WhosAI: A Contrastive Learning Framework for Machine-Generated Text Detection and Attribution

(Discussion Paper)

Lucio La Cava[1], Andrea Tagarelli[1]

[1]Dept. Computer Engineering, Modeling, Electronics, and Systems Engineering (DIMES),
University of Calabria, 87036 Rende (CS), Italy

Abstract

The rapid advancement of Large Language Models (LLMs) has increasingly blurred the line between human and machine-generated text (MGT), raising new societal challenges. As MGT content becomes more widespread and harder to detect, robust identification methods are crucial. In this work, we present WhosAI, a triplet-network contrastive learning framework designed to detect and attribute AI-generated text. Unlike most existing methods, our framework simultaneously learns semantic similarity representations from multiple text generators, enabling it to effectively handle both detection (human vs. machine) and authorship attribution (identifying the specific generator). Furthermore, WhosAI is model-agnostic and scalable, seamlessly adapting to new LLMs for text-generation by integrating their outputs into its learned embedding space. Experimental results on the TuringBench dataset of 200K news articles demonstrate that our framework excels in both the Turing Test and Authorship Attribution tasks, outperforming all existing methods on the TuringBench leaderboard.

Keywords

machine-generated text, AI detection, AI attribution, contrastive learning

1. Introduction

Recent breakthroughs in artificial intelligence (AI) have significantly advanced natural language processing (NLP), leading to powerful text generation models capable of producing highly fluent and human-like content [1]. As text generation models become increasingly realistic, distinguishing between machine-generated text (MGT) and human writing becomes a pressing challenge [2]. Without effective methods for identifying MGT, the preservation of truth, authenticity, and trustworthiness in online communication is at risk, posing several challenges to our society. Moreover, the growing presence of MGT in digital channels favors the risk of misinformation, manipulation, and deception [3]. Therefore, robust detection mechanisms are essential to prevent users from unknowingly consuming or spreading false or misleading information, compromising the integrity of public discourse and decision-making processes. Furthermore, as MGT closely mimics human language and creativity [4], ethical and societal concerns arise about authorship, intellectual property rights, and accountability, challenging established norms in intellectual property law and digital content creation in the absence of proper mechanisms for identifying the origin of text.

SEBD 2025: The 33rd Italian Symposium on Advanced Database Systems, June 16-19, 2025, Ischia, Italy

lucio.lacava@dimes.unical.it (L. La Cava); tagarelli@dimes.unical.it (A. Tagarelli)
0000-0003-3324-0580 (L. La Cava); 0000-0002-8142-503X (A. Tagarelli)

Related work. The aforementioned challenges determined growing interest in detecting whether and to what extent texts have been generated by humans or machines [5, 6]. One approach dubbed "watermarking" [7, 8] involves embedding specific signals into generated texts that remain invisible to humans but are algorithmically detectable. Statistical learning approaches for detecting the authorship of texts include probabilistic models [9, 10], log rank information [11], perplexity [12], discourse motifs [13], and other statistical approaches [14, 15, 16]. More recently, deep learning approaches have been proven promising in detecting or attributing MGT. These include exploiting LLMs to detect generated text [17, 18], using ChatGPT itself as a detector [19], or combining LLMs with topological aspects [20]. Similarly, contrastive representation learning has been proven particularly effective in NLP contexts, such as text classification [21, 22], hate-speech detection [23], unveiling intents [24], and MGT detection [25, 26]. Despite progress in MGT detection research, each of the above mentioned approaches faces notable challenges. Watermarking methods depend on the possibility to embed watermarks in text, statistical learning methods often require access to model internals or information that might be unavailable, and existing contrastive-learning-based methods typically perform best when trained separately for each generator.

