Mahalanobis OOD Detection for AI-Generated Text Classification

Dmitry Gorbunov
ITMO University, AI Talent Hub

2025

Abstract

We reproduce the AINL-Eval 2025 winning solution (sastsy, 91.22%) for detecting AI-generated scientific abstracts in Russian. The key challenge is identifying texts from unknown AI models not seen during training. We apply Mahalanobis distance-based OOD detection to Qwen2.5-7B with Dual-Head architecture, improving accuracy by +10.4% over softmax confidence (from 79.53% to 89.97%) with 76.25% unknown class recall. Mahalanobis also boosts lightweight ruBERT-tiny2 (29M params) to 85.25% – only 4.7% below Qwen, but 20x faster (15ms CPU) and 127x smaller. Code: `https://github.com/dpGorbunov/nlp-sem-project`.

1 Introduction

As LLMs become increasingly capable, distinguishing human-written scientific texts from AI-generated ones grows harder. A student can now generate a plausible abstract in seconds, and reviewers cannot reliably tell the difference. This threatens the integrity of scientific publishing.

The AINL-Eval 2025 shared task [5] tackles this challenge for Russian scientific abstracts. The task is not just binary (human vs AI) – it requires identifying *which* AI model generated the text, including an "unknown" class for models not seen during training. This makes the problem significantly harder: the system must generalize to new AI models.

The winning solution by team sastsy achieved 91.22% accuracy using GigaCheck [1] with a Dual-Head modification. We reproduce and extend this approach:

1. **Qwen2.5-7B**: stronger backbone than Mistral-7B on benchmarks
2. **Mahalanobis OOD Detection**: distance-based method for unknown class detection (+10.4% over baseline)
3. **Knowledge Distillation**: distillation to ruBERT-tiny for CPU inference

1.1 Team

Dmitry Gorbunov – model architecture design, experiments, report writing.

2 Related Work

GigaCheck [1] established a strong baseline for LLM-generated content detection using Mistral-7B fine-tuned with LoRA [3]. LoRA enables efficient adaptation by learning low-rank updates: $W' = W + \frac{\alpha}{r} \cdot BA$, where only matrices B and A are trained. GigaCheck uses EOS token pooling and a single classification head.

The sastsy solution [5] improved GigaCheck by splitting classification into two heads: binary (human vs AI) and multiclass (which AI model). This separation helps because detecting AI is easier than identifying the specific model.

3 Model Description

3.1 Architecture Overview

We follow the sastsy architecture but replace Mistral-7B with Qwen2.5-7B. Why Qwen? According to the Qwen2 Technical Report [2], it outperforms Mistral on standard benchmarks (Tab. 1), suggesting better text understanding.

The model has four components:

1. **Backbone**: Qwen2.5-7B fine-tuned with LoRA (r=8, alpha=16) – we only train 0.04% of parameters
2. **Pooling**: EOS token embedding (the last token captures the full sequence context)
3. **Shared Layer**: Linear + tanh + dropout (transforms embeddings before classification)
4. **Dual-Head**: Two classification heads working together

Benchmark	Qwen2-7B	Mistral-7B	Δ
MMLU	**70.3**	64.2	+6.1
HumanEval	**51.2**	29.3	+21.9
GSM8K	**79.9**	52.2	+27.7

Table 1: Qwen2-7B vs Mistral-7B benchmark comparison.

3.2 Dual-Head Architecture

The key insight from sastsy: separate easy and hard tasks.

Binary Head answers: "Is this AI-generated?" This is relatively easy – AI texts have subtle but consistent patterns.

Multiclass Head answers: "Which AI model?" This is harder – different LLMs produce similar outputs.

During training, both heads are optimized jointly:

$$\mathcal{L} = \mathcal{L}_{CE}^{bin} + \mathcal{L}_{CE}^{multi}$$

The multiclass loss ignores human samples (they have no AI model label).

During inference: if binary predicts "human" → output human. Otherwise, use multiclass prediction. The "unknown" class is not predicted directly – it is detected via Mahalanobis distance on embeddings.

