USTC-BUPT at SemEval-2024 Task 8: Enhancing Machine-Generated Text Detection via Domain Adversarial Neural Networks and LLM Embeddings

Zikang Guo[1], **Kaijie Jiao**[1], **Xingyu Yao**[2], **Yuning Wan**[1], **Haoran Li**[2], **Benfeng Xu**[1]
Licheng Zhang[1], **Quan Wang**[2], **Yongdong Zhang**[1] and **Zhendong Mao**[1*]
[1]University of Science and Technology of China, Hefei, China
[2]MOE Key Laboratory of Trustworthy Distributed Computing and Service,
Beijing University of Posts and Telecommunications, Beijing, China
zdmao@ustc.edu.cn

Abstract

This paper introduces the system developed by USTC-BUPT for SemEval-2024 Task 8. The shared task comprises three subtasks across four tracks, aiming to develop automatic systems to distinguish between human-written and machine-generated text across various domains, languages and generators. Our system comprises four components: DATeD, LLAM, TLE, and AuDM, which empower us to effectively tackle all subtasks posed by the challenge. In the monolingual track, DATeD improves machine-generated text detection by incorporating a gradient reversal layer and integrating additional domain labels through Domain Adversarial Neural Networks, enhancing adaptation to diverse text domains. In the multilingual track, LLAM employs different strategies based on language characteristics. For English text, the LLM Embeddings approach utilizes embeddings from a proxy LLM followed by a two-stage CNN for classification, leveraging the broad linguistic knowledge captured during pre-training to enhance performance. For text in other languages, the LLM Sentinel approach transforms the classification task into a next-token prediction task, which facilitates easier adaptation to texts in various languages, especially low-resource languages. TLE utilizes the LLM Embeddings method with a minor modification in the classification strategy for subtask B. AuDM employs data augmentation and fine-tunes the DeBERTa model specifically for subtask C. Our system wins the multilingual track and ranks second in the monolingual track. Additionally, it achieves third place in both subtask B and C.

1 Introduction

The burgeoning capabilities of large language models (LLMs), exemplified by ChatGPT (OpenAI, 2022), GPT-4 (OpenAI, 2023) and Llama (Touvron et al., 2023a), have made machine-generated text more fluent and human-like, which has led to an increasing concern about the abuse of LLMs such as misinformation spread (Bian et al., 2023; Hanley and Durumeric, 2024; Pan et al., 2023) and disruption in education system (Perkins et al., 2023; Vasilatos et al., 2023). So far humans perform only slightly better than chance when distinguishing between text generated by LLMs and human (Mitchell et al., 2023), so it calls for an automatic system to identify machine-generated text.

To this end, MBZUAI NLP department holds SemEval-2024 Task8, which consists of three subtasks. Subtask A focuses on determining whether a full text is human-written or machine-generated. The biggest challenge lies in the domain difference between the training set and test set while the multilingual track requires strong adaptation to text across various languages. It demands the model to have a strong capability of generalization in out-of-domain scenarios. Subtask B aims at doing multi-way machine-generated text detection and brings a new challenge of identifying the text source without knowing the domains in the test set. Subtask C proposes human-machine mixed text detection, which gives a text where the first part is human-written and the second part is machine-generated. The goal is to robustly determine the boundary where the change occurs using a fairly small training set. A detailed description can be found in the task description paper (Wang et al., 2024).

Currently, training-based machine-generated text detection strategies such as fine-tuning RoBERTa model (Solaiman et al., 2019) underperform in the out-of-domain scenario. Prominent zero-shot methods (Mitchell et al., 2023; Yang et al., 2023) can only discriminate whether a text is produced by a specific LLM or by a human. Sniffer (Li et al., 2023) and SeqXGPT (Wang et al., 2023a) appear to handle the problem of subtask B and C, but the claimed result is based on the assump-

*Corresponding author: Zhendong Mao.

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 1511–1522

tion that we know the origins of the text. Therefore, we propose our system, which consists of **D**omain-**A**daptive **Te**xt **D**etection (**DATeD**), **L**LM-Powered **L**anguage-**A**ware **M**odel (**LLAM**), **T**hree-stage **L**LM **E**mbeddings (**TLE**) and **Au**gmented **D**eBERTa **M**odel (**AuDM**). It performs outstandingly in out-of-domain scenarios, especially in subtask A.

In DATeD, we enhance performance by innovatively incorporating Domain Adversarial Neural Networks (DANN) (Ganin et al., 2016) into the task of machine-generated text detection. DANN consists of a feature extraction layer and a category predictor, forming the backbone network to predict classification labels. Furthermore, the domain classifier is connected to the backbone network through a gradient reversal layer for classifying domain labels. This enables the model to learn transferable features between the training and development set, effectively overcoming challenges in out-of-domain scenarios. We achieve **second** place out of **126** participants.

In LLAM, we handle text from various languages in a distinct manner. For English text, we employ LLM Embeddings, leveraging the powerful representation capabilities of LLM by directly extracting embeddings from the last layer of a proxy LLM, and we classify the text using a two-stage CNN. For text in other languages, we utilize LLM Sentinel, which reframes the classification task as a next-token prediction task. We win the **first** place in the track among **59** participants.

