

---

# 关于文本鉴伪的两篇论文

2026 年 02 月 12 日分享

LiuKai

---

2026 年 02 月 12 日

# 目 录

1 Ghostbuster 论文 .....	1
1.1 Introduction .....	1
1.2 Methodology .....	1
1.2.1 Probability Computation .....	1
1.2.2 Feature Selection .....	1
1.2.3 Classifier Training .....	2
1.3 Results .....	2
1.4 Analysis .....	3
1.4.1 Ablation Study .....	3
2 Raidar 论文介绍 .....	3
2.1 Introduction .....	3
2.2 Methodology .....	3
2.2.1 词袋编辑 (Bag-of-words edit) .....	3
2.2.2 Levenshtein 分数 (Levenshtein Score) .....	4
2.3 Results .....	4
2.4 Advantages .....	5
参考文献 .....	6

# 1 Ghostbuster 论文

## 1.1 Introduction

本文全称 “Ghostbuster: Detecting Text Ghostwritten by Large Language Models” [1], 发表在 ICLR 2024 上。文章主要提出了一种的方法来检测文本, 把文本经过 概率计算-特征选择-分类器计算 的流程进行判别。简单来说, 作者提出了一种方法, 先通过暴力枚举从所有特征中最值得计算的特征, 然后拿这些特征进行计算, 掌握这些特征之后, 就使用一个简单的线性分类器进行鉴别。

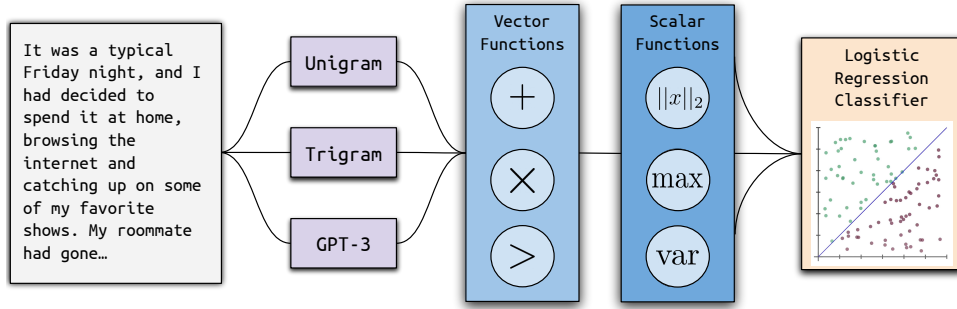


图 1 Ghostbuster 框架示意图

## 1.2 Methodology

### 1.2.1 Probability Computation

关于概率计算部分, 作者首先定义了一系列操作 (see 表 1), Vector Functions 之间可以自由组合, 对向量进行操作, 然后 Scalar Functions 是最后一步, 把操作出来的向量变成标量。

表 1 Ghostbuster 定义向量操作列表

Vector Functions	Scalar Functions
$f_{\text{add}} = p_1 + p_2$	$f_{\text{max}} = \max p$
$f_{\text{sub}} = p_1 - p_2$	$f_{\text{min}} = \min p$
$f_{\text{mul}} = p_1 \cdot p_2$	$f_{\text{avg}} = \frac{1}{ p } \sum_i p_i$
$f_{\text{div}} = \frac{p_1}{p_2}$	$f_{\text{avg-top25}} = \frac{1}{ p } \sum_{i \in T_p} p_i$
$f_{>} = \mathbb{1}_{p_1 > p_2}$	$f_{\text{len}} =  p $
$f_{<} = \mathbb{1}_{p_1 < p_2}$	$f_{\text{L2}} = \ p\ _2$
	$f_{\text{var}} = \frac{1}{n} \sum_i (p_i - \mu_p)^2$

