AI-Generated Text Detection using ISGraphs and Graph Neural Networks

Andric Valdez-Valenzuela, Helena Gómez-Adorno
Institute for Research in Applied Mathematics and Systems
National Autonomous University of Mexico
Mexico City, Mexico
andric.valdez@gmail.com, helena.gomez@iimas.unam.mx

Manuel Montes-y-Gómez
Department of Computational Sciences
National Institute of Astrophysics, Optics and Electronics
Puebla, Mexico
mmontesg@inaoep.mx

Abstract

The growing presence of AI-generated text in online environments has raised concerns around misinformation, academic fraud, and content manipulation. To address this, we propose a graph-based detection system that combines Integrated Syntactic Graphs with Graph Neural Networks to distinguish between human and machine-generated text. Our approach leverages syntactic dependency structures and contextual embeddings from pre-trained language models, showing strong performance across multiple test scenarios, including clean, short, Unicode, and paraphrased variants. Our results demonstrate the robustness and adaptability of the text graph approach in different AI-generated text detection scenarios. This study was part of our PANCLEF 2025 Voight-Kampff AI Detection Sensitivity submission, which ranked 2nd of 27.

1 Introduction

The rapid advancement and accessibility of Large Language Models (LLMs) have significantly increased the presence of AI-generated text in online environments, including social media, education, and academia. Although these models offer practical benefits, they also raise critical concerns such as misinformation, academic fraud, and content manipulation. This growing impact highlights the urgent need for automated detection systems capable of distinguishing between human and machine authored content, especially in scenarios where manual verification is insufficient (Nitu and Dascalu, 2024).

In response to these concerns, the PAN lab at CLEF 2025 introduced the Generative AI detection task (Bevendorff et al., 2025), organized in collaboration with the Voight-Kampff Task at ELOQUENT Lab. The task is divided into two subtasks: Subtask 1 challenges participants to build binary classifiers that distinguish between human and machine authored texts, even when LLMs attempt to mimic specific human writing styles. To test robustness, the organizers include new elements in the test set, such as different LLMs or unknown obfuscation techniques. Subtask 2 focuses on the classification of texts written by humans and LLMs collaboratively, aiming to assign each document to one of six categories reflecting the type of human and AI involvement.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.

Current approaches for detecting AI-generated text can be categorized into metric-based and model-based methods; the latter includes feature-based, neural network-based, zero-shot, and watermark-based approaches (Huang et al., 2025). Neural detectors built on pretrained LMs (e.g., BERT, RoBERTa, DeBERTa) are effective at separating human-written from model-generated text (e.g., GPT-3). More recently, a model-based alternative represents text as graphs and applies Graph Neural Networks (GNNs), which capture complex relational structure (Battaglia et al., 2018), by operating on nodes (such as words or documents) and edges encoding lexical, syntactic, or semantic relations. GNNs model local and global dependencies, making them well-suited to authorship analysis and AI-generated text detection.

In this work, we present our methodology and results for Subtask 1 of the PANCLEF 2025 Voight-Kampff AI Detection Sensitivity challenge. We propose a graph-based detector that combines Integrated Syntactic Graphs (ISGraphs) with Graph Neural Networks. Our system was containerized as Docker images and evaluated via the TIRA platform (Fröbe et al., 2023). We submitted five variants exploring architectural and configuration changes. On the final leaderboard (scored on an unseen test set), our best submission ranked 2nd of 27 systems. All code is publicly available[1]

2 Background

AI-generated text detection is challenging due to the variety of generators, domains, and languages. To structure the problem, benchmarks focus on three tasks: (i) Human vs. Machine (binary classification), (ii) Multi-Way Generator Detection (identifying the specific source, e.g., GPT-4, Cohere), and (iii) Change-Point Detection (locating human-machine transitions in hybrid texts). These challenges are the focus of dedicated shared tasks at major international conferences like PANCLEF, Autextification (Sarvazyan et al., 2024), SemEval (Wang et al., 2024), and COLING (Wang et al., 2025), which drive innovation and benchmark progress in the field.

