Research on Detection Methods for Text Generated by Large Language Models Based on Multi-Model Ensemble

Liang Tian[1,3,*], **Nan Jiang**[2,4]

[1]Business-intelligence of Oriental Nations Corporation Ltd, Beijing, China
[2]Goodwill E-Health Info Corporation Ltd, Beijing, China

[3]todd841026@163.com
[4]nancy.gooogle@gmail.com
*corresponding author

Abstract. The rapid development of Large Language Models (LLMs) has made their generated text almost indistinguishable from human writing, posing significant challenges to traditional human-machine recognition techniques. This paper proposes a detection method based on multi-model ensemble to accurately identify text generated by LLMs. Firstly, a large-scale, diverse, and heterogeneous dataset is constructed, covering student writings and texts generated by models such as GPT-3, GPT-2, CTRL, and XLM. Then, a multifaceted detection framework integrating linear models, deep learning models, and pre-trained language models is designed. The linear model utilizes an argumentative essay dataset (DAIGT V2 Train Dataset) similar in distribution to the competition dataset, combined with adaptive BPE tokenization, N-Gram, and TF-IDF features. It employs Multinomial Naive Bayes and SGDClassifier to train classifiers that capture shallow statistical features of the text. The deep learning model fine-tunes the DeBERTa-v3-small model on large-scale datasets (Pile, Ultra, Human vs. LLM Text Corpus) to learn deep semantic representations of the text. The pre-trained language model introduces a fine-tuned DistilRoBERTa model, enhancing detection capabilities using third-party datasets. Finally, the above models are integrated through a weighted average strategy, significantly improving the generalization and robustness of the detection results. Experimental results show that this method achieved a score of 0.967466 in the Kaggle competition, earning a silver medal and outperforming any single model. The study demonstrates the effectiveness of multi-source data and multi-model ensemble in detecting LLM-generated text, providing new ideas and practical references for research in this field.

Keywords: Large Language Models, text detection, heterogeneous dataset, deep learning.

1. Introduction

The rapid development of Large Language Models (LLMs) in recent years has continuously improved the quality of text generation, posing new challenges to traditional human-machine recognition techniques. LLMs like GPT-3 and BERT [1], through pre-training on massive corpora, can generate fluent and coherent text that is almost indistinguishable from human writing. LLMs have achieved great success in tasks such as text continuation, dialogue generation, and question-answering systems, showing broad application prospects.

However, the widespread application of LLMs has also brought a series of new problems. Firstly, the high-quality text generated by LLMs makes traditional human-machine recognition methods based on shallow features ineffective, demanding higher requirements for detection techniques. Secondly, the misuse of LLMs in the educational field may exacerbate academic dishonesty; in the news domain, it may accelerate the spread of misinformation, causing adverse social impacts. Therefore, accurately identifying text generated by LLMs is of great significance for maintaining the healthy development of content ecosystems.

Existing methods for detecting machine-generated text mainly include statistical modeling based on shallow features and end-to-end learning based on deep neural networks [2]. The former classifies by extracting surface features such as word frequency and n-gram distribution, having the advantages of strong interpretability and high computational efficiency but limited discrimination ability for high-quality text generated by LLMs [2]. The latter uses pre-trained language models like RoBERTa and ELECTRA, detecting machine-generated text by fine-tuning on downstream tasks, achieving good results on multiple datasets [3]. However, existing work mainly focuses on specific models or datasets, lacking comprehensive examination of different LLMs, genres, and topics, and rarely involves the fusion and integration of multiple techniques.

