Postprint: A Text Classification Method Integrating Text Graph Convolution and Ensemble Learning

Authors: Zhou Xuanlang, Qiu Weigen, Zhang Lichen

Date: 2022-05-10T11:22:57Z

Abstract

To improve text classification accuracy and address the issue of insufficient utilization of node features in text graph convolutional neural networks, a novel text classification model is proposed that intrinsically integrates the advantages of text graph convolution and Stacking ensemble learning methods. The model first employs text graph convolutional neural networks to learn global representations of documents and words as well as syntactic structure information of documents, then utilizes ensemble learning to perform secondary learning on the features extracted by text graph convolution, thereby compensating for the insufficient utilization of node features in text graph convolution and enhancing the accuracy of single-label text classification and the generalization capability of the entire model. To reduce the time consumption of ensemble learning, the K-fold cross-validation mechanism is removed from the ensemble learning process. The fusion algorithm establishes the connection between text graph convolution and Stacking ensemble learning methods, achieving classification performance improvements of over 1.5%, 2.5%, 11%, 12%, and 7% on the R8, R52, MR, Ohsumed, and 20NG datasets respectively compared to traditional classification models, and demonstrating superior performance among classification algorithms in the same domain.

Full Text

Preamble

Text Classification Combining Text Graph Convolution and Ensemble Learning

Zhou Xuanlang†, Qiu Weigen, Zhang Lichen
(Faculty of Computer, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: To improve the accuracy of text classification and address the problem of insufficient utilization of node features by text graph convolutional neural networks, this paper proposes a novel text classification model that intrinsically integrates the advantages of text graph convolution and Stacking ensemble learning. The model first learns global representations of documents and words, as well as grammatical structure information of documents, through a text graph convolutional neural network. It then performs secondary learning on the features extracted by text graph convolution via ensemble learning to compensate for the insufficient utilization of node features, thereby enhancing single-label text classification accuracy and the overall generalization capability of the model. To reduce the time consumption of ensemble learning, the k-fold cross-validation mechanism is removed. The fusion algorithm establishes a connection between text graph convolution and Stacking ensemble learning methods, achieving classification improvements of over 1.5%, 2.5%, 11%, 12%, and 7% on the R8, R52, MR, Ohsumed, and 20NG datasets respectively compared with traditional classification models. This method demonstrates superior performance when compared with classification algorithms in the same field.

Key words: text representation; text classification; text GCN; ensemble learning; fusion model

0 Introduction

In the era of big data, network text data is growing exponentially, making scientific management and organization of this data increasingly important. Consequently, numerous text processing methods have emerged. Text classification represents one of the most important research areas in natural language processing, with extensive applications in spam detection, news filtering, computational phenotyping, opinion mining, sentiment analysis, and document organization.

Text classification methods can be divided into traditional approaches and deep learning approaches. Traditional methods primarily employ machine learning techniques to study text representation and classification. Conventional text feature extraction methods such as n-grams produce inadequate text representations that lack word order relationships, limiting text representation flexibility and resulting in low classification accuracy due to their reliance on single classifiers. Deep learning-based text representation methods, such as those using Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) based on BiLSTM, learn local continuous word sequences for text representation, enabling more flexible text representation and improving classification effectiveness. However, these methods cannot capture syntactic structure information or global information, which restricts classification performance. Moreover, CNN and RNN models are limited to Euclidean-structured data and require additional processing for non-Euclidean data like text.

With the advancement of deep learning, graph neural networks have attracted increasing attention. Researchers have found that graph neural networks are

particularly suitable for processing non-Euclidean structured data such as text. The Text Graph Convolutional Network (Text GCN) model, for instance, can automatically learn embeddings for words and documents during training and integrate textual structural information to enhance representation capability. Nevertheless, graph neural network models do not fully utilize the features learned by the neural network for final classification.

To address these issues and improve text classification performance, this paper proposes a novel text classification model called TGCN-S (Text GCN-Stacking). This model employs Stacking ensemble learning to fit and train features obtained from text graph convolution, thereby solving the problem of insufficient feature utilization in text graph convolution and improving both classification effectiveness and model generalization capability. To accelerate ensemble learning, the cross-validation mechanism is removed. The effectiveness of this model is validated through experiments on the R8, R52, MR, Ohsumed, and 20NG datasets.

