TEXT CLASSIFICATION METHOD OF DONGBA CLASSICS BASED ON CATBOOST ALGORITHM

Rana Kinabadi
Kharazmi University
Tehran, Iran
rkinabadi@gmail.com

Abstract

The study focuses on preserving the Dongba culture of the Naxi nationality in Yunnan Province and proposes addressing the challenge of automatic text classification using artificial intelligence. The researchers establish a Dongba text dataset based on the Dongba classic collection from the Lijiang Dongba Culture Research Institute. They utilize the catboost machine learning algorithm to classify Dongba texts according to ceremonies. The experimental results, based on 300 datasets, indicate that the catboost algorithm achieves an 87.5% classification accuracy and an 86.7% recall rate for six ritual categories. While acknowledging imperfections in the dataset, the study concludes that the catboost algorithm is effective in classifying Dongba texts, demonstrating practical application value

.*Keywords*: ***Dongba classics, text classification, catboost, machine learning***

I. Introduction

Dongba civilization, a treasure in the national culture of the world, is a significant part of the spiritual civilization. The Naxi people, located on the southwestern border of the country, have attracted numerous scholars for their outstanding achievements in poetry, painting, sculpture, music, and dance. The Dongba culture is characterized by nearly a thousand kinds of Dongba scriptures and ancient books, which are used by the Naxi primitive religion Dongba priests. These ancient books are applied to various Dongba rituals, which contain profound cultural connotations and are closely related to the lives of the Naxi people.

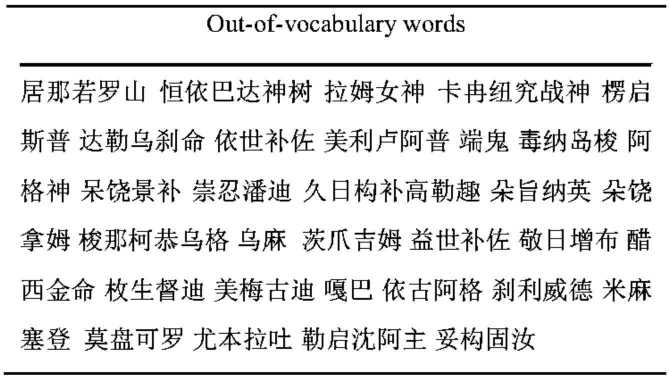
The ancient books of Dongba can be divided into blessings, disaster eliminations, funerals, divination, and more. In the process of inheriting and protecting Dongba classics, there is a need to identify and classify the types of Dongba classics. However, most cultural protection agencies lack deep processing and development and utilization technology, which is limited to shallow levels such as electronic scanning and database establishment. This paper uses machine learning algorithms and data features to identify the Dongba ritual category of the Dongba classic ancient book cataloging files and provides effective technical support for the classification of Dongba classics..

II. Preprocessing

A. *Text and Word Segmentation*

The article describes the preprocessing steps for word segmentation in Dongba text data. Initially, regular expressions are used to identify Chinese character strings, and numbers, punctuation, and paragraph characters are replaced with spaces to facilitate word segmentation. The jieba word segmentation method is then employed, known for its high accuracy and efficiency. This method involves building a prefix dictionary, creating a directed acyclic graph (DAG), and utilizing dynamic programming to calculate the maximum probability path for segmentation. The article addresses the presence of out-of-vocabulary (OOV) words in Dongba classics, such as the names of gods and ghosts. To handle this, the study customizes and adds these OOV words to the segmentation dictionary, aiming to enhance accuracy and efficiency in word segmentation. The customized OOV words are listed in Table 1.

Table 1 Out-of-vocabulary words



Out-of-vocabulary words
居那若罗山 恒依巴达神树 拉姆女神 卡冉纽究战神 楞启斯普 达勒乌刹命 依世补佐 美利卢阿普 端鬼 毒纳岛梭 阿格神 呆饶景补 崇忍潘迪 久日构补高勒趣 朵旨纳英 朵饶拿姆 梭那柯恭乌格 乌麻 茨爪吉姆 益世补佐 敬日增布 醋西金命 枚生督迪 美梅古迪 嘎巴 依古阿格 刹利威德 米麻塞登 莫盘可罗 尤本拉吐 勒启沈阿主 妥构固汝

III. Feature extraction

Feature extraction in text classification is the process of converting raw text data into a numerical representation suitable for machine learning algorithms. This enables algorithms to understand the content and meaning of the text for classification tasks.

