Coarse-to-Fine AI Text Detection: Multi-Granularity Contrastive Learning in Dual Stages

Anonymous Author(s)
Affiliation
Address
email

Abstract

As the ability of Large Language Models (LLMs) to imitate human writing becomes stronger and the diversity of machine texts' humanizing methods continuously increases, current AI text detectors have shown vulnerability. The monolithic threshold-based scoring mechanisms of many detectors exhibit severe performance degradation when confronting strategically humanized texts generated by advanced LLMs, which needs to be improved. Given that current humanizing methods can be roughly divided into four levels of character, word, sentence, and paragraph, we targetedly propose a Hierarchical and Multi-Granularity Contrastive Learning Detection framework and the detecor, HiGraDe. The coarse layer of HiGraDe filters out simple samples, while hard-to-detect samples like humanized machine texts will be fine-grainedly discriminated through the subdivision layer which applies a multi-granularity contrastive learning strategy. This hierarchical and multi-granularity framework makes up for the loophole that humanized machine texts can successfully escape the traditional detector after a single detection, and shows excellent robustness in the task of detecting humanized machine texts. In addition, our framework is flexiable, the subdivision layer can be deployed separately on the existing detector as a plug-and-play patch to tremendously improve their performance when facing large-scale humanized machine texts. We hope this multi-stage training framework can inspire new sparks in the field of AI-generated text detection. Our codes and datasets are open[1].

1 Introduction

The explosive rise of LLMs [6, 8, 16] makes it easy for people to get machine-generated texts. Just like a coin which has two sides, new problems emerge when the text generation function of LLMs facilitates people's work and life. Researchers have demonstrated various malicious applications of LLMs, including academic fraud [42], spam generation, and false information dissemination [20, 55]. In order to prevent machine-generated texts from leaning into the wrong direction, machine-generated text detectors (AI text detectors) come into being to correct the development of LLMs. Existing detectors include those based on statistics and mathematics [41, 49], watermarks [17, 25], classifiers [18, 51], to name a few. Among them, a considerable number of detectors judge whether the text is generated by machine based on the comparison of the final text's score with a certain threshold. For instance, when the test score of a text is greater than 0.5, it is considered to be written by a machine, otherwise it is considered to be written by a human. However, as large quantities of machine texts become easier to obtain and the humanizing methods become more diverse [59, 23], it is not enough for detectors to depend solely on this monolithic threshold-based scoring mechanisms, and further improvement is needed.

[1]Available at https://github.com/simonsshoot/Coarse-to-Fine-AI-Text-Detection

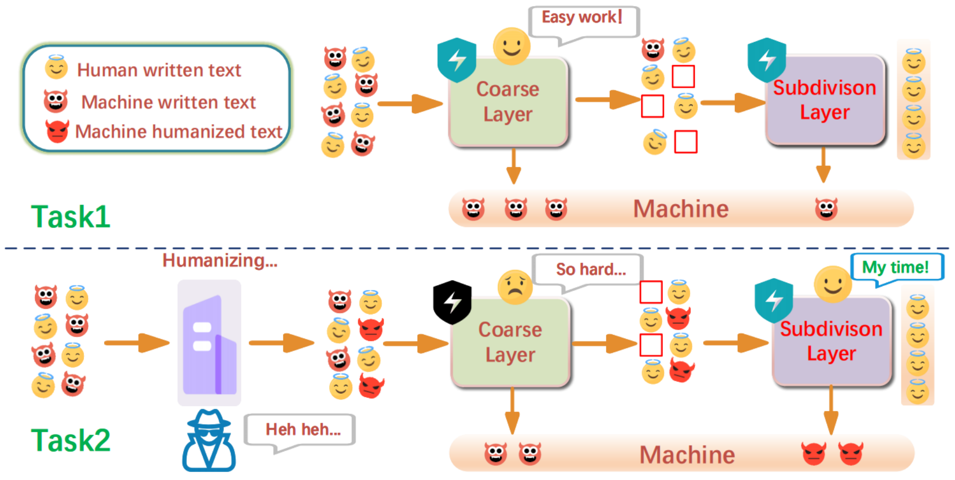

Figure 1: HiGraDe's framework overview under two scenarios. (a) Task 1: detecting human texts and machine texts. (b) Task 2: detecting human texts and humanized machine texts, original machine texts are also included. Task 1 is simple and traditional while Task 2 is more complex and realistic.

There are many ways of machine humanizing methods at present, such as simulating human spelling errors, word replacement, sentence back translation and so on, which can be roughly divided into four levels: character, word, sentence, and paragraph [59]. The vulnerability of detectors in the face of these attacks has also been pointed out by many researchers [12, 26]. Since single-stage detection can be easily bypassed by humanized machine texts, we believe it is necessary to conduct a dual stage and coarse-to-fine detection. To detailedly speak, when the scores of texts are within the machine category, these texts are unhumanized and easily detectable machine texts so the detection results are directly reached. When the texts' scores are within the human category, the texts may not only be human written, but also humanized machine texts, and a fine-grained detection is required. In other words, we add an extra gate to the detector, so that humanized machine texts not only need to pass the coarse detection, but also have to undergo an extra fine-grained inspection with multi-granularity contrastive learning, which greatly reduces the possibility of machine texts evading detection and improves the robustness of detector in the face of attacks.

Our detector, HiGraDe, consists of two layers. For the coarse screen layer, it is trained based on a general classifier framework, while for the subdivision layer, its training data comes from samples judged by the coarse screen layer as human writing (including human writing samples and humanized machine samples). In order to further distinguish human samples and machine samples humanized at different levels, we prescribe the right medicine by using multi-granularity contrastive learning to learn the similarities and differences between them. Concretely, for machine texts using the same humanizing method, their distance in the sample space should be close, while the distance from texts using different humanizing methods should be far; for texts humanized at the same level, their distances in the sample space should be close while far from samples humanized at other levels; for machine texts humanized at different levels along with original machine texts, their distances in the sample space should be close to each other and far from human texts. The final result can be obtained by integrating the results of the coarse detection and the fine-grained detection, which shows the vulnerability of the coarse screen layer (as a representative of the current common detectors) in the face of diversified attacks, the necessity and effectiveness of the subdivision layer, and the excellent performance of our detector that ultimately exceeds the baseline.

We not only propose a new detector with good performance, but also provide a new training paradigm for subsequent AI text detectors. While for existing AI text detectors, the subdivision layer can serve as a backup in the face of large-scale humanized machine texts detection, providing them with a possible plug-and-play patch to further improve their robustness.

In short, our work is multifaceted and can be summarized as follows:

- We propose a new framework for training detectors, with coarse stage targeting at detecting original machine texts and subdivison stage focusing on humanized machine texts, providing ideas for detector designers to resist the humanizing attacks.
- We use multi-granularity contrastive learning in subdivision layer to achieve distinction between human texts and machine texts humanized at different levels, which can also be used as a plug-and-play patch to improve the robustness of existing detectors.
- We conduct extensive experiments on two tasks of detecting original machine texts and humanized machine texts, results show that our detector has achieved State-Of-The-Art (SOTA) performance, we also apply our subdivision layer to current detectors and find improvement of F1-score up to 30%.

2 Related Works

Multi-stage detection. The advantage of multi-stage detection is that it can subdivide the detection process and make it more granular. Jiang [24] and Feng [13] divide the software vulnerability detection process into coarse-grained detection and fine-grained positioning. Cao [2] proposes a two-stage tax evasion detection. Concone [7] uses low-cost computing features to quickly filter easy-to-classify samples in spam detection and only performs fine-grained distinction on difficult samples. Duan [11] uses two classifiers to coperform object detection. The multi-stage detection framework is also widely used in detecting video event abnormality, network attack, conversation emotion, false information, and so on [53, 52, 29, 30, 28].

