDF feature set are 64.4%, 62.2%, and 65.7% respectively. The overall performance is lower than that of the neural network model using word embedding. The fundamental reason is that the model is a single level, and the complexity is lower than that of neural network learning. In addition, machine learning cannot perform a repetitive and suffix vocabulary prediction in advance of the text, making the final prediction result overall lower.

| Feature Set | Model            | Test accuracy |
|-------------|------------------|---------------|
| Feature set (TF-IDF unigram and bigram) | Logistic Regression | 0.644 |
|             | MultiomialNB     | 0.622         |
|             | LinearSVC       | 0.657         |
| Feature set (TF-IDF unigram) | LSTM             | 0.623         |
|             | Text-CNN (with glove) | 0.703        |
4. Conclusion

According to the analysis of the process described in this article, it is found that providing suitable data pre-processing and feature selection are more effectively to accuracy than model selection. Compared with the selection of model parameters, in addition, according to the scenarios and model parameters set in this paper it can be concluded that the CNN neural network model in natural language processing field, which has the highest accuracy of the text data submitted on Kaggle, which is 70.033%. This model is based on the current industry’s top technology BERT for further development and analysis. In this experiment, the deep learning model is more accurate than the machine learning model. In addition, the model developed based on BERT has better performance comparing with the version which launched from google, the prediction accuracy has more obvious advantages in terms of training time and accuracy for emotional text classification.

Reference

[1] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. Journal of Machine Learning Research, 3(Jan), 993-1022.

[2] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionally. Advances in Neural Information Processing Systems 26, 3111-3119.

[3] Miller, G. A. (1995). WordNet: a lexical database for English. Communications of the ACM, 38(11), 39-41.

[4] Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation. the 2014 Conference on Empirical Methods in Natural Language Processing, 1532-1543.

[5] Floridi L, Taddeo M. What is data ethics? Phil.Trans.R.Soc.A 373: 20160360. 2016

[6] Rokach, Lior. Maimon, Oded. Data Mining with Decision Trees; Theory and Applications. 2nd ed. World Scientific Publishing Co.

[7] Couronne, Raphael. Probst, Philipp.Boulesteix, Anne-Laure. Random forest versus logistic regression: a large-scale benchmark experiment. BMC Bioinformatics. 2018

[8] Britz, D. (2015). Implementing a cnn for text classification in tensorflow. In REINFORCEMENT LEARNING (pp. 1-5).

[9] Zhou, C., Sun, C., Liu, Z., & Lau, F. (2015). A C-LSTM neural network for text classification. arXiv preprint arXiv:1511.08630

[10] Susan, Li. (2017). Building A Logistic Regression in Python, Step by Step.

[11] Schank, Roger C.; Abelson, Robert P. (1977). Scripts, Plans, Goals, and Understanding: An Inquiry Into Human Knowledge Structures. Hillsdale: Erlbaum. ISBN 0-470-99033-3.

[12] Duan, Yucong; Cruz, Christophe (2011). "Formalizing Semantic of Natural Language through Conceptualization from Existence". International Journal of Innovation, Management and Technology. 2 (1): 37–42. Archived from the original on 2011-10-09.

[13] Mittal (2011). "Versatile question answering systems: seeing in synthesis" (PDF). International Journal of Intelligent Information and Database Systems. 5 (2): 119–142. doi:10.1504/IJIIDS.2011.038968

[14] Goldberg, Yoav (2016). "A Primer on Neural Network Models for Natural Language Processing". Journal of Artificial Intelligence Research. 57: 345–420. arXiv:1807.10854. doi:10.1613/jair.4992

[15] Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron (2016). Deep Learning. MIT Press.

[16] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144.