Several approaches for detecting AI-generated text have been proposed in recent years. For instance, Abburi et al. (2023) created an ensemble model that uses probabilities from pre-trained language models as features for a classic classifier, while Duran-Silva (2023) assessed text predictability using linguistic features and fine-tuned LLM representations; both approaches achieved the top performance in the Autextification 2023 task. Sarvazyan et al. (2024) built LLMIXTIC, which extracts token-level probabilistic features (log probability and entropy) from four LLaMA-2 models and feeds them into a Transformer Encoder for supervised training, achieving the highest ranking in the Human vs. Machine in the SemEval 2024 task. In the COLING 2025 GenAI shared task, Gritsai et al. (2024) proposed a multi-task model with a shared Transformer Encoder and multiple classification heads, distinguishing between human and machine-generated text while also classifying texts by domain.

In the PANCLEF 2025 edition, top systems proposed different approaches and data strategies. Macko (2025) fine-tuned Qwen3-14B via QLoRA for binary detection and improved generalization by obfuscating portions of the training data with homoglyph substitutions, then chose the final model using out-of-domain results on a 2,000-example, seven-language pool spanning 18 datasets. Liu et al. (2025) ensembled fine-tuned Qwen and ModernBERT under a contrastive-loss regime, augmenting coverage by LLM-paraphrasing human texts. Seeliger et al. (2025) employed a feature-engineering approach, constructing term–document matrices of cumulative binary correlation coefficients across uni/bi/tri-grams, offering a fine-tuned RoBERTa baseline, and analyzing both whole documents and temporal dynamics via cumulative per-word correlation trajectories.

Regarding the usage of Graph Neural Networks and text-graph representations is still limited but promising, motivating our exploration of this direction. GNNs have been widely applied to text classification (Wang et al., 2024). TextGCN (Yao et al., 2019) constructs a single corpus-level graph from word co-occurrence and document–word links and uses the resulting embeddings for classification; despite not using external word embeddings, it outperformed state-of-the-art methods and remained effective with limited training data. BertGCN (Lin et al., 2021) combines BERT with Graph Convolutional Networks, representing documents as nodes in a heterogeneous graph and using BERT embeddings as node features, achieving state-of-the-art results across multiple datasets. Text graphs have also been used for authorship analysis: Embarcadero-Ruiz et al. (2022) proposed a graph-based Siamese network for cross-topic, open-set verification, modeling texts with Part-of-Speech co-occurrence graphs and extracting structural features via GCNs; the architecture

[1]Code repository: https://anonymous.4open.science/r/GraphDeepLearning-FFB2/README.md

(graph convolutions, pooling, classification) explored three graph variants and incorporated stylistic features, yielding results on PANCLEF 2021 comparable to the state of the art.

Beyond these methods, the use of graph-based models is a promising but less explored option for AI-generated text detection. Building text graphs that show how words/sentences (or any token) connect could help find the subtle, overall patterns and mistakes that AI models often make. While early studies like TextGCN and BertGCN have shown that graph networks work well for general text classification, they have not been widely tested or specifically designed for detecting AI-generated text. This shows a need for more work to create and test new graph-based methods that can determine the authorship between humans and AI.

3 System Overview

In this section, we present the overall system architecture for AI-generated text detection. Section 3.1 describes our data preparation strategy, including cross-LLM partitioning and the creation of text set variants to improve model generalization. In Section 3.2, we describe the graph-based model architecture, which leverages the ISGraphs and GNNs to capture linguistic and structural patterns from the text.

3.1 Data Stratification

Table 1 shows the distribution of human and machine-generated texts across the training, validation, and test splits in the clean dataset version. The dataset is slightly imbalanced for machine-generated texts in all partitions, with an average document length of around 620 tokens. Texts are sampled from three genres: essays, fiction, and news.

In total, the dataset includes content from over 20 different LLMs, reflecting a diverse set of generation styles. The most frequently used models are GPT-4o-mini and GPT-4o. On the other end, less frequent models include mixtral-8x7b and llama-2-7b-chat. This wide range of LLM sources enriches the evaluation by introducing stylistic variety and generation complexity.

Partition	Class 0 (Human)	Class 1 (Machine)	Total	Avg. Doc Length
Train	3,460	5,368	8,828	620
Validation	1,077	1,652	2,729	608
Test	905	1,826	2,731	623

Table 1: Class distribution per dataset partition (clean version), including average document length in tokens.

To ensure robust evaluation and model generalization, we applied a data stratification strategy that partitions the corpus into three distinct sets: training, validation, and test, as detailed in the class distribution table. Importantly, we adopted a cross-LLM setup, meaning that each partition includes machine-generated texts from different LLMs. This strategy avoids the overlap of generation sources across partitions and encourages the model to generalize across unseen language models.