In TLE, we utilize the LLM Embeddings method mentioned above, with only a minor difference in the classification strategy. We rank **third** out of **70** participants.

In AuDM, we fine-tune a DeBERTa-base (He et al., 2021) model with a linear layer for token classification. This is an easy but effective system and ranks **third** out of **30** submissions.

In short, our contributions are as follows:

(1) We come up with a comprehensive system for machine-generated text detection in varies scenarios (Section §3), which significantly improves the performance compared to the baseline in all subtasks (Section §5).

(2) We utilize DANN in machine-generated text detection in DATeD (Section §3.1), and employ two adaptive strategies leveraging LLM capabilities in LLAM (Section §3.2).

(3) Extensive experimental analysis demonstrates the effectiveness of DATeD and LLAM (Section §5).

2 Related Work

2.1 Detecting LLM-Generated Text

The detection of machine-generated text is often expressed as a classification task. One way to solve this problem is to use supervised learning to train classification models on datasets that contain both machine-generated and human-written text. For example, GPTZero (Tian, 2023) collects human-written text from a variety of domains, including student-written articles, news articles, and question-and-answer datasets across multiple disciplines. G3Detector (Zhan et al., 2023) claims to be a general-purpose gpt generated text detector implemented by fine-tuning RoBERTa-large (Liu et al., 2019), however, the effect of text detection generated by multiple generators will be poor. T5-sentinel (Chen et al., 2023) trains RoBERTa and T5 (Raffel et al., 2023) classifiers on the OpenGPTText dataset they built, and then uses the T5 model's ability to predict the conditional probability of the next word to classify multiple text sources. SeqXGPT (Wang et al., 2023a) introduces the sentence-level detection challenge by synthesizing a dataset containing documents that have been polished with a Large Language Model. SeqXGPT uses sequence annotation methods to train its model and selects the most frequent class as sentence class, which provides a scheme for subtask C. However, a model explicitly trained to detect machine-generated text may overfit the training distribution of its domain (Bakhtin et al., 2019), resulting in poor generalization.

In addition, (Solaiman et al., 2019) notes the surprising power of a simple zero-shot method for machine-generated text detection, which thresholds candidate paragraphs based on their average log-probability under a generative model, a powerful baseline for many zero-shot learning machine-generated text detection tasks. DetectGPT (Mitchell et al., 2023) demonstrates that text sampled from LLMs tends to occupy regions of negative curvature of the model's log-probability function. Building upon this observation, a new curvature-based criterion is defined to determine whether a paragraph is generated by a given LLM. Its outstanding performance can only be guaranteed by a large disturbance function and a large number of perturbations, so more computational resources are required.

2.2 Domain Adversarial Neural Networks

Machine learning models typically assume that the training and test sets come from the same data distribution. However, labeled data is scarce, and it is the norm for unlabeled data, which may not align with the distribution of labeled data (HassanPour Zonoozi and Seydi, 2023), to constitute the majority of the data. In the task of machine-generated text detection, there are cases where the sources and generators differ between the source domain and the target domain. For instance, in the monolingual track of subtask A, the test set may include sources and models, such as BLOOMZ (Muennighoff et al., 2022), that are not present in the training set. Across data generated by different models, significant differences may exist in content style, text length, word frequency, and other distributions. This discrepancy results in models trained on labeled training data failing to generalize well to detect text data generated by other models. The core issue addressed by domain adversarial neural networks (Ganin et al., 2016) is mitigating the impact of inconsistent data distributions between the training and test sets on the performance of machine learning models. This enables models to learn sufficiently robust text representations and reduce differences in data distributions at the representation level. Introducing domain adversarial neural networks into monolingual generated text detection enhances the model's ability to generalize and transfer across different machine-generated text models (Chen et al., 2020).

3 System Overview

Our system consists of four components: DATeD, LLAM, TLE and AuDM. Each of these components addresses one of the four tracks of the task respectively.

In subtask A, we aim to do machine-generated text detection on mono/multilingual data. In the monolingual track, the domains in test set differ a lot from ones in training set. We innovatively apply DANN to the detection in DATeD. This is achieved by adding a gradient reversal layer on top of the base model. Additionally, besides category labels, we incorporate extra domain labels into the dataset (training set: 0, development set: 1), enabling the model to learn transferable features between the training and development set (Section §3.1).

In the multilingual track, another challenge is that the model is supposed to have robust generalization capabilities to adapt to the distinct characteristics of different languages, especially low-resource languages. We propose LLAM, which employs different methods for different languages. For text identified as English, we feed it into a proxy LLM to extract embeddings from the last layer and subsequently pass it through a two-stage CNN for classification. In the case of non-English text, we redefine the classification task as a next-token prediction task (Section §3.2).

