A Machine-Generated Text Detection Model Based on Text Multi-Feature Fusion

Notebook for the PAN Lab at CLEF 2024

Mingcan Guo, Zhongyuan Han*, Haoyang Chen and Jiangao Peng

Foshan University, Foshan, China

Abstract

In the current wave of rapid technological advancement, many cutting-edge large language models (LLMs) have emerged, such as GPT-4 and Llama. However, the ability of these LLMs to generate smooth and coherent text has led to concerns about potential misuse. Therefore, in practical applications, the ability to distinguish them from texts created by human hands becomes especially crucial. A model for detecting machine-generated text is proposed through the PAN Task 4 - Voight-Kampff Generative AI Authorship Verification task. The model emphasizes the extraction of additional semantic information from the text. Various pre-trained language models (PLMs) were employed in the experiments, and the incorporation of multiple text features, such as word frequency and perplexity, was explored to enhance the outcomes. Ultimately, four runs were submitted, with the highest-performing approach attaining a mean score of 0.884 across all test datasets.

Keywords

LLMs, machine-generated text, text features, word frequency, perplexity

1. Introduction

With the advent of the AI era, the content on the internet is increasingly being infiltrated by machine-generated text. The emergence of powerful LLMs like ChatGPT and its derivative versions has made the creation of high-quality text more accessible than ever before. However, this has also posed a challenging problem: How can we differentiate between machine-generated text and human creativity? This concerning issue is troubling people in various domains, such as conspiracy theories[1], plagiarism concerns[2], political biases[3], and misinformation[4]. Detecting machine-generated text is an urgent need to address the social issues arising from the misuse of these large models.

To tackle the challenges LLMs pose, PAN 2024[5] has intensely focused on authorship verification tasks based on LLMs. Among them, the Voight-Kampff Generative AI Authorship Verification 2024 task[6] is part of the tasks under the Conference and Labs of the Evaluation Forum (CLEF 2024). The builder-breaker collaboration combines the Generative AI Authorship Verification task with the Voight-Kampff task from the ELOQUENT Lab. The primary focus of the task is to comprehend and detect human and machine-generated text. Participants must build automated systems that accurately attribute the text to human authors by distinguishing between human and machine-generated text.

The primary emphasis of this study is on extracting valuable information from the original text. Drawing inspiration from the work of Davies[7] and Przybyla et al.[8], language modeling is investigated from two perspectives: text perplexity and word frequency. Furthermore, multiple PLMs were fine-tuned on an augmented dataset, and text features extracted from the modeling process were incorporated into a hybrid model. The hybrid models were fine-tuned independently and were subsequently assigned weights and ensembled to generate the final results.

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09-12, 2024, Grenoble, France

*Corresponding author.

gmc9812@163.com (M. Guo); hanzhongyuan@gmail.com (Z. Han); hoyo.chen.i@gmail.com (H. Chen); wyd1n910@gmail.com (J. Peng)

0000-0002-4977-2138 (M. Guo); 0000-0001-8960-9872 (Z. Han); 0000-0003-3223-9086 (H. Chen); 0009-0006-3780-5023 (J. Peng)

2. Related Work

With the advent of Language Models, detecting machine-generated text has become a popular research direction[9, 10]. Existing detection methods can generally be classified into three categories: probability-based detection methods, machine learning-based detection methods, and deep learning-based detection methods. Probability-based methods typically do not require training samples. Gehrmann et al.[11] designed a statistical tool called GLTR, highlighting the distribution differences between generated text and human-written text using different colors. GLTR detects the number of high, medium, and low probability words and presents the results in a visualized format. Mitchell et al.[12] proposed DetectGPT, which calculates log probabilities by perturbing text and comparing the changes between the original and perturbed text. It assumes that machine-generated text tends to lie within the negative log probability curve, while human-written text exhibits higher or lower probabilities compared to perturbed text. Machine learning-based methods, on the other hand, typically employ classical machine learning models. These methods often have fewer parameters and can be easily deployed. For instance, Solaiman et al.[13] applied the TF-IDF method to design a regression classifier that utilizes the Top-K sampling strategy to identify machine-generated text. Fröhling et al.[14] proposed a method that employs advanced features to simulate text's experience, syntax, and semantics. They utilized simple classification models for detection, achieving performance comparable to mainstream deep learning methods.