To simulate more realistic and complex scenarios, we generated multiple test and validation set variants:

- **Clean variant**: Text remains unchanged, preserving its original form.
- **Short variant**: Text is truncated to a maximum of 35 words to evaluate performance on limited context inputs.
- **Unicode variant**: To simulate obfuscation, 15 % of the characters are replaced with look-alike Unicode characters.
- **Paraphrased variant**: The text is rephrased using a google-t5/t5-base language model (Raffel et al., 2020) to test the classifier's resilience to semantic alterations.

3.2 Model Architecture

The pipeline architecture in Figure 1 shows the approach proposed for AI-generated text detection using the ISGraph and Graph Neural Networks. The system begins with raw text documents as input, which include both human-written and machine-generated content.

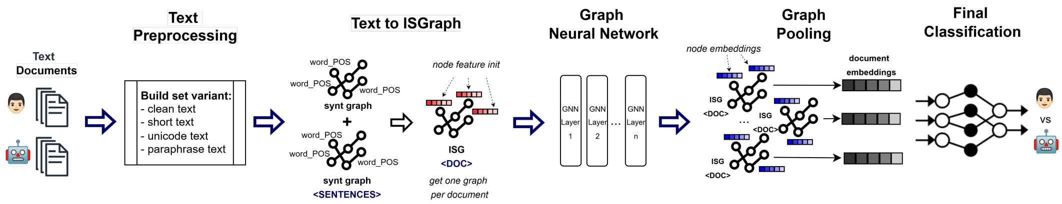

Figure 1: Pipeline architecture using ISGraphs and GNNs.

In the first phase, text preprocessing, the input text documents are divided into training, validation, and test partitions. As described in detail in the previous section, we adopt a cross-LLM strategy, where each partition includes texts generated by different language models. Additionally, we generate multiple set variants (with ofuscations) to simulate realistic scenarios and improve model generalization: the clean variant preserves the original text, the short variant truncates texts to 35 words, the unicode variant adds visually similar Unicode noise to 15% of the characters, and the paraphrased variant rephrases texts using a T5-base model.

A key component of our pipeline involves the building of the Integrated Syntactic Graph, proposed by Gómez-Adorno et al. (2016). This graph-based representation captures multilevel linguistic information, including lexical, morphological, and syntactic elements, within a unified structure. Each document is first divided into sentences, and a syntactic dependency tree is generated for each sentence using a parser tool. Each word (token) is represented as a node in the graph in the format token_POS, which includes the word and its part-of-speech tag. Edges between nodes represent grammatical dependencies (e.g., subject or object relations) and are weighted accordingly. These sentence-level graphs are then merged into a single document-level graph connected by a virtual root node (ROOT-0), resulting in a rich structural representation (see Figure 2).

After graph construction, we initialize node features to provide meaningful vector representations for each node. Several strategies exist for this step, including random initialization, static word embeddings like Word2Vec, and contextual embeddings from pre-trained language models. In this work, we adopt the latter approach, extracting token-level embeddings from models such as RoBERTa and DeBERTa. These contextualized embeddings help to capture rich semantic and syntactic patterns from the text, enhancing the model's ability to distinguish between human and machine text.

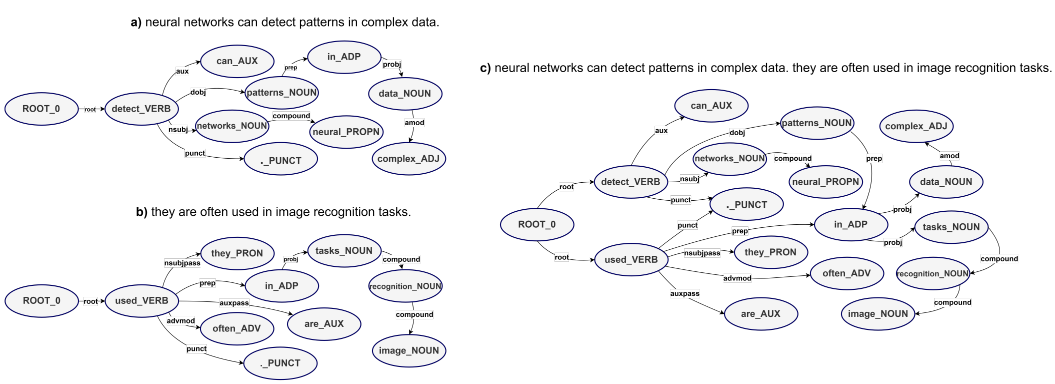

Figure 2: Integrated Syntactic Graph for the text document: *Neural networks can detect patterns in complex data. They are often used in image recognition tasks.*

Once the ISGraph is built, it is processed by a Graph Neural Network, which propagates and updates node representations through message passing. In this step, each node aggregates information from its neighbors, allowing the model to capture both local and global structural patterns. We experimented with two GNN architectures: Graph Convolutional Networks (GCNs) Yao et al. (2019), which apply convolutions over the graph, and Graph Attention Networks (GATs) Veličković et al. (2017), which use self-attention mechanisms to weigh neighboring nodes dynamically.