In summary, this paper proposes the TGCN-S model with the following main contributions: (1) We utilize Text GCN to obtain global information and structural information from text, addressing the inability of traditional models to capture textual structure and thereby enhancing feature expression. (2) We optimize the Stacking ensemble learning module by removing k-fold cross-validation, which reduces time consumption while maintaining classification efficiency. Replacing the softmax classifier with a Stacking ensemble learning classifier effectively solves the problem of insufficient feature utilization in text graph convolution and improves the classification performance and generalization capability of the entire model. (3) By integrating the advantages of text graph convolution and ensemble learning, we propose the novel TGCN-S model, which significantly improves text classification accuracy.

1 Related Work

1.1 Traditional Text Classification

Traditional text classification methods include Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Random Forest (RF), which primarily focus on text representation and corresponding algorithms such as bag-of-words and n-grams. The bag-of-words method divides documents into word collections and determines their frequency, but the final representation is independent of word order, resulting in loss of syntactic characteristics and word correlations. While n-grams can capture word correlations, they ignore syntactic features and lack flexibility, leading to inadequate text representation and loss of global information.

1.2 Deep Learning-Based Text Classification

Currently, most text classification methods are based on deep learning, with representative models including CNN for sentence classification, RNN based on

Bidirectional LSTM (BiLSTM), and BERT. Kim (2014) proposed CNN-based sentence classification, applying one-dimensional convolutions to text with good accuracy. Liu et al. (2016) applied LSTM to text classification to learn text representations while preserving longer word sequences. Devlin et al. (2018) introduced BERT, a pre-trained language representation model trained on large corpora that captures longer dependencies between words. While these models address the representation limitations of traditional methods, they fail to capture structural and global information. CNN and RNN focus on local continuous word sequences and cannot obtain global co-occurrence information or structural information, and they are limited to Euclidean-structured data.

As deep learning evolves, Graph Neural Networks (GNN) have gained attention. GNN is not only computationally efficient through parameter sharing but also well-suited for the non-Euclidean structure of word relationships in text, achieving breakthroughs in machine learning. GNN can extract multi-scale local spatial features and combine them into high-level features. Through graph embedding, GNN learns low-dimensional vector representations of nodes, edges, and subgraphs, eliminating the need for manual network structure design and improving flexibility. Cai et al. demonstrated that GNN can effectively handle tasks with rich relational structures while preserving global information. Kipf and Welling simplified GNN and proposed Graph Convolutional Networks (GCN), which capture high-order neighborhood features to improve text classification accuracy. Yao et al. applied GCN to text classification with the Text GCN model, constructing large heterogeneous graphs with sentences and words as nodes to learn embeddings and capture global word information and overall text structure.

Currently, both traditional and deep learning-based text classification methods use single classifiers (e.g., softmax) after feature extraction, which cannot correct classification errors. Ensemble learning combines multiple weak classifiers into a strong classifier, effectively addressing the limitations of single classifiers. Ensemble learning includes Boosting, Bagging, and Stacking algorithms, with Stacking offering superior flexibility and extensibility. While Stacking can classify text efficiently, its performance depends heavily on the input text features.

Based on these considerations, this paper proposes TGCN-S, a text classification method that fuses text graph convolution and Stacking ensemble learning. This approach uses text graph convolution for feature extraction and employs ensemble learning to compensate for insufficient feature utilization, thereby improving classification accuracy and model generalization. To reduce fitting time, we remove the cross-validation mechanism from Stacking.

2 Proposed Algorithm

Our method proposes a novel text classification algorithm called TGCN-S by fusing text graph convolution and Stacking ensemble learning. This model combines the advantages of both approaches to address insufficient feature utiliza-

tion in text graph convolution, improving classification accuracy and model generalization. To reduce time consumption, we remove the cross-validation mechanism from Stacking, enhancing fitting speed and classification efficiency.

2.1 TGCN-S Model Structure

The TGCN-S model is illustrated in Figure 1. The model consists of two components: feature extraction and Stacking ensemble classification. TGCN-S connects Text GCN and Stacking, using features extracted by Text GCN as input to Stacking. The Text GCN classification results are concatenated with the first-layer Stacking classification results as input to the second Stacking layer, forming residual connections. This skip-connection approach enhances the relationship between the two models and strengthens feature expression for the second Stacking layer. The final classification result is obtained through the second layer.

In Figure 1, black nodes represent documents, white nodes represent words, solid lines represent document-word connections, and dashed lines represent word-word connections. The adjacency matrix computed from this heterogeneous text graph serves as input to Text GCN.

