ConDA: Contrastive Domain Adaptation for AI-generated Text Detection

Amrita Bhattacharjee Tharindu Kumarage Raha Moraffah Huan Liu
School of Computing and AI
Arizona State University
{abhatt43, kskumara, rmoraffa, huanliu}@asu.edu

Abstract

Large language models (LLMs) are increasingly being used for generating text in a variety of use cases, including journalistic news articles. Given the potential malicious nature in which these LLMs can be used to generate disinformation at scale, it is important to build effective detectors for such AI-generated text. Given the surge in development of new LLMs, acquiring labeled training data for supervised detectors is a bottleneck. However, there might be plenty of unlabeled text data available, without information on which generator it came from. In this work we tackle this data problem, in detecting AI-generated news text, and frame the problem as an unsupervised domain adaptation task. Here the domains are the different text generators, i.e. LLMs, and we assume we have access to only the labeled source data and unlabeled target data. We develop a **Con**trastive **D**omain **A**daptation framework, called **ConDA**, that blends standard domain adaptation techniques with the representation power of contrastive learning to learn domain invariant representations that are effective for the final unsupervised detection task. Our experiments demonstrate the effectiveness of our framework, resulting in average performance gains of 31.7% from the best performing baselines, and within 0.8% margin of a fully supervised detector. All our code and data is available here.

1 Introduction

In recent years there have been significant improvements in the area of large language models that are capable of generating human-like text. Several variants of such language models are designed for specific tasks such as summarization, translation, paraphrasing, etc. Recent advancements in conversational language models such as ChatGPT and GPT-4 (OpenAI, 2023) have demonstrated how these language models can generate incredibly human-like text, along with serving as an AI assistant for several use cases such as creative writing, explanation of ideas and concepts, code generation and correction, solving mathematical proofs etc. (Bubeck et al., 2023). However, along with improved progress in machine generation of text, there is also a growing concern about how these technologies may be misused and abused by malicious actors. Given how convincing some of these machine-generated texts are, malicious actors may use these models to propagate misinformation/disinformation (Zellers et al., 2019), propaganda (Varol et al., 2017), or even spam/scams. With the accessibility and ease of use of newer language models that have public-facing APIs, the risk of these technologies being used for generating disinformation or misleading information at scale has increased significantly (De Angelis et al., 2023) and hence has prompted researchers to worry about detection and mitigation strategies (Zhou et al., 2023). For example, recently, there have been concerns about misleading news websites hosting fully AI-generated news articles[1]. Such unprecedented improvement in language generation capabilities

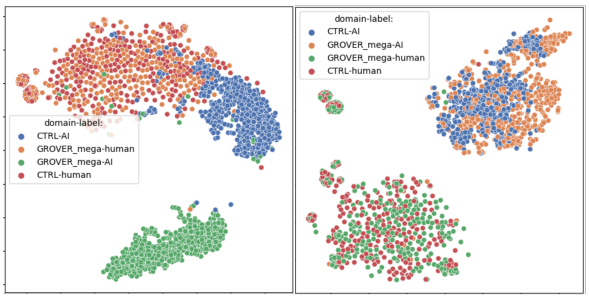


Figure 1: Text embeddings from (*left*) source-only model and (*right*) **ConDA** model on target domain CTRL with GROVER_mega as source. Each domain has both 'human' and 'AI' text. **ConDA** effectively removes domain-specific features while retaining task-specific features, increasing the separability between 'human' and 'AI' text, and decreasing the separability between source and target domains.

[1] https://www.newsguardtech.com/special-reports/newsbots-ai-generated-news-websites-proliferating/

Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 598–610
November 1–4, 2023.

hence naturally necessitates the development of detectors that can accurately and reliably classify such generated text. Motivated by this, we focus on the sub-problem of AI-generated news detection.

A major issue surrounding building a supervised classifier for AI-generated text is the sheer variety of large language models that are available for use. Prior work (Jawahar et al., 2020) has demonstrated that detectors built to identify text generated by a particular generator struggle with text from other generators. Furthermore, for newer generators, it might even be impossible to collect and curate labeled training datasets, since access to such models might be limited or even forbidden. Given this data problem, in this paper, we consider the situation where we have access to text from a generator but we do not know which generator it came from. However, we do have labeled data from some generators. In this context, we propose a framework for AI-generated text detection that can perform well on target data in the absence of labels. We frame this problem as an unsupervised domain adaptation problem, assuming we have labeled data from a source generator and unlabeled data from (perhaps newer) target generators. Our framework also uses a contrastive loss component that acts as a regularizer and helps the model learn invariant features and avoid overfitting to the particular generator it was trained on, hence improving performance on the unknown generator (Figure 1). For news text, our model achieves performance with a 0.8% margin of a fully supervised detector. Our main contributions in this paper are:

1. We propose a novel AI-generated text detection framework, **ConDA**, that uses unsupervised domain adaptation and self-supervised contrastive learning to effectively leverage labeled source domain and unlabeled target domain data.

2. Through extensive evaluations on benchmark human/AI-generated news datasets, spanning a variety of LLMs, we show that **ConDA** effectively solves the problem of label scarcity, and achieves state-of-the-art performance for unsupervised detection.

3. Furthermore, we create our own ChatGPT-generated data and via a case study, show the efficacy of our model on text generated using new conversational language models.

2 Related Work

Generated Text Detection The burgeoning progress in the generation capabilities of large language models has led to a corresponding increase in research and development efforts in the field of detection. Several recent efforts look at methods, varying from simple feature-based classifiers to fine-tuned language model-based detectors, in order to classify whether a piece of input text is human-written or AI-generated (Ippolito et al., 2019; Gehrmann et al., 2019; Mitchell et al., 2023), along with methods that specifically focus on AI-generated news (Zellers et al., 2019; Bogaert et al., 2022; Bhattacharjee and Liu, 2023; Kumarage et al., 2023). A related direction of work is that of authorship attribution (AA). While older AA methods focused on human authors, more recent efforts (Uchendu et al., 2020; Munir et al., 2021) build models to identify the generator for a particular input text. Recent work also shows how AI-generated text can deceive state-of-the-art AA models (Jones et al., 2022), thus making the task of detecting such text even more important.