Deep learning-based detection methods have gained widespread application in recent research and often outperform the previous two categories[15]. The emergence of the Transformer architecture has provided many advanced methods for deep learning. For example, Chen et al.[16] employed RoBERTa and T5 architectures to design a feature extraction and discrimination process between human-written text and text generated by ChatGPT. They trained two text classification models for text detection. Gambini et al.[17] demonstrated that fine-tuning XLNET on tweets written by humans, GPT-2, and earlier generation techniques, such as Markov chains and RNNs, can effectively identify GPT-3-generated tweets with an accuracy of 82.1%. It shows that fine-tuning XLNET on a mixed dataset can be a successful method for detecting machine-generated text.

It has been shown through practical applications that incorporating text features into PLMs is an effective method[18, 19, 20]. Some researchers have observed that incorporating features such as word embeddings[21], predictability[19], linguistic inquiry and word count[22], and other linguistic features can yield text encodings that contain high-quality semantic information. This feature integration enhances the models' representation power and improves their performance in text detection tasks. The present study derives inspiration from the methods above and employs a similar approach by incorporating multiple features with text encoding to improve the model's performance.

3. Methods

This section describes the model employed for automatic differentiation between human and machine-generated text. It includes an overview of the explored fusion features and the final ensemble architecture utilized. The architecture of the model is depicted in Figure 1. A PLM encodes the text and derives a pooled output for text representation. To capture features, an LSTM is utilized for language modeling [23, 24], generating text-based feature vectors. The pooled vectors and feature vectors are concatenated and inputted into a linear layer to obtain fusion logits. Subsequently, the softmax function is applied to derive the output probabilities. Alternatively, the pooled vector can be directly passed through a linear layer to yield PLM logits, which are then softmaxed to obtain the output probabilities.

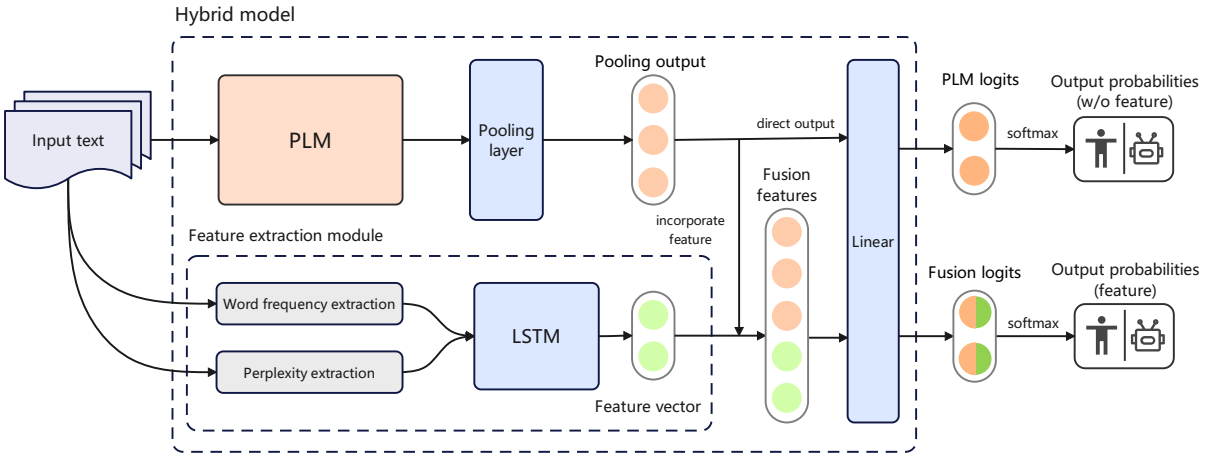

Figure 1: The model architecture includes two versions of outputs: with features and without features.

