Detecting Machine-Generated Text in Polish Using Fine-Tuned Qwen Models

Konrad Pierzyński
Adam Mickiewicz University
ul. Uniwersytetu Poznańskiego 4
61-614 Poznań, Poland
konrad.pierzynski@amu.edu.pl

Abstract

This paper presents a system submitted to the PolEval 2025 ŚMIGIEL shared task on detecting machine-generated Polish text. Within the CONSTRAINED setting, several Qwen3 models were fine-tuned using only the organisers' data and publicly available checkpoints. The study details the dataset, formulates an adaptation of a decoder-only language model for binary classification, and describes the end-to-end training pipeline. The best model, Qwen/Qwen3–8B, attains 93.11% accuracy on Test A and 92.53% on the hidden Test B. These results show that multilingual decoder-based LLMs can be strong discriminators of human- and machine-authored Polish text when appropriately adapted. The discussion also outlines areas in which robustness can be improved and points to avenues that may enable accuracy to surpass the reported results.

1 Introduction

Large language models have transformed modern NLP, enabling fluent text generation across languages, registers, and domains. However, these same capabilities also make it increasingly difficult to distinguish machine-generated text (MGT) from human-written content. Reliable discrimination is important for applications ranging from misinformation mitigation to academic integrity and content authenticity systems.

The ŚMIGIEL shared task at PolEval 2025 (Przybyła et al., 2025) provides a controlled benchmark for MGT detection in Polish—a morphologically rich West Slavic language whose inflectional complexity and syntactic flexibility challenge both generators and discriminators. This paper examines whether a multilingual decoder-only model, trained primarily for next-token prediction, can be effectively repurposed as a binary classifier given only supervised fine-tuning data.

This paper presents a complete pipeline for such adaptation using Qwen3 models. Work is carried out within the *CONSTRAINED* track, where only the organisers' dataset and publicly available pretrained checkpoints may be used. The dataset, architecture modification, training design, evaluation results, and practical observations from model behaviour are described in detail.

All code used in experiments, is publicly available at https://github.com/kpierzynski/poleval-smigiel.

2 Task Description

The ŚMIGIEL task is a binary classification problem. Given an input text, the system must assign:

- **0** – Human (human-written),
- **1** – AI (machine-generated).

The official evaluation metric is accuracy:

$$\text{Accuracy} = \frac{|\text{correct predictions}|}{|\text{all samples}|}.$$

Accuracy reflects the overall proportion of correctly classified instances and serves as the primary ranking criterion.

Although accuracy is intuitive, additional metrics might provide complementary insight into error types. Precision is defined as

$$\text{Precision} = \frac{TP}{TP + FP},$$

indicating how often predicted AI-generated texts are correct. Recall is given by

$$\text{Recall} = \frac{TP}{TP + FN},$$

capturing the fraction of AI-generated texts that are successfully detected. Their harmonic mean,

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}},$$

balances the two and is particularly informative under class imbalance or asymmetric error costs.

Evaluation is carried out on two datasets:

Proceedings of the PolEval 2025 Workshop, pages 16–20
November 21, 2025 ©2025 Association for Computational Linguistics

- **Test A** — public,
- **Test B** — private (labels withheld).

3 Dataset

3.1 Structure

The dataset consists of two files: `data.tsv` and `labels.tsv`. Each line in the label file corresponds to the same line in the data file, with 35,763 examples in total. The distribution is nearly balanced: 17,890 AI samples and 17,873 Human samples.

3.2 Domains

Human-written data comes from:

- Wikipedia passages,
- literary texts,
- social media posts,
- product reviews.

Machine-generated data was created using a diverse collection of open-source LLMs. According to the organisers' description, the training set includes generations from multiple model families such as LLama, Mistral, Bielik, PLLuM and larger Gemma. This diversity is crucial: it prevents the classifier from overfitting to idiosyncratic traits of a single generator and encourages learning broader stylistic cues indicative of synthetic text.

3.3 Example Data

Below are examples from the training dataset that were used in finetuning.