Contrastive Learning for Text Classification Following the success of contrastive representation learning in the computer vision domain, several recent works in natural language have used contrastive learning for text classification, often for benefits such as robustness (Zhang et al., 2022; Ghosh and Lan, 2021; Pan et al., 2022), generalizability (Tan et al., 2020; Kim et al., 2022) and also in few-shot scenarios (Jian et al., 2022; Zhang et al., 2021; Chen et al., 2022a). Authors in (Qian et al., 2022; Chen et al., 2022b) also use ideas from contrastive learning to leverage label information to learn better representations for the classification task.

Domain Adaptation for Text Classification Domain adaptation (DA) is a paradigm that aims to tackle the distribution shift between training and testing distributions, by learning a discriminative classifier, that is invariant to domain-specific features (Sener et al., 2016). Along with labeled source data, DA methods may use either unlabeled target data (unsupervised DA) or a few labeled target samples (semi-supervised DA). In our work, we consider the unsupervised DA setting (Ganin et al., 2016). In the domain of language, unsupervised domain adaptation has been used in a variety of tasks (Ramponi and Plank, 2020), such as senti-

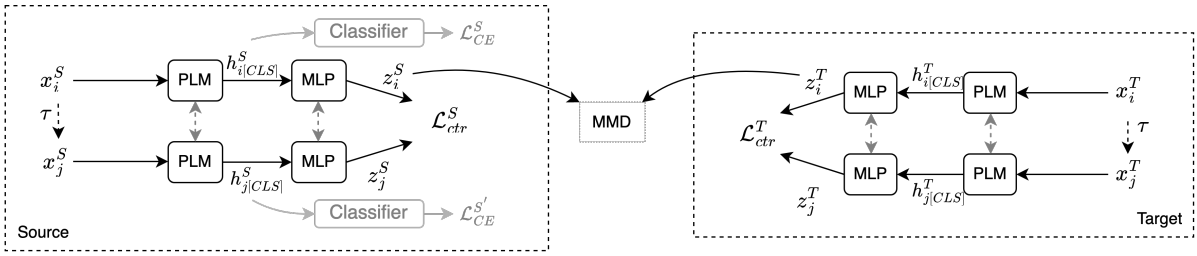


Figure 2: Our ConDA framework. PLM refers to the pre-trained language model (here, RoBERTa); PLM and MLP weights are shared across all four instances.

ment classification (Glorot et al., 2011; Trung et al., 2022), question answering (Yue et al., 2021), event detection (Trung et al., 2022), sequence tagging or labeling (Han and Eisenstein, 2019), etc.

In this work, we frame the problem of detecting AI-generated news text from multiple generators as an unsupervised domain adaptation task, where the different generators are the different data domains. Our proposed framework combines the representational power of self-supervised contrastive learning and a principled method for unsupervised domain adaptation to solve the AI-generated text detection problem. To the best of our knowledge, we are the first to propose this kind of a formulation for AI-generated text detection, along with a novel framework for this task. In the following section, we describe our framework in detail, along with our training objective.

3 Model

In this work, we consider a setting where we have labeled data from the source generator and only unlabeled samples from the target generator[2]. More formally, the source domain dataset is denoted by $\mathbf{S} = \{(x_i^S, y_i^S)\}_{i=1}^{N^S}$ where $y_i^S \in \{0, 1\}$ corresponding to 'human-written' or 'AI-generated' labels, and N^S is the number of source domain samples. The target domain is denoted by $\mathbf{T} = \{(x_i^T)\}_{i=1}^{N^T}$, where N^T is the number of target domain samples. Note that all domains share the same label space.

3.1 ConDA Framework

We show our framework in Figure 2. For the detector, we use a pre-trained RoBERTa model (`roberta-base`) from Huggingface[3], with a classifier head on top of it. As the input, we have two articles: x_i^S from the source and x_i^T from the target. We perform a text transformation τ on this text whereby we get the transformed samples x_j^S and x_j^T. In order to input both the original and the transformed (also referred to as 'perturbed' throughout this paper), we use a Siamese network (Bromley et al., 1993; Neculoiu et al., 2016; Reimers and Gurevych, 2019) where the RoBERTa model weights are shared across the two branches. For the two input texts, we take the hidden layer representation of the `[CLS]` token: $h_{i[CLS]}^S$ and $h_{j[CLS]}^S$. Following the methodology in (Chen et al., 2020), we pass these embeddings through a projection layer that consists of a multilayer perceptron (MLP) with one hidden layer and compute a contrastive loss in the lower dimensional projection space. The MLP can be represented as a function $g(\cdot) : \mathbb{R}^{d_h} \mapsto \mathbb{R}^{d_p}$, where d_h is the size of the hidden layer embedding: 768 for `roberta-base`, and we set d_p as 300, following (Pan et al., 2022). For the source domain, we also compute the cross-entropy losses for binary classification of both the original and transformed text. Furthermore, we have a domain discrepancy component between the projected representations of the source and target text. We elaborate on the losses and related design choices in the following section.

3.2 Training Objective

Source Classification Loss: We leverage the availability of the source labels and compute the binary cross-entropy (CE) losses for the original and the perturbed text:

$$\mathcal{L}_{CE}^S = -\frac{1}{b}\sum_{i=1}^{b}[y_i \log p(y_i|h_{i[CLS]}^S) + (1-y_i)\log(1-p(y_i|h_{i[CLS]}^S))] \quad (1)$$

$\mathcal{L}_{CE}^S$ denotes the CE loss for the original text, b denotes the batch size. Similarly, we compute $\mathcal{L}_{CE}^{S'}$ for the perturbed text, and we skip the equation

[2]We use the terms 'LLM' and 'generator' interchangeably.
[3]https://huggingface.co/roberta-base

for brevity. Inspired by the training objective in (Pan et al., 2022), we use CE losses for both the original and perturbed samples in the final training objective. The transformation performed on the original text (i.e. synonym replacement in our experiments) preserves the semantics of the text and hence is label-preserving. In such a case we would want a classifier to be able to detect text with such minor, semantic-preserving perturbations as well. Not only is this supposed to improve the robustness of the classifier, but in turn also the generalizability of the detector (Xu and Mannor, 2012), which is essential for our use-case.