3.1. Word Frequency Extraction

The inclusion of word frequency features is considered in the text encoding process. The word frequency corpus used in this work is sourced from the English dataset of the Google Book Corpus Ngrams[1]. This dataset encompasses textual data from millions of books published between 1500 and 2008, providing a comprehensive resource for word frequency information. The data is presented as "n-grams," consecutive sequences of n words. It can be utilized to study the distribution of token frequencies in machine-generated text compared to human-authored text. Specifically, a word frequency dictionary is created using the Google Book Corpus, where each word group is mapped to its corresponding frequency of occurrence. Then, the tokenizer will be used to segment the input text. Look up the dictionary for each word group w to get its counts $tf(w)$. If it does not exist in the dictionary, take 1. Finally, take the log of $tf(w)$. The process is expressed as Equation 1.

$$\text{Freq} = \log\left(\max\left(1, tf(w)\right)\right) \tag{1}$$

When encountering a lengthy word group comprising multiple words, the complete count is assigned to each word within the group. Ultimately, the word frequency feature representation of the text is obtained through the LSTM.

3.2. Perplexity Extraction

Perplexity is used to evaluate the predictive ability of a set of sample data. It measures the uncertainty or confusion level of the given data. Research by Tang et al. [10] has shown that language models tend to focus on common patterns in the text they are trained on, resulting in lower perplexity scores for text generated by LLMs. In contrast, human authors can express themselves in multiple styles, making it more challenging for language models to make accurate predictions and resulting in higher perplexity values for text created by humans.

GPT2 is considered the underlying generative model for perplexity features since most LLMs are based on the same Transformer architecture. GPT2 tends to assign low perplexity to common text and higher perplexity to text with varied styles. Perplexity can be measured by the entropy of the probability distribution. For a given sequence of n tokens t_i, the probability distribution entropy $H(t)$ can be represented as Equation 2, where p represents the probability of generating the token corresponding to the vocabulary, and eps is the deviation term.

$$H(t) = -\sum_{i=1}^{n} p\left(t_i\right) \log_2\left(p(t_i) + \text{eps}\right) \tag{2}$$

[1] https://storage.googleapis.com/books/ngrams/books/datasetsv3.html

In addition, for a given t_i, t_{i+1} represents the next token immediately following t_i, t_i' represents the GPT2-predicted token for t_i (the token with the highest probability in the vocabulary). Therefore, when considering the token probabilities across the entire vocabulary in GPT2, the logarithmic probabilities of the occurrence of the succeeding token t_{i+1} and the predicted token t_i in the vocabulary are also taken into account. As shown in Equation 3 and Equation 4, these measures assess the probability of the context token occurring and the model's confidence in predicting the token, respectively.

$$L(t) = logp\left(t_{i+1}\right) \tag{3}$$

$$C(t) = logp\left(t_i'\right) \tag{4}$$

The number of context probability and prediction confidence values is n-1. Missing values are filled with 0 to align with the number of entropy values for the probability distribution. Finally, these features are concatenated together, and the LSTM network is used to obtain the perplexity feature representation of the text.

3.3. Hybrid Model

Due to the excellent performance of Transformer-based PLMs in downstream classification tasks [15], three different variants of PLMs are explored: BERT, BERT-Large, and Roberta-Large. The [CLS] token representation with a length of either 768 or 1024 is utilized, and the output is extracted from the pooling layer.

The primary focus is on incorporating text features. As shown in Figure 1, the hybrid model designed includes LSTM and PLM:

- Text features are extracted using the word frequency and perplexity extraction introduced in the previous two sections and concatenated.
- The concatenated features are fed into the LSTM for encoding, and the hidden layer's state values are used as the feature vector.
- The feature vector is concatenated with the PLM pooled output to obtain fusion features, which are then passed through a linear classification layer.

During the training phase, the hybrid model's logits are directly outputted as the binary classification scores for each text after applying the softmax function. Since the provided samples are text pairs during the prediction phase, potential prediction biases caused by individual texts are minimized by predicting the probabilities for text1 and text2 separately.