In multi-way text detection, texts originate from human, ChatGPT, Cohere, Davinci, BLOOMZ and Dolly (Conover et al., 2023). The challenge lies in distinguishing texts generated by various LLMs. Therefore we conduct a three-stage classification based on the LLM Embeddings method (TLE) mentioned in the multilingual track to better fit the scenario of multi-classification (Section §3.3).

In human-machine mixed text detection, the target is to do fine-grained detection. Inspired by the wide use of BERT (Devlin et al., 2019) in sequence labeling task, we fine-tune a DeBERTa model with data augmentation (AuDM) to classify each token in a text (Section §3.4).

3.1 Domain-Adaptive Text Detection

The overall process of Domain-Adaptive Text Detection is illustrated in Figure 1. The model comprises three components: a feature extraction layer (such as RoBERTa) acquires text representation, a category predictor determines whether the given text is machine-generated, and a domain classifier is employed to mitigate differences in data distribution between the training and development sets, thereby enhancing the generalization capability of machine-generated text detection.

The feature extractor and label predictor constitute a feedforward neural network serving as the backbone of machine-generated text detection, utilized to classify data from the source domain. The label predictor employs MLP for classification, aiming to predict labels accurately. Following the feature extractor, we append an additional branch called the domain classifier. The domain classifier is interconnected through a gradient reversal layer to classify data in the feature space, determining whether it originates from the source domain or the target domain.

Forward propagation During forward propagation, given an input text of $\boldsymbol{n}$ tokens $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, we initially input the text $\boldsymbol{x}$ into the model. We opt for RoBERTa as the

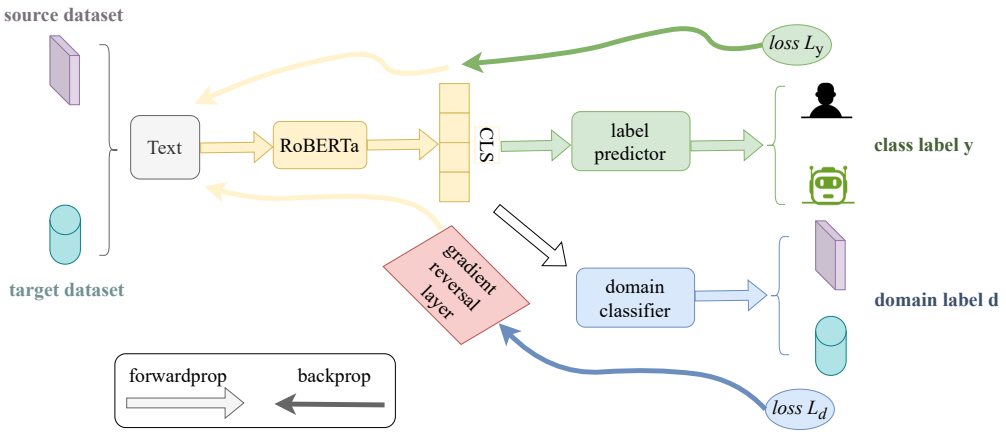

Figure 1: Overall architecture of DATeD

feature extractor $G_f(\cdot)$, utilizing the output vector corresponding to the CLS token as the semantic representation of the input text, denoted as $G_f(\boldsymbol{x})$. Subsequently, $G_f(\boldsymbol{x})$ is simultaneously fed into the label predictor $G_y(\cdot)$ and the domain classifier $G_d(\cdot)$, yielding $G_y(G_f(\boldsymbol{x}))$ and $G_d(G_f(\boldsymbol{x}))$, representing the category label $\boldsymbol{y}$ and the domain label $\boldsymbol{d}$, respectively (as shown in the green and blue sections depicted in Figure 1).

Back propagation During back propagation, the cross-entropy loss function is computed by comparing the category labels $\boldsymbol{y}$ predicted by the category predictor with the actual labels in the source domain, resulting in category loss (Ganin and Lempitsky, 2015). Additionally, we calculate the cross-entropy loss function by comparing the domain labels $\boldsymbol{d}$ classified by the domain classifier with all data from both the source and target domains, obtaining domain loss. It is worth mentioning that the gradient reversal layer behaves like a feedforward neural network during forward propagation. However, during backward propagation, the gradients are reversed (we achieved by multiplying by a negative identity matrix). Finally, the model updates its label predictor and domain classifier by summing up their respective losses (as shown in Formula 1). This setup enables the label predictor to distinguish categories in the source domain data (Ganin et al., 2016), while rendering the domain classifier unable to discern the origin domain of the data.

$$\mathcal{L}_{all} = \mathcal{L}_y + \lambda \mathcal{L}_d \tag{1}$$

$\mathcal{L}_y$ represents the label predictor loss in the source domain, $\mathcal{L}_d$ represents the domain classifier loss, λ is a hyperparameter.

3.2 LLM-Powered Language-Aware Model

In the multilingual machine-generated text detection task, we propose our model LLAM, which employs the language identification tool langdetect [1] to determine the language of the input text first. Then, we utilize LLM Embeddings and LLM Sentinel to detect English and non-English text respectively. The overall architecture of LLAM is depicted in Figure 2.