Contributions. In this paper, we discuss WhosAI, a recently proposed contrastive-learning framework conceived to address binary/multi-class prediction tasks of texts written by humans or machine text-generation models [27]. The core idea behind WhosAI is to integrate the power of Transformer-based pretrained language models (PLMs) into a similarity learning framework optimizing a contrastive triplet loss function to learn deep semantic subspaces that maximize the cohesiveness among similar texts and the separation between dissimilar texts. Compared to existing MGT detection methods, WhosAI offers the following key advantages. First, unlike watermarking methods, WhosAI does not require editing texts, accessing models' internals, or assume specific linguistic features, being able to *deal with any type of texts.* Second, WhosAI is conceived to be *versatile* w.r.t. the particular PLM used at the core of the learning framework, and is *general-purpose*, eliminating the need for separate models tailored to specific tasks or generators.Finally, our contrastive learning framework makes WhosAI *model-agnostic* and *scalable* to the release of new AI text-generators—simply incorporating new data into training enables generalization to new models. The effectiveness of WhosAI has been demonstrated through a comprehensive evaluation on the widely recognized *TuringBench* benchmark dataset, which includes 200K articles that are either human-written or machine-generated by 19 different AI text-generation models. WhosAI achieves excellent results in terms of both classification performance and internal validity criteria, outperforming all the methods appearing in the benchmark's leaderboard, for both the *Turing Test* and *Authorship Attribution* tasks.

In the remainder of this paper, we summarize and discuss the main findings drawn from the development and evaluation of WhosAI. For comprehensive details on the design of WhosAI and in-depth discussion of results, the interested reader is referred to [27].

2. Problem Statement

We are given a set of discrete labels (categories) $\mathcal{C} = \{c_j\}_{j=1}^{M}$, with $M \geq 2$, and a collection of text data objects $\mathcal{D} = \{D_i\}_{i=1}^{N}$, such that each text object in $\mathcal{D}$ is assigned to one of the

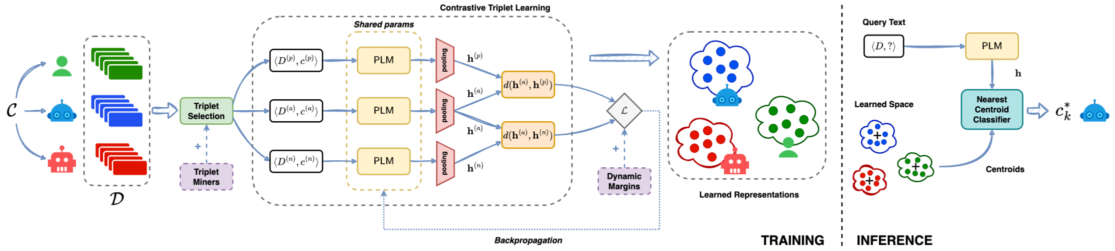

Figure 1: Overview of the WhosAI framework, at training time (left) and inference time (right) [27].

categories in $\mathcal{C}$, indicating whether it was authored by a human or a machine. The authorship of all texts in $\mathcal{D}$ are a priori known.

Our goal is to learn a model, supervisedly trained on $\langle \mathcal{D}, \mathcal{C} \rangle$, that can predict the category from $\mathcal{C}$ for a given text whose authorship is unknown. Specifically, we tackle the following problems: (i) **Turing Test** (TT), a binary classification task to determine whether the author of a text is a human or an AI text-generator, and (ii) **Authorship Attribution** (AA), a multi-class classification task, that identifies the specific author of a text, choosing between a human or an AI text-generator. Following literature, our setting does not differentiate between human authors in either task, with 'human' always corresponding to one class in $\mathcal{C}$. Also, the identity of a particular machine text-generator must be unveiled for the Authorship Attribution task only, therefore $M - 1$ categories are available that correspond to either *any* AI text-generator (for TT) or *a specific* AI text-generator (for AA).

3. The WhosAI Framework

Overview. WhosAI is a deep learning framework for detecting and attributing open-ended texts, distinguishing between AI-generated and human-written content.

Figure 1 illustrates the main components and data flow of the framework. It is trained on labeled text data, where authorship is categorized as either human or AI-generated, and consists of three key components: (i) a pretrained language model (PLM) that learns deeply contextualized text representations in an unsupervised manner, (ii) a Triplet Network that applies contrastive learning to structure a similarity space for PLM embeddings, and (iii) a nearest centroid classifier that predicts the authorship of query texts.