To solve the above problems and promote the development of detection technology for text generated by LLMs, this paper proposes a detection method based on multi-model ensemble. Our main contributions are as follows:

- **Constructing a large-scale, diverse, and heterogeneous dataset**: The dataset includes student writings and texts generated by various LLMs (such as GPT-3, GPT-2, CTRL, XLM), surpassing existing work in scale and diversity, providing rich data support for model training and evaluation.
- **Designing a multifaceted detection framework integrating linear models, deep learning models, and pre-trained models**: This framework characterizes the features of LLM texts from both shallow language patterns and deep semantics, improving detection accuracy and generalization ability.
- **Introducing cross-domain corpora to fine-tune pre-trained models and performing model integration**: By fine-tuning pre-trained models on large-scale cross-domain corpora and integrating multiple heterogeneous models using weighted averaging, the generalization and robustness of detection results are significantly improved.
- **Achieving excellent experimental results**: Our method achieved a score of 0.967466 in the Kaggle competition, earning a silver medal and outperforming any single model, providing new ideas and practical references for the task of detecting LLM-generated text.

The structure of this paper is arranged as follows: Section 2 introduces related work, Section 3 describes the detection methods in detail, Section 4 reports experimental results and analysis, and Section 5 summarizes the full text and prospects future research directions.

2. Related Work

Existing methods for detecting machine-generated text mainly fall into two categories: statistical modeling based on shallow features and end-to-end learning based on deep neural networks.

2.1. Statistical Modeling Based on Shallow Features

Early detection methods mainly relied on shallow text features. Hunter and Dale used Unix command-line tools to detect software-generated fake reviews by analyzing surface patterns such as sentence length and function word usage. Lavergne et al. statistically analyzed word transition probability matrices and trained Support Vector Machine (SVM) models to identify machine-written scientific papers. Uchendu et al. [2] designed a multivariate method integrating six types of linguistic features (e.g., word frequency, readability), achieving good results on Yelp and Amazon review datasets. However, these methods based on manual feature engineering are difficult to characterize the deep patterns of text generated by LLMs and have limited discrimination ability for high-quality AI-generated text.

2.2. End-to-End Learning Based on Deep Neural Networks

With the development of pre-trained language models, deep learning methods have gradually become the mainstream for detecting machine-generated text. Zellers et al. [3] proposed the Grover framework, using generative-discriminative adversarial networks to detect fake news, achieving 89.2% accuracy on the RealNews dataset. Bhat et al. improved the adversarial training method based on RoBERTa, introducing Bayesian optimization to automatically adjust hyperparameters, increasing the F1 score on the GLTR dataset to 95.2%. Zhong et al. designed a zero-shot detection method using the latent space differences of GPT-2, which can discriminate text generated by the model without training, achieving results superior to traditional methods on multiple open-source datasets. However, existing work mainly focuses on specific models or datasets, lacking comprehensive examination of different LLMs, genres, and topics, and rarely involves the fusion and integration of multiple techniques.

2.3. Innovations of This Study

To address the above issues, this paper proposes a detection method for LLM-generated text based on multi-model ensemble. Compared with existing work, our main innovations are:

- **Constructing a large-scale, diverse heterogeneous dataset**: Introducing multiple datasets such as DAIGT V2 [4], Pile [5], Ultra, and Human vs. LLM Text Corpus [6], covering student writings and texts generated by various LLMs, surpassing existing research in scale and diversity.
- **Integrating multiple models in a detection framework**: Designing a multifaceted detection framework that integrates linear models, deep learning models, and pre-trained language models, characterizing the features of LLM texts from both shallow statistical features and deep semantic representations.
- **Improving the generalization and robustness of the model**: By fine-tuning pre-trained models on large-scale cross-domain corpora and integrating multiple heterogeneous models using weighted averaging, the generalization and robustness of detection results are significantly improved.

3. Methodology

This section will detail the proposed detection method for LLM-generated text based on multi-model ensemble. Figure 1 illustrates the overall research idea and technical route. We first construct a heterogeneous dataset containing student writings and texts generated by various LLMs. Then, we design linear classification models, deep learning models, and pre-trained language models based on shallow features and deep semantics, respectively. Finally, we achieve the fusion of different models through weighted averaging.