A. *Importence*

- Raw text is unstructured and difficult for algorithms to process directly.
- Feature extraction helps capture relevant information like word presence, frequency, relationships, and semantic meaning.
- This information is then used to train classifiers to distinguish between different categories of text.

B. *TF-IDF(Term Frequency – Inverse Document Frequency)*

- TF-IDF (Term Frequency-Inverse Document Frequency) is a feature extraction technique used in various applications like information retrieval and text classification.

- TF-IDF is a widely used and well-understood feature extraction technique that performs well in various text analysis tasks.

C. *Word2Vec Model : Capturing Semantic Word Relationships*

Google's Word2Vec introduced a groundbreaking method for generating word vectors that capture semantic relationships and similarities between words. Trained on vast amounts of unlabeled text data, Word2Vec uses the Continuous Bag-of-Words (CBOW) model to predict the target word based on its surrounding context..

IV. Python Code Explanation

The Dongba dataset is currently unavailable; therefore, the Chinese digit MNIST dataset is utilized as an alternative..

A. *Data Preparation*

- Loads the Chinese Digit Mnist dataset from a CSV file.
- Converts the 'label' column to strings.
- Splits the data into input features (X) and target labels (y), followed by a train-test split.

B. *Data Processing*

- Defines two feature extraction methods: Tf-Idf and Word2Vec.
- tfidf_feature_extraction: Uses TfidfVectorizer for text data.
- word2vec_feature_extraction: Tokenizes and pads sequences, then trains a Word2Vec model for generating embeddings

C. *Classifiers*

- Defines three classifiers: CatBoost, K-Nearest Neighbors (KNN), and Decision Tree

Possible Issues

1. **Data Preprocessing:**

   - The **preprocess_text** function may not handle all edge cases or special characters appropriately. It might be beneficial to conduct a more comprehensive text preprocessing, including handling emojis, URLs, and special characters.

2. **Data Imbalance:**

   - The code does not address potential issues related to class imbalance in the sentiment labels. If one sentiment class significantly outweighs the other, it can affect the model's performance.

3. **Tokenization and Padding:**

   - The choice of a fixed sequence length (maxlen=100) for padding might lead to information loss for longer texts. Dynamic padding or truncation strategies could be explored.

4. **Hyperparameter Tuning:**

   - The code lacks a comprehensive hyperparameter tuning process. Optimizing hyperparameters, such as learning rate, batch size, or the number of epochs, could improve model performance.

5. **Model Evaluation:**

   - While the code evaluates the model accuracy, additional metrics such as precision, recall, and F1 score could provide a more comprehensive understanding of the model's performance.

6. **Overfitting:**

   - The model's performance on the training set and validation set should be monitored for signs of overfitting. Techniques like dropout layers or regularization may be applied to mitigate overfitting.

7. **Documentation and Comments:**

   - The code lacks detailed comments and documentation, making it challenging for others (or even the author) to

UNDERSTAND THE RATIONALE BEHIND SPECIFIC DECISIONS OR STEPS.

Future Improvment

1. **Comprehensive Text Preprocessing:**
    - Enhance the **preprocess_text** function to handle a broader range of text preprocessing tasks, including addressing special characters, handling emojis, and removing URLs.
2. **Addressing Class Imbalance:**
    - Check for class imbalance in the sentiment labels and consider techniques such as oversampling, undersampling, or using weighted loss functions to address the imbalance.
3. **Dynamic Padding or Truncation:**
    - Instead of using a fixed sequence length, consider using dynamic padding or truncation based on the distribution of text lengths in the dataset.
4. **Experiment with Different Model Architectures:**
    - Explore alternative model architectures, such as Bidirectional LSTM or CNN layers, to identify the most effective structure for sentiment analysis on your specific dataset.
5. **Hyperparameter Tuning:**
    - Conduct hyperparameter tuning to optimize key model parameters, including learning rate, batch size, and the number of epochs. Techniques like grid search or random search can be employed.
6. **Incorporate Additional Evaluation Metrics:**
    - Evaluate the model using additional metrics such as precision, recall, and F1 score to gain a more comprehensive understanding of its performance, especially in the context of imbalanced datasets.
7. **Monitor for Overfitting:**
    - Introduce techniques to monitor and prevent overfitting, such as adding dropout layers or applying regularization.
8. **Documentation and Comments:**
    - Add detailed comments and documentation throughout the code to explain the rationale behind specific decisions, steps, and parameters. This enhances code readability and facilitates collaboration.
9. **Handling OOV Words:**
    - Continuously update the custom list of out-of-vocabulary (OOV) words based on the evolving dataset. Consider using pre-trained word embeddings to capture semantic information for OOV words.
10. **Handling Unknown Classes:**
    - Implement a mechanism to handle unknown classes in the testing set gracefully. This could involve predicting a special class for unknown instances or employing zero-shot learning techniques.
11. **Regular Model Evaluation:**
    - Regularly evaluate the model's performance on new data to ensure its effectiveness over time. This can involve monitoring performance metrics and retraining the model as needed.
12. **Consider Pre-trained Models:**
    - Explore the use of pre-trained language models (e.g., BERT, GPT) that have been trained on large corpora. Fine-tuning such models on your sentiment analysis task may lead to improved performance.