During training, WhosAI learns a deep semantic representation space, where distinct regions capture the characteristics of human-written and AI-generated texts. Contrastive learning allows for capturing the underlying data-similarity structure, by grouping embeddings from the same author while separating those from different authors. This learned similarity space hence facilitates classification by defining clear decision boundaries. In this setting, the nearest centroid classifier provides an efficient and effective approach to authorship attribution, ensuring robust predictions for previously unseen texts.

WhosAI is designed to be versatile and modular. Its versatility lies in the flexibility to select different PLMs as the core of the Triplet Network, experiment with various Triplet Network architectures, and use alternative instance-based classification models. Additionally, WhosAI follows a modular design, allowing for targeted enhancements to improve specific aspects of the framework. These enhancements include: (i) optimizing the contrastive learning component for better efficiency and generalization, (ii) refining class separation in the learned representation space to better distinguish different text creators, and (iii) increasing robustness by introducing perturbations to the input textual data [27].

3.1. Training

Transformer-based Pre-trained Language Models (PLMs) are the well-established NLP tools to build deeply contextualized text-representation learning models. Given a text data $D_i \in \mathcal{D}$, a token sequence $T_i = [\tau_{i,1}, \ldots, \tau_{i,|T_i|}]$ is produced as initial representation of D_i through a *tokenization* process typically associated with a PLM. Each token sequence is deeply contextualized by mapping it onto a dense, relatively low dimensional space of size f, based on the PLM. The resulting output is the *token embeddings* of D_i, denoted as $\mathsf{PLM}(T_i) \in \mathbb{R}^{f \times |T_i|}$. Eventually, a pooling function *pooling*$(\cdot)$ is applied to the token embeddings of each object D_i to yield a single embedding vector $\mathbf{h}_i$ of size f:

$$\mathbf{h}_i = \textit{pooling}(\mathsf{PLM}(T_i)) \in \mathbb{R}^f. \tag{1}$$

Typically, this pooled output is an average embedding over all token embeddings of a data object. The embeddings $\mathbf{h}_i$ are commonly referred to as *sentence embeddings*.

Similarity Learning. The deeply contextualized representations produced by a PLM lend themselves particularly suited to enable semantic comparisons between the input text objects. In this respect, we leverage the similarity space induced from the sentence embeddings. *Similarity learning* aims to train a model to distinguish between similar and dissimilar pairs of objects. More specifically, if we consider objects whose relative similarity follows a predefined orde—i.e., for any triplet of objects, the first object is assumed to be more similar to the second object than to the third object–the goal becomes to learn a *contrastive loss* function, so that it favors small distances between pairs of objects labeled as similar, and large distances for pairs labeled as dissimilar. This is certainly our case since it is expected that a human-written text to be similar to another human-written text than an AI-generated text, or texts generated by the same AI model to be similar to each other than to texts generated from other AI models.

Contrastive learning is often performed by using a *Siamese Network* architecture [28], which contains two PLM instances sharing the same weights while being trained in parallel on two input objects to compute comparable outputs. When using a contrastive triplet loss, Siamese Network is commonly referred to as *Triplet Network*.

Training process. Our training process starts with mining *triplets* $\langle D^{(a)}, D^{(p)}, D^{(n)} \rangle$ of text data objects from $\mathcal{D}$ to be fed into our triplet network. Such triplets are formed in such a way that, for a given *anchor* $D^{(a)}$, $D^{(p)}$ and $D^{(n)}$ are selected as *positive* and *negative* sample, respectively, i.e., such that $c^{(a)} = c^{(p)}$ and $c^{(a)} \neq c^{(n)}$, where symbols $c^{(\cdot)}$ are here used to denote the category associated with an anchor, positive or negative object.

The embeddings $\mathbf{h}^{(a)}, \mathbf{h}^{(p)}, \mathbf{h}^{(n)}$ of the anchor, positive and negative objects, respectively, are next computed according to Eq. 1. Note that the text annotations, i.e., associated categories, are not required when computing the embeddings, since the PLM is an unsupervised learner.