Label: AI
"Nie do końca tak jest. Program Smart to usługa oferowana przez Allegro, która zapewnia kupującym darmową dostawę i zwrot produktów przy zakupie u danego sprzedawcy. Sprzedawca musi spełnić określone . . . "

Label: Human
"Rozwalone składy owszem są bardzo ważne i na jakiś czas faktycznie zatrzymały strzelanie, ale ilość rozwalonych w ten sposób pocisków nie sądzę aby była duża (w porównaniu do zużycia 50k dziennie). Ukraińcy rozwalali składy przejściowe, które z założenia mają magazynować amunicję dla lokalnego odcinku frontu . . . "

Label: AI
"Ciekawe, że niektórzy senatorowie wyrażali wątpliwości co do jej skuteczności w ochronie prywatności obywateli. Czy uważacie, że ta ustawa jest wystarczająco skuteczna w zapewnieniu bezpieczeństwa . . . "

3.4 Observations

The dataset covers a broad range of domains in which AI-text detection systems are practically relevant. Both human and machine generated samples appear in multiple styles and levels of formality, which makes the task realistic and prevents reliance on superficial cues. The data preparation procedure—in particular the balanced label distribution and sufficient volume—provides a solid basis for training a classifier with good generalisation ability. As a result, the dataset is well created for developing models that can indeed detect AI-generated Polish text across real world scenarios.

4 Methods

4.1 Base Architecture: Qwen3

Qwen3 is a versatile family of decoder-only Transformer models offered in multiple sizes (from 0.6B to 235B), built on a proven architecture and widely adopted due to strong performance across tasks. The models provide reliable multilingual coverage, including Polish, which makes them a practical choice for downstream applications (Yang et al., 2025).

Although Qwen3 is trained for next-token prediction, its internal representations transfer well to classification. By replacing the generative LM Head with a simple classifier, the pretrained model becomes an effective text encoder suitable for wide range of text classifcation tasks.

4.2 Adapting a Generative LM for Classification

A decoder-only LLM typically ends with a large linear layer projecting hidden states to vocabulary logits. This LM Head is aligned with next-token prediction, not sequence-level classification.

For classification, token-level generation is unnecessary. Instead, the model should produce a single binary label based on the entire input sequence.

The transformation is conceptually simple:

- Generative LM Head is removed.
- A Classification Head is added, a linear projection from the final hidden state to two logits.

This setup parallels how generative models encode semantics: transformer layers still build a contextual representation of the entire sequence, but instead of using it to predict the next token,

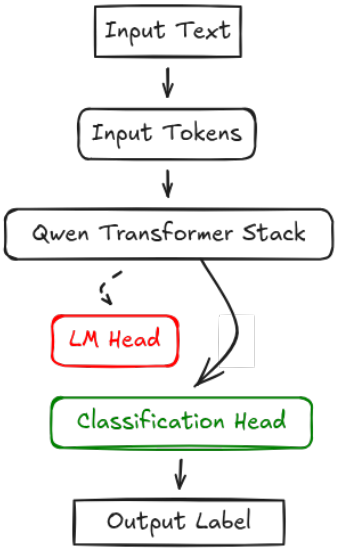

Figure 1: Modified model architecture

the final hidden state is repurposed as the input to a classifier. Last-token pooling was chosen both for its simplicity and because decoder-only models tend to compress sequence-level information into the representation of the final token (Radford et al., 2018).

4.3 Training Pipeline

Before listing the procedural steps, it is useful to describe the overall philosophy guiding finetuning. The goal is to preserve the strong multilingual representations already present in Qwen while allowing a lightweight classifier to specialise in the AI-vs-human distinction. Given the dataset size and the model's multilingual pretraining, training target stable convergence within a single epoch.

The core idea is that almost all parameters belong to the transformer backbone, which already captures syntax, semantics, coherence, and discourse structure, while classification head is comparatively tiny. As a result:

- the model adapts quickly,
- overfitting is unlikely within one epoch,
- most learning consists of shaping decision boundaries in the last-layer representation space.

Fine tuning steps

1. **Tokenizer preparation.** Load the pretrained tokenizer and remap the padding token to the EOS token to ensure consistent padding behaviour.
2. **Model loading.** The original LM Head is replaced by a Classification Head, using `AutoModelForSequenceClassification` with `num_labels=2` (Wolf et al., 2019).
3. **Dataset construction.** Load `data.tsv` and `labels.tsv` using a custom dataset class tailored to the task.
4. **Tokenisation.** Apply truncation to 256 tokens and pad sequences to a fixed length to enable efficient batching.
5. **Dataloaders.** A batch size of 4 provides a good balance between optimisation stability, memory limits, and generalisation behaviour.
6. **Hyperparameters.**

Hyperparameter	Value
Learning rate	5×10^{-5}
Epochs	1
Batch size	4
Optimizer	AdamW

Table 1: Fine tuning hyperparameters.

