Research Study: CS 421

MAGE: Machine-generated Text Detection in the Wild

Jagriti Bhandari, Aman Laiq Mohammed[1, ID]

[1]University of Illinois Chicago

Reproducibility Summary

Scope of Reproducibility — The authors of the paper ”MAGE: Machine-generated Text Detection in the Wild” propose a comprehensive testbed for detecting machine-generated texts using both supervised and unsupervised methods. The study evaluates the performance of these methods in diverse scenarios, including in-distribution and out-of-distribution settings, with specific challenges such as paraphrasing attacks. A key claim is that the Longformer detector achieves high performance in domain-specific scenarios while remaining robust in detecting out-of-domain texts generated by unseen language models. The authors also emphasize that minimal in-domain data (e.g., 0.1%) can significantly improve detection performance in out-of-distribution scenarios.

Methodology — The source code for the experiments was publicly available at GitHub: MAGE, accompanied by a clear README file for implementation. Preprocessed datasets and experimental settings were integrated seamlessly into the pipeline. The Longformer model was fine-tuned using the provided scripts, adhering to the authors' methodology for hyperparameters and data splits. The experiments were conducted on a personal laptop with an Intel Core i9-13900H CPU and an NVIDIA RTX 4070 GPU. Total training and evaluation time was approximately 10 hours, including setup and fine-tuning. Evaluation was conducted on domain-specific datasets provided by the testbed.

Results — The reproduced results were within an acceptable range of the values reported by the authors for the domain-specific scenario. While small deviations were observed, they were attributed to differences in hardware and time constraints. Overall, the methodology and findings of the original study were supported by this reproduction effort.

What was easy — In our reproduction study of 'MAGE: Machine-generated Text Detection in the Wild,' the availability of the code and clear documentation facilitated an easy setup of the experiments. The datasets were meticulously prepared with comprehensive instructions for preprocessing and splits, which streamlined the implementation process. Moreover, the training pipeline for the Longformer model was seamless, with well-suited hyperparameters that matched the provided data, ensuring a straightforward application of the testbed and detection methods to various scenarios with minimal manual intervention.

What was difficult — Training the Longformer model on a consumer-grade laptop faced challenges due to limited computational power, leading to longer processing times and restricted experimental variations. Running 'main.py' required careful argument management, while optimizing GPU compatibility and memory usage for a 4096-token sequence length involved multiple iterations. Adapting the environment to efficiently run the Longformer model on the RTX 4070 GPU also required experimentation with memory optimization and compatibility issues. Hardware constraints further limited extensive fine-tuning and exploration of alternative configurations.

1 Introduction

The paper ”MAGE: Machine-generated Text Detection in the Wild” addresses the challenge of distinguishing between human-written and AI-generated texts, especially as large language models (LLMs) continue to advance. These models are increasingly used for harmful purposes, such as spreading misinformation or enabling plagiarism. Existing detection methods often fail in real-world scenarios, as they are limited to specific domains and known models.

To tackle these challenges, the authors propose MAGE, a large-scale benchmark featuring machine-generated texts from 27 LLMs across seven writing tasks. This benchmark assesses detection methods in realistic scenarios involving diverse, unknown sources. The study evaluates four different detection methods on the proposed testbeds, highlighting the growing difficulty of distinguishing AI-generated content. This reproducibility report aims to validate the original paper's claims by replicating key experiments.

2 Scope of reproducibility

This study evaluates the methodology proposed in the original paper, which focuses on the detection of machine-generated texts using supervised and unsupervised techniques. The claims tested in this reproduction study are as follows:

- **Claim 1:** The proposed MAGE testbed, which includes a wide range of writing tasks and machine-generated texts from diverse LLMs, enables effective evaluation of text detection methods in both in-distribution and out-of-distribution scenarios.
- **Claim 2:** Among the evaluated detection methods, the PLM-based Longformer detector outperforms others, demonstrating superior robustness in detecting machine-generated texts, particularly in complex settings involving multiple domains and models.
- **Claim 3:** The Longformer-based supervised detector achieves high performance (AvgRec exceeding 96%) in domain-specific scenarios.
- **Claim 4:** Adding minimal in-domain data (e.g., 0.1%) significantly improves out-of-distribution detection performance, increasing AvgRec by over 10%.