Humanizing methods of machine texts. Many researchers have pointed out the vulnerability of current detectors, indicating even a small perturbation attack can cause the performance of the detector to drop sharply [59, 12, 26, 32, 23, 54]. Specifically, Liu [32] points out that DetectGPT relies on the threshold setting of the logit regression module (coincides with our motivation), which is sensitive to the detection results, and the perturbations of deletion, duplication, insertion, replacement imposed on the test data cause the performance of AI text detector to drop significantly; Dugan [12] designs a variety of perturbations such as local vocabulary replacement, syntactic structure adjustment, semantic preservation rewriting, finding that with only 5% of the text content being modified, the detection accuracy drops by an average of 37.2%; Zhou [59] concludes humanizing methods into four levels of character, word, sentence, and paragraph on a variety of detectors (we apply this idea), pointing out that current detectors need to be trained with adversarial texts. Their works show that lacerating the mask of humanized texts is an urgent affair which current detectors need to solve.

3 Model and Methodology

3.1 Workflow Under two Scenarios

Zhou and Huang's work [59, 23] extensively explore the vulnerability of detectors to different attacks. Based on their work, we classify the existing common humanizing methods into four levels of char, word, sentence, paragraph, which consistent with Zhou's summary of present attacks [59], detailed in Appendix C. We define two tasks for AI text detectors, in which Task 1 is relatively simple, Task 2 is more realistic:

$$\begin{aligned}
&\text{Task 1}: \quad \text{Detecting human texts and machine texts}\\
&\text{Task 2}: \quad \text{Detecting human texts, machine texts and humanized machine texts}
\end{aligned}$$

Given a set of unhumanized machine texts $X = \{x_1, x_2, ..., x_k\}$ and human writing texts set $Y = \{y_1, y_2, ..., y_k\}$, where k represents the number of texts, an attacker with ulterior motives may humanize the original texts of machine at character, word, sentence, or paragraph level to try to evade detection, thus there will be sets of humanized machine texts:

$$X_{level} = \{x'_1, x'_2, \ldots, x'_k\}, \quad level \in \{char, word, sentence, paragraph\}$$

Given a sample space S in real world detection,we have $S = X_{char} + X_{word} + X_{sentence} + X_{paragraph} + X + Y$. For a sample $s \in S$, our goal is to ultimately determine whether it is machine generated. Sample s will first pass through the coarse screen layer, the final text's score is compared

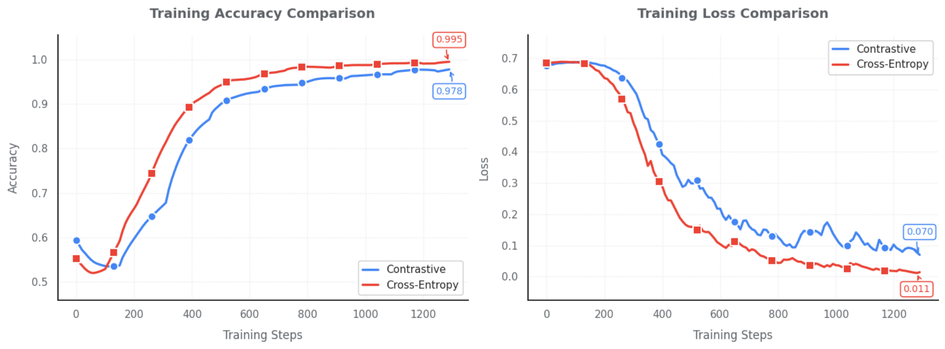

Figure 2: Contrastive loss VS cross-entropy loss. Model loss based on cross entropy converges faster and more stably, and its training accuracy is higher than that of the model based on contrastive learning for Task 1 and 2.

with the threshold and if it is in the machine range, then the conclusion that s is machine generated is directly reached, indicating that some simple machine samples are successfully identified by the first layer. If the text is judged to be human written, the text may be real human text or machine written text with humanizing methods. Sample s which is judged as human by the first layer will undergo the subdivision layer with multi-granularity contrastive learning for fine-grained differentiation, ultimately determining whether s is indeed a humanized machine text.

3.2 Two-layer Combined Training Paradigm

Blending the strengths of both——our training idea: With Concone's work [7] in spam detection and Duan's work in object detection [11] guiding the way, we leverage cross entropy and contrastive learing. For Task 1, the classifier based on cross-entropy loss has the advantages of stable optimization process, rapid convergence, and applicability to mutually exclusive scenarios [10, 38, 56]. Therefore, we choose it as the main body in the coarse screen layer, hoping to take advantage of its strengths and filter out the original machine texts which are simple. For the more realistic task of detecting human texts and humanized machine texts (Task 2), cross-entropy loss lacks sensitivity to intra-class differences [33, 48]. Therefore, we introduce multi-granularity contrastive learning as the subdivision layer, hoping it can further learn the feature representation of humanized machine texts and human texts.

In general, the coarse screen layer quickly and roughly screens texts, solving the problem that the detector based on contrastive learning has relatively high training computational overhead when directly facing large-scale text detection and limitation in learning human texts' feature representation when facing data imbalance [33, 43]. The subdivision layer based on multi-granularity contrastive learning, in turn, solves the problem of coarse screen layer being insensitive to intra-class differences. Therefore, our two-layer detector HiGraDe can achieve better performance in both detecting original machine texts and detecting humanized machine texts. Better performance and lower costs are the reasons we choose "two" rather than "one", shown in Figure 2 and Appendix E.

3.3 Coarse Screen Layer

The coarse screen layer is a traditional classifier-based detector. Its purpose is to perform coarse-grained screening of input batch texts. This layer can also be used as a representative of the current detector that has not been specially trained with text perturbations. A good coarse screen layer should be able to filter out most of the original machine texts that have not been humanized. Its loss function is as follows,in which l represents true label, p represents predicted label:

$$\mathcal{L}_{ce} = -\frac{1}{N}\sum_{i=1}^{N} l_i \cdot log(p_i) + (1 - l_i) \cdot log(1 - p_i), \tag{1}$$

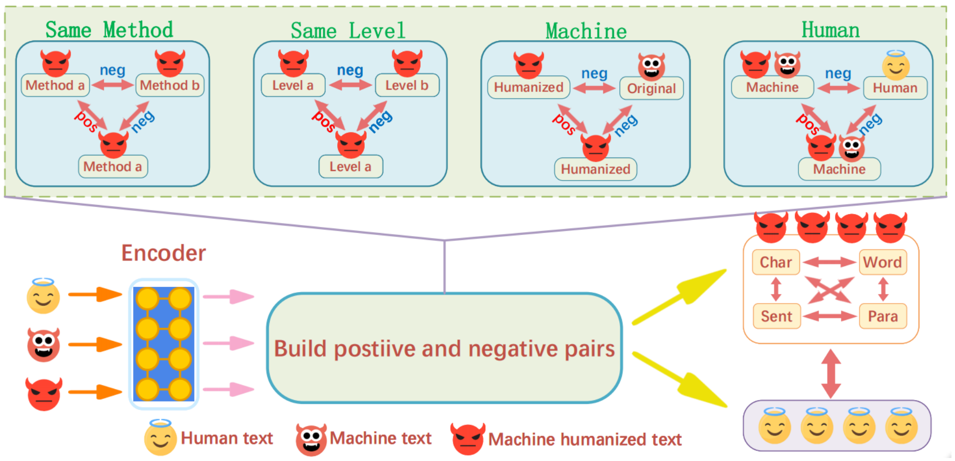

Figure 3: Overview of the subdivision layer. After the filtered texts are encoded by the encoder, their deeper features are learned through multi-granularity contrastive learning, which pulls in the same positive samples and pushes away the negative samples to distinguish human texts and machine texts.