Given a triplet, the Triplet Network computes the distance between the embedding of the anchor object and the embedding of the positive object (*positive pair*), and the distance between the embedding of the anchor object and the embedding of the negative object (*negative pair*). The *triplet loss* minimizes the distance between an anchor and a positive, both having the same category, and maximizes the distance between the anchor and a negative of a different category:

$$\mathcal{L} = \sum_{\langle D^{(a)}, D^{(p)}, D^{(n)} \rangle} \max(d(\mathbf{h}^{(a)}, \mathbf{h}^{(p)}) - d(\mathbf{h}^{(a)}, \mathbf{h}^{(n)}) + \lambda, 0) \tag{2}$$

where $d(\cdot, \cdot)$ is a distance function and $\lambda \in \mathbb{R}^+$ is a margin between positive and negative pairs. This loss defines the *triplet constraint* as the requirement that the distance of negative pairs should be larger than the distance of positive pairs.

3.2. Inference

At inference time, WhosAI exploits an off-line step that consists in precomputing the *centroids* in $\mathcal{D}$ for each category $c_k \in \mathcal{C}$, defined as $\mathbf{c}_k = (1/|\mathcal{D}_k|) \sum_{D_i \in \mathcal{D}_k} \mathbf{h}_i$, where $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing data objects of category c_k.

Given a previously unseen data object D, WhosAI computes its embedding $\mathbf{h}$ (Eq. 1), which is then compared to each of the centroids in such a way that D is assigned to the category c_{k^*} that corresponds to the least distant centroid:

$$k^* = \arg\min_{k=1..M} d(\mathbf{h}, \mathbf{c}_k). \tag{3}$$

3.3. Optimizations

We discuss here a set optimization techniques as enhancements of key components in WhosAI, namely improved triplet mining, dynamic margin scheduling, and data corruption.

Improving Triplet Mining. A straightforward implementation of the triplet mining process involves gathering triplets before each training epoch and feeding batches of these triplets into the Triplet Network, essentially as an "offline" process. However, this approach might have two main drawbacks: (i) not all generated triplets may contain the valuable information needed for minimizing the loss (Eq. 2), and (ii) triplets regarded as "informative" in an earlier stage of training might quickly become "uninformative" as the model's weights undergo updates.

Within this view, it becomes crucial for the triplet mining process to prioritize an *online* identification of the most informative triplets for each training epoch. These should be the most unexpected ones, i.e., triplets that most violate the margin constraints enforced by the loss function. This strategy can improve the mining process as it enhances the generalization capabilities and training stability, and it makes training more efficient by avoiding the inclusion of the uninformative triplets.

The above requirements can effectively be fulfilled by the pair mining scheme adopted in the *multi-similarity miner* method [29]. Essentially, the pair mining consists in sampling informative pairs through the relative similarity between the negative and positive pairs sharing a common anchor. More specifically, a negative pair is selected as one having lower distance than the hardest positive pair (i.e., the one with the highest distance):

$$d(\mathbf{h}^{(a)}, \mathbf{h}^{(n)}) < \max_{D^{(p)}} d(\mathbf{h}^{(a)}, \mathbf{h}^{(p)}) + \varepsilon. \tag{4}$$

A positive pair is selected as one having higher distance than the hardest negative pair (i.e., the one with the lowest distance):

$$d(\mathbf{h}^{(a)}, \mathbf{h}^{(p)}) > \min_{D^{(n)}} d(\mathbf{h}^{(a)}, \mathbf{h}^{(n)}) - \varepsilon. \tag{5}$$

Dynamic Margin Scheduling. Another improvement we consider is to make the training of WhosAI progressively harder. Specifically, by *dynamically increasing* the margin λ in our loss function (Eq. 2), we require the model to focus on harder negative pairs as the training goes on, in order to produce an enhanced separation between classes.

To this aim, we revise the loss function with a dynamic margin that follows a linear schedule dependent on the training step time $t \geq 0$, which is defined as $\lambda^{(t)} = \lambda_{\text{min}} + \lambda_{\Delta}(t \bmod \delta)$, where $\lambda_{\text{min}} \in \mathbb{R}^+$ is the initial margin, $\lambda_{\Delta} \in \mathbb{R}^+$ denotes the margin increment, and δ represents the step size of the increment. We begin with an initial, relatively low margin, λ_{min}, to facilitate manageable gradients during early optimization; in fact, at early stage, a model can exhibit some discriminative ability, however, large margins during this stage would lead to excessively large gradients, hindering learning. As the optimization progresses and the distance constraints are enforced, the importance of loss-based gradients gradually diminishes. To prevent stagnation, the margin is periodically increased by λ_{Δ} every δ training steps. Based on the definition of $\lambda^{(t)}$, WhosAI integrates a dynamic margin scheduling with the triplet loss:

$$\mathcal{L}^{(t)} = \sum_{\langle D^{(a)}, D^{(p)}, D^{(n)} \rangle} \max(d(\mathbf{h}^{(a)}, \mathbf{h}^{(p)}) - d(\mathbf{h}^{(a)}, \mathbf{h}^{(n)}) + \lambda^{(t)}, 0). \tag{6}$$