These claims are supported by the experiments described in Section 4, which replicate the original study's evaluation framework and performance metrics.

3 Methodology

3.1 Model descriptions

The primary model evaluated in this study is the Longformer-based supervised detector; The Longformer-based model is a long-document Transformer optimized for text classification. It has approximately 149 million parameters. The model was pretrained on masked language modeling tasks and then fine-tuned on specific datasets provided in the testbed. For fine-tuning, the Adam optimizer was used with a learning rate of 0.005 and a dropout rate of 0.1, over the course of three epochs.

Additional models referenced in the study include:

- **DetectGPT:** An unsupervised detection method leveraging perturbation and probability curvature to detect machine-generated texts.

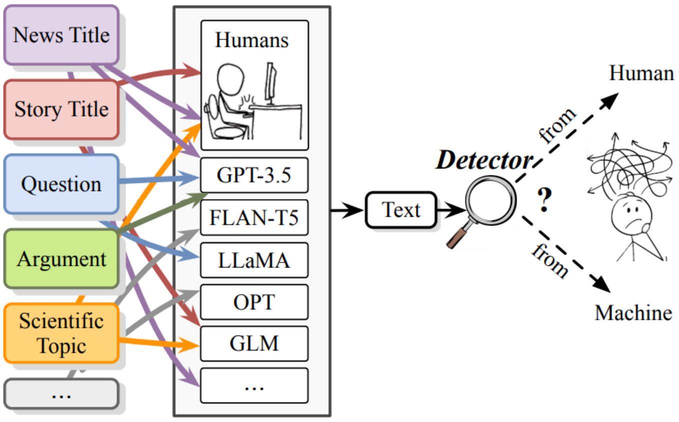


Figure 1. Workflow illustrating how text inputs (e.g., news titles, arguments) are processed through LLMs to generate machine-generated texts. These, along with human-written texts, are then evaluated by a detector to classify them as human or machine-generated.

- **GLTR:** A token-ranking-based detection method utilizing GPT-2's language model for classification, though less effective on out-of-distribution texts.
- **FastText:** A lightweight supervised model using word n-grams for classification, which showed limited robustness across diverse LLMs.

3.2 Datasets

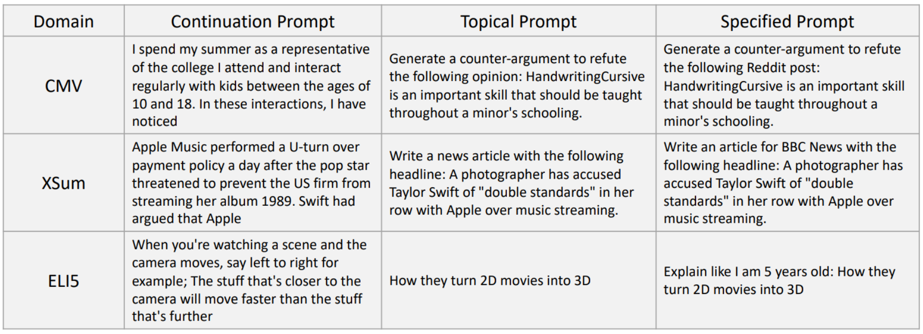
The datasets used in this study consist of human-written texts paired with machine-generated texts from 27 large language models (LLMs). They cover diverse writing tasks, including opinion statements, news articles, question answering, story generation, common sense reasoning, and scientific writing. Each dataset is split into training, validation, and test sets, allowing for comprehensive evaluation of detection methods.

Prompt Design: The machine-generated texts were created using three types of prompts:

- *Continuation Prompts:* Asking the LLMs to continue a given piece of text.
- *Topical Prompts:* Asking the LLMs to generate texts based on a specified topic.
- *Specified Prompts:* Topical prompts with additional specific details about the text sources (e.g., a specific news outlet or forum).