For input $s \in S$, the text encoding $\Phi(s)$, the output is:

$$Prediction(s) = \begin{cases} human, score(\Phi(s)) > threshold \\ machine, scores(\Phi(s)) < threshold \end{cases} \tag{2}$$

3.4 Subdivision Layer

As elaborated in subsection 3.1 and Figure 1, there are three levels of multi-granularity contrastive learning in the subdivision layer: machine texts using the same humanizing method and machine texts with different humanizing methods, machine texts using the same level of humanizing method and machine texts with different levels of humanizing methods (not humanized is seen as a special level of humanizing method), human texts and machine texts. Given a text label triple (p, q, l), where p represents the humanizing method used, q represents the level of the humanizing method, and l represents the source of the sample (human or machine), we have the cosine similarity constraints, where $s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9 \in S$:

$$\begin{cases} Sim(\Phi(s_1), \Phi(s_2)) < Sim(\Phi(s_1), \Phi(s_3)), p(s_1) = p(s_2), p(s_1) \neq p(s_3) \\ Sim(\Phi(s_4), \Phi(s_5)) < Sim(\Phi(s_4), \Phi(s_6)), q(s_4) = q(s_5), q(s_4) \neq q(s_6) \\ Sim(\Phi(s_7), \Phi(s_8)) < Sim(\Phi(s_7), \Phi(s_9)), l(s_7) = l(s_8), l(s_7) \neq l(s_9) \end{cases} \tag{3}$$

For contrastive learning at a specific level, we use the contrastive loss based on Guo's framework [19], which takes the form of a negative logarithmic aggregation function, we have the loss expression Eq. 4, in which s represents targeted sample, S_{K+} is a set of positive samples, S_{K-} is a set of negative samples, τ is the temperature coefficient.

$$\mathcal{L}_p = -\log \frac{\exp\left(\sum_{k\in K^+} \frac{s(p,k)}{\tau} / S_{K^+}\right)}{\exp\left(\sum_{k\in K^+} \frac{s(p,k)}{\tau} / S_{K^+}\right) + \sum_{k\in K^-} \exp\left(\frac{s(p,k)}{\tau}\right)}. \tag{4}$$

We take contrastive loss in label p as example,label q and l's loss are consistent with the above equation.

The final contrastive learning loss should be the sum of the contrastive loss at different levels above, so we have Eq. 5, where l_i represents the label l that whether the i_{th} sample belongs to human or machine, K indicates the sum of all samples entering the subdivision layer, and $\mathcal{L}_l$, $\mathcal{L}_p$, and $\mathcal{L}_q$ represents the loss of the subdivision layer at above level:

$$\mathcal{L}_{contrastive-tot} = \sum_{i=1}^{K} l_i \cdot \mathcal{L}_{l-human} + (1 - l_i) \cdot (\mathcal{L}_p + \mathcal{L}_q + \mathcal{L}_{l-machine}). \tag{5}$$

Through multi-granularity contrastive loss function propagation, the model can distinguish the differences between machine texts humanized at different levels and the similarities between machine texts humanized at the same level in a fine-grained manner. We introduce the cross-entropy loss function Eq. 1 to drive the model to improve performance in the final binary classification task, the final loss is:

$$\mathcal{L}_{final-loss} = \mathcal{L}_{contrastive-tot} + \mathcal{L}_{ce}. \tag{6}$$

4 Experiments

4.1 Experimental Setup

Datasets. We use three datasets that are widely used for detector training and detection. Detailed dataset information in Appendix D. **HC3** [18]: A high-quality dataset specifically for fine-tuning dialogue models, containing QA question-answer pairs in multiple fields, each question corresponds to at least one human answer and one machine-generated answer, focusing on multiple open-ended questions such as finance and medicine. **SeqXGPT-Bench** [51]: A benchmark dataset designed specifically for sentence-level AI generated text detection tasks, containing text generated from multiple LLMs (such as GPT-2, GPT-Neo, GPT-J, LLaMa, and GPT-3). The dataset uses feature alignment design to align word-level log probabilities to a common vocabulary. **CheckGPT** [36]: This dataset contains 900,000 samples, generated by ChatGPT based on prompts, covering multiple fields such as news, reviews, and literatures.

Evaluating Metrics. In order to systematically and thoroughly evaluate our work and the work of others, we use Accuracy (ACC), F1-score (F1), and Recall as the standard. ACC: The proportion of correctly predicted samples to the total number of samples, which directly reflects the overall prediction accuracy of the model, but may be distorted when the categories are unbalanced; Recall: The proportion of correctly predicted samples among samples that are actually positive, which measures the model's coverage of positive samples, but may have a high false alarm rate; F1: The harmonic average of precision and recall, which balances Precision and Recall and avoids the one-sidedness of a single indicator. We use the three in combination with the hope of evaluating model performance more comprehensively.

Baseline Detectors. To verify the effectiveness of our method, we select the following five representative detectors as baselines and compare them with HiGraDe. Considering that machine texts may be humanized in reality, we specially select three baseline detectors that have gone through adversarial training or text perturbation training. **SimpleAI** [18]: Fine-tune the pre-trained RoBERTa model, filter the patterned words in the training data to improve generalization ability, and add sentence-level data to enable the model to capture local features. **Watermark** [25]: The watermark embeds the signal by biasing the "green token list" at generation time, and the detector counts the actual number of green tokens in the texts. It is completely independent of the generation model, avoiding the overhead of traditional model training while ensuring the robustness and interpretability of the detection. **CoCo** [34]: By constructing a coherence graph to capture the entity interaction structure of the text and introducing a supervised contrastive learning framework, the model's understanding of language patterns is enhanced. **RADAR** [22]: Using the adversarial learning framework of generator and discriminater along with some instruction tuning, model shows excellent robustness and transferability. **PECOLA** [32]: The noise introduced by random perturbations is reduced through selective perturbation strategies, the key features of the text are retained, and the contrastive learning strategy is further used to enhance the robustness.