4. Experimental Methodology

Data. We used the publicly available benchmark *TuringBench* [30, 31], containing 200K news articles, where 10K are human-written and the others are machine-generated news articles equally distributed over 19 AI text-generation models. These correspond to different sizes and implementations of GPT-1 [32], GPT-2 [33], GPT-3 [34], GROVER [35], CTRL [36], XLM [37], XLNET [38], FAIR [39], TRANSFORMER [40], and PPLM [41]. To foster reproducibility, we followed the pre-defined train, validation and test set splits provided by TuringBench.

Assessment Criteria and Model Settings. To validate the performance of WhosAI in detecting and attributing AI-generated text, we resort to standard statistics based on the confusion matrices derived from testing WhosAI predictions w.r.t. the ground-truth under the Turing Test task and w.r.t. the ground-truth under the Author Attribution task, respectively. These include the weighted average (i.e., averaging over the support-weighted mean per class) of *precision* (P), *recall* (R), and F_1*-score* (F_1). We also calculate distance-based quantitative criteria to measure how well the learned space aligns with the predefined categorization of the training texts, in terms of *compactness* within same-category groups of objects and *separation* between groups of objects of different categories. Following the most widely used approach to sentence

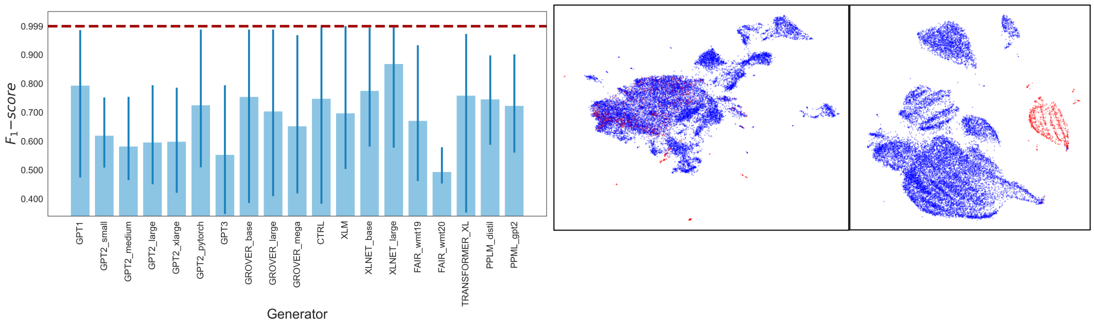

Figure 2: *Turing Test evaluation.* (On the left) Barchart of the average F_1 score from the TuringBench Leaderboard, for each TuringBench subset (i.e., generator). The horizontal red dashed line corresponds to the F_1 score achieved by WhosAI (best-performing setting) over the entire TuringBench test set. (On the right) cosine-distance-based 2D UMAP visualization [44] of the semantic space produced by WhosAI *before* (left) and *after* training (right). Colors denote human (red) vs. AI-generated (blue) texts [27].

embedding [42], we used BERT [43] as our reference PLM;[1] despite being a baseline model, we will show that this choice is sufficient to demonstrate the strong performance achieved by WhosAI without relying on more complex architectures.

5. Results

Here we summarize our main results on the Turing Test (TT) and Authorship Attribution (AA) tasks, respectively, achieved by WhosAI (best-performing setting) and competitors. For additional details, the interested reader is referred to [27].