Contrastive Loss: To learn a better representation of the input text, we use contrastive losses, for both the source and target texts (Figure 2). We use a loss similar to the one in (Chen et al., 2020): the only difference is that, instead of computing the loss between two transformed views of the text, we use the transformed text and the original anchor text. For our transformation, we use synonym replacement (more details regarding implementation are in the Appendix). The contrastive loss for the source is denoted by:

$$\mathcal{L}^S_{ctr} = -\sum_{(i,j)\in b} \log \frac{exp(sim(z^S_i, z^S_j)/t)}{\sum_{k=1}^{2|b|} \mathbb{1}_{[k\neq i]} exp(sim(z^S_i, z^S_k)/t)} \quad (2)$$

z^S_i and z^S_j denote the projection layer embeddings for the original (anchor) and the transformed text, t is the temperature, b is the current mini-batch, $sim(\cdot,\cdot)$ is a similarity metric which is cosine similarity in our case. Similar to (Chen et al., 2020), we do not sample or mine negatives explicitly, we simply consider the remaining $2(|b|-1)$ samples in the mini-batch b as negatives. We have a similar contrastive loss for the target domain, denoted by $\mathcal{L}^T_{ctr}$, and we skip the equation here for brevity. The objective of these contrastive losses is to bring the positive pairs, i.e. anchor and the transformed sample, closer in the representation space, and well separated from the negative samples.

Since the performance of contrastive learning depends significantly on the transformation used to generate the positive sample (Tian et al., 2020), we take a principled approach to choosing a transformation out of several possible ones (Bhattacharjee et al., 2022). To choose one transformation for the main experiments, we evaluate a simple detection model (only one domain) over different choices of transformations and choose the one that gives the best performance, and therefore, is the most discriminative. In the input space, we use random swap, random crop, and synonym replacement as the choices. In the latent space, we have paraphrasing and summarization as the choices. Based on detection performance, we finally choose synonym replacement as the transformation that we use throughout the remainder of the paper.

Maximum Mean Discrepancy(MMD): Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) is a metric to measure the distance between two distributions, which in our case refers to two different generators. Formally, let $S = \{x^S_1, x^S_2, ..., x^S_{N^S}\}$ and $T = \{y^T_1, y^T_2, ..., y^T_{N^T}\}$ be two sets of samples drawn from distribution $\mathcal{S}$ and $\mathcal{T}$, respectively. The MMD distance between the distributions $\mathcal{S}$ and $\mathcal{T}$ is defined as the distance between means of two samples mapped to the Reproducing Kernel Hilbert Space (RKHS) (Steinwart, 2001). Following past work (Pan et al., 2010; Long et al., 2015), we compute the MMD between text embeddings in a lower dimensional space, i.e. between z^S_i and z^T_i. Formally,

$$MMD(\mathcal{S},\mathcal{T}) = \| \frac{1}{N^S}\sum_{i=1}^{N^S} \phi(z^S_i) - \frac{1}{N^T}\sum_{i=1}^{N^T} \phi(z^T_i) \|_{\mathcal{H}}, \quad (3)$$

where $\phi: \mathcal{S} \mapsto \mathcal{H}$ and $\mathcal{H}$ represents the RKHS space.

The final training objective for our main framework is:

$$\begin{aligned}\mathcal{L} =& \frac{(1-\lambda_1)}{2}[L^S_{CE} + L^{S'}_{CE}] + \\ & \frac{\lambda_1}{2}[L^S_{ctr} + L^T_{ctr}] + \lambda_2 MMD(S,T)\end{aligned} \quad (4)$$

where λ_1 and λ_2 are hyper-parameters.

4 Experimental Settings

In this section we describe the datasets, baselines and the training details we use for our experiments.

4.1 Dataset

Since our task requires news text from multiple generators, we use the publicly available TuringBench[4] dataset (Uchendu et al., 2021), which contains human-written and machine-generated news articles from 19 generators, spanning over 10 different language model architectures (including different sizes for some of the generators). For a full list of labels, check Appendix B.1. Out of the 10 different architectures available in the dataset, we

[4]https://turingbench.ist.psu.edu/

Model	# of Parameters	Shorthand used
CTRL	1.5B	C
FAIR_wmt19	656M	F19
GPT-2_xl	1.5B	G2X
GPT-3	175B	G3
GROVER_mega	1.5B	GM
XLM	550M	X

Table 1: List of generators we used for our evaluation.

sample a representative set of 6 different generators, in order to evaluate our model (Table 1). For most of the architectures, if there were multiple parameter sizes available, we choose the largest one, to make the detection task more challenging for our model. We briefly go over the architectural details of each of the generators used:

CTRL (Keskar et al., 2019) is a transformer-based language model, that is developed for controllable generation of text based on control codes for style, content, and task-specific generation. The model is pre-trained on a variety of text types, including web-text, news, question-answering datasets, etc. **FAIR_wmt19** (Ng et al., 2019) is FAIR's model that was developed for the WMT19 news translation task. Texts in TuringBench are from the English version of the FAIR_wmt19 language model. **GPT2-XL** (Radford et al., 2019) is the 1.5B size version of GPT-2, which is also a transformer-based language model built upon the architecture of the original GPT model (Radford et al., 2018), with further modifications. **GPT-3** (Brown et al., 2020) is the successor of the GPT-2 model, and is the largest model we use in our evaluation, with a size of 175B parameters. **GROVER_mega** (Zellers et al., 2019) is the largest version of the GROVER model, which is a transformer-based model, similar in architecture to GPT-2, but trained to conditionally generate news articles. **XLM** (Lample and Conneau, 2019) is also a transformer-based language model designed for cross-lingual tasks.