4.2 Results and Analysis

In order to better illustrate the effectiveness of our method, we retrain the existing baseline detectors on the three datasets of HC3, SeqXGPT and CheckGPT according to the method described in their papers and compare them with HiGraDe. Among them, RADAR does not provide source code, so we use the API they provide. In addition, the watermark-based detector does not need to be trained, its processing work is to add watermarks to the input dataset. We first conduct tests on Task 1. The results are shown in Table 1. For HC3 and SeqXGPT datasets, our method outperforms all baseline detectors in all evaluating metrics. For the CheckGPT dataset, we achieve the best in F1, ACC, and

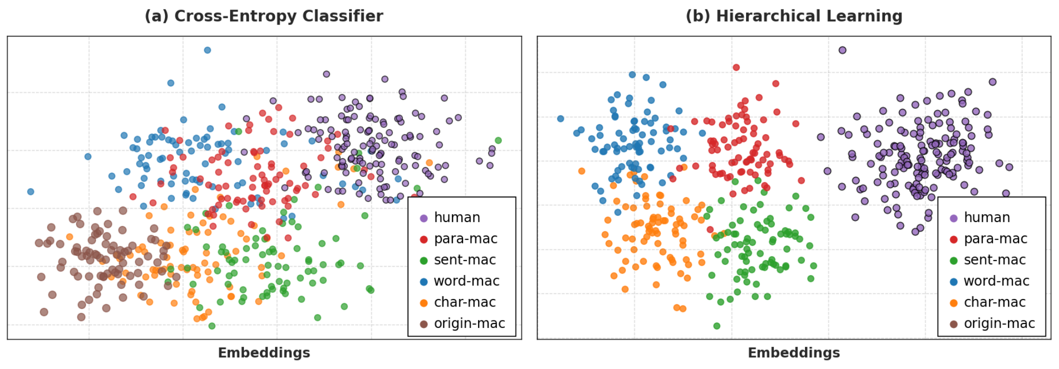

Figure 4: Illustration of contrastive learning and cross entropy on final texts' embeddings. Para-mac reffers to text using humanizing method of paragraph level, rest are the same.

also achieve the second best in overall recall. Further, using the comprehensive evaluating metric of F1 to illustrate, HiGraDe is 0.72% higher than the second place on the SeqXGPT dataset and 4.11% higher than the second place on the CheckGPT dataset. For the HC3 dataset, although it is released relatively early and the recognition difficulty may be relatively low, all baseline detectors perform well, HiGraDe still achieves a certain breakthrough with F1 0.54% higher than the second place. The above results show that the cross-data adaptability of our two-layer framework is commendable, and acquires the most advanced performance in Task 1.

Dataset	Detectors	Origin			Humanized		
		Recall	F1	ACC	Recall	F1	ACC
HC3	SimpleAI	94.32	94.31	94.32	71.58	79.25	96.44
	Watermark	94.75	95.13	94.88	76.03	69.05	55.16
	CoCo	99.31	98.30	98.42	58.18	95.09	94.44
	RADAR	89.57	90.39	89.57	79.20	89.40	81.75
	PECOLA	99.25	99.24	99.23	64.03	70.59	95.44
	HiGraDe	**99.78**	**99.78**	**99.80**	**96.26**	**95.99**	**99.52**
SeqXGPT	SimpleAI	94.38	94.36	94.37	81.74	86.18	97.04
	Watermark	96.30	95.92	96.07	76.22	68.80	54.61
	CoCo	82.36	79.54	80.67	93.65	93.06	98.16
	RADAR	61.15	54.16	61.37	66.77	72.08	58.51
	PECOLA	90.83	90.82	90.82	74.98	80.30	96.06
	HiGraDe	**96.70**	**96.64**	**96.66**	**97.23**	**94.02**	**99.21**
CheckGPT	SimpleAI	87.82	88.78	88.77	88.58	70.21	90.52
	Watermark	**97.06**	72.26	75.69	76.54	69.56	56.22
	CoCo	84.90	85.97	84.55	85.63	**98.21**	96.60
	RADAR	63.26	63.01	63.04	72.71	75.35	61.94
	PECOLA	87.16	86.79	86.82	69.14	75.63	96.90
	HiGraDe	93.33	**92.89**	**93.55**	**96.08**	95.71	**99.63**

Table 1: Combined performance results across original and humanized datasets. The best number is highlighted in **bold**, while the second best one is underlined.

To further illustrate that our two-layer training paradigm can train a more robust detector, we humanize the machine texts in the HC3, SeqXGPT and CheckGPT datasets at different levels, and retrain the baseline detector based on the humanized dataset to compare with HiGraDe. Results correspond to the humanized column in Table 1. Our detector HiGraDe has achieved SOTA performance on the

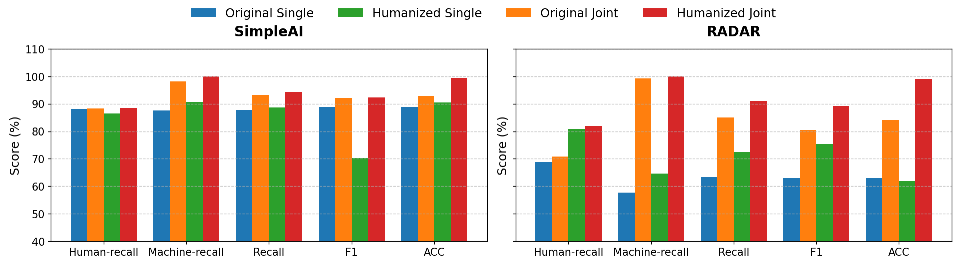

Figure 5: Illustration of "patch" result for baseline detectors SimpleAI and RADAR on CheckGPT dataset, original (humanized) represents the original (humanized) dataset, single represents the original detector, joint represents detecor using our patch. Results prove that our patch can effectively improve the performance of the current baseline detectors. Detailed results are shown in Appendix H.

three humanized datasets. In terms of recall, F1, and ACC, HiGraDe is 17.06%, 0.90%, and 3.08% higher than the second place on the HC3 dataset, and 3.58%, 0.96%, and 1.05% higher than the second place on the SeqXGPT dataset. On the CheckGPT dataset, recall and ACC are both the first place, and F1 reaches the runner-up performance. Furthermore, for the detector CoCo, which also has outstanding performance, although its performance on the CheckGPT dataset is comparable to ours, it relies on the extraction of entities in texts and the construction of a coherence graph. If the machine texts are relatively concise and short, its performance will suddenly drop because of the failure in building coherence graphs. In our experiments on the HC3 dataset, we find that 10,083 out of 24,000 data do not have corresponding coherence graphs, which leads to a sudden collapse of CoCo's recall metric (58.18%). Overall, HiGraDe performes well on all three datasets, and demonstrates excellent generalization and robustness in Task 2.

Method	Type	Detectors	Metrics				
			Human-recall	Machine-recall	Recall	F1	ACC
Contrastive Learning	Original	Single	**89.48**	92.25	90.86	90.45	91.44
		Joint	88.13	**98.99**	**93.56**	**93.15**	**93.77**
	Humanized	Single	**86.90**	94.13	90.52	55.78	93.81
		Joint	85.91	**99.80**	**92.86**	**90.39**	**99.18**
Fine-tuned DeBERTa	Original	Single	94.17	89.17	91.67	91.62	91.70
		Joint	**94.43**	**90.09**	**92.26**	**91.92**	**92.05**
	Humanized	Single	91.64	91.57	91.60	50.12	91.65
		Joint	**93.36**	**99.97**	**96.67**	**95.32**	**99.60**

Table 2: Performance results of "patching" results for contrastive learning and fine-tuned DeBERTa.

We further explore the effect of our framework in "patching" existing detectors, given that the coarse screen layer plays an important role in the overall detection effect, its initial screening of data directly affects the training and detection of the subdivision layer. We replace the first layer based on traditional classification loss with fine-tuned DeBERTa (Wang's framework) [54] and detecor based on supervised contrastive learning (Chen's framework) [3], along with baseline detectors SimpleAI [18] and RADAR [22].

Results are shown in Table 2 and Figure 5. For the existing detectors based on contrastive learning, the F1 reaches 90.45% and the ACC reaches 91.44% in Task 1, which shows that a relatively ideal effect has been achieved in the first layer. On this basis, we further use our subdivision layer to play the role of the second gate in the hope of achieving a more refined effect. The results are consistent with our expectations. The patched detector further improves the F1 to 93.15% and the ACC to 93.77%. For Task 2, the detector without the patch has only 55.78% of F1, which shows a large number of humanized machine texts are misjudged as human. However, using our patch, a second checkpoint is set up in time to capture these humanized texts that bypass detection, and the F1 score is increased by a huge span of 34.61% to 90.39%. This further proves our original intention of using two-layer detection: To give the existing detector robustness when facing large-scale humanized machine texts, and further improve its recognition ability of machine texts so that it can effectively capture both original and humanized texts. Overall, by applying our patch, we achieve a significant improvement in most evaluation metrics, with only a minor sacrifice of 1% in human texts recall, resulting in a substantial overall performance optimization.