An ensemble of two models in Figure 2 is considered. For instance, two hybrid models with PLM using BERT-Large and Roberta-Large, respectively, are fine-tuned. The corresponding logits for text1 and text2 are obtained from the models and weighted. The final logits for each text are calculated as the weighted sum, as depicted in Equation 5, where λ represents the weight coefficient.

$$\text{logits}_{\text{final}} = \lambda \cdot \text{logits}_{\text{BERT\_Large}} + (1 - \lambda) \cdot \text{logits}_{\text{Roberta\_Large}} \tag{5}$$

Finally, the final logits of text1 and text2 are averaged after applying the softmax function. Specifically, the logits of text1 and (1 - logits) of text2 are added together and averaged. This results in the output probabilities during the prediction phase.

4. Experiments and Results

4.1. Data Preprocessing

The official train dataset is derived from a collection of real and fake news articles from multiple news headlines in the United States in 2021. On the other hand, the test dataset includes various types of texts, such as news articles, Wikipedia summaries, or fan fiction. The training set consists of one portion

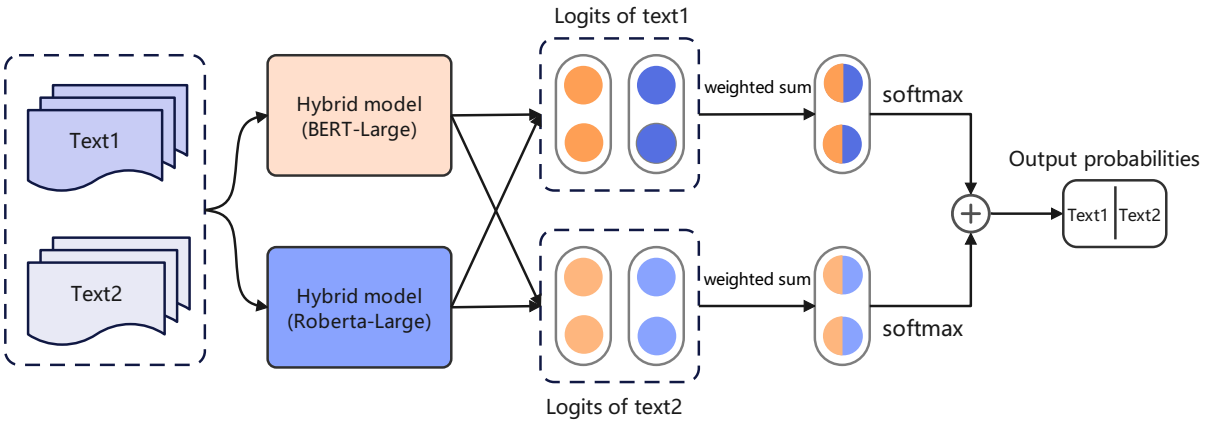

Figure 2: During the prediction phase, the logits of two hybrid models, each based on a different PLM, are ensemble combined to generate the final output.

of human-authored data and 13 portions of data generated by different LLMs, each corresponding to a human sample topic. Each text data portion contains 1,087 samples and is provided in a dictionary format separated by a set of newline characters, such as *{"id": "...", "text": "..."}*. The test set will be provided in a different format, where each line contains a pair of texts, such as *{"id": "...", "text1": "...", "text2": "...}*. One of the texts is human-authored, while the other is machine-generated.

Figure 3 displays the text lengths and their corresponding probability density distributions plotted as KDE graphs for each data portion. Most text lengths are distributed between 300 and 500, implying that when utilizing PLMs, the information loss resulting from truncating excessively long texts does not need to be excessively concerned with.