Turing Test. As shown in Figure 2 (left), according to the TuringBench leaderboard[2] there is a substantial disparity in the F_1 scores for TT, which implies that texts from some generators are more easily detectable than others. By contrast, WhosAI is able to learn a deep semantic space for the whole set of generators at once achieving an impressive F_1 score of 0.999 on the whole TT test set supplied by the TuringBench benchmark, setting a new best performance on the TT. Our remarkable F_1 score is further corroborated by a qualitative analysis based on the visualization provided in Figure 2 (right): while at the beginning of the training the semantic representation directly induced by the PLM does not adequate separate the human and AI subspaces, the final trained WhosAI shows its ability to learn perfectly to recognize the two classes for the TT. This couples with the remarkable results (not shown) in terms of average within-category compactness (0.931) and average across-category separation (-0,808).

Authorship Attribution. Our first finding derives from a comparison between WhosAI results against those reported on the TuringBench leaderboard for the AA task, whose top-5 best-performing models are shown in Figure 3 (left). RoBERTa [45] with a multi-class classification setting turns out to be the best model in the leaderboard for the AA task, with a F_1 score of 0.811, followed by other BERT-based approaches, as well as the official OpenAI detector and machine

[1] https://huggingface.co/google-bert/bert-base-uncased
[2] Available at https://turingbench.ist.psu.edu/

Detection method	P	R	F_1
WhosAI	**0.990**	**0.990**	**0.990**
RoBERTa	0.821	0.813	0.811
BERT	0.803	0.802	0.800
BERTAA	0.780	0.775	0.776
OpenAI detector	0.781	0.781	0.774
SVM (3-grams)	0.712	0.722	0.715

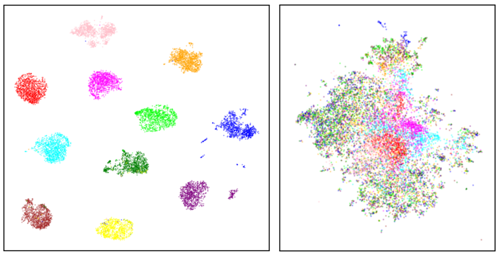

Figure 3: *Authorship Attribution evaluation.* (On the left) Results achieved by WhosAI vs. the top-5 models from the TuringBench Leaderboard. (On the right) 2D UMAP visualization of the semantic space produced by WhosAI (left) and SBERT (right). Colors denote human (blue) and the various AI text generators [27].

learning-based models. The winner method from the leaderboard is however outperformed by WhosAI, which achieves a striking average weighted F_1 score, precision and recall of 0.990, thus demonstrating almost perfect capabilities of authorship prediction. As previously found for the TT task, the striking F_1 scores achieved by WhosAI couple with an evidence of highest cohesiveness (0.938) and separation (-0.012) of the subspaces associated with the various text authorships, as shown in Fig. 3 (left).

It is also worth noting that the outstanding performance by WhosAI in the AA task is not paired by a state-of-the-art sentence-embedding method for semantic-similarity-related tasks like SBERT [42], based on a Siamese network using BERT at its core: indeed, as shown in Fig. 3 (right), the intra-class cohesiveness and inter-class separation of the semantic space learned by SBERT are clearly worse than those achieved by WhosAI.

6. Conclusions and Future Work

We tackled the challenge of detecting and attributing AI-generated text through WhosAI, a novel PLM-based framework that leverages contrastive learning to induce a semantic similarity space texts written by humans or AI text-generation models. This similarity space is efficiently exploited at inference time by means of a nearest centroid classifier to predict the authorship of unlabeled texts. Extensive experimentation on the well-known *TuringBench* dataset has revealed state-of-the-art performances of WhosAI on both TT and AA tasks.

Our ongoing work includes OpenTuringBench [46], an Open-model-based benchmark and framework for machine-generated text detection and attribution. Moreover, future work involves extending WhosAI to other text domains, comparing with advanced yet commercially licensed AI detection tools (e.g., GPTZero), improving training efficiency [47], and investigating explainability aspects of WhosAI.

Acknowledgements. AT, resp. LLC, was supported by project “Future Artificial Intelligence Research (FAIR)” spoke 9 (H23C22000860006), resp. project SERICS (PE00000014), both under the MUR National Recovery and Resilience Plan funded by the EU - NextGenerationEU.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] M. Jakesch, J. T. Hancock, M. Naaman, Human heuristics for ai-generated language are flawed, Proceedings of the National Academy of Sciences 120 (2023) e2208839120.