4.3 Ablation Studies

First, we directly apply the first layer, the second layer, and two-layer detecor HiGraDe to Task 1 (Original) and Task 2 (Humanized). Results are shown in Table 3. For Task 1, coarse screen layer performes well overall, which confirms its applicability and practicality in the simple binary classification task of detecting human texts and original machine texts. The subdivision layer performes relatively averagely, with an F1 of 63.88% and an ACC of only 47.18%. This phenomenon is well explained: second layer is trained with samples screened by the first layer as data set. If it is directly applied to the original texts detection, the effect may not be satisfactory. This is also the reason why the F1 and ACC scores of the second-stage detector plummets in Task 2. For task 2, although the coarse screen layer bears a good recall rate, combined with the imbalance of the dataset under this task (the humanizing methods of machine texts are varied, Appendix D) and its poor performance in F1 score (55.78%), indicating that it will misjudge a large number of humanized machine texts as human, which is consistent with our expectation that "the robustness of the first layer is fragile when facing large-scale humanized texts", and is one of the reasons why we introduce the subdivision layer.

Type	Detectors	Direct Apply			Direct Train		
		Recall	F1	ACC	Recall	F1	ACC
Original	First	90.52	90.07	90.60	88.75	90.07	90.60
	Second	87.15	63.88	47.18	88.16	87.09	88.40
	Combined	**93.33**	**92.89**	**93.55**	**93.33**	**92.89**	**93.55**
Humanized	First	91.46	57.22	94.05	83.43	74.95	96.94
	Second	89.67	8.50	5.15	88.42	85.51	97.83
	Combined	**96.08**	**95.71**	**99.63**	**96.08**	**95.71**	**99.63**

Table 3: Merged results of ablation study. Direct Apply represents results from direct application, Direct Train represents results from direct training.

To further verify the effectiveness of the two-layer joint training framework, we directly trains the first layer and second layer on the CheckGPT dataset in Task 1 and 2. For the coarse screen layer, even after training with humanized texts, it still has the phenomenon of "wrongly accusing good guys" when detecting humanized machine texts, which reveals the limitations of single stage detection in adversarial scenarios. For subdivision layer, in Task 1, its multi-granularity contrastive learning will degenerate into single-granularity contrastive learning, while in Task 2, its ability to discriminate human texts is slightly weak due to the fact that in the face of a variety of large-scale humanized machine texts in reality, contrastive learning learns limited features of human texts.

Overall, whether it is detecting original machine text or humanized machine text, whether it is directly applied or trained, deleting any component will lead to a decrease in the overall performance of HiGraDe and it is undoubtedly worthwhile to sacrifice the recall rate of human texts slightly in exchange for a significant improvement in overall performance.

5 Conclusion

In this paper, we propose a coarse-to-fine AI generated text detector model and a novel training paradigm. The coarse screen layer quickly screens the original machine texts, and the subdivision layer uses multi-granularity contrastive learning to carefully distinguish different levels of humanized machine texts. Our detector HiGraDe has achieved SOTA performance in the task of detecting human texts from original machine texts and the more realistic task of detecting human texts from humanized machine texts, proving the effectiveness of each component and the correctness of the "blending the strengths of both" training paradigm. Moreover, our subdivision layer is a plug-and-play patch that can be easily applied to existing detectors to further improve their performance. We hope HiGraDe can better assist AI text detection in real life, and that this hierarchical and multi-granularity contrastive learning framework can bring new maps and ideas to researchers in this field.

References

[1] Shuyang Cai and Wanyun Cui. Evade chatgpt detectors via a single space, 2023. URL https://arxiv.org/abs/2307.02599.

[2] Shuzhi Cao, Jianfei Ruan, Bo Dong, Bin Shi, and Qinghua Zheng. Rr-pu: A synergistic two-stage positive and unlabeled learning framework for robust tax evasion detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8246–8254, Mar. 2024. doi: 10.1609/aaai.v38i8.28665. URL https://ojs.aaai.org/index.php/AAAI/article/view/28665.

[3] Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xu. Contrastnet: A contrastive learning framework for few-shot text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10492–10500, Jun. 2022. doi: 10.1609/aaai.v36i10.21292. URL https://ojs.aaai.org/index.php/AAAI/article/view/21292.

[4] Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. Token prediction as implicit classification to identify llm-generated text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, page 13112–13120. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.810. URL http://dx.doi.org/10.18653/v1/2023.emnlp-main.810.

[5] Xuxin Cheng, Bowen Cao, Qichen Ye, Zhihong Zhu, Hongxiang Li, and Yuexian Zou. ML-LMCL: Mutual learning and large-margin contrastive learning for improving ASR robustness in spoken language understanding. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6492–6505, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.406. URL https://aclanthology.org/2023.findings-acl.406/.

[6] Claude AI, 2024. URL https://www.ibm.com/think/topics/claude-ai.

[7] Federico Concone, Giuseppe Lo Re, Marco Morana, and Sajal K. Das. Spade: Multi-stage spam account detection for online social networks. *IEEE Transactions on Dependable and Secure Computing*, 20(4):3128–3143, 2023. doi: 10.1109/TDSC.2022.3198830.

[8] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

[10] Matthew C. Dickson, Anna S. Bosman, and Katherine M. Malan. *Hybridised Loss Functions for Improved Neural Network Generalisation*, page 169–181. Springer International Publishing, 2022. ISBN 9783030933142. doi: 10.1007/978-3-030-93314-2_11. URL http://dx.doi.org/10.1007/978-3-030-93314-2_11.

[11] Kaiwen Duan, Lingxi Xie, Honggang Qi, Song Bai, Qingming Huang, and Qi Tian. Corner proposal network for anchor-free, two-stage object detection, 2020. URL https://arxiv.org/abs/2007.13816.

[12] Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12463–12492, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.674. URL https://aclanthology.org/2024.acl-long.674/.

[13] Siyue Feng, Yueming Wu, Wenjie Xue, Sikui Pan, Deqing Zou, Yang Liu, and Hai Jin. FIRE: Combining Multi-Stage filtering with taint analysis for scalable recurring vulnerability detection. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 1867–1884, Philadelphia, PA, August 2024. USENIX Association. ISBN 978-1-939133-44-1. URL https://www.usenix.org/conference/usenixsecurity24/presentation/feng-siyue.

[14] Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552/.

[15] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. Gltr: Statistical detection and visualization of generated text, 2019. URL https://arxiv.org/abs/1906.04043.

[16] Gemini. Gemini: A family of highly capable multimodal models, 2024. URL https://arxiv.org/abs/2312.11805.

[17] Chenxi Gu, Chengsong Huang, Xiaoqing Zheng, Kai-Wei Chang, and Cho-Jui Hsieh. Watermarking pre-trained language models with backdooring. *arXiv preprint arXiv:2210.07543*, 2022.

[18] Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arxiv:2301.07597*, 2023.

[19] Xun Guo, Shan Zhang, Yongxin He, Ting Zhang, Wanquan Feng, Haibin Huang, and Chongyang Ma. Detective: Detecting ai-generated text via multi-level contrastive learning. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 88320–88347. Curran Associates, Inc., 2024.