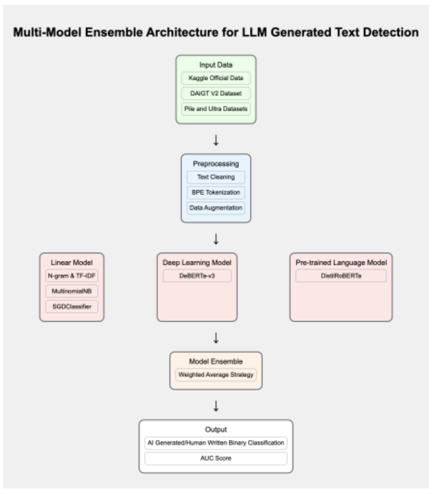


Figure 1. Architecture of Large Language Model Generated Text Detection Method Based on Multi-Model Ensemble

3.1. Construction of Heterogeneous Dataset

To enhance the generalization ability of the model, we constructed a heterogeneous dataset containing student writings and texts generated by various LLMs. The dataset sources include:

- **Kaggle Competition Dataset**: Contains 1,378 English texts, of which 1,375 are student writings and 3 are LLM-generated.
- **DAIGT V2 Training Set**: DAIGT V2 Train Dataset [4] includes argumentative essays written by students and articles generated by different LLMs (e.g., GPT-2, CTRL, XLM), totaling about 50,000 pieces. We used t-SNE dimensionality reduction visualization to analyze the distribution structure of these samples, intuitively showing the differences between student writings and LLM-generated texts.

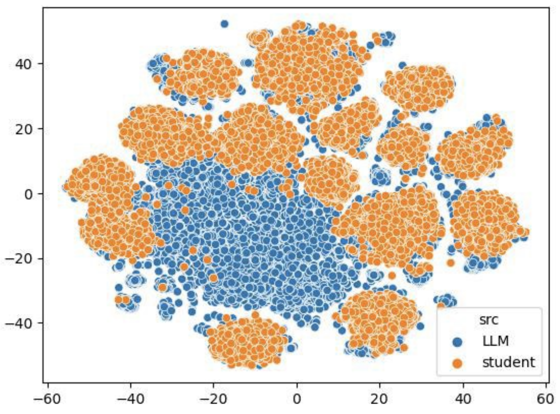


Figure 2. Visualization of DAIGT V2 Train Dataset(t-SNE Dimentionality Reduction)

- **Pile and Ultra Datasets**: Pile and Ultra Pile [5] is a large-scale multi-domain English corpus, including texts from books, GitHub code, web pages, and more. Ultra is a multi-turn dialogue dataset from Tsinghua University, generating natural and fluent dialogue texts from multiple ChatGPT models. We extracted human-written texts and LLM-generated texts from them, enriching the scale of the dataset.
- **Human vs. LLM Text Corpus**: Human vs. LLM Text Corpus [6] contains human-written texts and LLM-generated texts, covering various topics.

To ensure data quality, we performed the following processing:

- **Data Cleaning**: Normalized punctuation, corrected spelling errors, and removed redundant information from all texts [7].
- **Length Filtering**: Removed texts that were too short (less than 100 words) or too long (more than 1,000 words).
- **Deduplication**: Used TF-IDF and cosine similarity to filter near-duplicate texts with similarity exceeding 0.92.
- **Data Splitting**: Employed stratified sampling to split the data into training and validation sets at an 8:2 ratio, ensuring consistent distribution of various texts in different subsets.

3.2. Linear Classification Model Based on Shallow Features

3.2.1. Text Preprocessing and Adaptive Tokenization

To capture the shallow statistical features of the text, we needed to tokenize the text. Due to vocabulary differences from different text sources, we did not directly use a general tokenizer but adaptively trained a BPE (Byte-Pair Encoding) tokenizer [7] based on the training and validation texts. The specific steps are:

- **Training a Custom Tokenizer:**
  - **Data Source**: Used text data from the training and validation sets to ensure the tokenizer adapts to the data distribution the model will process.
  - **Tokenizer Settings**: Adopted the BPE algorithm with a vocabulary size set to 5,000 to capture common words and phrases.
- **Tokenization Processing:**
  - **Unified Tokenization**: Used the trained tokenizer to tokenize all texts in the training set, validation set, and test set, ensuring vocabulary consistency and obtaining unified sequence representations.
  - **Vocabulary Coverage**: Since the tokenizer is trained based on the training and validation sets, it may encounter unseen words in the test set. The BPE tokenizer can effectively handle this situation by breaking unknown words into known subwords or characters.