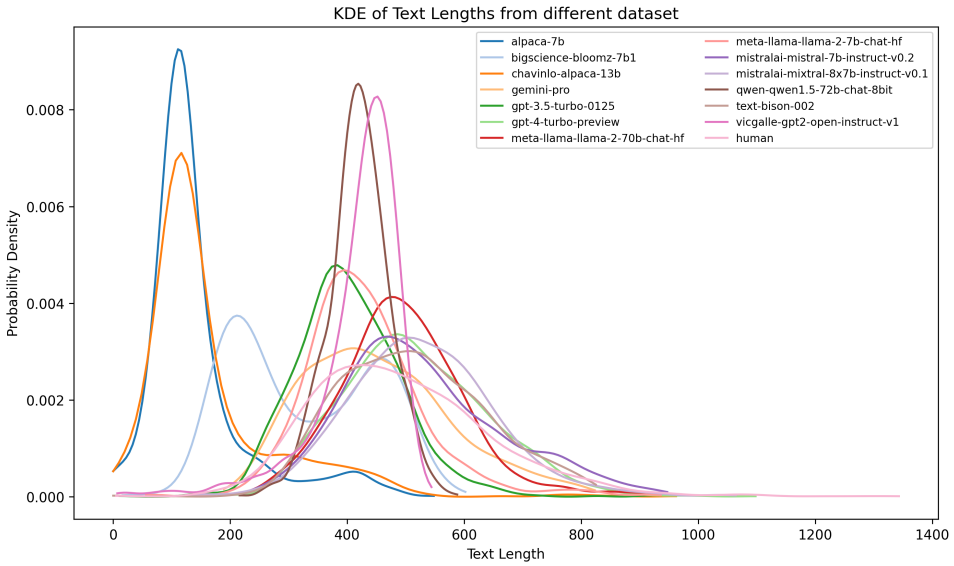

Figure 3: The length of texts generated by humans and different LLMs and their corresponding probability density distribution.

Considering the imbalance between human-authored and machine-generated texts (1:13), data augmentation techniques are employed to expand the original training set, ensuring the quality of the model's learning. Expressly, a benchmark dataset[2] proposed by Sarvazyan et al.[25], similar in the domain to the provided human dataset and consisting of Wikipedia and news articles, is referred to. Since the LLMs used for machine-generated data differ from the training set, it was chosen not to utilize

[2]https://huggingface.co/datasets/symanto/autextification2023

the machine-generated text portion of this benchmark dataset. Instead, only human-authored data was extracted to augment and align the dataset with the training set.

A pre-trained sentence-transformers model[3], capable of tasks such as similarity judgment or semantic search, was employed. Similarity comparisons were conducted between each text in the training set and the reference dataset. By encoding the texts using the sentence-transformers model, cosine similarity was calculated, and the top 12 texts with the highest similarity scores were selected. In the end, 13,044 similar texts were extracted for expansion.

4.2. Experimental Setting

The training set was re-divided into train and test sets using a 7:3 ratio. BERT, BERT-Large, and Roberta-Large were utilized as the base PLMs. The model was developed based on the PaddlePaddle framework for the framework and parameter selection. The batch size was 64, the max length was 512, and the learning rate was 2e-5. The model was trained for 10 epochs using the AdamW optimizer on an environment with an A800 GPU.

Additionally, the method introduces two parameters, λ, and $is\_feature$, which specify the ensemble models' weight proportion and decide whether the hybrid model outputs fusion logits or PLM logits. It provides us with the flexibility to generate different approaches.

The approach's effectiveness was validated by running four different approaches on the final test set of the TIRA platform [26]. Their introductions are as follows:

- rapid-pole: As a baseline for comparison, using only the prediction results from BERT, without including any word frequency or perplexity feature, $is\_feature$ set to False.
- savory-plate: Similar to rapid-pole, but employs the Bert-Large model without any word frequency or perplexity feature, $is\_feature$ set to False.
- lazy-iteration: Ensembles Bert-Large and Roberta-Large hybrid models, with the weighting coefficient λ in Equation 5 set to 0.9, $is\_feature$ set to True.
- gritty-producer: Similar to lazy-iteration, but with the weighting coefficient λ in Equation 5 set to 0.1, $is\_feature$ set to True.