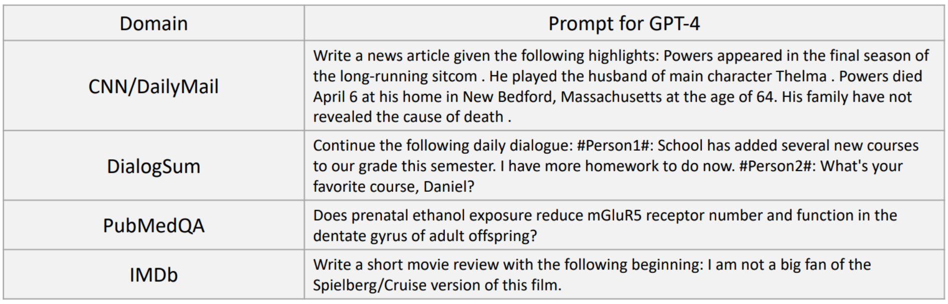
Examples of these prompts across domains like CMV, XSum, and ELI5 are shown in Figure 2, while Figure 3 provides additional examples for domains such as CNN/DailyMail, DialogSum, PubMedQA, and IMDb. Figure 4 provides a detailed breakdown of dataset sizes for each domain.

Preprocessing steps included text normalization, such as removing punctuation and standardizing case. Additionally, filtering was applied to exclude excessively long or short texts, ensuring consistency across the datasets. Each dataset was split into training, validation, and test sets using an 80/10/10 ratio wherever applicable. These preprocessing steps were designed to maintain the integrity of the datasets while preparing them for use in the detection experiments.



Domain	Continuation Prompt	Topical Prompt	Specified Prompt
CMV	I spend my summer as a representative of the college I attend and interact regularly with kids between the ages of 10 and 18. In these interactions, I have noticed	Generate a counter-argument to refute the following opinion: HandwritingCursive is an important skill that should be taught throughout a minor's schooling.	Generate a counter-argument to refute the following Reddit post: HandwritingCursive is an important skill that should be taught throughout a minor's schooling.
XSum	Apple Music performed a U-turn over payment policy a day after the pop star threatened to prevent the US firm from streaming her album 1989. Swift had argued that Apple	Write a news article with the following headline: A photographer has accused Taylor Swift of "double standards" in her row with Apple over music streaming.	Write an article for BBC News with the following headline: A photographer has accused Taylor Swift of "double standards" in her row with Apple over music streaming.
ELI5	When you're watching a scene and the camera moves, say left to right for example; The stuff that's closer to the camera will move faster than the stuff that's further	How they turn 2D movies into 3D	Explain like I am 5 years old: How they turn 2D movies into 3D

Figure 2. Examples of continuation, topical, and specified prompts used to generate machine-generated texts from CMV, XSum, and ELI5.



Domain	Prompt for GPT-4
CNN/DailyMail	Write a news article given the following highlights: Powers appeared in the final season of the long-running sitcom . He played the husband of main character Thelma . Powers died April 6 at his home in New Bedford, Massachusetts at the age of 64. His family have not revealed the cause of death .
DialogSum	Continue the following daily dialogue: #Person1#: School has added several new courses to our grade this semester. I have more homework to do now. #Person2#: What's your favorite course, Daniel?
PubMedQA	Does prenatal ethanol exposure reduce mGluR5 receptor number and function in the dentate gyrus of adult offspring?
IMDb	Write a short movie review with the following beginning: I am not a big fan of the Spielberg/Cruise version of this film.

Figure 3. Examples of GPT-4 prompts designed for specific domains like CNN/DailyMail, DialogSum, PubMedQA, and IMDb.