LLM Embeddings For English text, we use Llama-2-70B (Touvron et al., 2023b) as the proxy LLM to obtain embeddings of the input text. Given an input text of $\boldsymbol{n}$ tokens $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, we initially input the text $\boldsymbol{x}$ into the proxy LLM to get the token embeddings from the last layer. Subsequently, we calculate the average of token embeddings $\boldsymbol{h}$ to serve as the text representation. This representation is then inputted into a two-stage CNN for classification. In the first stage, the CNN extracts relevant features from the input representation $\boldsymbol{h}$. This process is accomplished through the utilization of three convolutional and pooling layers. In the second stage, the extracted feature is fed into three fully connected linear layers to output class probabilities $\boldsymbol{p}$. The model is trained by minimizing the cross-entropy loss.

LLM Sentinel (Chen et al., 2023) have proposed utilizing the base LLM's inherent next-token prediction ability for detection, advancing the field by choosing the T5 model as the base LLM. Drawing inspiration from this approach, for non-English text, we choose the mT5-large model as our proxy LLM. LLM Sentinel relies on the LLM's capability to predict the conditional probability of the next token. Given an input text of $\boldsymbol{n}$ tokens

[1] https://pypi.org/project/langdetect/

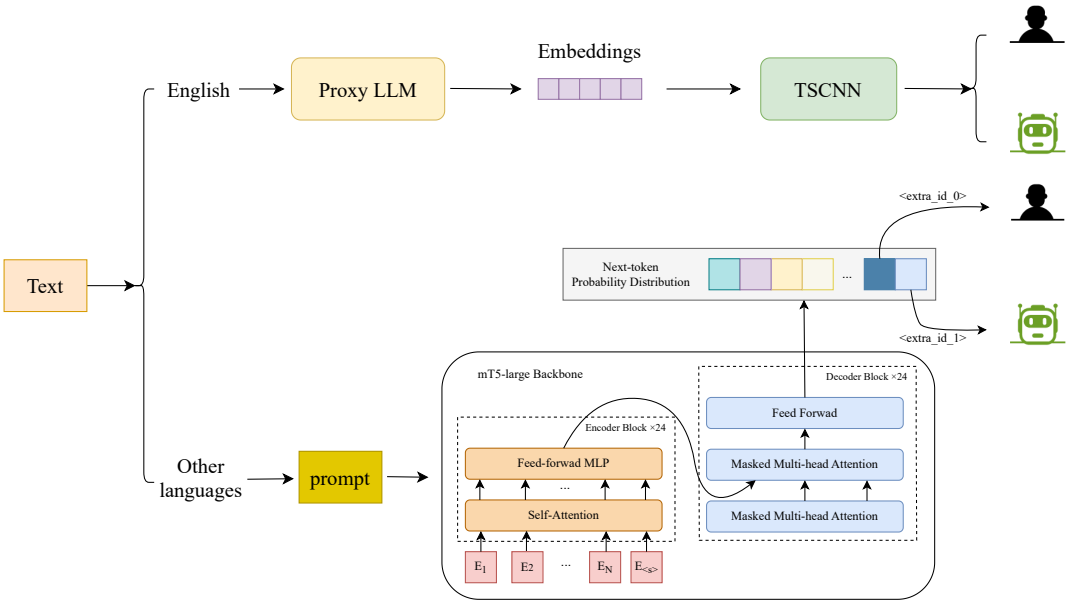

Figure 2: Overall architecture of LLAM

$\boldsymbol{x} = \{x_1, \cdots, x_n\}$, let Y denote the set of labels in this particular task, which contains "human" and "machine". We can establish a bijection $f : Y \rightarrow \mathcal{Y}$, where $\mathcal{Y}$ serves as a stand-in for the labels. Consequently, we reframe the binary classification task $\boldsymbol{x} \rightarrow Y$ as a next-token prediction task $\boldsymbol{x} \rightarrow \mathcal{Y}$. To accomplish this, we employ reserved tokens, $\mathcal{Y}$, which are not present in the text dataset. Specifically, We use $< extra\_id\_0 >$ for human and $< extra\_id\_1 >$ for machine. Therefore, the binary classification task can be effectively tackled using the LLM:

$$\hat{y} = f^{-1}\left(\arg\max_{y \in \mathcal{Y}} \mathbb{P}(y|\boldsymbol{x})\right) \tag{2}$$

Besides, in order to adapt to the characteristics of natural language, our team add a prompt before the detected text to fine-tune the LLM to perform it. The prompt used by our team is "Discern whether the following text is authored by a human or a machine. If human-written, respond with $< extra\_id\_0 >$; if machine-generated, respond with $< extra\_id\_1 >$: 'text'". Regarding the model's output, the decoding space consists of token probabilities for the entire dictionary. We simply need to extract the probability distribution over the set $\mathcal{Y}$, corresponding to the token probabilities for $< extra\_id\_0 >$ (human) and $< extra\_id\_1 >$ (machine). We then compare their magnitudes and select the larger of the two as the prediction result.