4.3. Metrics and Baselines

This section discusses the official metrics and baselines used in the task. The metrics employed encompass six different dimensions:

- ROC-AUC: The area under the ROC (Receiver Operating Characteristic) curve.
- Brier: The complement of the Brier score (mean squared loss).
- C@1: A modified accuracy score that assigns non-answers (score = 0.5) the average accuracy of the remaining cases.
- F1: The harmonic mean of precision and recall.
- F0.5u: A modified F0.5 measure (precision-weighted F measure) that treats non-answers (score = 0.5) as false negatives.
- The arithmetic mean of all the metrics above.

The LLM detection baseline includes seven implementations: PPMd Compression-based Cosine (PPMd CBC), Authorship Unmasking, Binoculars, DetectLLM LRR and NPR, DetectGPT, Fast-DetectGPT, and Text length. PPMd CBC and Authorship Unmasking utilize a bag-of-words model, while Binoculars, DetectLLM, and DetectGPT employ LLMs to measure text perplexity. Text length serves as a randomness indicator for data integrity checks. The results of the partial baseline implementations can be found in the section 4.4.

[3] https://huggingface.co/annakotarba/sentence-similarity

4.4. Results

The external results of the model can be seen in Table 1 and 2. Table 1 shows the results that the official baselines provided by the PAN organizers and summary statistics of all submissions to the task (i.e., the maximum, median, minimum, and 95-th, 75-th, and 25-th percentiles over all submissions to the task). Table 2 shows the summarized results averaged (arithmetic mean) over 10 variables of the test dataset.

The best-performing approach, "gritty-producer," achieved an average score of 0.966, surpassing all baseline methods. In the quantile results, the scores of participating teams are arranged in ascending order. This approach falls within the range of 75% to 95%.

As expected, the performance of approaches that incorporate features (lazy-iteration and gritty-producer) outperforms the individual PLM models (rapid-pole and savory-plate) on both datasets.

The results show that the scheme focusing on using Roberta_Large probability with language features (gritty-producer) is better than the scheme focusing on using Bert_Large probability with language features (lazy-iteration), which shows that the choice of PLM is also important. The model with more advanced pre-training skills is usually better for this use case.

This task uses PLMs to maintain a lead over almost all advanced baselines, including DetectLLM and DetectGPT using LLM, so introducing LLM to achieve the task was not explored.

Table 1
Overview of the accuracy in detecting if a text is written by an human in task 4 on PAN 2024 (Voight-Kampff Generative AI Authorship Verification). We report ROC-AUC, Brier, C@1, F_1, $F_{0.5u}$ and their mean.

Approach	ROC-AUC	Brier	C@1	F_1	$F_{0.5u}$	Mean
rapid-pole	0.978	0.935	0.954	0.908	0.918	0.939
savory-plate	0.947	0.899	0.913	0.888	0.891	0.908
lazy-iteration	0.979	0.925	0.955	0.948	0.948	0.951
gritty-producer	0.989	0.949	0.965	0.963	0.962	0.966
Baseline Binoculars	0.972	0.957	0.966	0.964	0.965	0.965
Baseline Fast-DetectGPT (Mistral)	0.876	0.8	0.886	0.883	0.883	0.866
Baseline PPMd	0.795	0.798	0.754	0.753	0.749	0.77
Baseline Unmasking	0.697	0.774	0.691	0.658	0.666	0.697
Baseline Fast-DetectGPT	0.668	0.776	0.695	0.69	0.691	0.704
95-th quantile	0.994	0.987	0.989	0.989	0.989	0.990
75-th quantile	0.969	0.925	0.950	0.933	0.939	0.941
Median	0.909	0.890	0.887	0.871	0.867	0.889
25-th quantile	0.701	0.768	0.683	0.657	0.670	0.689
Min	0.131	0.265	0.005	0.006	0.007	0.224

Table 3 showcases the final results achieved by the model in the task. The individual effectiveness scores are aggregated across all test datasets and corrected by half a standard deviation to penalize unstable classification performance. The "gritty-producer" approach achieved a mean score of 0.884 and ranked fourth among 30 teams.