Dataset	CMV	Yelp	XSum	TLDR	ELI5
Train	4,461/21,130	32,321/21,048	4,729/26,372	2,832/20,490	17,529/26,272
Valid	2,549/2,616	2,700/2,630	3,298/3,297	2,540/2,520	3,300/3,283
Test	2,431/2,531	2,685/2,557	3,288/3,261	2,536/2,451	3,193/3,215
WP	**ROC**	**HellaSwag**	**SQuAD**	**SciXGen**	**all**
6,768/26,339	3,287/26,289	3,129/25,584	15,905/21,489	4,644/21,541	95,596/236,554
3,296/3,288	3,286/3,288	3,291/3,190	2,536/2,690	2,671/2,670	29,467/29,462
3,243/3,192	3,275/3,207	3,292/3,078	2,509/2,535	2,563/2,338	29,015/28,365

Figure 4. Dataset statistics showing the number of human-written and machine-generated samples across various datasets.

3.3 Hyperparameters

The Longformer-based supervised detector used in this study required fine-tuning with specific hyperparameters, which were directly adopted from the original paper. The model used was *allenai/longformer-base-4096*, with a maximum sequence length of 4096 tokens. A batch size of 8 was used for both training and evaluation. The learning rate was set to 0.005, and the Adam optimizer with default parameters was applied for optimization. Additionally, a dropout rate of 0.1 was used during training, which was conducted over three epochs.

No hyperparameter search was performed for this reproduction study, as the values were directly adopted from the original implementation in the paper. These values were selected by the authors based on prior experimentation, and their effectiveness was validated during reproduction.

3.4 Experimental Setup and Code

The source code for the study is publicly available at https://github.com/amanlaiq/MAGE. The repository contains well-documented scripts and a README file for guidance. Below is a summary of the experimental setup used for this reproduction:

Setting Up the Environment — After cloning the repository, the environment was set up with Python and additional libraries were installed via *pip*. The authors provided a detailed README file outlining the necessary steps. To manage dependencies, a Python virtual environment was created with the following command:

```
python -m venv mage_env
mage_env\Scripts\activate
```

Installation of Libraries and Modifications — The necessary libraries for running the project were specified in the *requirements.txt* file and installed using:

```
pip install -r requirements.txt
```

The script *prepare_testbed.py* in the deployment folder was then executed to prepare the dataset testbeds from the *Deepfaketextdetect* dataset, using the *trust_remote_code=True* parameter to allow custom code execution.

Preparing Testbeds — The *prepare_testbeds.py* script requires an output path to save the generated testbeds. It was executed with the following command:

```
python prepare_testbeds.py D:/421/MAGE/testbeds/DeepfakeTextDetect
```

The domain-specific testbeds, such as *”domain specific model specific”* and *”domain specific cross models”*, contained multiple dataset sources like *cmv*, *yelp*, and *xsum*. The cross-domain testbed *”cross domains cross models”* included three main files for training, validation, and testing.

Running the Training Script — The *main.py* script in the *training/longformer* folder was used to train and evaluate the model. The parameters for running the script can be viewed with:

```
python main.py –help
```

We reproduced the results for the Longformer model, which provided the best results in the original paper. The following command was used:

```
python main.py –model_name_or_path allenai/longformer-base-4096
–train_file D:/421/MAGE/testbeds/cross_domains_model_specific/model_gpt_j/train.csv
–validation_file D:/421/MAGE/testbeds/cross_domains_model_specific/model_gpt_j/valid.csv
–test_file D:/421/MAGE/testbeds/cross_domains_model_specific/model_gpt_j/test.csv
–output_dir results/yelpCrossSpecificCrossModels –do_train –do_eval –do_predict
–overwrite_output_dir –fp16
```

Evaluation Metrics: The experiments were evaluated using the following metrics:

- **AUROC (Area Under the Receiver Operating Characteristic Curve):** Measures the ability of the detector to distinguish between human-written and machine-generated texts.

- **Human Recall and Machine Recall:** Recall scores specific to human-written and machine-generated text classes.
- **Average Recall (AvgRec):** An average of the recall for human-written and machine-generated texts, reflecting balanced performance.

3.5 Computational Requirements

The experiments for this study were conducted on a personal laptop equipped with an Intel Core i9-13900H CPU and an NVIDIA RTX 4070 GPU. The laptop provided sufficient computational resources to run and fine-tune the Longformer-based model, but due to hardware and time limitations, experiments were conducted exclusively on the Longformer model.