To obtain a fixed-size representation for each document graph, we apply mean graph pooling, where node embeddings are averaged to create a single graph-level embedding. This step summarizes the structural and semantic information of the entire document in a unified form suitable for downstream classification.

Finally, the resulting graph embeddings are passed through a dense neural network, which performs the binary classification task, predicting whether a document was written by a human or generated by a machine. The classification layer outputs a probability score between 0.0 and 1.0 for each document. If a text is predicted as human-written, we use the score as-is; if it's predicted as machine-generated, we compute the 1-score. If the score is exactly 0.5 (±0.01), it is considered undecidable and left unchanged.

4 Results

Table 2 summarizes the average performance across five PAN metrics (ROC-AUC, Brier, C@1, F_1, $F_{0.5u}$) [2] for the four test variants: clean, paraphrased, short, and unicode, using different versions of the ISGraph approach (V0–V4). These ISG variants were built using different configurations, including the type of graph neural network architecture (e.g. GCN, GAT), the choice of pre-trained language model for node feature initialization, such as DeBERTa or RoBERTa, as well as variations in the number of GNN layers, attention heads, learning rate, vocabulary size, preprocessing, and other hyperparameters.

Overall, the clean and Unicode variants consistently show the highest mean scores, indicating strong performance under standard and obfuscation conditions. In particular, ISG-V2 achieves the best overall results in both the clean (0.981) and Unicode (0.981) settings, along with strong performance under the short inputs (0.917).

In contrast, the paraphrased variant yields the lowest scores in all versions, with ISG-V4 showing the weakest result (0.533), suggesting that paraphrasing significantly challenges model robustness. Interestingly, ISG-V3 demonstrates the best generalization to paraphrased inputs (0.797), even though it slightly lags on the clean and Unicode cases. This comparison highlights how each ISG version handles different types of test obfuscations, with ISG-V2 offering the most balanced and highest mean performance overall.

Approach	**mean-text-clean**	**mean-text-paraph**	**mean-text-short**	**mean-text-unicode**
ISG-GNN-V0	0.978	0.625	0.890	0.969
ISG-GNN-V1	0.969	0.586	0.872	0.963
ISG-GNN-V2	0.981	0.741	0.917	0.981
ISG-GNN-V3	0.959	0.797	0.907	0.958
ISG-GNN-V4	0.972	0.533	0.815	0.963

Table 2: Average (ROC-AUC, Brier, C@1, F_1, $F_{0.5u}$) performance on test set variants for different versions of the ISG approach.

Table 3 shows the final PAN-CLEF leaderboard for the AI-Generated Text Detection subtask 1. Our proposed GNN-based approach, ISG-GNN-V3, achieved a strong second place with a mean score of 0.929. The winning team, Macko, secured first place with a mean score of 0.989. Our method demonstrated robust performance across all evaluation metrics, including ROC-AUC (0.939), $F1$ (0.926), and $F_{0.5u}$ (0.960), consistently positioning it at the top of the competitive field.

[2]**ROC-AUC**: The area under the Receiver Operating Characteristic curve; **Brier**: The complement of the Brier score (mean squared loss); **C@1**: A modified accuracy score that assigns non-answers (score = 0.5) the average accuracy of the remaining cases; $\mathbf{F}_1$: The harmonic mean of precision and recall; $\mathbf{F}_{0.5u}$: A modified $F_{0.5}$ measure (precision-weighted F measure) that treats non-answers (score = 0.5) as false negatives; **Mean**: The arithmetic mean of all previous measures

#	Team	ROC-AUC	Brier	C@1	F_1	$F_{0.5u}$	Mean
1	Macko (Macko, 2025)	0.995	0.984	0.982	0.989	0.993	**0.989**
2	**ISG-GNN-V3 (our)**	**0.939**	**0.902**	**0.897**	**0.926**	**0.960**	**0.929**
3	Liu (Liu et al., 2025)	0.962	0.891	0.889	0.923	0.963	**0.928**
4	Seeliger (Seeliger et al., 2025)	0.912	0.898	0.896	0.930	0.959	**0.925**
5	Voznyuk	0.899	0.898	0.898	0.929	0.962	**0.924**
10	Marchitan	0.945	0.890	0.869	0.905	0.952	**0.916**
27	Liang	0.734	0.694	0.694	0.752	0.827	**0.751**

Table 3: PAN-CLEF Final leaderboard for AI-Generated Text Detection subtask 1.