### 1.2.2 Feature Selection

定义了以上操作, 作者先用 Unigram, Trigram, GPT-3 Ada, Davinci 模型的 Token 序列作为输入向量, 然后经过 Vector Functions 和 Scalar Functions 的任意组合, 暴力枚举出所有可能的特征组合, 最后得到一个特征集合 S。得到 S 之后, 作者根据 表 2 进行枚举, 每一轮只选一个让当前模型性能提升最大的特征, 一直枚举直到加任何新特征都不能提升性能。除了电脑枚举, 作者还加入了人工选择的特征进行矫正, 最终你会得出最终特征集合 S。

---

Algorithm 1 Subroutine FIND-ALL-FEATURES

---

Require: The previously picked feature  $p$ , depth  $d < \text{max\_depth}$ , vectors  $V$  of token probabilities (from unigram, trigram, ada, and davinci models), scalar functions  $F_s$ , vector functions  $F_v$

Ensure: A list of all possible features

Let  $S = \emptyset$

for all scalar functions  $f_s \in F_s$  do

    Add  $f_{s(p)}$  to  $S$

end for

    for all combinations of features and vector functions  $(p', f_v) \in V \times F_v$  do

        Add Find-All-Features( $f_{v(p,p')}, d+1$ ) to  $S$

    end for

---

### 1.2.3 Classifier Training

拿到特征集合  $S$  之后，作者使用一个简单的线性分类器 (Logistic Regression) 进行训练，最终得到一个鉴别文本是否由 LLM 生成的模型。关于这里为什么作者只使用最简单的线性分类器，我们在后续的 Ablation Study 中进行分析。

## 1.3 Results

在 Metric 上，作者使用了 F1 Score 来评测模型性能，结果如图 2 所示，Ghostbuster 在所有数据集上都显著优于之前的 SOTA 方法。同时还表现出了很好的泛化能力，例如在新闻上训练，在作文上测试仍然可行，说明 Ghostbuster 是学到了文本生成的通用特征进行鉴别。

Model	Prompts (F1)	Claude (F1)	Lang8 (Acc.)	TOEFL 11 (Acc.)	TOEFL 91 (Acc.)
Perplexity only	85.3	84.1	98.6	98.1	13.2
DetectGPT	70.8	64.2	98.6	<b>100.0</b>	63.7
GPTZero	96.1	75.6	<b>99.2</b>	<b>100.0</b>	92.3
RoBERTa	97.4	87.8	98.6	98.1	<b>96.7</b>
<b>Ghostbuster</b>	<b>99.5</b>	<b>92.2</b>	95.5	99.9	74.7

Table 3: Additional generalization results. We evaluated model performance across a variety of prompting strategies (example prompts in Table 8). We also evaluated our model’s ability to detect essays generated by Claude. Finally, we evaluated the model accuracy on a set of three datasets of text written by non-native English speakers. For all conditions, we trained the RoBERTa and Ghostbuster models on all three domains of human and ChatGPT-written text; we applied oracle thresholding on the DetectGPT and GPTZero models only for the prompt and model generalization experiments. For generalization across models and prompts, we report F1; for the non-native English speaker data, we report accuracy (equivalent to precision, since there is no corresponding AI-generated text).

Ablation	In-Domain				Out-of-Domain		
	All Domains	News	Creative Writing	Student Essays	News	Creative Writing	Student Essays
Handcrafted features only	80.5	79.6	78.2	83.6	75.8	77.2	77.2
Limited search (depth = 1)	93.7	96.9	89.6	93.9	93.7	81.3	87.3
Limited search (depth = 2)	98.3	98.1	98.1	98.8	95.9	95.2	93.1
Further search (depth = 4)	98.3	<b>99.5</b>	97.8	99.4	96.4	<b>95.8</b>	<b>97.7</b>
Without ada and davinci	88.2	91.8	93.7	96.5	70.1	78.5	75.5
Without davinci	98.8	99.3	<b>99.5</b>	<b>99.8</b>	97.3	90.3	91.9
Without handcrafted features	98.9	99.0	98.9	99.5	97.8	93.4	97.4
With random features	97.8	98.7	97.3	99.4	94.3	93.1	94.3
Ghostbuster (full model)	<b>99.0</b>	<b>99.5</b>	98.4	99.5	<b>97.9</b>	95.3	<b>97.7</b>