3.2.2. Feature Representation Based on TF-IDF
Based on tokenization, we extracted N-Gram features of the text, specifically 3-gram to 5-gram. Then, we calculated the TF-IDF values of these N-Grams to obtain feature vectors for each text. TF-IDF can measure the importance of words in the text, helping to capture statistical features of the text.

3.2.3. Training of Linear Classifiers
We used two linear models, Multinomial Naive Bayes and SGDClassifier, for training:

- **Multinomial Naive Bayes**: Suitable for multinomial distributions, performs well on high-dimensional sparse data.
- **SGDClassifier**: A linear classifier optimized using Stochastic Gradient Descent (SGD), allowing for different loss functions (e.g., hinge loss corresponds to linear SVM, log loss corresponds to logistic regression).

Training Steps:

- **Similarity Filtering**: Filtered out highly repetitive data (similarity greater than 0.92) based on cosine similarity to ensure data diversity.
- **Feature Extraction**: Tokenized texts using the custom tokenizer and extracted N-Gram TF-IDF features.
- **Model Training**: Input the above features into MultinomialNB and SGDClassifier to build an ensemble classifier for training.
- **Model Prediction**: Made predictions on the dataset to obtain results.

3.3. Pre-trained Models Based on Deep Learning

3.3.1. Fine-tuning the DeBERTa Model
To capture deep semantic features of the text, we chose the DeBERTa-v3-small pre-trained model [8] and fine-tuned it using large-scale datasets:

- **Data Preparation:**
  - **Data Collection**: Collected a large amount of human-written texts and LLM-generated texts from Pile [5], Ultra datasets, and open-source datasets like Human vs. LLM Text Corpus [6].
  - **Data Processing**: Performed simple preprocessing on the large-scale data to construct a binary classification dataset for fine-tuning.
- **Model Fine-tuning**: Input the processed data into the DeBERTa-v3-small model for fine-tuning on the binary classification task to obtain optimized model weights.

- **Model Inference**: Used the fine-tuned model to make predictions on the Kaggle competition dataset, generating results.

3.3.2. Introducing the DistilRoBERTa Model

We also introduced the open-source DistilRoBERTa model [9], fine-tuning it using third-party datasets to enhance detection capabilities. The specific steps are:

- **Model Fine-tuning**: Fine-tuned the DistilRoBERTa model [9] using third-party datasets.
- **Model Prediction**: Used the fine-tuned model to make predictions on the Kaggle competition dataset, obtaining results.

3.4. Model Ensemble and Prediction

To fully utilize the advantages of different models, we integrated the prediction results of the three models. The detailed methods and formulas for model ensemble are as follows.

3.4.1. Standardization of Prediction Results

Since the output ranges and scales of different models may differ, we first standardized the prediction scores of each model to ensure they are fused on the same scale. We adopted Rank Scaling, mapping each model's prediction scores to the [0, 1] interval.

For model $M_i (i = 1, 2, 3)$, its prediction score for sample x_j is $s_i(x_j)$. We performed ascending sorting on all prediction scores of model M_i to obtain the rank $r_i(x_j)$. Then, we calculated the standardized score $\hat{s}_i(x_j)$:

$$\hat{s}_i(x_j) = \frac{r_i(x_j) - 1}{N - 1}$$

where N is the total number of samples in the test set, and $r_i(x_j)$ is the rank of sample x_j in the sorted prediction scores of model M_i.