3.3 Three-stage LLM Embeddings

Building upon the LLM Embeddings approach outlined in Section 3.2, TLE additionally employs a three-stage classification process to address subtask B. Firstly, we distinguish between human-generated and machine-generated text. Subsequently, we categorize ChatGPT and Cohere as a single class for a four-class classification, differentiating them from Davinci, BLOOMZ, and Dolly. Given the challenges we encountered in distinguishing between Cohere and ChatGPT in our initial experiments, we proceed with a binary classification specifically focusing on ChatGPT and Cohere.

3.4 Augmented DeBERTa Model

In human-machine mixed text detection, we set up a model with a DeBERTa-base layer and a linear layer. The human token is classified as 0 and the machine token is classified as 1. The boundary is where the change of 0 to 1 occurs. We also perform data augmentation by employing Llama-2-7B to further generate training data.

4 Experimental setup

4.1 Datasets and Evaluation Metrics

Datasets The dataset for this task is an extension of the M4 (Wang et al., 2023b) dataset. Unlike the M4 dataset, this task samples human data to ensure data balance. New domains, generators, and languages appear in the test set to evaluate the generalization

ability of the algorithm. See Table 1 and Table 4 for the division of the dataset. See Appendix A for more details.

	Train	Dev	Test
Human	63351	2500	16272
Machine	56406	2500	18000
Total	119757	5000	34272

Table 1: Dataset division of monolingual track for subtask A.

	Train	Dev	Test
Human	83846	2000	20238
Machine	88571	2000	22140
Total	172417	4000	42378

Table 2: Dataset division of multilingual track for subtask A.

	Train	Dev	Test
Human	11997	500	3000
ChatGPT	11995	500	3000
Cohere	11336	500	3000
Davinci	11999	500	3000
BLOOMZ	11998	500	3000
Dolly	11702	500	3000
Total	71027	3000	18000

Table 3: Dataset division of subtask B.

Train	Dev	Test
3649	505	11123

Table 4: Dataset division of subtask C.

Evaluation Metrics The evaluation metrics for subtask A and B are accuracy. Accuracy is the ratio of the number of samples that the model predicts correctly to the total number of samples. Subtask C is evaluated using the MAE metric, which calculates the absolute difference between the predicted and actual boundary positions for each sample and takes the average value. The performance is better when MAE is smaller.

4.2 Training

Domain-Adaptive Text Detection In the monolingual track of subtask A, we initially define the training set and development set as the source domain and target domain respectively. Apart from the class labels provided by the dataset, we augment both the source and target domain datasets with additional domain labels. Specifically, the domain labels for the source domain samples are categorized into one class (e.g., labeled as 0), while the domain labels for the target domain samples are categorized into another class (e.g., labeled as 1). Since the overall loss computation includes both the label loss from the source domain and the domain loss from both the source and target domains, an equal number of samples from both the source and target domains is necessary to calculate the total loss. Therefore, it is imperative to balance the proportion of samples between the source and target domains. Please refer to Appendix B.1 for detailed settings.

LLM-Powered Language-Aware Model LLAM is composed of two parts: the LLM Embeddings model and the LLM Sentinel model. During the training process of our LLM Embeddings model, we split the training set into new training and development sets in a 9:1 ratio. The highest accuracy attained on the new development set during training is used to select the best checkpoint. Our final LLM Sentinel model is trained with the complete training set of this subtask and undergoes validation on the entire development set of the same subtask after each epoch. The final model chosen is the one demonstrating the optimal performance on the development set. For more details about the experiment, please refer to Appendix B.2.

Three-stage LLM Embeddings Our final model is trained in three stages, utilizing data from the corresponding categories in the subtask B dataset for each stage. See more details in Appendix B.3.

Augmented DeBERTa Model For our final submission, we do augmentation by adding the development set to the training set and using Llama-2-7B to continue generating based on training data. The checkpoint with the lowest MAE on the development set is chosen for submission. More details are in Appendix B.4.

5 Results

In this section, we report our results on all three subtasks and discuss our findings of the current

System	Accuracy
Baseline	88.46
Genaios	**96.88**
mail6djj	95.76
L3i++	85.83
QUST	84.16
USTC-BUPT (ours)	96.10

Table 5: Performance on subtask A: monolingual track.

System	Accuracy
Baseline	80.89
FI Group	95.84
KInIT	95.00
priyansk	93.77
L3i++	92.87
USTC-BUPT (ours)	**95.99**
w/o LLM Embeddings	92.03
w/o LLM Sentinel	82.05

Table 6: Performance on subtask A: multilingual track.

work. We provide the final submission results, as well as the results from several top-ranked systems.

5.1 Subtask A: Monolingual Track

In this part, we present a portion of the official results from the monolingual track and analysis of the selection process for target domain data.