Table 4: Model ablations (F1). We first evaluated the performance of Ghostbuster using only handcrafted features, without the structured search procedure defined in Section 4.2. We also experimented with only allowing one or two operations during structured search, effectively limiting the space of potential features. Finally, we evaluated the performance of our model without access to ada and/or davinci, or without access to any of the handcrafted features, finding that a model which uses only n-gram and ada features (i.e., the without davinci condition) nearly matched the performance of our full model.

图 2 Ghostbuster 在不同数据集上的表现

## 1.4 Analysis

### 1.4.1 Ablation Study

关于这里，作者主要分析了三方面：`max_depth` 和 `feature selection` 的重要性和分类器的选择。关于 `max_depth`，作者发现当 `max_depth = 3` 时性能最好，说明过于复杂的特征组合反而会导致过拟合。关于 `feature selection`，作者发现不使用人类提供的特征会导致模型泛化能力下降，说明人工选择的特征在模型训练中起到了重要的矫正作用。而分类器的选择上，使用复杂的分类方式（例如神经网络）反而导致性能下降，这是因为输入的特征本身经过了复杂模型处理的高级非线性信号了，再使用神经网络会导致过拟合。

## 2 Raidar 论文介绍

### 2.1 Introduction

Raidar [2] 论文的方法有点类似于 DNA-GPT [3]，DNA-GPT 是给 LLM 一段文本，然后补写，然后比对补写部分与原先部分的区别。而 Raidar 论文就是把目标文本输入给 LLM，然后让 LLM 进行 Polish（润色），然后比对润色前后文本的区别，来判断文本是否是 LLM 生成的。其中的底层原理是，LLM 对自己写的文本都很自信，然后认为人类写的文本不够好，所以会进行大幅度的润色，而对于 LLM 生成的文本，LLM 认为已经很好了，所以润色的幅度就比较小。

图 3 Raidar 原理展示图

### 2.2 Methodology

设  $F(\cdot)$  是 LLM，给定输入文本是  $x$ ，以后输出分类标签  $y$ ，方法观察结果是给定相同重写提示词，LLM 写的文本会被视为高质量输入，修改很少而人类文本会被进行更多的编辑。其中的提示词类似：

txt

1. Help me polish this:
2. Rewrite this for me:
3. Refine this for me please:

同时论文假设当多次重写的时候，LLM 生成的文本将比人类撰写的文本更稳定。作者还定义了输出的方差作为一种检测的度量。

$$U = \sum_{i=1}^{K-1} \sum_{j=i}^K D(x'_i, x'_j) \quad (1)$$

然后就需要计算两段文本之间的差异了，Raidar 使用了两种方法：

#### 2.2.1 词袋编辑 (Bag-of-words edit)

计算逻辑如下：

1. 提取：将“原始文本”和“重写后的文本”都拆解成一个个的  $n$ -词块（比如 3 个词一组）。
2. 找共同点：统计有多少个词块是同时出现在两个文本里的。

3. 算比例： 用共同词块的数量除以输入文本的长度。

### 2.2.2 Levenshtein 分数 (Levenshtein Score)

Levenshtein 分数就是用于衡量将一个字符串改写为另一个字符串所需的最小单字符编辑次数。论文中使用标准动态规划算法计算 Levenshtein 分数，分数越高表示两个字符串越相似。设重写输出  $s_k = F(p_k, x)$ 。然后计算比率：

$$D_k(x, s_k) = 1 - \frac{\text{Levenshtein}(s_k, x)}{\max(\text{len}(s_k), \text{len}(x))} \quad (2)$$

## 2.3 Results

下面是 Raidar 论文的结果展示，Raidar 在所有数据集上都显著优于之前的 SOTA 方法，同时不同的重写提示词上表现也很稳定，说明 Raidar 的方法是比较鲁棒的。

Table 1: F1 score for detecting machine-generated paragraphs. The results are in domain testing, where the model has been trained on the same domain. We **bold** the best performance on in-distribution and out-of-distribution detection. Our method achieved over 8 points of improvement over the established state-of-the-art.