[20] Julian Hazell. Spear phishing with large language models, 2023. URL https://arxiv.org/abs/2305.06972.

[21] Abe Hou, Jingyu Zhang, Yichen Wang, Daniel Khashabi, and Tianxing He. k-SemStamp: A clustering-based semantic watermark for detection of machine-generated text. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1706–1715, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.98. URL https://aclanthology.org/2024.findings-acl.98/.

[22] Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. Radar: Robust ai-text detection via adversarial learning, 2023. URL https://arxiv.org/abs/2307.03838.

[23] Guanhua Huang, Yuchen Zhang, Zhe Li, Yongjian You, Mingze Wang, and Zhouwang Yang. Are AI-generated text detectors robust to adversarial perturbations? In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6005–6024, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.327. URL https://aclanthology.org/2024.acl-long.327/.

[24] Yuan Jiang, Yujian Zhang, Xiaohong Su, Christoph Treude, and Tiantian Wang. Stagedvulbert: Multigranular vulnerability detection with a novel pretrained code model. *IEEE Transactions on Software Engineering*, 50(12):3454–3471, 2024. doi: 10.1109/TSE.2024.3493245.

[25] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kirchenbauer23a.html.

[26] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36, 2024.

[27] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703/.

[28] Jiale Li, Hang Dai, Ling Shao, and Yong Ding. Anchor-free 3d single stage detector with mask-guided attention for point cloud. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, page 553–562, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450386517. doi: 10.1145/3474085.3475208. URL https://doi.org/10.1145/3474085.3475208.

[29] Jiang Li, Xiaoping Wang, Guoqing Lv, and Zhigang Zeng. Ga2mif: Graph and attention based two-stage multi-source information fusion for conversational emotion detection. *IEEE Transactions on Affective Computing*, 15(1):130–143, 2024. doi: 10.1109/TAFFC.2023.3261279.

[30] Ying-Dar Lin, Shin-Yi Yang, Didik Sudyana, Fietyata Yudha, Yuan-Cheng Lai, and Ren-Hung Hwang. Two-stage multi-datasource machine learning for attack technique and lifecycle detection. *Computers and Security*, 142:103859, 2024. ISSN 0167-4048. doi: https://doi.org/10.1016/j.cose.2024.103859. URL https://www.sciencedirect.com/science/article/pii/S0167404824001603.

[31] Aiwei Liu, Leyi Pan, Xuming Hu, Shu'ang Li, Lijie Wen, Irwin King, and Philip S. Yu. An unforgeable publicly verifiable watermark for large language models, 2024. URL https://arxiv.org/abs/2307.16230.

[32] Shengchao Liu, Xiaoming Liu, Yichen Wang, Zehua Cheng, Chengzhengxu Li, Zhaohan Zhang, Yu Lan, and Chao Shen. Does DetectGPT fully utilize perturbation? bridging selective perturbation to fine-tuned contrastive learning detector would be better. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1874–1889, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.103. URL https://aclanthology.org/2024.acl-long.103/.

[33] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 507–516, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/liud16.html.

[34] Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Hang Pu, Yu Lan, and Chao Shen. Coco: Coherence-enhanced machine-generated text detection under data limitation with contrastive learning, 2023. URL https://doi.org/10.48550/arXiv.2212.10341.

[35] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019. URL https://arxiv.org/abs/1907.11692.

[36] Zeyan Liu, Zijun Yao, Fengjun Li, and Bo Luo. On the detectability of chatgpt content: Benchmarking, methodology, and evaluation through the lens of academic writing, 2024. URL https://arxiv.org/abs/2306.05524.

[37] Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. An entropy-based text watermarking detection method. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11724–11735, Bangkok, Thailand, August 2024.

Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.630. URL https://aclanthology.org/2024.acl-long.630/.

[38] Chao Ma, Daniel Kunin, Lei Wu, and Lexing Ying. Beyond the quadratic approximation: the multiscale structure of neural network loss landscapes, 2022. URL https://arxiv.org/abs/2204.11326.

[39] Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. Efficient detection of llm-generated texts with a bayesian surrogate model, 2024. URL https://arxiv.org/abs/2305.16617.

[40] Niloofar Mireshghallah, Justus Mattern, Sicun Gao, Reza Shokri, and Taylor Berg-Kirkpatrick. Smaller language models are better black-box machine-generated text detectors, 2024. URL https://arxiv.org/abs/2305.09859.

[41] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/mitchell23a.html.

[42] Roe Perkins, Mike. Detection of gpt-4 generated text in higher education: Combining academic judgement and software to identify generative ai tool misuse. *Journal of Academic Ethics*, 22(1), October 2023. ISSN 1572-8544. doi: 10.1007/s10805-023-09492-6. URL http://dx.doi.org/10.1007/s10805-023-09492-6.

[43] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf.

[44] Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. Release strategies and the social impacts of language models, 2019. URL https://arxiv.org/abs/1908.09203.

[45] Rafael Alberto Rivera Soto, Kailin Koch, Aleem Khan, Barry Y. Chen, Marcus Bishop, and Nicholas Andrews. Few-shot detection of machine-generated text using style representations. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=cWiEN1plhJ.

[46] Rafael Rivera Soto, Kailin Koch, Aleem Khan, Barry Chen, Marcus Bishop, and Nicholas Andrews. Few-shot detection of machine-generated text using style representations, 2024. URL https://arxiv.org/abs/2401.06712.

[47] Jinyan Su, Terry Yue Zhuo, Di Wang, and Preslav Nakov. Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text. *arXiv preprint arXiv:2306.05540*, 2023.

[48] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[49] Edward Tian and Alexander Cui. Gptzero: Towards detection of ai-generated text using zero-shot and supervised methods, 2023. URL https://example.com.

[50] Jörg Tiedemann and Santhosh Thottingal. OPUS-MT – building open translation services for the world. In André Martins, Helena Moniz, Sara Fumega, Bruno Martins, Fernando Batista, Luisa Coheur, Carla Parra, Isabel Trancoso, Marco Turchi, Arianna Bisazza, Joss Moorkens, Ana Guerberof, Mary Nurminen, Lena Marg, and Mikel L. Forcada, editors, *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages

479–480, Lisboa, Portugal, November 2020. European Association for Machine Translation. URL https://aclanthology.org/2020.eamt-1.61/.

[51] Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. SeqXGPT: Sentence-level AI-generated text detection. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1144–1156, Singapore, December 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.emnlp-main.73/.

[52] Qian Wang, Weilong Wang, Yan Wang, Jiadong Ren, and Bing Zhang. Multi-stage network attack detection algorithm based on gaussian mixture hidden markov model and transfer learning. *IEEE Transactions on Automation Science and Engineering*, 22:3470–3484, 2025. doi: 10.1109/TASE.2024.3395355.

[53] Siqi Wang, Yijie Zeng, Qiang Liu, Chengzhang Zhu, En Zhu, and Jianping Yin. Detecting abnormality without knowing normality: A two-stage approach for unsupervised video abnormal event detection. In *Proceedings of the 26th ACM International Conference on Multimedia*, MM '18, page 636–644, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356657. doi: 10.1145/3240508.3240615. URL https://doi.org/10.1145/3240508.3240615.

[54] Yichen Wang, Shangbin Feng, Abe Hou, Xiao Pu, Chao Shen, Xiaoming Liu, Yulia Tsvetkov, and Tianxing He. Stumbling blocks: Stress testing the robustness of machine-generated text detectors under attacks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2894–2925, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.160. URL https://aclanthology.org/2024.acl-long.160/.