5 Conclusion

In this work, we presented a graph-based system for AI-generated text detection using Integrated Syntactic and Graphs and Graph Neural Networks. Our approach was evaluated in Subtask 1 of the PANCLEF 2025 Voight-Kampff challenge, where we submitted five system variants via the TIRA platform. Through a combination of linguistic graph modeling and contextualized node embeddings from pre-trained language models, our system demonstrated strong and stable performance across multiple test variants, including standard, short, Unicode, and paraphrased inputs. The results confirm the effectiveness and robustness of syntactic graph representations for this task and open promising directions for future work on interpretability and multilingual detection.

6 Limitations

Although we evaluate in a robust dataset, all text documents are in English and primarily focused on academic or instructional content. The applicability of our method to other languages, informal texts, or unseen LLMs remains unexplored. Regarding computational cost, the graph construction and model training are resource-intensive. To improve scalability, we used PyG's NeighborLoader for the heterogeneous graph and DataLoader for the co-occurrence graphs. While these strategies helped manage resources, further optimization remains an important direction for future work. Finally, our current analysis does not fully leverage interpretability techniques (e.g., attention score/heatmaps, GNNExplainer); future work could better explore how decisions are made across layers and how syntactic structures influence predictions.

A Experimental Setup

All experiments were executed on a machine with two NVIDIA RTX A5000 GPUs (24GB each), using CUDA 12.2, PyTorch Geometric 2.5, and Python 3.10. We used a 2-layer GCNs and GATs for model configuration with 2 attention heads, 128 hidden dimensions, ReLU activations, and LayerNorm. The models were trained using AdamW (learning rate 2e-5, weight decay 0.001, batch size 16, for 100 epochs with early-stopper optimization).

References

M. Nitu, M. Dascalu, Beyond lexical boundaries: Llm-generated text detection for romanian digital libraries, Future Internet 16 (2024). URL: https://www.mdpi.com/1999-5903/16/2/41. doi:10.3390/fi16020041.

J. Bevendorff, D. Dementieva, M. Fröbe, J. Karlgren, M. Mayerl, P. Nakov, A. Panchenko, M. Potthast, A. Shelmanov, E. Stamatatos, B. Stein, Y. Wang, M. Wiegmann, E. Zangerle, Overview of PAN 2025: Generative AI Authorship Verification, Multi-Author Writing Style Analysis, Multilingual Text Detoxification, and Generative Plagiarism Detection, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fourteenth International Conference of the CLEF Association (CLEF 2025), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2025.

B. Huang, C. Chen, K. Shu, Authorship attribution in the era of llms: Problems, methodologies, and challenges, 2025. URL: https://arxiv.org/abs/2408.08946. arXiv:2408.08946.

P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, et al., Relational inductive biases, deep learning, and graph networks, arXiv preprint arXiv:1806.01261 (2018).

M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous Integration for Reproducible Shared Tasks with TIRA.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. URL: https://link.springer.com/chapter/10.1007/978-3-031-28241-6_20. doi:10.1007/978-3-031-28241-6_20.

A. M. Sarvazyan, J. Á. González, F. Rangel, P. Rosso, M. Franco-Salvador, Overview of iberautextification at iberlef 2024: Detection and attribution of machine-generated text on languages of the iberian peninsula, Procesamiento del Lenguaje Natural 73 (2024).

Y. Wang, J. Mansurov, P. Ivanov, j. su, A. Shelmanov, A. Tsvigun, O. Mohammed Afzal, T. Mahmoud, G. Puccetti, T. Arnold, C. Whitehouse, A. F. Aji, N. Habash, I. Gurevych, P. Nakov, Semeval-2024 task 8: Multidomain, multimodel and multilingual machine-generated text detection, in: Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 2041–2063. URL: https://aclanthology.org/2024.semeval2024-1.275.