The settings in Table 1 were chosen based on prior experience to balance convergence speed and memory footprint.

7. **Training loop.** For each batch, perform:

forward pass → compute loss → backward pass → parameter update.

Training one epoch on Qwen3–8B required approximately 1 hour on a single A100 80GB GPU.

8. **Saving the model.** Save the final checkpoint in the standard Hugging Face format for straightforward downstream inference.

5 Results

Table 2 reports the accuracy obtained by the evaluated Qwen3 models of different sizes, showing consistently high performance across all variants. All variants achieve high accuracy on Test A, with results ranging from 89.0% for Qwen3–0.6B to 93.11% for Qwen3–8B. The intermediate model, Qwen3–1.7B, attains 91.55% accuracy, placing it between the smaller and larger variants.

A consistent increase in performance is observed as model size grows. The largest model, Qwen3–8B, achieves the highest accuracy on both evaluation sets, with 93.11% on Test A and 92.53% on Test B. These results indicate a clear scaling trend within the tested Qwen3 family, where larger models benefit from increased parameter capacity.

Model	Params	Test A	Test B
Qwen3–8B	8B	**0.9311**	**0.9253**
Qwen3–1.7B	1.7B	0.9155	–
Qwen3–0.6B	0.6B	0.8900	–

Table 2: Evaluation results.

5.1 Training Behaviour

Even a single epoch was sufficient for convergence, likely due to:

- strong multilingual pretraining of Qwen,
- relatively large dataset,
- straightforward binary classification objective.

Fast adaptation. Within a few thousand steps the model transitioned from near-random predictions to highly stable behaviour. This is characteristic of finetuning decoder-only LLMs on tasks where a linear classifier is sufficient to separate high-level representations.

No overfitting within one epoch. Because only the upper layer is replaced, memorisation is limited; the backbone already encodes broad linguistic patterns, so the classifier mostly learns to interpret them for this specific task.

Generalisation patterns. Qwen-8B performance on Test B closely tracks Test A, suggesting robustness across domains and generator types. It appears to capture subtle stylistic signatures that transcend dataset-specific artefacts.

6 Discussion

6.1 Choice of Qwen3–8B for Test B

Since Qwen3–8B consistently outperformed smaller variants on Test A, it was selected for the final evaluation. Its additional parameter capacity appears to enhance semantic sensitivity and capture soft signals correlated with AI generation.

6.2 Potential Improvements

Although 92.53% accuracy is competitive, crossing 95% or approaching 99% would require substantial refinements. Several avenues merit exploration:

- **More complex classification head.** Currently, only one linear layer is used. Adding more advanced classifier could better capture decision boundaries.

- **Hyperparameter tuning.** Further tuning the hyperparameters could slightly improve model accuracy. Among them, the number of epochs and the batch size will likely have the biggest impact.

- **More epochs with regularisation.** While one epoch prevents overfitting, multiple epochs with dropout or layer-wise learning rate decay might systematically improve model quality.

- **Intermediate model sizes (4B).** This would illuminate whether performance scales smoothly or non-linearly between 1.7B and 8B, helping choose an optimal cost-accuracy trade-off.

- **Very large Qwen3 models (14B–32B).** Testing bigger backbones could provide insights into the upper limit of this architecture for spotting AI generated texts.

Collectively, these steps could improve the classification capabilities of the given architecture and provide insights into the cost-efficiency trade-off, helping to select the best solution.

7 Conclusion

This paper presented a complete system for detecting machine-generated Polish text using finetuned Qwen3 models. By replacing the LM Head with a simple classifier, the models can effectively discriminate between human and machine authorship. Qwen3–8B achieved the strongest results, generalising well to the hidden Test B set. The findings demonstrate that decoder-only multilingual LLMs can serve as accurate detectors even when trained solely on task-provided data. Future work can explore larger models, deeper classification heads, and more advanced training regimes to push accuracy beyond current levels.

References

Piotr Przybyła, Jakub Strebeyko, and Alina Wróblewska. 2025. PolEval 2025 Task 1 Śmigiel: Spotting Machine-Generated Text from LLMs for Polish. In *Proceedings of the PolEval 2025 Workshop*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, and 1 others. 2018. Improving language understanding by generative pre-training.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.