5. Conclusion

This paper proposes a text-based multi-feature fusion hybrid model for addressing the Voight-Kampff Generative AI Authorship Verification 2024 task. Experiments were conducted using various Transformer-based PLMs, and detailed insights into different feature extraction methods are provided. These methods effectively enhance the model's performance, resulting in the task's mean score of 0.884. For future work, improving the selection of hyperparameters, such as the output weights for each model, can be focused on, where grid search techniques can help identify better values. Additionally, further exploration can be done by incorporating additional text features to enhance the final output results.

Table 2
Overview of the mean accuracy over 9 variants of the test set. We report the minumum, median, the maximum, the 25-th, and the 75-th quantile, of the mean per the 9 datasets.

Approach	Minimum	25-th Quantile	Median	75-th Quantile	Max
rapid-pole	0.540	0.711	0.936	0.950	0.984
savory-plate	0.386	0.732	0.908	0.950	0.977
lazy-iteration	0.659	0.849	0.951	0.977	0.990
gritty-producer	0.743	0.959	0.966	0.990	0.996
Baseline Binoculars	0.342	0.818	0.844	0.965	0.996
Baseline Fast-DetectGPT (Mistral)	0.095	0.793	0.842	0.931	0.958
Baseline PPMd	0.270	0.546	0.750	0.770	0.863
Baseline Unmasking	0.250	0.662	0.696	0.697	0.762
Baseline Fast-DetectGPT	0.159	0.579	0.704	0.719	0.982
95-th quantile	0.863	0.971	0.978	0.990	1.000
75-th quantile	0.758	0.865	0.933	0.959	0.991
Median	0.605	0.645	0.875	0.889	0.936
25-th quantile	0.353	0.496	0.658	0.675	0.711
Min	0.015	0.038	0.231	0.244	0.252

Table 3
Final ranking results

Rank	Team	Approach	ROC-AUC	Brier	C@1	F_1	$F_{0.5u}$	Mean
1	marsan	staff-trunk	**0.961**	**0.928**	0.912	0.884	**0.932**	**0.924**
2	you-shun-you-de	charitable-mole_v3	0.931	0.926	**0.928**	**0.905**	0.913	0.921
3	baselineavengers	svm	0.925	0.869	0.882	0.875	0.869	0.886
4	g-fosunlpteam	gritty-producer	0.889	0.875	0.887	0.884	0.884	0.884
	(26 more)							

Acknowledgments

This work is supported by the Social Science Foundation of Guangdong Province, China (No.GD24CZY02)

References

[1] S. Levy, M. Saxon, W. Y. Wang, Investigating memorization of conspiracy theories in text generation, in: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, 2021, pp. 4718–4729.

[2] N. Dehouche, Plagiarism in the age of massive generative pre-trained transformers (gpt-3), Ethics in Science and Environmental Politics 21 (2021) 17–23.

[3] D. Rozado, The political biases of chatgpt, Social Sciences 12 (2023) 148.

[4] G. Spitale, N. Biller-Andorno, F. Germani, Ai model gpt-3 (dis) informs us better than humans, Science Advances 9 (2023) eadh1850.

[5] A. A. Ayele, N. Babakov, J. Bevendorff, X. B. Casals, B. Chulvi, D. Dementieva, A. Elnagar, D. Freitag, M. Fröbe, D. Korenčić, M. Mayerl, D. Moskovskiy, A. Mukherjee, A. Panchenko, M. Potthast, F. Rangel, N. Rizwan, P. Rosso, F. Schneider, A. Smirnova, E. Stamatatos, E. Stakovskii, B. Stein, M. Taulé, D. Ustalov, X. Wang, M. Wiegmann, S. M. Yimam, E. Zangerle, Overview of PAN 2024: Multi-Author Writing Style Analysis, Multilingual Text Detoxification, Oppositional Thinking Analysis, and Generative AI Authorship Verification, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of