Methods	Datasets					
	News	Creative Writing	Student Essay	Code	Yelp Reviews	Arxiv Abstract
GPT Zero-Shot Verma et al. (2023)	54.74	20.00	52.29	62.28	66.34	65.94
GPTZero (Tian, 2023)	49.65	61.81	36.70	31.57	25.00	45.16
DetectGPT Mitchell et al. (2023)	37.74	59.44	45.63	67.39	69.23	66.67
Ghostbuster Verma et al. (2023)	52.01	41.13	42.44	65.97	71.47	76.82
Ours (Invariance)	<b>60.29</b>	<b>62.88</b>	<b>64.81</b>	<b>95.38</b>	<b>87.75</b>	81.94
Ours (Equivariance)	58.00	60.27	60.07	80.55	83.50	75.74
Ours (Uncertainty)	60.27	60.27	57.69	77.14	81.79	<b>83.33</b>

Table 2: F1 score for detecting machine-generated paragraph following the out-of-distribution setting in Verma et al. (2023). We use logistic regression classifier for all ours. Our method achieved over 22 points of improvement over the established state-of-the-art.

Methods	Datasets		
	News	Creative Writing	Student Essay
Ghostbuster Verma et al. (2023)	34.01	49.53	51.21
Ours (Invariance)	56.47	55.51	<b>52.77</b>
Ours (Equivariance)	<b>56.87</b>	<b>59.47</b>	51.34
Ours (Uncertainty)	55.04	52.01	47.47

图 4 Raidar 的结果展示

对于用一个模型进行训练，检测其他模型生成的文本，Raidar 也表现出了很好的泛化能力。下面的表格是使用 GPT 3.5 进行训练。

Table 4: Robustness in detecting outputs from various language models. Using the same GPT-3.5-Turbo rewriting model, we present F1 detection scores for detecting text from five generation models across three diverse tasks. In the in-distribution experiment, detectors are trained and tested on the same model. For out-of-distribution, detectors are trained on text from other generators. Overall, our method effectively detects machine-generated text in both scenarios.

LLM Model Used for Text Generation	Raidar (Ours)						DetectGPT		
	In Distribution			Out of Distribution			Code	Yelp	arXiv
Ada	96.88	<b>96.15</b>	97.10	62.06	72.72	70.00	67.39	70.59	69.74
Text-Davinci-002	84.85	65.80	76.51	75.41	51.06	60.00	66.82	71.36	66.67
GPT-3.5-turbo	95.38	87.75	81.94	<b>91.43</b>	71.42	48.74	67.39	69.23	66.67
GPT-4-turbo	80.00	83.42	84.21	83.07	79.73	74.02	70.97	66.94	66.99
LLaMA 2	<b>98.46</b>	89.31	<b>97.87</b>	70.96	<b>89.30</b>	<b>74.41</b>	68.42	67.24	66.67

图 5 Raidar GPT 3.5 训练效果

对于不同模型进行重写，如果不用类似 GPT-3.5 的大模型，使用小模型，效果依旧显著。

Table 5: Effectiveness of detection using various large language models for rewriting. We present detection F1 scores for the same input data rewritten by Ada, Text-Davinci-002, and GPT-3.5. Among these, GPT-3.5-turbo yields the highest performance in rewriting for detection.