5.1.1 Main Results

There are 126 teams that participate in the monolingual track. Due to the limited space, we only compare our system with the systems from teams Genaios, mail6djj, QUST and L3i++. The official results are shown in table 5. Our system achieves an accuracy of 96.10% and secures second place in the official ranking, surpassing the baseline of 88.46% by 7.64%. This indicates that adding domain adversarial neural networks solves the impact of inconsistent data distribution between the training and test set, enabling the model to learn transferable features between the two sets, thus significantly improving model performance. Upon reviewing the methods published by other participants, we find that our result (96.10%) is not far from Genaios (96.88%). Notably, while Genaios utilizes the larger Llama-2-13B model, we achieve similar performance using the smaller RoBERTa-base model.

5.1.2 Target Domain Data Selection

We conduct a series of experiments regarding qualitative and quantitative data selection. **Qualitative analysis** aims to verify whether the target domain utilizes the development set or the test set. This is because the generative models utilized in the training set and the test set may overlap. Using the test set directly as the target domain could result in texts generated by the same generative model (belonging to the same domain) being assigned different domain labels. According to our submitted results, training with the development set as the target domain results in the best performance with an accuracy of 96.10%. However, training with the test set as the target domain results in an accuracy of only 88.70%. Observing the distribution of generative models in the test set further validates these findings. The test set comprises existing models such as ChatGPT, Cohere, and also features the emergence of a new generation model, GPT-4. Thus, utilizing the test set as the target domain could lead to misdefined domain labels.

Quantitative analysis aims to explore how many repetitions of the target domain could yield better results. This is because updating the loss necessitates domain label loss from both the source domain and target domain, as discussed in Section §4.2 about DATeD, which requires an equal number of samples from each. Hence, the number of target domain samples to be duplicated needs exploration. According to our submitted results, repeating the target domain 15 times yields nearly the same number of samples as the source domain, resulting in the best performance with an accuracy of 96.10%. When repeated three times, the detection accuracy in the test set decreased to 91.75%.

5.2 Subtask A: Multilingual Track

In this part, we offer partial results from the leaderboard in the multilingual track and perform an ablation study.

5.2.1 Main Results

There are 59 teams that participate in the multilingual track. Due to the limited space, we only compare our system with the systems from teams FI Group, KInIT, priyansk, L3i++. The official results are shown in Table 6. Our system achieves the best result in the official ranking with 95.99% accuracy, surpassing the baseline by 15.10%. This showcases the powerful representational capacity of LLMs and encourages us to explore further strategies to leverage it.

System	Accuracy
Baseline	74.61
AISPACE	**90.85**
Unibuc-NLP	86.96
dianchi	83.48
L3i++	83.12
USTC-BUPT (ours)	84.33
w/o three-stage strategy	80.94

Table 7: Part of the official results for subtask B.

System	MAE
Baseline	21.535
TM-TREK	**15.684**
AIpom	15.940
Fine-tuned RoBERTa-large	20.876
Fine-tuned DeBERTa-large	18.075
USTC-BUPT (ours)	17.702
with Bi-LSTM	16.556

Table 8: Part of the official results for subtask C.

5.2.2 Ablation Study

We conduct extensive ablation experiments to show the effectiveness of LLM Embeddings and LLM Sentinel respectively. The results are shown in Table 6. When we remove the language discriminator and only use the LLM Sentinel method, the accuracy drops to 92.03%. Since mT5 is designed for multilingual text-to-text tasks, its training corpus may be more biased towards encompassing texts in diverse languages rather than focusing on a specific language, such as English. Consequently, this could result in inferior performance on English text classification tasks due to the model's lack of exposure to a sufficient amount of English texts for optimal training. However, when we solely utilize the LLM Embeddings method, the accuracy decreases to 82.05%. Since most of the training data for Llama-2-70B is in English, its ability to comprehend other languages is limited. The potential of other multilingual LLMs awaits exploration in future research. A more detailed discussion is in the Appendix C.

5.3 Subtask B: Multi-Way Track

Our final submission achieves an accuracy of 84.33%, marking a 9.72% improvement over the baseline, as shown in Table 7. This indicates that LLM can capture subtle differences between different models, allowing for classification based on these distinctions. Furthermore, rather than directly implementing multi-classification, we embrace a three-stage strategy. This results in an enhancement in model performance from 80.94% to 84.33%, suggesting that we can prioritize the classification of categories with a significant gap before handling the others.

5.4 Subtask C: Mixed Track

After the release of the golden label, we test the performance of our model with a Bi-LSTM (Zhou et al., 2016) layer. The MAE of our final submission is 17.702, and the new result is 16.556, only slightly larger than the SOTA benchmark by TM-TREK but saliently smaller than the baseline as seen in Table 8. This shows the strong ability of DeBERTa to extract effective contextualized features, while LSTM (Hochreiter and Schmidhuber, 1997) helps process sequential information in the text. The result compared with DeBERTa-large also shows that the effectiveness of the encoder model is not linear with the scale.

6 Conclusion

In conclusion, this paper presents the development and performance of our system for the SemEval-2024 Task 8. Our system wins the multilingual track and secures second place in the monolingual track. Additionally, we attain third place in both subtask B and subtask C. We demonstrate the efficacy of incorporating DANN, which significantly enhances out-of-domain accuracy by introducing a gradient reversal layer and integrating additional domain labels. Leveraging LLM embeddings proves to be a straightforward yet effective method, harnessing the representation capabilities of LLM without fine-tuning. Furthermore, our implementation of LLM Sentinel exhibits remarkable performance, especially in low-resource language scenarios. In the future, we plan to investigate the application of DANN to multi-label classification scenarios and explore more effective strategies to leverage LLM embeddings.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.62232006, 62222212, 62121002, 62376033).