[55] Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Taxonomy of risks posed by language models. FAccT '22, page 214–229, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393522. doi: 10.1145/3531146.3533088. URL https://doi.org/10.1145/3531146.3533088.

[56] Danny Wood, Tingting Mu, and Gavin Brown. Bias-variance decompositions for margin losses. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 1975–2001. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/wood22a.html.

[57] Dejiao Zhang, Wei Xiao, Henghui Zhu, Xiaofei Ma, and Andrew Arnold. Virtual augmentation supported contrastive learning of sentence representations. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 864–876, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.70. URL https://aclanthology.org/2022.findings-acl.70/.

[58] Yanzhao Zhang, Richong Zhang, Samuel Mensah, Xudong Liu, and Yongyi Mao. Unsupervised sentence representation via contrastive learning with mixing negatives. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11730–11738, Jun. 2022. doi: 10.1609/aaai.v36i10.21428. URL https://ojs.aaai.org/index.php/AAAI/article/view/21428.

[59] Ying Zhou, Ben He, and Le Sun. Navigating the shadows: Unveiling effective disturbances for Modern AI content detectors. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10847–10861, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.584. URL https://aclanthology.org/2024.acl-long.584/.

[60] Biru Zhu, Lifan Yuan, Ganqu Cui, Yangyi Chen, Chong Fu, Bingxiang He, Yangdong Deng, Zhiyuan Liu, Maosong Sun, and Ming Gu. Beat LLMs at their own game: Zero-shot LLM-generated text detection via querying ChatGPT. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7470–7483, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.463. URL https://aclanthology.org/2023.emnlp-main.463/.

A Broader Impacts

The rapid development of LLMs has enabled a large amount of machine-generated texts to be obtained quickly and at low cost. Given that it may lead to academic fraud, phishing emails, the spread of false information and other problems, detecting and monitoring AI-generated texts is undoubtedly a top priority. However, due to the fragility of current AI content detectors and the diversity of text humanizing methods, machine texts can easily bypass detection after humanizing. Therefore, the development of robust AI content detectors is urgent. Our paper introduces a new robust AI content detector training paradigm, which demonstrates SOTA performance in multiple benchmarks. These advances will bring the green development and use of LLMs with new power. In addition, our second-stage detector can be used as a "patch" to further improve the performance of current detectors when facing large-scale humanized machine texts, which shows that our method has broad prospects for practical application and rich significance.

B Limitations and Future Work

In this paper, we take into consideration that machine texts may use different humanizing methods to evade the detector and thus use multi-granularity contrastive learning to strengthen the detector in a targeted manner. However, the original machine texts generated by different models often has certain differences, which may affect the performance of the detector. In addition, we did not introduce some latest humanizing strategies (such as adding emoticons to machine texts) and did not train on a larger corpus. In the future, we will continue to work in this direction and further improve the performance of the model.

C Detailed Humanizing Methods and Its Levels

Following Zhou's work [59], four types of humanizing methods are classified below:

1. **char level**: Attacks at this level include space deletion, space addition [1], capitalization typo simulation, punctuation deletion, and random word merging.
2. **word level**: Attacks at this level include keyboard spelling errors, which replace characters in similar keyboard positions; swaping adjacent characters, inserting irrelevant characters, and deleting specific characters, thereby simulating human negligence when typing; word spelling errors, which simulate users' incorrect spelling of words through a predefined spelling error dictionary; adverb perturbations, which randomly insert relevant adverbs before verbs in the original text; word replacement, which uses the BERT model [9] to replace words in the text with synonyms.
3. **sentence level**: Attacks at this level include adding irrelevant sentences; repeating parts of sentences; randomly selecting sentences for back-translation; and sentence-level replacement, which randomly masks 2 to 5 sentences in the original text and replaces them using the BART-large model [27].
4. **paragraph level**: Attacks at this level include rewriting using the Dipper interpreter [26]; back-translation using the Helsinki-NLP model [50]; and rearrangement of paragraph structure.

Given original machine texts X, we thus have humanized machine texts set:

$$X_{humanized} = \{X_{char}, X_{word}, X_{sent}, X_{para}\}$$

which is illustrated in section 3.1.

D Detailed Construction of Dataset

For the simple binary classification task of detecting human texts and original machine texts, the distribution of our samples is shown in Table 4. We ensure the distribution of human texts and machine texts is approximately 1:1. The two-tuple (human, machine) in the table represents the number of human texts and the number of machine texts. For the watermark detector, since it focuses

Dataset	Train	Test	Valid
CheckGPT	(2000,2000)	(1921,2078)	(2500,2500)
HC3	(5000,5000)	(5000,5000)	(2000,2000)
SeqXGPT	(2467,2533)	(1928,1872)	(1005,995)

Table 4: Detailed composition of the dataset for detecting human texts and original machine texts.

Watermark	Human	Machine
CheckGPT	566	570
HC3	438	538
SeqXGPT	600	520

Table 5: Detailed composition of the dataset for watermarks.

Dataset	Train	Test	Valid
CheckGPT	(500,8500)	(1101,23887)	(500,8490)
HC3	(500,7500)	(1500,22500)	(500,7500)
SeqXGPT	(500,7500)	(1500,21004)	(500,6494)

Table 6: Detailed composition of the dataset for detecting human texts and humanized machine texts.

Watermark	Human	Machine
CheckGPT	566	8550
HC3	438	8098
SeqXGPT	600	520

Table 7: Detailed composition of the humanized dataset for watermarks.

on the hidden singal embedded in the data and does not require training, it only needs to build a test set. While it takes a long time to process the watermark on the dataset, we did not generate a large test set. The data are shown in Table 5. For the more realistic task of detecting human texts and humanized machine texts, the distribution of our samples is shown in Table 6. We select 500 human texts and 500 original machine texts from the dataset respectively, and perform 16 humanizing methods on the machine texts at the character, word, sentence, and paragraph levels, thereby constructing a perturbation dataset containing human texts, original machine texts, and humanized machine texts. Therefore, the perturbed datasets are mostly composed of machine texts, which are consistent with the current trend of a variety of machine text humanizing methods and a flood of generation sources. For the watermark detector, similarly, after the machine texts are injected with the watermark, we humanize them in sixteen different ways and explore whether these humanizing attacks will cause the watermark to be covered and invalid. The data distribution is shown in Table 7.

E Comparison of Detectors in Costs

Detectors	Preprocess Time	Preprocess Memory	Train Time	Train Memory
PECOLA	22:17:04	642	00:08:55	9228
CoCo	04:42:56	10520	00:33:27	14556
Watermark	153:21:22	1290	00:01:31	840
Contra	no need	no need	00:12:40	10840
Coarse	no need	no need	00:08:40	10840
Subdivision	no need	no need	00:01:36	10840

Table 8: Detailed comparison of the detectors in terms of costs. Preprocess time means data should be preprocessed before training.