The general Text GCN uses softmax directly on GCN-extracted features for classification as final output, which does not fully utilize the learned features. Our TGCN-S model integrates Stacking ensemble classification with Text GCN advantages. During softmax classification of GCN features, Stacking ensemble learning performs secondary fitting on the features learned by GCN through various base classifiers, followed by fusion classification to obtain the final text classification result.

Unlike traditional Stacking, the TGCN-S ensemble learning component consists of a base classifier layer and a fusion layer. The first base classifier layer comprises five base classifiers. The second fusion layer directly uses classification results from base classifiers and integrates Text GCN output results (as shown in Equation (6)), forming skip connections. These connections strengthen the relationship between text graph convolution and Stacking while incorporating Text GCN prediction effectiveness into the second layer. To reduce time consumption, we remove Stacking' s cross-validation mechanism to improve fitting speed.

2.2 Feature Extraction

We use Text GCN as the feature extractor for the first model component. During graph construction, words and documents serve as graph nodes. Document-word connection weights are represented by TF-IDF, while word-word connection weights use Pointwise Mutual Information (PMI). PMI is calculated as:

$$PMI(i,j) = \log \frac{P(i,j)}{P(i)P(j)}$$

where $P(i,j)$ is the probability that nodes i and j co-occur in a sliding window, $P(i)$ is the probability of node i appearing in a sliding window, and N is the total number of sliding windows. The edge weight A_{ij} between nodes i and j is defined as:

$$A_{ij} = \begin{cases} PMI(i,j) & \text{if } i,j \text{ are words and } PMI(i,j) > 0 \\ TFIDF_{ij} & \text{if } i \text{ is a document and } j \text{ is a word} \\ 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

where $TFIDF_{ij}$ represents the TF-IDF value of word j in document i. Positive PMI values indicate high semantic correlation between words in the corpus, while negative values indicate low or no correlation. During heterogeneous graph construction, edges are only added between node pairs with positive PMI. The weighted graph is then fed into a simple two-layer GCN for learning. The second GCN layer produces word-document embeddings with dimensions equal to the number of label categories.

The extracted features Z can be calculated using:

$$Z = \text{ReLU}(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}XW_0)$$

where $\tilde{A} = A + I_N$ is the n-order adjacency matrix with added self-connections, I_N is the n-order identity matrix, and n is the number of vertices. $\tilde{D}$ is the degree matrix of $\tilde{A}$ with $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. X is the feature matrix composed of features from n nodes, and W_0, W_1 are trainable weight matrices specific to the first and second layers respectively. ReLU serves as the activation function between layers.

Finally, node embeddings are passed through a softmax function to obtain temporary classification output Y:

$$Y = \text{softmax}(Z)$$

2.3 Ensemble Learning Component

The second component of TGCN-S is Stacking ensemble learning. Traditional Stacking models train multiple base classifiers $C_k(k = 1, 2, \ldots, m)$, then use them to predict training data, obtaining training predictions $P_i(i = 1, 2, \ldots, m)$ and test predictions $p_j(j = 1, 2, \ldots, m)$. These predictions are combined to form new datasets, where predictions from all base classifiers for the same sample serve as new features.

For base classifier selection, we follow the principle of "accurate yet different" —base classifiers should exhibit diversity. Our Stacking base classifier layer employs five types: Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), k-Nearest Neighbors (kNN), and Gaussian Naive Bayes (Gaussian NB). These five classifiers are considered fundamental, with most other classification methods being improvements based on one or more of them. To control training time overhead, we limit the first layer to these five classifiers. The second-layer classifier uses voting to produce final classification results based on individual machine learning classifier predictions.

3 Experimental Results and Analysis

This section validates and analyzes the classification effectiveness of our proposed model through experiments and comparative analysis with other state-of-the-art models.

3.1 Datasets

We evaluate TGCN-S on five datasets: R8, Ohsumed, MR, R52, and 20NG.

R8 Dataset: Derived from the Reuters corpus with 8 categories, containing 5,485 training documents and 2,189 test documents.

Ohsumed Dataset: A medical literature database maintained by the National Library of Medicine. We extract single-category data, resulting in 3,357 training documents and 4,043 test documents (7,400 total).

MR Dataset: A movie review dataset where each review contains only one sentence, with 5,331 positive and 5,331 negative reviews.

R52 Dataset: Also from the Reuters corpus with 52 categories, containing 6,532 training and 2,568 test instances.

20NG Dataset: A newsgroup dataset with 20 categories, containing 11,314 training documents and 7,532 test documents.