Furthermore, given the challenge of detecting text from the more recent conversational language models, we augment the TuringBench dataset with ChatGPT news articles. Following a similar data generation procedure as in (Uchendu et al., 2021), we use a subset of around $9,000$ news articles from The Washington Post and CNN (more details in Appendix B.2), and use the headlines to generate articles using ChatGPT. For this paper, we used the OpenAI API with the `gpt-3.5-turbo` model (version as on March 14, 2023). After experimenting with a few different prompt types, we finally used the following prompt for each news `headline`: "Generate a news article with the headline '`<headline>`'." Finally, we have a balanced dataset of approximately 9k human-written articles, and 9k articles generated using ChatGPT (after accounting for null values and API request errors). For simplicity, we name this dataset **ChatGPT News** and we use this dataset for a case study on ChatGPT generated news articles, in Section 6.

4.2 Baselines

For a fair comparison, we compare our method with baselines that do not require labeled data. We use two open-source AI-generated text detectors, namely GLTR (Gehrmann et al., 2019) and the more recent DetectGPT (Mitchell et al., 2023), as our unsupervised baseline models.

GLTR utilizes a proxy language model to calculate the token-wise log probability of the input text. It employs four statistical tests: (i) log probabilities $(\log p(x))$, (ii) average token rank (Rank), (iii) token log-rank (LogRank), and (iv) predictive entropy (Entropy). The first test assumes that a higher average log probability in the input text indicates AI generation. The second and third tests follow a similar assumption, where input texts with lower average rank are more likely to be generated by AI. The last test is based on the hypothesis that AI-generated texts tend to exhibit less diversity and surprises, resulting in low entropy.

DetectGPT also utilizes a proxy language model to calculate the token-wise log probability. However, its decision function is based on comparing the log probability of the original input text with the log probability of a set of n perturbed versions of the input text. These perturbations are generated using the mask-filling language model T5(T5-base) (Raffel et al., 2020). The decision function assumes that if the log probability difference between the input text and the perturbed text is positive with high probability, then the input text is likely to be AI-generated.

In addition to these zero-shot baselines, we include the off-the-shelf **OpenAI-GPT2 detector** as one of the baselines in our study. The OpenAI-GPT2 detector is a RoBERTa model fine-tuned specifically for detecting GPT2-generated text. It was trained on a GPT-2-output dataset [5] comprising

[5] https://github.com/openai/gpt-2-output-dataset

Task	Source Only	ConDA	ΔF1	Source Only Avg.	ConDA Avg.
F19 → C	93	96	3		
G2X → C	61	81	20		
G3 → C	61	69	8	57.2	**84.6**
GM → C	22	99	77		
X → C	49	78	29		
C → F19	40	73	33		
G2X → F19	83	83	0		
G3 → F19	62	63	1	41.2	**54.8**
GM → F19	4	27	23		
X → F19	17	28	11		
C → G2X	77	77	0		
F19 → G2X	90	98	8		
G3 → G2X	73	69	-4	74.4	**86.6**
GM → G2X	81	95	14		
X → G2X	51	94	43		
C → G3	60	81	21		
F19 → G3	48	89	41		
G2X → G3	87	82	-5	64.8	**83.2**
GM → G3	74	77	3		
X → G3	55	87	32		
C → GM	26	95	69		
F19 → GM	10	68	58		
G2X → GM	66	92	26	37.6	**73.4**
G3 → GM	67	68	1		
X → GM	19	44	25		
C → X	81	94	13		
F19 → X	77	95	18		
G2X → X	87	94	7	81	**90.2**
G3 → X	73	69	-4		
GM → X	87	99	12		

Table 2: Performance of **ConDA** on unlabeled target domains. Source-only model for each task S → T refers to zero-shot evaluation of a model trained on S and evaluated on test set of T. ΔF1 is increase (or decrease, in a few cases) in F1 scores of the **ConDA** model over the source-only model. Avg. scores in **bold** indicate where ConDA out-performs the source-only model.

250k documents from the WebText test set (Radford et al., 2019) as human-written text. Then as the AI text, this dataset contains 250k GPT-2 generated text with a temperature of 1 with no truncation and another 250k samples generated with top-k 40 truncation. Note that for our evaluation, this model may be considered unsupervised for all target domains except GPT-2_xl.

5 Results

To understand and investigate the effectiveness of our model, we try to answer the following research questions:

- RQ1: Does **ConDA** perform well on unknown target domains in comparison to a source-only model (Table 2) and a supervised model fine-tuned on the target (Table 3)?

- RQ2: How well does **ConDA** perform in comparison to unsupervised-baselines (Table 4)?

- RQ3: Are each of the loss components beneficial in training (Table 5)?

All results are reported as an average over 3 training runs with 3 different random seeds.

5.1 Performance of ConDA on unlabeled target data

To evaluate the performance of ConDA on each of the target domains, i.e. generators, we first look at how our model improves over a source-only model. Table 2 shows the results for this experiment, grouped by target domain. We report F1 scores for the ConDA framework and a source-only model, along with scores averaged over sources, for each target. The source-only model is a pre-trained RoBERTa (roberta-base) fine-tuned only on the source domain S. The source-only scores provide an estimate of how well a model trained just on the source transfers to the target domain. Although a few of the source-only models have satisfactory performance on the target, using our ConDA framework, we achieve performance gains over the source-only model in almost all tasks (rows with positive ΔF1 values). Particularly interesting are the cases where we use a smaller generator as the source, a larger one as the target, and still get high performance gains: 58 F1 points for FAIR_wmt19 (656M)→ GROVER_mega (1.5B), and 41 F1 points for FAIR_wmt19 (656M)→ GPT-3 (175B). This may suggest that, with our ConDA framework, even having unlabeled data from newer and possibly larger generators can improve performance if we use a suitable generator as the source.