3.3 Mahalanobis OOD Detection

The multiclass head can only predict known classes (GPT-4, Llama, Gemma). To detect unknown AI models, we use Mahalanobis distance [6] in embedding space:

$$D_M(x, c) = \sqrt{(x - \mu_c)^T \Sigma^{-1} (x - \mu_c)}$$

Unlike softmax confidence, which only considers output logits, Mahalanobis measures geometric distance to class centroids accounting for correlations. Samples far from all known classes are flagged as "unknown" – exactly what we need for unseen AI models.

4 Dataset

The AINL-Eval 2025 dataset [5] contains Russian scientific abstracts from four sources: human-written, GPT-4-Turbo, Llama-3.3-70B, and Gemma-2-27B (Tab. 2).

	Train	Dev	Test
Samples	35,158	10,979	6,169
Classes	4	5	5

Table 2: AINL-Eval 2025 dataset statistics.

The critical twist: dev and test sets include a fifth class – "unknown" – generated by models *not present in training* (GigaChat-Lite in dev, DeepSeek-V3 in test). A classifier trained on four classes has never seen these models and must somehow recognize "this looks like AI, but not any AI I know."

This is the core OOD detection challenge. Standard softmax confidence fails here: the model confidently misclassifies unknown samples as one of the known AI models. Mahalanobis distance solves this by measuring distance in embedding space – unknown samples are far from all class centroids.

Interesting observation: Human texts are longer (126 words on average) and contain 10x more digits than AI-generated ones [5]. This suggests simple features could help, but our TF-IDF baseline shows they are not enough.

5 Experiments

5.1 Metrics

Primary metric: **Accuracy** (as per competition rules).

We also report precision, recall, and F1-score per class, and visualize results with confusion matrices.

5.2 Experiment Setup

- GPU: NVIDIA A100 40GB
- Precision: bfloat16
- Batch: 16
- Learning rate: 3e-5
- Epochs: 10, Early stopping: patience=3
- LoRA: r=8, alpha=16, targets: q_proj, v_proj

5.3 Baselines

- TF-IDF + Logistic Regression
- ruBERT-tiny fine-tuned
- sastsy [5]: 1st place winner (GigaCheck-based)

6 Results

Tab. 3 shows our main results. The "Base" column is accuracy without any unknown detection – the model predicts one of 4 known classes for every sample. "Confidence" uses softmax threshold: if max probability is below a threshold, predict "unknown". "Mahalanobis" uses distance in embedding space.

Why does Mahalanobis work so much better? Softmax confidence measures how "sure" the model is about its prediction, but a model trained only on GPT-4/Llama/Gemma will confidently assign GigaChat samples to one of these classes – it has no concept of "none of the above." Mahalanobis distance, computed on the shared layer embeddings, measures geometric distance to class clusters. Unknown AI models produce embeddings that are far from all known class centroids, making them detectable.

Key findings:

Method	Base	Confidence	Mahalanobis
AINL-Eval 2025 results [5]:			
TF-IDF baseline (competition)	80.81%	–	–
sastsy (1st place) [5]	91.22%	–	–
Our experiments:			
TF-IDF + LogReg	76.85%	81.06%	–
ruBERT-tiny (fine-tuned)	78.29%	80.29%	85.25%
Qwen2.5 + Dual-Head (ours)	79.53%	82.61%	**89.97%**

Table 3: Comparison of methods with different OOD detection strategies.

1. Mahalanobis boosts accuracy from 79.53% to 89.97% (+10.4%). The binary head alone achieves 95.38% (human vs AI is easy), and unknown recall reaches 76.25%.
2. ruBERT-tiny2 with Mahalanobis achieves 85.25% – only 4.7% below Qwen, but 20x faster and 127x smaller. This makes real-time CPU deployment feasible.

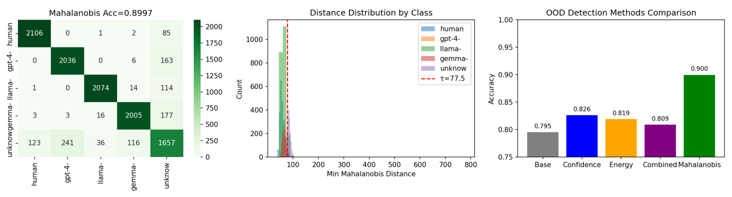

Figure 1: OOD detection methods comparison. Mahalanobis distance achieves the best accuracy (89.97%) on the dev set.