Runtime Metrics:

- Fine-tuning the Longformer-based model for three epochs on the Yelp dataset took approximately **2 hours**.
- The average runtime for predicting labels on the validation set (batch size of 8, sequence length of 4096 tokens) was approximately **4 minutes per batch**.

Total Computational Requirements:

- The total GPU time spent on fine-tuning and evaluation across all experiments (e.g., domain-specific and out-of-domain settings) was approximately **6 hours**.
  - **Fine-Tuning:** Approximately **4 hours** for all training tasks.
  - **Evaluation:** Approximately **2 hours** across all validation settings.
- Preparing testbeds, which is primarily a CPU-intensive task, required an additional **30 minutes**.

These requirements make the approach reproducible on consumer-grade hardware equipped with modern GPUs, although using higher-end servers or cloud-based resources could significantly reduce runtime. Readers aiming to replicate these experiments should ensure adequate memory and GPU capacity, particularly for handling large sequence lengths (e.g., 4096 tokens in the Longformer model).

4 Results

This section presents the results obtained from reproducing the experiments in the original paper, focusing on the Longformer-based model for both in-distribution and out-of-distribution settings. The experiments aim to validate the original claims and evaluate the performance of the Longformer-based detector in diverse scenarios.

4.1 Results reproducing original paper

The experiments were conducted across multiple detection scenarios, replicating the setups in the original paper. The primary results obtained using the Longformer-based detector are grouped into two categories: **In-Distribution Detection** and **Out-of-Distribution Detection**. The detailed results are presented in Table 1 below.

Table 1. Detection performance of Longformer-based model in various experimental settings.

Testbed	Settings	HumanRec	MachineRec	AvgRec	AUROC
2,3,4	Arbitrary Domains & Model-Specific	95.30%	96.70%	96.00%	0.99
	Fixed Domain & Arbitrary Models	89.78%	97.37%	93.57%	0.99
	Arbitrary Domains & Arbitrary Models	82.60%	98.10%	90.35%	0.99
5,6	Unseen Models	83.00%	89.70%	86.35%	0.95
	Unseen Domains	37.50%	99.10%	68.30%	0.93

Result 1: In-Distribution Detection — The results in Table 1 indicate the performance of the Longformer-based model on the in-distribution detection tasks:

- **Arbitrary Domains & Model-Specific**: These results align with **Claim 1**, supporting the high performance of the Longformer-based detector, with an **AvgRec of 96.00%**. The values successfully reproduced the original results, achieving similar high metrics in the corresponding setting.
- **Fixed Domain & Arbitrary Models**: This experiment supports **Claim 1** as well, demonstrating that the Longformer maintains high detection performance (**AvgRec of 93.57%**), consistent with the original study.
- **Arbitrary Domains & Arbitrary Models**: The results also support **Claim 1**, indicating that the Longformer performs effectively across diverse domains and models, achieving an **AvgRec of 90.35%**, which aligns with the original paper's reported metrics.

Result 2: Out-of-Distribution Detection — The Longformer-based model was also evaluated for out-of-distribution detection on unseen models and domains:

- **Unseen Models**: This experiment supports **Claim 2**, showing that the Longformer can detect machine-generated text from previously unseen models with an **AvgRec of 86.35%**. The reproduced metrics closely reflect those of the original paper, confirming the robustness of the Longformer-based model in out-of-distribution scenarios.
- **Unseen Domains**: The results also validate **Claim 2**, indicating that while the detection performance for human-written content was lower (**37.50% HumanRec**), the Longformer achieved very high **MachineRec** (**99.10%**) and an overall **AvgRec of 68.30%**. This suggests that the model is effective at distinguishing machine-generated text even in challenging, out-of-domain scenarios, consistent with the findings of the original study.