Next, we compare the performance of our model with a fully-supervised detector trained on the target domain in Table 3. For ConDA, we show the test performance for all target-source pairs. For the supervised model, we use a pre-trained RoBERTa (roberta-base) fine-tuned on the target data. We then evaluate the model on the test set of the same target domain, and essentially this is our upper bound performance. ConDA achieves test performance comparable to fully-supervised models. In particular, for targets CTRL and XLM, ConDA (with GROVER_mega as source) achieves upper bound performance. For targets GROVER_mega and GPT-2_xl, ConDA performs within 3 and 6 F1 points of the fully-supervised model.

Interestingly, for target generator GPT-3, all the ConDA models perform better than the fully-

Target	Supervised (Fine-tuned RoBERTa)		ConDA model with Source as													
			C		F19		G2X		G3		GM		X		*Average*	
	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1	AUROC
C	98	1		–	96	0.998	81	0.949	69	0.783	**99**	**1**	78	0.991	84.6	0.9442
F19	98	0.999	73	0.894		–	**83**	**0.966**	63	0.607	27	0.826	28	0.766	54.8	0.8118
G2X	92	0.998	77	0.946	**98**	**0.998**		–	69	0.902	95	0.991	94	0.991	86.6	0.9656
G3	72	0.988	81	0.938	**89**	0.975	82	0.962		–	77	**0.982**	87	0.981	83.2	0.9676
GM	98	0.996	**95**	**0.988**	68	0.961	92	0.984	68	0.819		–	44	0.98	73.4	0.9464
X	99	1	94	0.985	95	0.999	94	0.988	69	0.683	**99**	**0.999**		–	90.2	0.9308

Table 3: Performance of our ConDA model on each of the target domains, with each of the other domains as source. Numbers in **bold** are the best performing ConDA models for each target domain, i.e. closest to fully supervised performance.

supervised performance, with the best F1 (from ConDA with source FAIR_wmt19) being 27 points higher than the supervised performance. Furthermore, when GPT-3 is used as the source domain, we get mediocre performance for all target domains. We suspect that this might be due to the following reason: The GPT-3 data in TuringBench might be noisy and therefore lack good quality, discriminative signals that can guide the detector. The performance improvement that occurs when ConDA is evaluated on GPT-3 as target, with any other domain as source, is possibly due to the effective transfer of discriminative signals from the labeled source data, hence improving the performance on GPT-3 data even in the absence of labels.

5.2 Performance compared to unsupervised baselines

We compare our ConDA framework with relevant unsupervised baselines and report results in Table 4. Out of the four GLTR measures ($\log p(x)$, Rank, Log Rank, and Entropy), the first three fare quite well for detecting CTRL-generated text, but performance on other generators is quite poor. DetectGPT, which is the most recent method we evaluate, performs poorly on almost all generators, with some satisfactory performance on CTRL and XLM. Surprisingly, the OpenAI GPT-2 Detector performs poorly on the GPT-2_xl data from TuringBench, although it can be considered supervised for this particular target. Finally, we see ConDA outperforms all the baselines in terms of maximum AUROC, and all but one in terms of average AUROC.

Interestingly, we see that ConDA models trained with GROVER_mega as the source perform very well for several target domains. This might be because GROVER (Zellers et al., 2019) was designed and trained in order to generate news articles. Since our task here is to specifically detect human vs. AI written news articles, training models on data generated using GROVER_mega is useful and this data possibly has good discriminative signals.

5.3 Ablation: Effectiveness of loss components

We evaluate variants of the ConDA model, by removing one component at a time and compare these in Table 5. **ConDA \CEs** removes the two cross-entropy losses, i.e. no supervision even for the source. **ConDA \contrast** removes the contrastive loss components for both source and target. **ConDA \MMD** removes the MMD loss between source and target. Hence the only component that makes use of the unlabeled target domain data is the target contrastive loss. Finally, **ConDA** is the full model. We see that the full model outperforms all the variants, implying that all three types of components are essential for detection performance in this problem setting. Combined with source supervision, the contrastive losses and the MMD objective effectively tie the power of self-supervised learning and unsupervised domain adaptation resulting in superior performance across target domains.

6 A Case Study on ChatGPT

Given recent concerns surrounding OpenAI's ChatGPT and GPT-4 (OpenAI, 2023), it is important to create detectors for text generated by these conversational language models. With the incredible fluency and writing quality these language models possess, not only can such text easily fool humans (Else, 2023) but can also be extremely difficult for detectors to identify. Even OpenAI's detector struggles to detect AI-generated text reliably[6]. Hence in this case study, we are interested

[6]https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text

Target	GLTR				DetectGPT	OpenAI-GPT2 Detector	ConDA *(ours)*	
	log p(x)	Rank	LogRank	Entropy			*Avg.*	*Max. (Source)*
C	0.951	0.849	**0.956**	0.379	0.793	0.366	0.9442	**1.00** (GM)
F19	0.558	0.618	0.546	0.656	0.5045	0.464	**0.8118**	**0.966** (G2X)
G2X	0.485	0.508	0.48	0.631	0.529	0.48	**0.9656**	**0.998** (F19)
G3	0.362	0.356	0.341	0.756	0.5485	0.73	**0.9676**	**0.982** (GM)
GM	0.434	0.469	0.434	0.592	0.5415	0.659	**0.9464**	**0.988** (C)
X	0.473	0.762	0.442	0.696	0.7355	0.873	**0.9308**	**0.999** (GM,F19)

Table 4: Performance of ConDA in comparison to unsupervised baselines, as AUROC. For ConDA, we report the average AUROC over all sources (for each target) and also the maximum AUROC (across all sources), along with the corresponding source in parentheses. **Bold** shows superior performance across each target.