As text data cannot be directly used for model training, preprocessing is required. After preprocessing, the statistical information for each dataset is shown in Table 1, detailing training set size, test set size, vocabulary size, number of nodes, and number of classes.

3.2 Comparison with Baseline Models

We compare our model with several text classification approaches:

CNN: Convolutional neural network for text classification proposed by Kim (2014), performing sentence-level classification on pretrained word vectors.

LSTM: Long Short-Term Memory model using the last hidden state as text representation, applied to text classification by Liu et al. (2016).

Bi-LSTM: Bidirectional LSTM variant using pretrained word embeddings as input.

FastText: A simple yet effective text classification model proposed by Joulin et al. (2017), using averaged word n-gram embeddings fed into a linear classifier.

Text GCN: Text graph convolution proposed by Yao et al. (2019), constructing large heterogeneous text graphs based on word co-occurrence and document-word relationships for classification.

Table 2 presents the prediction accuracy of each model across datasets. Our TGCN-S model achieves the best test accuracy on all five datasets with varying degrees of improvement. For R8, TGCN-S outperforms the best Text GCN by 1.5 percentage points and other algorithms by at least 2 percentage points. For R52, it surpasses other models by over 2.5 percentage points, improving CNN by 13 percentage points. On Ohsumed, TGCN-S exceeds Text GCN by 12 percentage points and other models by over 20 percentage points. For MR, TGCN-S improves upon Text GCN by nearly 11 percentage points and other models by approximately 14 percentage points. On the larger 20NG dataset, TGCN-S outperforms Text GCN by 7 percentage points and other models by over 12 percentage points.

Figure 4 visually demonstrates that our proposed model consistently outperforms comparison models across all datasets. These results confirm that Stacking ensemble learning can more effectively utilize features learned by text graph convolution, improving classification performance to varying degrees.

The accuracy improvements vary across datasets due to specific characteristics. MR contains polarized reviews (e.g., “This movie has a rich story but is too scary”), and Ohsumed includes interrelated medical literature describing cases with associated medications. While graph neural networks capture global information, they cannot obtain word order features within sentences, limiting detailed feature extraction. Nevertheless, our method shows substantial improvement over single classifiers, demonstrating that the voting mechanism in Stacking effectively enhances classification even for texts with multiple polarities.

Accuracy alone cannot fully determine model quality. We therefore evaluate models using Macro-F1 and Micro-F1 scores, which consider both precision and recall. Higher values indicate better model quality and classification performance. Since Text GCN outperforms CNN, LSTM, BiLSTM, and FastText, we primarily compare Macro-F1 and Micro-F1 between Text GCN and TGCN-S-vote (Table 3). Our model achieves higher scores across all datasets, indicating superior quality and classification effectiveness.

To compare convergence, we examine accuracy per epoch until stabilization (Figure 5). Our model converges faster than Text GCN on all datasets, reaching stable states earlier while achieving higher final accuracy, further validating its effectiveness.

3.3 Cross-Validation Removal in Ensemble Learning

To simplify the ensemble learning module and improve training speed, we removed the k-fold cross-validation mechanism from all base classifiers, using only random shuffling of training and test sets. Table 4 compares results with and without cross-validation.

In Table 4, Kfoldt and KP represent time consumption and accuracy with k-fold cross-validation, while nKfoldt and nKP represent time and accuracy without it. Removing cross-validation reduces time consumption because it eliminates K-1 model fittings. Classification accuracy without cross-validation is not lower than with it, as cross-validation primarily benefits small datasets by improving generalization. For larger datasets like ours, cross-validation provides minimal benefit while significantly impacting fitting speed. Therefore, our model removes cross-validation to reduce time costs while maintaining good accuracy.

3.4 Fusion Layer Comparison in Ensemble Learning

To analyze the impact of the second-layer fusion classifier, we experimented with nine common machine learning methods as Stacking fusion classifiers: Gaussian Naive Bayes (GaussianNB), Linear Regression, Logistic Regression, Decision Tree, LightGBM, SVM, AdaBoost, Bagging, and Voting. Results on R8, R52, Ohsumed, MR, and 20NG are shown in Table 5.

Different fusion classifiers yield varying classification effectiveness. Except on Ohsumed where LightGBM slightly outperforms voting, voting achieves the highest accuracy on all other datasets, demonstrating its universality and simplicity. Therefore, our model adopts voting as the Stacking fusion classifier.