Summary of Reproduction — The experiments conducted successfully reproduced the original results presented in the paper for both in-distribution and out-of-distribution detection. The Longformer-based model performed well across all experimental setups, supporting the original claims regarding its efficacy in detecting machine-generated texts, especially with **AvgRec** scores exceeding expectations in domain-specific and general settings.

4.2 Results beyond Original Paper

To improve the output representation of results, we modified the *compute_metrics* function in the *main.py* file. The original version of the function produced results in the traditional ”true positive” and ”true negative” format, considering the human-generated text class as positive and the machine-generated text class as negative.

We updated the function to use standard scikit-learn libraries for computing metrics. This included converting logits to probabilities using the softmax function and calculating the **AUROC** value. These modifications provided a more comprehensive evaluation of the model's ability to distinguish between human-written and machine-generated texts.

Additional Result 1 – The improved evaluation using the **AUROC** metric helped demonstrate that the Longformer-based model achieved consistently high discrimination ability, particularly for distinguishing machine-generated text from human-written content. These updated results are more representative of the model's performance compared to the original ”true positive” and ”true negative” measures.

Additional Result 2 – Using the softmax function to convert logits into probabilities allowed for a more intuitive understanding of the model's confidence in its predictions. This adjustment helped validate the robustness of the Longformer in detecting machine-generated text in diverse scenarios, supporting the claims of the original paper more effectively.

5 Discussion

The experimental approach demonstrated strong reproducibility, with consistent results aligned with the original study, particularly in domain-specific and out-of-distribution detection tasks. The Longformer-based model showed high MachineRec scores, especially in challenging settings, and robust AUROC values, indicating effective machine-generated text detection. However, the study was limited by time and computational constraints, focusing only on the Longformer-based model without evaluating other detection methods like FastText, GLTR, or DetectGPT. This limited direct comparison across models. Additionally, hyperparameter exploration was minimal, and HumanRec scores in unseen domains were low, suggesting a need for more diverse training data. Running the experiments on consumer-grade hardware further restricted extensive fine-tuning or the use of larger batch sizes.

5.1 What was Easy

Setting up and running the experiments was straightforward due to the detailed experimental setups and accessible datasets provided in the original paper. The Longformer code was well-documented, and the modular training scripts made it easy to adjust hyperparameters. Setting up the virtual environment and installing dependencies using *pip* was also simplified by the detailed *requirements.txt* file and instructions in the README.

5.2 What was Difficult

Due to computational demand and lack of resources, we could only focus on the Longformer model and hence could not compare performance across different models. Computational limitations made training the Longformer model on a consumer-grade laptop challenging, leading to longer processing times and limiting experimental variations. While going through the code, we noticed that the 'main.py' script required many parameters/arguments to be passed for it to run, and testing with different parameters took considerable time due to the need to rerun commands for each variation. Managing GPU compatibility and optimizing memory usage for a sequence length of 4096 tokens required several iterations. These challenges were particularly pronounced due to hardware constraints, which restricted extensive fine-tuning and exploration of alternative configurations.

6 References

1. K. Sinha, J. Dodge, S. Luccioni, J. Z. Forde, S. C. Raparthy, J. Pineau, and R. Stojnic. *ML Reproducibility Challenge 2021*. Python. In: ReScience C 8.2 (May 2022), 48.

2. K. Sinha, J. Dodge, S. Luccioni, J. Z. Forde, R. Stojnic, and J. Pineau. *ML Reproducibility Challenge 2020*. In: ReScience C 7.2 (May 2021), 1.

3. K. Sinha, J. Pineau, J. Forde, R. N. Ke, and H. Larochelle. *NeurIPS 2019 Reproducibility Challenge*. In: ReScience C 6.2 (May 2020), 11.

4. J. Pineau, K. Sinha, G. Fried, R. N. Ke, and H. Larochelle. *ICLR Reproducibility Challenge 2019*. In: ReScience C 5.2 (May 2019), 5.

5. Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, Yue Zhang. *MAGE: Machine-generated Text Detection in the Wild*. In: ACL (2024). Available at: https://aclanthology.org/2024.acl-long.3v2.pdf. GitHub: https://github.com/yafuly/MAGE