Model variant	Target (avg. across Sources)					
	C		F19		G2X	
	F1	AUROC	F1	AUROC	F1	AUROC
ConDA \CEs	60.4	0.5268	41.6	0.4914	60.25	0.4822
ConDA\contrast	62.6	0.898	44.2	0.687	85.4	0.9594
ConDA\MMD	69.8	0.7826	39.8	0.6272	65	0.852
ConDA	**84.6**	**0.9442**	**54.8**	**0.8118**	**86.6**	**0.9656**

Table 5: Comparison of different model variants; **bold** shows best performance. We randomly chose 3 target domains to show in this table due to space constraints.

in evaluating our ConDA framework on ChatGPT-generated news articles, in an unsupervised manner. Since there is no existing dataset of ChatGPT-generated vs. human-written text or news, we create our own dataset as explained in Section 4.1. We assign ChatGPT as the unlabeled target domain and assume that we have labeled data from the 6 other generators (Table 1). Therefore we emulate a real-world scenario where labeled data from older generators may be available, but it might be hard to find labeled samples for newer LLMs. We sample 4k articles from our **ChatGPT News** dataset and evaluate the same 3 unsupervised models as in Section 4.2 (upper row block in Table 6), and our ConDA framework over 6 source generators (lower row block in Table 6) on this data. For GLTR, we report the average over the 4 statistical measures. Although we see satisfactory performance across most methods, our ConDA framework with source as FAIR_wmt19 and GPT2_xl has the best and the second best performance, respectively. However, we would like the reader to note that such good performance on our ChatGPT News dataset does not imply similar performance on any other type of text generated by ChatGPT (see Section 8). For text embedding visualizations from our ConDA model for this ChatGPT case study, check Appendix C.

Baselines		
GLTR(avg.)	DetectGPT	OpenAI Detector
0.72925	0.7735	0.715
ConDA model with Source as		
C	F19	G2X	G3	GM	X
0.653	**0.877**	<u>0.831</u>	0.679	0.73	0.626

Table 6: Results on our ChatGPT News dataset using unsupervised baselines (upper row) and ConDA (lower row). Scores are AUROC. **Bold** shows best and <u>underline</u> shows second best performance.

7 Conclusion & Future Work

In this work, we address the problem of AI-generated text detection in the absence of labeled target data. We propose a contrastive domain adaptation framework that leverages the power of both unsupervised domain adaptation and self-supervised representation learning, in order to tackle the task of AI-generated text detection. Our experiments focus on news text, and show the effectiveness of the framework, as well as superior performance when compared to unsupervised baselines. We also perform a case study to evaluate our framework on our dataset of ChatGPT-generated news articles and achieve satisfactory performance. Our framework can be easily extended to other forms of text beyond news and our results suggest that such a framework may be effectively used for detection of AI-generated text when labels are unavailable, such as in the case of newly emerging generators. Future work can investigate more challenging variations of this problem, such as domain adaptation across multiple unlabeled target generators, generalization to fully unseen generators, etc., along with exploring other types of text such as scientific articles, medical literature, etc.

8 Limitations

Problem Formulation & Model: Despite the impressive performance of our ConDA model, there are several limitations that we go over in this section. First, our model and evaluations only focus on news text and performance may vary widely across other types of text such as creative writing, scientific articles, blog-style articles, etc. Second, our model simply tries to detect whether an input news article is generated by an LLM or not. AI generation does not necessarily imply malice. A dimension that our model does not consider is that of factuality: not all AI-generated news is factually inaccurate, and not all human-written news is factually correct. Incorporating factuality, perhaps in the form of a fact-checking module, could possibly improve the usefulness of our model. Third, our model, along with most other AI-generated text detectors, is not explainable. The discrete input space of natural language also makes it difficult to identify specific features that result in detection. Furthermore, given the black-box nature of LLMs, any detector that uses some LLM as the backbone, trade off explainability for performance gains.

ChatGPT Case Study: As we elaborated in Section 4.1, we create our own ChatGPT-generated news article dataset, following a procedure similar to (Uchendu et al., 2021). However, the data we generated is conditioned on the sample of human-written news articles we randomly selected. We suppose the performance of our model on this ChatGPT data hence is dependent on this sample. The high performance scores for ChatGPT-generated articles could also stem from the inherent structure of news articles; our data is specifically constrained to the style of journalistic news articles. Therefore, good performance on our news article dataset for ChatGPT does not necessarily imply similar performance across text from other areas. For this, more thorough evaluation is needed, which would be an interesting direction for future work.

9 Ethical Considerations

We go over some of the ethical considerations surrounding this work and similar directions.

9.1 Potential to Penalize Benign Use of LLMs

Recent articles have demonstrated how the newer language models including ChatGPT, GPT-4 (OpenAI, 2023), Bing Chat[7], etc. can be used to improve productivity, spur creative thinking, help with writing essays or cover letters or even explain concepts and help in homework. As these LLMs become more pervasive, standard use of these as writing or brain-storming assistants may become commonplace. In such a case, we may encounter an increasing amount of text generated by these LLMs online. Such text, if used for benign purposes such as the ones mentioned above, should not be penalized by a detector such as ours. This brings another dimension to this already challenging problem: the issue of intent. Flagging AI-generated content without characterizing the intent behind that could wrongfully penalize users of LLMs. Therefore, the nuances surrounding this need to be considered while using such a detector.

9.2 Danger of Misuse in High Stakes Areas

We discuss the issue of model misuse, by taking education as an example. Given the accessibility of ChatGPT and other recent AI-text generators, educators have expressed concerns (Tlili et al., 2023) over students cheating or plagiarising via these new technologies. There are already commercial detectors for AI-generated content such as GPTZero[8] and one from Copyleaks[9] that educators may use. However, similar to our model, there is always a margin of error on such detectors. Performing plagiarism checks and subsequently implementing punitive action based solely on such detectors may be detrimental in case of false positives. Legitimate work by a student may be misclassified by these detectors, and potentially impact their career. Eventually, this also diminishes trust in these detectors. Hence, before the widespread use of such AI-generated text detectors, thorough studies on error analysis and reliability need to be performed, along with policy changes to accommodate for the rapidly evolving landscape of AI technologies.