References

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *ArXiv*, abs/1906.03351.

Ning Bian, Hongyu Lin, Peilin Liu, Yaojie Lu, Chunkang Zhang, Ben He, Xianpei Han, and Le Sun. 2023. Influence of external information on large language models mirrors social cognitive patterns.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Token prediction as implicit classification to identify LLM-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13112–13120, Singapore. Association for Computational Linguistics.

Zhuyun Chen, Guolin He, Jipu Li, Yixiao Liao, Konstantinos Gryllias, and Weihua Li. 2020. Domain adversarial transfer network for cross-domain fault diagnosis of rotary machinery. *IEEE Transactions on Instrumentation and Measurement*, 69(11):8702–8712.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks.

Hans W. A. Hanley and Zakir Durumeric. 2024. Machine-made media: Monitoring the mobilization of machine-generated articles on misinformation and mainstream news websites.

Mahta HassanPour Zonoozi and Vahid Seydi. 2023. A survey on adversarial domain adaptation. *Neural Processing Letters*, 55(3):2429–2469.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Linyang Li, Pengyu Wang, Ke Ren, Tianxiang Sun, and Xipeng Qiu. 2023. Origin tracing and detecting of llms.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. 2022. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue.

OpenAI. 2023. Gpt-4 technical report.

Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. 2023. On the risk of misinformation pollution with large language models.

Mike Perkins, Jasper Roe, Darius Postma, James McGaughran, and Don Hickerson. 2023. Detection of gpt-4 generated text in higher education: Combining academic judgement and software to identify generative ai tool misuse. *Journal of Academic Ethics*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Neha Sengupta, Sunil Kumar Sahu, Bokang Jia, Satheesh Katipomu, Haonan Li, Fajri Koto, Osama Mohammed Afzal, Samta Kamboj, Onkar Pandit, Rahul Pal, et al. 2023. Jais and jais-chat: Arabic-centric foundation and instruction-tuned open generative large language models. *arXiv preprint arXiv:2308.16149*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *ArXiv*, abs/1908.09203.

Edward Tian. 2023. GPTZero: An ai text detector.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Christoforos Vasilatos, Manaar Alam, Talal Rahwan, Yasir Zaki, and Michail Maniatakos. 2023. Howkgpt: Investigating the detection of chatgpt-generated university student homework through context-aware perplexity analysis.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. 2023a. Seqxgpt: Sentence-level ai-generated text detection.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, jinyan su, Artem Shelmanov, Akim Tsvigun, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Chenxi Whitehouse, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. Semeval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 2041–2063, Mexico City, Mexico. Association for Computational Linguistics.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, and Preslav Nakov. 2023b. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. *ArXiv*, abs/2305.14902.

Xianjun Yang, Wei Cheng, Linda Petzold, William Yang Wang, and Haifeng Chen. 2023. Dna-gpt: Divergent n-gram analysis for training-free detection of gpt-generated text. *ArXiv*, abs/2305.17359.

Haolan Zhan, Xuanli He, Qiongkai Xu, Yuxiang Wu, and Pontus Stenetorp. 2023. G3detector: General gpt-generated text detector.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 207–212.

A Data

In the monolingual track of subtask A, the dataset contains both human-written and machine-generated text. Different from the training and development set, all the data in the test set comes from a new area: student essays Outfox, and a new generator GPT-4 appears to generate machine text.

In the multilingual track, the text is not only in English, but also in Chinese, German, Russian and other languages. Compared to the training and the development set, two new fields of Outfox and Italian text appear in the test set, and new generators Llama-2-finetune and Jais-30B (Sengupta et al., 2023) are used to generate machine text.

In the dataset of subtask B, the generators remain the same in the training set, development set and test set, including Human, ChatGPT, Cohere, Davinci, BLOOMZ and Dolly. However, the text of the test set is only from the student essays Outfox instead of wikiHow, etc.

Each text of the subtask C dataset is composed of human-written text and machine-generated text, and its label is an index, representing the boundary where the change occurs.

Tables 9 to 12 detail the data sources and the distribution of the model, which is conducive to evaluating the model's generalization ability.

	Train	Dev	Test
wikiHow	✓	✓	
Wikipedia	✓	✓	
Reddit	✓	✓	
arXiv	✓	✓	
PeerRead	✓	✓	
Outfox			✓

Table 9: Source distribution of subtask A monolingual track.

	Train	Dev	Test
Human	✓	✓	✓
ChatGPT	✓		✓
Cohere	✓		✓
Davinci	✓		✓
Dolly	✓		✓
BLOOMZ		✓	✓
GPT-4			✓

Table 10: Model distribution of subtask A monolingual track.