[2] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi, Can ai-generated text be reliably detected?, arXiv:2303.11156 (2023).

[3] C. Chen, K. Shu, Can LLM-Generated Misinformation Be Detected?, arXiv:2309.13788 (2024).

[4] L. L. Cava, A. Tagarelli, Open Models, Closed Minds? On Agents Capabilities in Mimicking Human Personalities through Open Large Language Models, in: Proc. AAAI 2025, AAAI Press, 2025, pp. 1355–1363. URL: https://doi.org/10.1609/aaai.v39i2.32125.

[5] G. Jawahar, M. Abdul-Mageed, L. V. Lakshmanan, Automatic detection of machine generated text: A critical survey, arXiv:2011.01314 (2020).

[6] R. Tang, Y.-N. Chuang, X. Hu, The science of detecting llm-generated text, Communications of the ACM 67 (2024) 50–59.

[7] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, T. Goldstein, A watermark for large language models, in: Proc. ICML Conf., 2023, pp. 17061–17084.

[8] K. Yoo, W. Ahn, J. Jang, N. Kwak, Robust multi-bit natural language watermarking through invariant features, in: Proc. ACL Conf., 2023, pp. 2092–2115.

[9] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-generated text detection using probability curvature, in: Proc. ICML Conf., PMLR, 2023, pp. 24950–24962.

[10] P. Wang, L. Li, K. Ren, B. Jiang, D. Zhang, X. Qiu, Seqxgpt: Sentence-level ai-generated text detection, arXiv:2310.08903 (2023).

[11] J. Su, T. Y. Zhuo, D. Wang, P. Nakov, Detectllm: Leveraging log rank information for zero-shot detection of machine-generated text, arXiv:2306.05540 (2023).

[12] C. Vasilatos, M. Alam, T. Rahwan, Y. Zaki, M. Maniatakos, Howkgpt: Investigating the detection of chatgpt-generated university student homework through context-aware perplexity analysis, arXiv:2305.18226 (2023).

[13] Z. M. Kim, K. H. Lee, P. Zhu, V. Raheja, D. Kang, Threads of subtlety: Detecting machine-generated texts through discourse motifs, arXiv:2402.10586 (2024).

[14] S. Gehrmann, H. Strobelt, A. Rush, GLTR: Statistical detection and visualization of generated text, in: Proc. ACL Conf.: System Demonstrations, 2019, pp. 111–116.

[15] E. Tulchinskii, K. Kuznetsov, K. Laida, D. Cherniavskii, S. Nikolenko, E. Burnaev, S. Barannikov, I. Piontkovskaya, Intrinsic dimension estimation for robust detection of AI-generated texts, in: Proc. NIPS Conf., 2023.

[16] S. Venkatraman, A. Uchendu, D. Lee, Gpt-who: An information density-based machine-generated text detector, arXiv preprint arXiv:2310.06202 (2023).

[17] D. Ippolito, D. Duckworth, C. Callison-Burch, D. Eck, Automatic detection of generated text is easiest when humans are fooled, arXiv:1911.00650 (2019).

[18] V. Verma, E. Fleisig, N. Tomlin, D. Klein, Ghostbuster: Detecting text ghostwritten by large language models, arXiv:2305.15047 (2023).

[19] A. Bhattacharjee, H. Liu, Fighting Fire with Fire: Can ChatGPT Detect AI-generated Text?, SIGKDD Explor. Newsl. 25 (2024) 14–21.

[20] A. Uchendu, T. Le, D. Lee, Toproberta: Topology-aware authorship attribution of deepfake texts, arXiv preprint arXiv:2309.12934 (2023).

[21] L. Pan, C.-W. Hang, A. Sil, S. Potdar, Improved text classification via contrastive adversarial training, in: Proc. AAAI Conf., volume 36, 2022, pp. 11130–11138.

[22] J. Chen, R. Zhang, Y. Mao, J. Xu, Contrastnet: A contrastive learning framework for few-shot text classification, in: Proc. AAAI-IAAI-EAAI Conf., 2022, pp. 10492–10500.