Y. Wang, A. Shelmanov, J. Mansurov, A. Tsvigun, V. Mikhailov, R. Xing, Z. Xie, J. Geng, G. Puccetti, E. Artemova, jinyan su, M. N. Ta, M. Abassy, K. A. Elozeiri, S. E. D. A. E. Etter, M. Goloburda, T. Mahmoud, R. V. Tomar, N. Laiyk, O. M. Afzal, R. Koike, M. Kaneko, A. F. Aji, N. Habash, I. Gurevych, P. Nakov, Genai content detection task 1: English and multilingual machine-generated text detection: Ai vs. human, 2025. URL: https://arxiv.org/abs/2501.11012. arXiv:2501.11012.

H. Abburi, M. Suesserman, N. Pudota, B. Veeramani, E. Bowen, S. Bhattacharya, Generative ai text classification using ensemble llm approaches, arXiv preprint arXiv:2309.07755 (2023).

N. Duran-Silva, I've seen things you machines wouldn't believe: Measuring content predictability to identify automatically-generated text., 2023.

A. M. Sarvazyan, J. Á. González, M. Franco-salvador, Genaios at SemEval-2024 task 8: Detecting machine-generated text by mixing language model probabilistic features, in: A. K. Ojha, A. S. Doğruöz, H. Tayyar Madabushi, G. Da San Martino, S. Rosenthal, A. Rosá (Eds.), Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 101–107. URL: https://aclanthology.org/2024.semeval-1.17/. doi:10.18653/v1/2024.semeval-1.17.

G. Gritsai, A. Voznyuk, I. Khabutdinov, A. Grabovoy, Advacheck at genai detection task 1: Ai detection powered by domain-aware multi-tasking, 2024. URL: https://arxiv.org/abs/2411.11736. arXiv:2411.11736.

D. Macko, mdok of kinit: Robustly fine-tuned llm for binary and multiclass ai-generated text detection, 2025. URL: https://arxiv.org/abs/2506.01702. arXiv:2506.01702.

J. Liu, L. Kong, Z. Peng, F. Chen, Generative AI authorship verification based on contrastive-enhanced dual-model decision system, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2025.

M. Seeliger, P. Styll, M. Staudinger, A. Hanbury, Human or not? light-weight and interpretable detection of ai-generated text, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum, CEUR-WS.org, 2025.

K. Wang, Y. Ding, S. C. Han, Graph neural networks for text classification: A survey, Artificial Intelligence Review 57 (2024) 190.

L. Yao, C. Mao, Y. Luo, Graph convolutional networks for text classification, in: Proceedings of the AAAI conference on artificial intelligence, volume 33, 2019, pp. 7370–7377.

Y. Lin, Y. Meng, X. Sun, Q. Han, K. Kuang, J. Li, F. Wu, Bertgcn: Transductive text classification by combining gcn and bert, arXiv preprint arXiv:2105.05727 (2021).

D. Embarcadero-Ruiz, H. Gómez-Adorno, A. Embarcadero-Ruiz, G. Sierra, Graph-based siamese network for authorship verification, Mathematics 10 (2022) 277.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

H. Gómez-Adorno, G. Sidorov, D. Pinto, D. Vilariño, A. Gelbukh, Automatic authorship detection using textual patterns extracted from integrated syntactic graphs, Sensors 16 (2016) 1374.

P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903 (2017).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the contribution of a graph-based system using ISGraphs and GNNs for AI-text detectio

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses the scope, language focus, computational cost, and lack of interpretability analysis.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper is empirical and does not present any theoretical results or proofs.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: In Appendix A we provide a brief description of the experimental setup used in the experiments

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: In Section 1 we provide the link for the code repository: https://anonymous.4open.science/r/GraphDeepLearning-FFB2/README.md

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Section 3.1 mention the data stratification and appendix A describe de experimental setup

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The results in Table 2 are presented as single scores without any measures of variance, such as error bars or confidence intervals.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix A shows the hardware use for the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research involves benchmarking existing models on a provided task for detection.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper focuses on technical methodology and results but does not discuss any potential positive or negative societal impacts of the AI detection technology.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper does not involve the release of a new model or dataset that poses a high risk of misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All existing models, datasets, and tools (e.g., RoBERTa, DeBERTa, T5) are properly cited in the References section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not introduce any new datasets, code, or models that are released as part of this work.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The research does not involve crowdsourcing or experiments with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The research does not involve human subjects, thus IRB approval is not applicable.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs were used as objects of study (to generate text for detection) and not as a component of the core detection methodology itself.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.