The baseline detectors SimpleAI and RADAR are similar to coarse layer (which are also the subjects of our patching experiments) thus are not considered here. Shown in Table 8, Contra means we directly use contrastive learning to train the detector. The data processing and training time are formed in hour: minute :second, and the memory occupied is in MB. Watermark does not require additional training of the detector, and its training time is temporarily expressed as the detection time. We uniformly set the training batch size to 16, the training round to 15, and trained on NVIDIA RTX A6000. The high cost of contrastive learning is that it needs to calculate the similarity between all sample pairs, which has a time complexity of $O(n^2)$ while cross entropy of $O(n)$, which will mainly affect training time rather than memory usage. Compared with using contrastive learning directly, the subdivision layer only needs to process the filtered samples, and the training time is reduced from 12 minutes to 1 minute 36 seconds. In addition, although PECOLA, CoCo and Watermark have achieved good performance in detecting machine texts, their cumbersome and lengthy data preprocessing process deserves high attention and needs to be considered seriously. Taking CoCo as an example, it is slow in extracting entity graphs from texts, takeing nearly 5 hours to extract 8,000 training samples. When large-scale machine text needs to be detected in real life, such as the 23005 samples in our test data set, its data preprocessing takes an astonishing 17 hours, shown in Table 10. These situations remind us that if we want to make an AI content detector with advanced performance and practical significance, the time issue of data processing needs to be paid great attention to.

PECOLA	Augment	Select	Total
Train	00:03:44	22:13:20	22:17:04
Eval	00:03:16	19:26:45	19:30:01
Test	00:09:48	56:56:28	57:06:16

Table 9: Detailed time cost for PECOLA in building train, eval and test sets. Augment means data augmentation, Select means its selecting strategy.

CoCo	Extract	Build	Total
Train	04:41:04	00:00:31	04:42:56
Eval	02:03:20	00:00:23	02:03:43
Test	17:06:38	00:01:21	17:07:59

Table 10: Detailed time cost for CoCo in building train, eval and test sets. Extract means extracting entity, Build means building graphs according to the extracted entities.

F Machine-Generated Text Detectors

In order to prevent machine-generated texts from being abused, numbers of detectors have been proposed by researchers, thus consolidating the defense line of text detection. We classify the existing detectors into the following four categories:

Statistical and mathematical based detectors: Using information entropy, cross perplexity, word frequency statistics and other features to perform zero-shot detection. Mithcell [41] quantifies the difference between machines and human in word selection via conditional probability curvature. Su [47] applies log-rank information to detect. Open source detecting platform GPTZero and GLTR [49, 15] are also included.

Watermark based detectors: Watermark detection algorithms in machine texts detection track the source of generated texts by embedding invisible identifiers. Representative works include: [17, 31, 21, 37]. Among them, Kirchenbauer [25] adds a fixed weight to the logit value of the predefined "green word list" and determine whether the text is generated by the model by counting the proportion of green words in the texts.

Classifier based detectors: Researchers [4, 39, 40, 51, 36] typically employ RoBERTa [35] as the backbone architecture for training supervised binary classifiers. Notable developments are seen in OpenAI's official detection toolkit [44] and RADAR [22], which enhances adversarial robustness against perturbation attacks through paraphrase-based adversarial training.

Other methods based detecors: Soto [46] uses style representations, Huang [23] takes advantage of siamese neural network, Krishna [26] achieves success through retrieval methods. Zhu [60] innovatively queries LLM to detect LLM-generated texts.

G Contrastive Learing in Detectors

There have been works showing that contrastive learning has excellent performance in the field of natural language processing [5]. MixCSE [58], SimCSE [14], VaSCL [57] use unsupervised contrastive learning framework to enhance the semantic discrimination ability of the model; CoCo [34] use supervised contrastive learning to make the model pay more attention to difficult negative samples in low-resource scenarios; Soto and Guo [45, 19] use contrastive learning to distinguish the style features of human and machine writing. By narrowing the distance between positive samples and

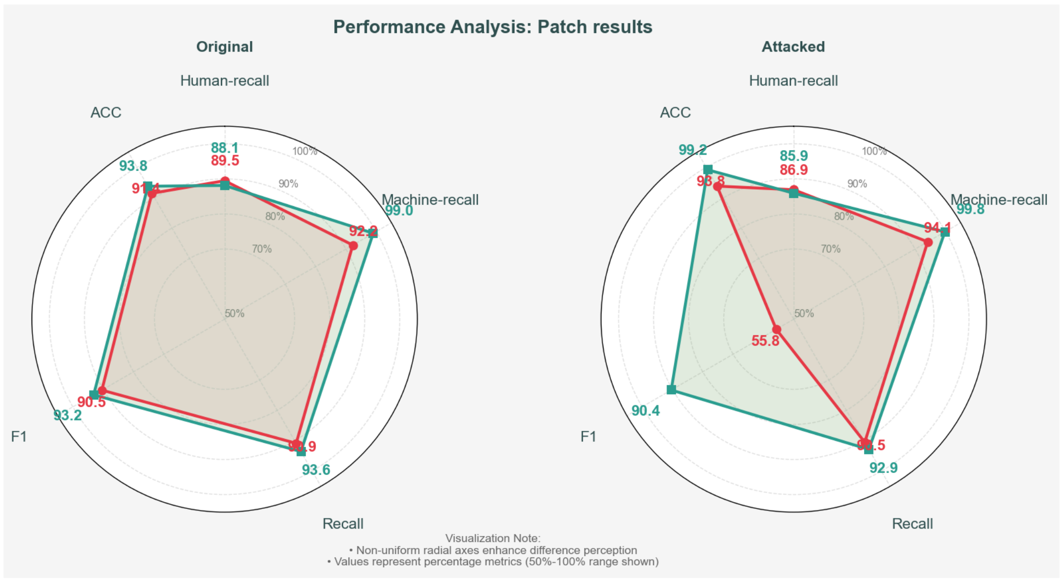

Figure 6: Illustration of "patch" result on supervised contrastive learning.

increasing the distance between negative samples, contrastive learning has shown great potential in training AI content detectors [32].

H Detailed Patching Results

Type	Detectors	Metrics				
		Human-recall	Machine-recall	Recall	F1	ACC
Original	Single	88.11	87.54	87.82	88.78	88.77
	Patch	96.94	84.56	90.75	97.30	95.33
	Joint	88.35	98.08	93.22	92.22	92.92
Humanized	Single	86.55	90.61	88.58	70.21	90.52
	Patch	98.88	99.01	98.94	98.32	98.96
	Joint	88.55	99.91	94.23	92.37	99.35

Table 11: Performance results of "patching" results of SimpleAI.

Type	Detectors	Metrics				
		Human-recall	Machine-recall	Recall	F1	ACC
Original	Single	68.77	57.75	63.26	63.01	63.04
	Patch	98.64	98.18	98.41	98.71	98.45
	Joint	70.83	99.23	85.03	80.43	84.15
Humanized	Single	80.84	64.58	72.47	75.35	61.94
	Patch	97.20	99.77	98.48	97.46	99.54
	Joint	82.00	99.91	90.96	89.17	99.11

Table 12: Performance results of "patching" results of RADAR.

Similarly, we add our patch (subdivision layer) to the baseline detectors SimpleAI and RADAR, the results are shown in Tables 11 and 12. In the table, single represents the result without using the patch, patch represents the further screening effect of our patch on the filtered human text, and joint represents the final effect of the detector after using the patch. It can be seen that after using our patch, all metrics of detectors have achieved breakthroughs (the yellow part in the table). Taking the

F1 score as an example, for Task 1, the two detectors still have steady improvements, SimpleAI from 88.78% to 92.22%, an increase of 3.44%; RADAR from 63.01% to 80.43%, an increase of 17.42%. Furthermore, when facing humanized data sets, the performance improvement is particularly obvious, such as SimpleAI's F1 from 70.21% to 92.37%, a huge leap of 22.16%. The above results show that our patch has well blocked the loophole that humanized machine text escapes detection after one screening. This patch is undoubtedly an extremely useful remedial tool for existing AI text detectors, and it also confirms the wide impact of our method.