3.4.2. Weighted Fusion

After obtaining the standardized prediction scores of each model, we fused the model's prediction results through a weighted average strategy to obtain the final prediction probability. Set the weight of each model as w_i, satisfying $\sum_{i=1}^{3} w_i = 1$. Then, for sample x_j, the final prediction probability $P(y = 1 \mid x_j)$ is calculated as:

$$P(y = 1 \mid x_j) = \sum_{i=1}^{3} w_i \hat{s}_i(x_j)$$

where:

- $\hat{s}_i(x_j)$ is the standardized prediction score of model M_i for sample x_j.
- w_i is the weight of model M_i, reflecting the importance of the model in the fusion.

3.4.3. Determining Weights

To determine the optimal weights wiw_iwi of each model, we performed tuning on the validation set. The specific steps are:

- **Initializing Weight Range**: Set the initial range of weights w_i to [0, 1], satisfying $\sum_{i=1}^{3} w_i = 1$.

- **Grid Search/Bayesian Optimization**: Used grid search or Bayesian optimization methods to traverse possible weight combinations, calculating the model's performance on the validation set for each set of weights.
- **Selecting Optimal Weights**: Chose the weight combination (w_1^*, w_2^*, w_3^*) that achieved the best performance on the validation set.

3.4.4. Summary of Model Ensemble Algorithm

In summary, the algorithm flow of model ensemble is:

1. **Obtaining Prediction Scores of Each Model**: For each sample x_j in the test set, calculate the prediction scores $s_i(x_j)$ of each model.
2. **Standardizing Prediction Results**: Perform Rank Scaling on each model's prediction scores to obtain standardized scores $\hat{s}_i(x_j)$.
3. **Weighted Fusion**: Calculate the final prediction probability using the determined weights w_i:

$$P(y = 1 \mid x_j) = \sum_{i=1}^{3} w_i \hat{s}_i(x_j)$$

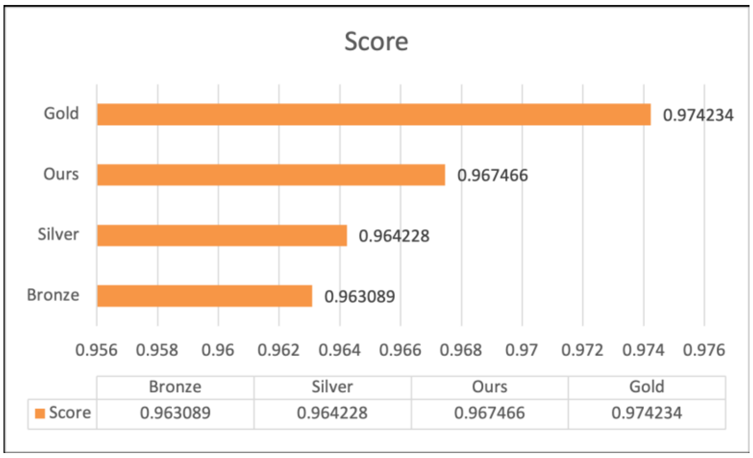
4. **Outputting Final Prediction Results**: Convert the prediction probability $P(y = 1 \mid x_j)$ into class labels based on the required threshold, or directly use the probabilities for evaluation metric calculation.

3.4.5. Advantages of Model Ensemble

Through the above model ensemble method:

- **Comprehensive Utilization of Different Models' Strengths**: Linear models are good at capturing shallow statistical features of the text, while deep learning models and pre-trained language models can capture deep semantic features.
- **Improved Generalization Ability**: By fusing predictions from multiple models, the risk of overfitting in a single model can be reduced, enhancing adaptability to unknown data.
- **Enhanced Model Robustness**: Errors from different models have certain complementarity; ensemble can reduce overall errors.