	Train	Dev	Test
wikiHow	✓		
Wikipedia	✓		
Reddit	✓		
arXiv	✓		
PeerRead	✓		
Bulgarian	✓		
Urdu	✓		
Indonesian	✓		
Chinese	✓		
Russian		✓	
Arabic		✓	✓
German		✓	✓
Outfox			✓
Italian			✓

Table 11: Source distribution of subtask A multilingual track.

	Train	Dev	Test
Human	✓	✓	✓
ChatGPT	✓	✓	✓
Cohere	✓		✓
Davinci	✓	✓	✓
Dolly	✓		✓
BLOOMZ	✓		✓
Llama 2			✓
Jais-30B			✓

Table 12: Model distribution of subtask A multilingual track.

B Detailed Experimental Setup

B.1 Domain-Adaptive Text Detection

Typically, the number of samples in the training set far exceeds that in the development set, which can also be observed in this competition dataset. The imbalance between the domain labels of the source and target domains is substantial, with 119,757 samples in the source domain and only 5000 samples in the target domain. To address this issue, we innovatively repeat the target domain 15 times to achieve a nearly 1:1 ratio of domain labels between the source and target domains, without compromising the genuine domain and classification label values of the target domain.

We opt to utilize the pre-trained model RoBERTa-base as the feature extraction layer for DANN to extract features from the text to be detected. Subsequently, we input the text information separately into the label classifier and the domain classifier. Apart from the batch size, which is set to 32, which differs from the baseline, all other hyperparameters remain consistent with the baseline. Specifically, the learning rate is set to 2e-5, and the optimizer selected is AdamW (Loshchilov and Hutter, 2017). The maximum token truncation length for the text is set to 512 tokens. We conduct training for 10 epochs on a single NVIDIA A40 40GB GPU.

B.2 LLM-Powered Language-Aware Model

LLM Embeddings We utilize the int8 quantized variant of Llama-2-70B as the proxy LLM for obtaining embeddings on a single NVIDIA A800 80GB GPU, with the maximum length set to 1024. For the two-stage CNN, the input channel is set to 1. A total of three convolutional layers are employed, with the number of kernels being 32, 64, 96 respectively. The sizes of their corresponding kernels are 24, 16, 8. We use the AdamW optimizer with a linear warmup decay learning schedule and a dropout of 0.1. The batch size and learning rate are set to 128 and 3e-4, and models are trained for 50 epochs.

LLM Sentinel We fine-tune the mT5-large model for 15 epochs using two NVIDIA A40 GPUs. Throughout this process, we utilize the Adafactor optimizer (Shazeer and Stern, 2018) to minimize GPU memory usage and expedite training. The optimizer utilizes the following hyperparameters: a learning rate of 1e-3, stability parameters of (1e-30, 1e-3), gradient clipping threshold of 1.0, learning rate decay rate of -0.8, momentum parameter set to None, weight decay of 0.0, relative step set to False, parameter scaling set to False, and warm-up initialization set to False. The maximum length constraint is set to 1024.

B.3 Three-stage LLM Embeddings

The hyper-parameters of this experiment are consistent with the method mentioned above for LLM embeddings. Please refer to Appendix B.2 for more details.

B.4 Augmented DeBERTa Model

All hyper-parameters synchronize with the baseline. We only change the model structure and fine-tune it using a single NVIDIA GeForce RTX 3090 GPU.

Method	Accuracy
T5-small (directly)	77.06
T5-small (prompt)	**87.76**
LLM Embeddings (English)	**97.30**
LLM Sentinel (English)	82.04
mT5-large	**90.58**
mT5-xl	73.56

Table 13: Performance comparison of different methods on the development set.

C More Analysis of Multilingual Track

C.1 Prompt Impact

In the initial stages of the experiment, we compared the impact of adding prompts on model performance for monolingual binary classification tasks. We trained and tested the T5 model using the monolingual training and development sets, respectively. The experimental results (Table 13) indicate that adding prompts could effectively enhance the model's performance on this task. Therefore, for multilingual task, we directly adopt the approach of adding prompts.

C.2 Language-Aware Strategy

During the experimental phase, we compared the performance of LLM Embeddings and LLM Sentinel on English texts. We trained them using the monolingual training set of subtask A and validated the monolingual development set. The experimental results are presented in Table 13. The results demonstrate that LLM Embeddings outperform LLM Sentinel. Consequently, for English text, we opt for LLM Embeddings.

C.3 LLM Selection

In the later stages of the experiment, we also explored larger models, such as mT5-xl. Considering that the test set in the competition does not include Russian, we evaluated the performance of mT5-xl on languages other than Russian for this task. We trained the mT5-xl model using the whole multilingual training set and utilized texts from languages other than Russian in the multilingual development set as the new development set. The training was conducted with the same experimental parameters as mT5-large (see details in Appendix B.2). We compare the best accuracy results of mT5-large and mT5-xl on the new development set (Table 13). The experimental results indicate that employing larger models with more parameters does not yield better experimental outcomes for this task. Therefore, we choose mT5-large as our LLM.