6.1 Knowledge Distillation

Qwen2.5-7B requires GPU and is too slow for real-time applications. Following DisRanker [4], which showed LLM knowledge can be distilled to BERT with 10x speedup, we compress to ruBERT-tiny2 (29M params):

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{KL}(p_s, p_t) \cdot T^2 + (1 - \alpha) \cdot \mathcal{L}_{CE}(p_s, y)$$

where $T = 4$ (temperature), $\alpha = 0.7$.

We compare two approaches:

- Fresh BERT + KD: training from scratch with distillation
- Fine-tuned BERT + KD: further training of already fine-tuned model

Observation: Distillation from fresh BERT achieves lower accuracy than fine-tuning baseline. This is expected since fresh BERT requires more training to learn from scratch. Fine-tuned BERT + KD achieves the same accuracy as the baseline, suggesting the model has already converged.

Model	Size	Inference	Raw Acc	+Mahalanobis
Qwen2.5-7B (teacher)	15 GB	~300ms (GPU)	79.53%	**89.97%**
ruBERT-tiny2 (fine-tuned)	118 MB	~15ms (CPU)	78.29%	85.25%
Fresh BERT + KD	118 MB	~15ms (CPU)	74.21%	80.44%
Fine-tuned BERT + KD	118 MB	~15ms (CPU)	77.01%	85.25%

Table 4: Teacher vs Student comparison. Qwen: 7.61B params [2], ruBERT-tiny2: 29M params [7] (260× fewer params, 127× smaller file size). Inference times based on [8].

7 Conclusion

The main takeaway from this work: **detecting unknown AI models is hard, but Mahalanobis distance makes it tractable**. Standard confidence-based methods fail because neural networks are confidently wrong on out-of-distribution samples. Mahalanobis operates in embedding space where "unknown" means "far from everything I've seen" - a much more robust criterion.

Our best model (Qwen2.5-7B + Dual-Head + Mahalanobis) achieves 89.97% accuracy, with 76.25% recall on the unknown class. But perhaps more interesting is that ruBERT-tiny2 - a model 260x smaller - reaches 85.25% with the same Mahalanobis approach. This suggests that the OOD detection method matters more than model size for this task.

Practical implications: A 29M parameter model running in 15ms on CPU can detect AI-generated scientific abstracts with reasonable accuracy. This enables deployment in resource-constrained environments - browser extensions, email filters, or mobile apps - without GPU infrastructure.

Limitations: Mahalanobis requires computing class statistics from training data embeddings upfront. If the distribution of AI models shifts (new models appear), these statistics need recomputation. Future work could explore online or adaptive OOD detection methods.

References

[1] Tolstykh, I., Tsybina, A., Yakubson, S., Gordeev, A., Dokholyan, V., Kuprashevich, M. (2024). GigaCheck: Detecting LLM-generated Content. *arXiv preprint arXiv:2410.23728.*

[2] Qwen Team. (2024). Qwen2 Technical Report. *arXiv preprint arXiv:2407.10671.*

[3] Hu, E. J., et al. (2022). LoRA: Low-Rank Adaptation of Large Language Models. *ICLR 2022.*

[4] Ye, D., et al. (2024). Best Practices for Distilling Large Language Models into BERT for Web Search Ranking. *arXiv preprint arXiv:2411.04539.*

[5] Batura, T., Bruches, E., Shvenk, M., Malykh, V. (2025). AINL-Eval 2025 Shared Task: Detection of AI-Generated Scientific Abstracts in Russian. *arXiv preprint arXiv:2508.09622*. `https://codalab.lisn.upsaclay.fr/competitions/21895`

[6] Lee, K., et al. (2018). A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks. *NeurIPS 2018*.

[7] Dale, D. (2022). ruBERT-tiny2: Russian Sentence Encoder. *Habr*. `https://habr.com/ru/post/669674/`, `https://huggingface.co/cointegrated/rubert-tiny2`

[8] Boudier, M., Music, D. (2022). Scaling up BERT-like model Inference on modern CPU. *Hugging Face Blog*. `https://huggingface.co/blog/bert-cpu-scaling-part-1`