4 Conclusion

This paper proposes TGCN-S, a text classification method fusing Text GCN and Stacking ensemble learning to address insufficient feature utilization in Text GCN and improve classification accuracy. Unlike traditional single-classifier approaches or direct softmax classification of Text GCN features, TGCN-S employs Stacking ensemble learning for secondary learning of Text GCN features, removes cross-validation from base classifiers to accelerate fitting, and obtains final classification through the fusion layer. TGCN-S achieves accuracies of 98.58%, 96.04%, 88.28%, 80.90%, and 93.02% on R8, R52, MR, Ohsumed, and 20NG datasets respectively, showing substantial improvements over other models. Experimental results demonstrate high recognition effectiveness and feasibility.

Future work includes optimizing base classifier parameters (currently set empirically) to further improve classification performance. Additionally, since graph convolution lacks word order relationships in sentences, enriching text feature representation remains an important research direction.

References

[1] Kowsari, Meimandi J, Heidarysafa, et al. Text Classification Algorithms: A Survey [J]. Information, 2019, 10 (4): 150.

[2] Li Qian, Peng Hao, Li Jianxin, et al. A Survey on Text Classification: From Shallow to Deep Learning [J/OL]. 2020. [2022-04-15]. https://doi.org/10.48550/arXiv.2008.00364.

[3] Kim, Yoon. Convolutional Neural Networks for Sentence Classification [C]. EMNLP. 2014. (2014-09-03) [2022-04-15]. https://doi.org/10.48550/arXiv.1408.5882.

[4] Jin Chen, Li Weihua, Ji Chen, et al. Bi-directional Long Short-term Memory Neural Networks for Chinese Word Segmentation [J]. Journal of Chinese Information Processing, 2018, 32 (02): 29-37.

[5] Liu Pengfei, Qiu Xipeng, Huang Xuanjing. Recurrent Neural Network for Text Classification with Multi-Task Learning [J]. Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016, 2873-2879.

[6] Zhou Jie, Cui Ganqu, Hu Shengding, et al. Graph neural networks: A review of methods and applications [J]. AI Open, 2020, 1: 57-81.

[7] Yao Liang, Mao Chengsheng, Luo Yuan. Graph Convolutional Networks for Text Classification [J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2019, 33: 7370-7377.

[8] Cai Hongyun, Zheng Vincent W, Chang Chenchuan. A Comprehensive Survey of Graph Embedding: Problems, Techniques and Applications [J]. IEEE Transactions on Knowledge & Data Engineering, 2018, 30 (9): 1616-1637.

[9] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding [J]. arXiv preprint arXiv:1810.04805, 2018.

[10] Kipf, T, & Welling, M. Semi-Supervised Classification with Graph Convolutional Networks [C]. ICLR 2017. (2017-02-22) [2022-04-15]. https://doi.org/10.48550/arXiv.1609.02907.

[11] Mehrotra K G, Mohan C K, Huang H, et al. Ensemble Methods [G]// Terrorism, Security, and Computation. Terrorism, Security, and Computation, 2017: 135–152.

[12] Xu Jiwei, Yang Yun. Integrated Learning Methods: Research Review [J]. Journal of Yunnan University: Natural Sciences Edition, 2018, 40 (06): 1082-1092.

[13] Ran Yaxin, Han Hongqi, Zhang Yunliang, et al. Large-scale Text Hierarchical Classification Method based on Stacking Ensemble Learning [J]. Information Theory and Practice, 2020, 43 (10): 171-176+182.

[14] Wu Dangping, Zhang Zhonglin, Cao Tingting. Research on Stability Classifier Combination Model Based on Stacking Strategy [J]. Small Microcomputer System, 2019, 40 (05): 135-139.

[15] Joulin A, Grave E, Bojanowski P, et al. Bag of Tricks for Efficient Text Classification [C]// Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. 2017.

[16] Ke G, Meng Q, Finley T, et al. Lightgbm: A highly efficient gradient boosting decision tree [J]. Advances in neural information processing systems, 2017, 30.

[17] Rehman Javed A, Jalil Z, Atif Moqurrab S, et al. Ensemble adaboost classifier for accurate and fast detection of botnet attacks in connected vehicles [J]. Transactions on Emerging Telecommunications Technologies, 2020: e4088.

[18] Wang Qi, Luo Zhihao, Huang Jincai, et al. A Novel Ensemble Method for Imbalanced Data Learning: Bagging of Extrapolation-SMOTE SVM [J]. Computational Intelligence and Neuroscience, 2017, 2017: 1827016.

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