the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2024.
[6] J. Bevendorff, M. Wiegmann, J. Karlgren, L. Dürlich, E. Gogoulou, A. Talman, E. Stamatatos, M. Potthast, B. Stein, Overview of the "Voight-Kampff" Generative AI Authorship Verification Task at PAN and ELOQUENT 2024, in: G. Faggioli, N. Ferro, P. Galuščáková, A. G. S. de Herrera (Eds.), Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS.org, 2024.
[7] M. Davies, Making google books n-grams useful for a wide range of research on language change, International Journal of Corpus Linguistics 19 (2014) 401–416.
[8] P. Przybyła, Detecting bot accounts on twitter by measuring message predictability, 2019.
[9] E. Crothers, N. Japkowicz, H. L. Viktor, Machine-generated text: A comprehensive survey of threat models and detection methods, IEEE Access (2023).
[10] R. Tang, Y.-N. Chuang, X. Hu, The science of detecting llm-generated text, Communications of the ACM 67 (2024) 50–59.
[11] S. Gehrmann, H. Strobelt, A. M. Rush, Gltr: Statistical detection and visualization of generated text, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, 2019, pp. 111–116.
[12] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-generated text detection using probability curvature, in: International Conference on Machine Learning, PMLR, 2023, pp. 24950–24962.
[13] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, J. Wang, Release strategies and the social impacts of language models (2019).
[14] L. Fröhling, A. Zubiaga, Feature-based detection of automated language models: tackling gpt-2, gpt-3 and grover, PeerJ Computer Science 7 (2021) e443.
[15] D. Macko, R. Moro, A. Uchendu, J. Lucas, M. Yamashita, M. Pikuliak, I. Srba, T. Le, D. Lee, J. Simko, et al., Multitude: Large-scale multilingual machine-generated text detection benchmark, in: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 2023, pp. 9960–9987.
[16] Y. Chen, H. Kang, V. Zhai, L. Li, R. Singh, B. Raj, Gpt-sentinel: Distinguishing human and chatgpt generated content, arXiv preprint arXiv:2305.07969 (2023).
[17] M. Gambini, T. Fagni, F. Falchi, M. Tesconi, On pushing deepfake tweet detection capabilities to the limits, in: Proceedings of the 14th ACM Web Science Conference 2022, 2022, pp. 154–163.
[18] A. Palmer, N. Schneider, N. Schluter, G. Emerson, A. Herbelot, X. Zhu, Proceedings of the 15th international workshop on semantic evaluation (semeval-2021), in: Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), 2021.
[19] P. Przybyła, N. Duran-Silva, S. Egea-Gómez, I've seen things you machines wouldn't believe: Measuring content predictability to identify automatically-generated text, in: Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2023). CEUR Workshop Proceedings, CEUR-WS, Jaén, Spain, 2023.
[20] P. Fivez, W. Daelemans, T. Van de Cruys, Y. Kashnitsky, S. Chamezopoulos, H. Mohammadi, A. Giachanou, A. Bagheri, W. Poelman, J. Vladika, et al., The clin33 shared task on the detection of text generated by large language models, Computational Linguistics in the Netherlands Journal 13 (2024) 233–259.
[21] E. Ferracane, S. Wang, R. Mooney, Leveraging discourse information effectively for authorship attribution, in: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2017, pp. 584–593.
[22] A. Uchendu, T. Le, K. Shu, D. Lee, Authorship attribution for neural text generation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 8384–8395.
[23] M. Lippi, M. A. Montemurro, M. Degli Esposti, G. Cristadoro, Natural language statistical features of lstm-generated texts, IEEE Transactions on Neural Networks and Learning Systems 30 (2019) 3326–3337.

[24] M. Sundermeyer, R. Schlüter, H. Ney, Lstm neural networks for language modeling., in: Interspeech, volume 2012, 2012, pp. 194–197.
[25] A. Mikael Sarvazyan, J. Ángel González, M. Franco-Salvador, F. Rangel, B. Chulvi, P. Rosso, Overview of autextiflcation at iberlef 2023: Detection and attribution of machine-generated text in multiple domains., Procesamiento del Lenguaje Natural 71 (2023).
[26] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. doi:10.1007/978-3-031-28241-6_20.