References

[1] [1]Shiying, Y., Hong, W.: 'Analysis of the inheritance status
[2] and existing problems of Dongba culture', Youth and Society,
[3] 2014, 21,326-327
[4] [2] Nishida, T.: 'Living hieroglyphs---Naxi culture', (New
[5] Book of Zhong Gong, 1966)
[6] [3] Ming, G.: 'The Naxi Dongba Ancient Books', Hubei
[7] Archives., 2015,(03),50
[8] [4]Bowen, Z., Lingjiao, W., Hua, G.: 'Weighted Naive
[9] Bayesian Text Classification Algorithm Based on Poisson
[10] Distribution', Computer Engineering.,2019, 1-7
[11] [5]Yong, L., Yanyun, X.: 'Research and application of text
[12] classification based on improved random forest algorithm',
[13] Computer System Applications., 2019, (5),220-225
[14] [6]Weiyin, G., Li, W.: 'Text classification based on
[15] convolutional neural network and XGBoost',
[16] Communication Technology. 2018, (10), 2337-2342
[17] [7]Ruixuan, L., Jingjing, X., Yi Z., et al.:' PKUSEG: A
[18] Toolkit for Multi-Domain Chinese Word Segmentation',
[19] Arxiv. 2019
[20] [8]You, Y., Yu, F., Xiaoping, W.: 'A review of Chinese text
[21] classification methods', Journal of Network and Information
[22] Security. 2019, (5),1-8
[23] [9] Jiang, B.: 'Chinese phonetic retrieval method based on
[24] stop word processing', Harbin: Harbin Institute of
[25] Technology. 2008
[26] [10]Yao, H., Shunmiao, Z.: 'Research on Improving the
[27] Performance of Central Classification by Using Unmarked
[28] Documents', Computer Knowledge and Technology:
[29] Academic Exchange. 2007, (16),1125-1126
[30] [11]Mingxia, G., Jingwei, L.: 'Chinese short text
[31] classification method based on word2vec word model',
[32] Journal of Shandong University (Engineering Science
[33] Edition), 2019, 49,(02), 34-41
[34] [12]Ming, T., Lei, Z., Xianchun, Z.: 'A document vector
[35] representation based on Word2Vec', Computer Science, 2016,
[36] 43,(06), 214-217+269

[37] [13]Fulin, X., Yihao, D., Xiaosheng, T.: 'The core
[38] architecture of Word2vec and its application', Journal of
[39] Nanjing Normal University (Engineering Technology
[40] Edition), 2015, 15,(01),43-48
[41] [14]MIKOLOV, T., SUTSKEVER, I., CHEN, K., et al.:
[42] 'Distributed representations of words and phrases and their
[43] compositionality', Advances in Neural Information
[44] Processing Systems, 2013, 26, 3111-3119
[45] [15]Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013).
[46] 'Efficient estimation of word representations in vector space',
[47] arXiv preprint arXiv:1301.3781.
[48] [16]'Word2vec Code',
[49] http://word2vec.googlecode.com/svn/trunk/, accessed 18
[50] September 2015
[51] [17]Lian, Z.: 'The working principle and application of
[52] Word2vec', Science and Technology Information
[53] Development and Economy, 2015, 25,(2), 145-148Y. Yorozu, M.