[23] Y. Kim, S. Park, Y.-S. Han, Generalizable implicit hate speech detection using contrastive learning, in: Proc. COLING Conf., 2022, pp. 6667–6679.

[24] J. Zhang, T. Bui, S. Yoon, X. Chen, Z. Liu, C. Xia, Q. H. Tran, W. Chang, P. Yu, Few-shot intent detection via contrastive pre-training and fine-tuning, arXiv:2109.06349 (2021).

[25] A. Bhattacharjee, T. Kumarage, R. Moraffah, H. Liu, ConDA: Contrastive domain adaptation for AI-generated text detection, in: Proc. IJCNLP Conf., 2023, pp. 598–610.

[26] A. Bhattacharjee, R. Moraffah, J. Garland, H. Liu, Eagle: A domain generalization framework for ai-generated text detection, arXiv:2403.15690 (2024).

[27] L. L. Cava, D. Costa, A. Tagarelli, Is contrasting all you need? contrastive learning for the detection and attribution of ai-generated text, in: ECAI 2024 - 27th European Conference on Artificial Intelligence, 19-24 October 2024, Santiago de Compostela, Spain, volume 392 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, 2024, pp. 3179–3186. doi:10.3233/FAIA240862.

[28] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, R. Shah, Signature verification using a siamese time delay neural network, in: Proc. NIPS Conf., 1993, pp. 737–744.

[29] X. Wang, X. Han, W. Huang, D. Dong, M. R. Scott, Multi-similarity loss with general pair weighting for deep metric learning, in: Proc. CVPR Conf., 2019, pp. 5022–5030.

[30] A. Uchendu, Z. Ma, T. Le, R. Zhang, D. Lee, TURINGBENCH: A benchmark environment for Turing test in the age of neural text generation, in: Findings of the EMNLP Conf., 2021, pp. 2001–2016.

[31] A. Uchendu, T. Le, K. Shu, D. Lee, Authorship attribution for neural text generation, in: Proc. EMNLP Conf., 2020, pp. 8384–8395.

[32] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al., Improving language understanding by generative pre-training (2018).

[33] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (2019) 9.

[34] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, Proc. NIPS Conf. 33 (2020) 1877–1901.

[35] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, Y. Choi, Defending against neural fake news, in: Proc. NIPS Conf., 2019.

[36] N. S. Keskar, B. McCann, L. R. Varshney, C. Xiong, R. Socher, Ctrl: A conditional transformer

language model for controllable generation, arXiv:1909.05858 (2019).

[37] G. Lample, A. Conneau, Cross-lingual language model pretraining, arXiv:1901.07291 (2019).

[38] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, Proc. NIPS Conf. 32 (2019).

[39] N. Ng, K. Yee, A. Baevski, M. Ott, M. Auli, S. Edunov, Facebook fair's wmt19 news translation task submission, arXiv:1907.06616 (2019).

[40] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, R. Salakhutdinov, Transformer-XL: Attentive language models beyond a fixed-length context, in: Proc. ACL Conf., 2019, pp. 2978–2988.

[41] S. Dathathri, A. Madotto, J. Lan, J. Hung, E. Frank, P. Molino, J. Yosinski, R. Liu, Plug and play language models: A simple approach to controlled text generation, arXiv:1912.02164 (2019).

[42] N. Reimers, I. Gurevych, Sentence-BERT: Sentence embeddings using Siamese BERT-networks, in: Proc. EMNLP-IJCNLP Conf., 2019, pp. 3982–3992.

[43] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: Proc. NAACL-HLT Conf., 2019, pp. 4171–4186.

[44] L. McInnes, J. Healy, J. Melville, Umap: Uniform manifold approximation and projection for dimension reduction, 2020. arXiv:1802.03426.

[45] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: A robustly optimized bert pretraining approach, 2019. arXiv:1907.11692.

[46] L. L. Cava, A. Tagarelli, OpenTuringBench: An Open-Model-based Benchmark and Framework for Machine-Generated Text Detection and Attribution, arXiv:2504.11369 (2025).

[47] F. Scala, S. Flesca, L. Pontieri, Play it straight: An intelligent data pruning technique for green-ai, in: Discovery Science, Springer Nature Switzerland, Cham, 2025, pp. 69–85.