4. Experimental Results and Model Performance



	Bronze	Silver	Ours	Gold
■ Score	0.963089	0.964228	0.967466	0.974234

Figure 3. The benchmark scores of gold, silver, and bronze medals in the LLM - Detect AI Generated Text competition. Our final score surpasses the silver medal benchmark

To better demonstrate quantitative results, we compared our model's score in the Kaggle competition with the medal thresholds, as shown in Figure 3. The ensemble model achieved a score of 0.967466 on the dataset, surpassing the bronze medal standard (0.964223–0.963089) and falling within the silver medal range (0.973915–0.964228). This indicates that the method has strong detection capabilities. Although it did not reach the gold medal standard (0.990206–0.974234), this achievement is quite competitive on the Kaggle competition leaderboard, fully demonstrating the potential of multi-model ensemble in complex text detection tasks.

4.1. Performance of Individual Models

- **Linear Model**: The linear model trained on the DAIGT V2 dataset performed excellently, effectively capturing shallow features of the text and providing a good baseline for the final ensemble model.
- **Deep Learning Model (DeBERTa-v3-small)**: The DeBERTa-v3-small model fine-tuned on large-scale datasets effectively learned deep semantic features of the text, significantly improving detection accuracy.
- **Pre-trained Model (DistilRoBERTa)**: By fine-tuning on third-party datasets, the DistilRoBERTa model balanced computational efficiency and performance, providing strong support for multi-model fusion.

4.2. Model Ensemble and Fusion

The ensemble strategy performed weighted rank normalization fusion on the prediction results of each model and conducted grid search on the validation set to determine the optimal weights. Results show that this ensemble strategy can greatly improve the detection accuracy and stability of the model. The AUC value corresponding to the Kaggle competition test data reached 0.967466.

Through this multi-model ensemble method, the model exhibited outstanding performance in handling diverse texts and complex LLM-generated text detection tasks, outperforming any single model.

5. Conclusion

In summary, this paper proposes a detection method for LLM-generated text based on multi-model ensemble. The method combines the advantages of linear models, deep learning models, and pre-trained language models, successfully achieving effective detection of text generated by LLMs. Scoring 0.967466 in the Kaggle competition, it surpasses the bronze medal standard and falls within the silver medal range, demonstrating high generalization ability and robustness.

In this study, the linear model primarily captures shallow statistical features of the text, the deep learning model (DeBERTa-v3-small) provides deeper semantic representations, and the pre-trained DistilRoBERTa model further enhances detection capabilities through knowledge distillation. By weighting and fusing these models, we significantly improved overall detection performance.

This research provides a new solution for detecting LLM-generated text, helping to strengthen the identification of different types of text, especially in fields like education and media. The multi-model ensemble strategy in this study not only achieved good results in this competition but also provides important references for future text detection in broader scenarios. Future research can further explore different model architectures and feature fusion to handle multilingual texts and the continuously evolving LLM models, further improving detection accuracy and robustness.

References

[1] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems* (Vol. 33, pp. 1877-1901). Curran Associates, Inc.

[2] Uchendu, A., Suresh, H., Xu, W., & Lee, S. (2020). Authorship Attribution for Neural Text Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 8384-8395). Association for Computational Linguistics.

[3] Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., & Choi, Y. (2019). Defending Against Neural Fake News. In *Advances in Neural Information Processing Systems* (Vol. 32, pp. 9051-9062). Curran Associates, Inc.

[4] DAIGT V2 Train Dataset. (2022). Kaggle. Available at: https://www.kaggle.com/datasets/thedrcat/daigt-v2-train-dataset

[5] Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., ... & Leahy, C. (2020). The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv preprint arXiv:2101.00027*. Available at: https://www.kaggle.com/datasets/canming/piles-and-ultra-data

[6] Human vs. LLM Text Corpus. (2023). Kaggle. Available at: https://www.kaggle.com/datasets/starblasters8/human-vs-llm-text-corpus

[7] Kudo, T., & Richardson, J. (2018). SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 66-71). Association for Computational Linguistics.

[8] He, P., Liu, X., Gao, J., & Chen, W. (2021). DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In *International Conference on Learning Representations (ICLR) 2021*. Available at: https://openreview.net/forum?id=XPZIaotutsD

[9] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.