LLM for Rewriting	News	Creative Writing	Student Essay	Code	Yelp	Arxiv
Ada	55.73	62.50	57.02	77.42	73.33	71.75
Text-Davinci-002	55.47	60.59	58.96	82.19	75.15	59.25
GPT 3.5 turbo	<b>60.29</b>	<b>62.88</b>	<b>64.81</b>	<b>95.38</b>	<b>87.75</b>	<b>81.94</b>
LLaMA 2	56.26	61.88	60.48	85.33	74.85	72.59

图 6 Raidar 小模型重写效果

同时，Raidar 也表现出了关于文本长度越长，鉴别效果也好的趋势。

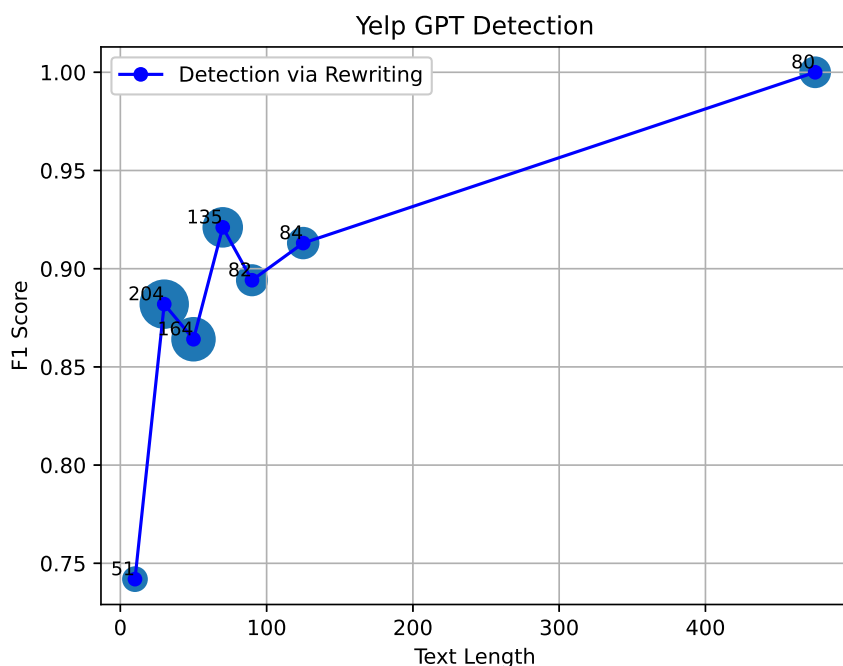


图 7 Raidar 文本长度与鉴别效果的关系

## 2.4 Advantages

- 这种方法优点之一是相比 Detect-GPT [4] 不需要 LLM 内部的 Token 概率，只需要文本输入输出即可，适用范围更广。
- 同时 Levenshtein 分数是离散的，也就是不可微。攻击者无法简单的通过梯度下降来优化输入文本来欺骗模型。
- Raidar 是给予离散符号进行运作的，一个词没有概率分布，不存在中间状态，要么是 A 要么是 B, 这样使算法对输入微小的噪声脱敏，只要整体结构和用词没有变化，检测结果不会发生大的改变。

## 参考文献

- [1] V. Verma, E. Fleisig, N. Tomlin, 和 D. Jurafsky, 《Ghostbuster: Detecting Text Ghostwritten by Large Language Models》, 收入 The Twelfth International Conference on Learning Representations (ICLR), 2024. [在线]. 载于: <https://openreview.net/forum?id=5tyoByPepe>
- [2] B. Chen 等, 《Raidar: geneRative AI Detection via Rewriting》, 收入 Proceedings of the International Conference on Learning Representations (ICLR), 2024.
- [3] X. Hu 等, 《DNA-GPT: Divergent N-Gram Analysis for Training-Free Detection of GPT-Generated Text》, arXiv preprint arXiv:2305.17359, 2024.
- [4] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, 和 C. Finn, 《DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature》, 收入 Proceedings of the 40th International Conference on Machine Learning (ICML), 2023, 页 24950~24962.