Acknowledgements

This work is supported by the DARPA SemaFor project (HR001120C0123), Office of Naval Research (N00014- 21-1-4002), Army Research Office (W911NF2110030) and Army Research Lab (W911NF2020124). The views, opinions and/or findings expressed are those of the authors.

[7] https://www.bing.com/new
[8] https://gptzero.me/
[9] https://copyleaks.com/ai-content-detector

References

Amrita Bhattacharjee, Mansooreh Karami, and Huan Liu. 2022. Text transformations in contrastive self-supervised learning: a review. *arXiv preprint arXiv:2203.12000*.

Amrita Bhattacharjee and Huan Liu. 2023. Fighting fire with fire: Can chatgpt detect ai-generated text? *arXiv preprint arXiv:2308.01284*.

Jérémie Bogaert, Marie-Catherine de Marneffe, Antonin Descampe, and Francois-Xavier Standaert. 2022. Automatic and manual detection of generated news: Case study, limitations and challenges. In *Proceedings of the 1st International Workshop on Multimedia AI against Disinformation*, pages 18–26.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Junfan Chen, Richong Zhang, Yongyi Mao, and Jie Xu. 2022a. Contrastnet: A contrastive learning framework for few-shot text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10492–10500.

Qianben Chen, Richong Zhang, Yaowei Zheng, and Yongyi Mao. 2022b. Dual contrastive learning: Text classification via label-aware data augmentation. *arXiv preprint arXiv:2201.08702*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Luigi De Angelis, Francesco Baglivo, Guglielmo Arzilli, Gaetano Pierpaolo Privitera, Paolo Ferragina, Alberto Eugenio Tozzi, and Caterina Rizzo. 2023. Chatgpt and the rise of large language models: the new ai-driven infodemic threat in public health. *Frontiers in Public Health*, 11:1567.

Holly Else. 2023. Abstracts written by chatgpt fool scientists. *Nature*, 613(7944):423–423.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Aritra Ghosh and Andrew Lan. 2021. Contrastive learning improves model robustness under label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2703–2708.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. 2012. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.

Xiaochuang Han and Jacob Eisenstein. 2019. Unsupervised domain adaptation of contextualized embeddings for sequence labeling. *arXiv preprint arXiv:1904.02817*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Automatic detection of generated text is easiest when humans are fooled. *arXiv preprint arXiv:1911.00650*.

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks VS Lakshmanan. 2020. Automatic detection of machine generated text: A critical survey. *arXiv preprint arXiv:2011.01314*.

Yiren Jian, Chongyang Gao, and Soroush Vosoughi. 2022. Contrastive learning for prompt-based few-shot language learners. *arXiv preprint arXiv:2205.01308*.

Keenan Jones, Jason RC Nurse, and Shujun Li. 2022. Are you robert or roberta? deceiving online authorship attribution models using neural text generators. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 429–440.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Youngwook Kim, Shinwoo Park, and Yo-Sub Han. 2022. Generalizable implicit hate speech detection using contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6667–6679.

Tharindu Kumarage, Joshua Garland, Amrita Bhattacharjee, Kirill Trapeznikov, Scott Ruston, and Huan Liu. 2023. Stylometric detection of ai-generated text in twitter timelines. *arXiv preprint arXiv:2303.03697*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint arXiv:2301.11305*.

Shaoor Munir, Brishna Batool, Zubair Shafiq, Padmini Srinivasan, and Fareed Zaffar. 2021. Through the looking glass: Learning to attribute synthetic text generated by language models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1811–1822.

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 148–157.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook fair's wmt19 news translation task submission. *arXiv preprint arXiv:1907.06616*.

OpenAI. 2023. Gpt-4 technical report. *arXiv*.

Lin Pan, Chung-Wei Hang, Avirup Sil, and Saloni Potdar. 2022. Improved text classification via contrastive adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11130–11138.

Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. 2010. Domain adaptation via transfer component analysis. *IEEE transactions on neural networks*, 22(2):199–210.

Tao Qian, Fei Li, Meishan Zhang, Guonian Jin, Ping Fan, and Wenhua Dai. 2022. Contrastive learning from label distribution: A case study on text classification. *Neurocomputing*, 507:208–220.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey. *arXiv preprint arXiv:2006.00632*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ozan Sener, Hyun Oh Song, Ashutosh Saxena, and Silvio Savarese. 2016. Learning transferrable representations for unsupervised domain adaptation. *Advances in neural information processing systems*, 29.

Ingo Steinwart. 2001. On the influence of the kernel on the consistency of support vector machines. *Journal of machine learning research*, 2(Nov):67–93.

Reuben Tan, Bryan A Plummer, and Kate Saenko. 2020. Detecting cross-modal inconsistency to defend against neural fake news. *arXiv preprint arXiv:2009.07698*.

Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? *Advances in neural information processing systems*, 33:6827–6839.

Ahmed Tlili, Boulus Shehata, Michael Agyemang Adarkwah, Aras Bozkurt, Daniel T Hickey, Ronghuai Huang, and Brighter Agyemang. 2023. What if the devil is my guardian angel: Chatgpt as a case study of using chatbots in education. *Smart Learning Environments*, 10(1):15.

Nghia Ngo Trung, Linh Ngo Van, and Thien Huu Nguyen. 2022. Unsupervised domain adaptation for text classification via meta self-paced learning. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4741–4752.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395.

Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. Turingbench: A benchmark environment for turing test in the age of neural text generation. *arXiv preprint arXiv:2109.13296*.

Onur Varol, Emilio Ferrara, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2017. Online human-bot interactions: Detection, estimation, and characterization. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 280–289.

Huan Xu and Shie Mannor. 2012. Robustness and generalization. *Machine learning*, 86:391–423.

Zhenrui Yue, Bernhard Kratzwald, and Stefan Feuerriegel. 2021. Contrastive domain adaptation for question answering using limited text corpora. *arXiv preprint arXiv:2108.13854*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems*, 32.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021. Few-shot intent detection via contrastive pre-training and fine-tuning. *arXiv preprint arXiv:2109.06349*.

Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Ré. 2022. Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. *arXiv preprint arXiv:2203.01517*.

Jiawei Zhou, Yixuan Zhang, Qianni Luo, Andrea G Parker, and Munmun De Choudhury. 2023. Synthetic lies: Understanding ai-generated misinformation and evaluating algorithmic and human solutions. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–20.

A Reproducibility

A.1 Training Details & Hyper-parameters Used

We perform all experiments using PyTorch, on a single NVIDIA A100 GPU. We use an Adam optimizer with a learning rate of 2×10^{-5}. All models are trained for 5 epochs with early stopping, to avoid overfitting. We provide the full list of hyper-parameter values in Table 7 to facilitate reproducibility.

For values of λ_1 and λ_2 in Equation 3, we use 0.5 and 1.0 respectively, after choosing these values from a small hyper-parameter search. We randomly choose 4 tasks: {F19 $\rightarrow$ G3, C $\rightarrow$ X, G2X $\rightarrow$ GM, and G2X $\rightarrow$ F19} and evaluate models with $\lambda_1 = \{0.2, 0.5, 0.8\}$ and $\lambda_2 = \{0.2, 0.5, 1.0\}$. We finally choose $\lambda_1 = 0.5$ and $\lambda_2 = 1.0$ based on these evaluation performances.

A.2 Synonym Replacement Implementation

In order to implement the synonym replacement transformation, we perturb 10% of the words in each sentence in an article by replacing these with their synonyms. Out of these words, we only perform synonym replacement for words that have a `NOUN`, `ADVERB`, `ADJECTIVE` or `VERB` POS tag. Synonyms are based on WordNet Synsets from the nltk[10] package. If a word has multiple synonyms, we choose one from that list, uniformly at random.

Hyper-parameter	Description	Value
λ_1	Weight for both source & target contrastive losses in final objective function (Eq. 3)	0.5
λ_2	Weight for MMD loss in final objective function (Eq. 3)	1
t	Temperature for contrastive loss in Eq 3	0.5
lr	Learning rate	2×10^{-5}
$epochs$	Number of epochs for training	5
max_seq_len	Maximum input sequence length	256
$weight\_decay$	Weight decay for Adam optimizer	0
d_p	Embedding size of the MLP projection space	300
$\|b\|$	Batch size for training the model	16

Table 7: Hyper-parameter values we used for all our experiments.

B TuringBench Details

B.1 Labels

TuringBench (Uchendu et al., 2021) has 200k samples across 20 labels. These labels include 'human' and 19 different generators, which are: { Human, GPT-1, GPT-2_small, GPT-2_medium, GPT-2_large, GPT-2_xl, GPT-2_PyTorch, GPT-3, GROVER_base, GROVER_large, GROVER_mega, CTRL, XLM, XLNET_base, XLNET_large, FAIR_wmt19, FAIR_wmt20, TRANSFORMER_XL, PPLM_distil, PPLM_gpt2}.

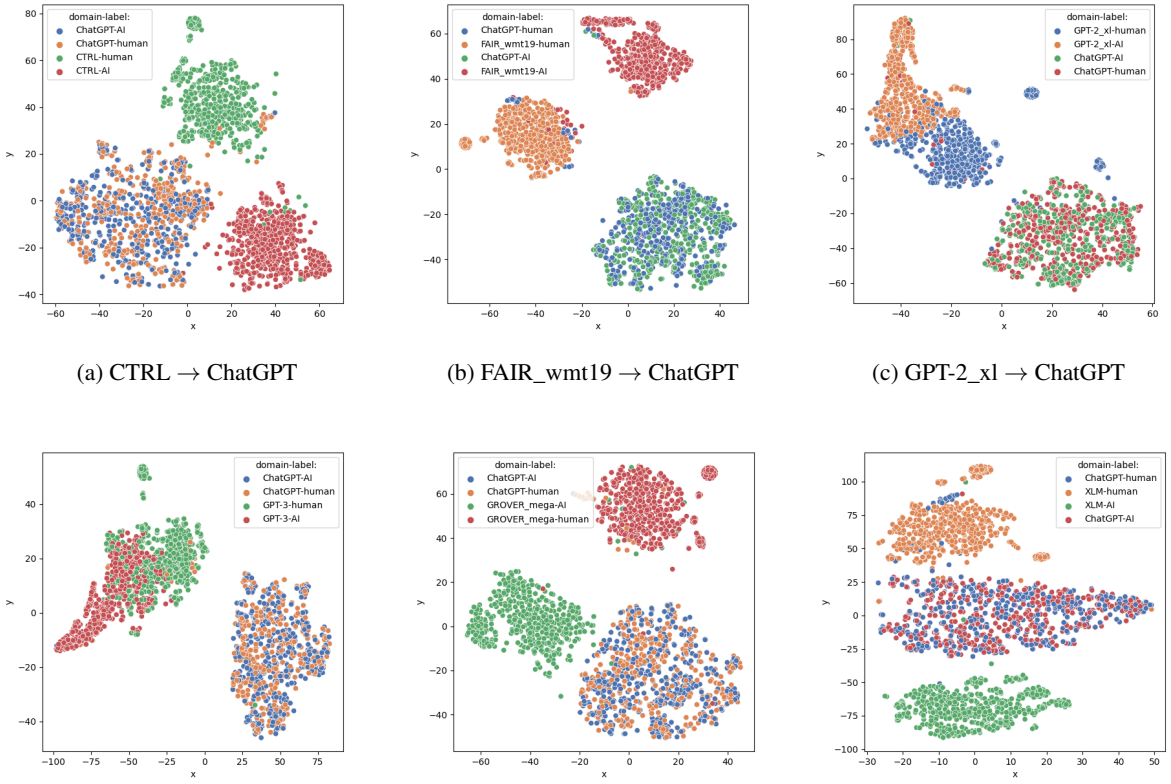
B.2 Human-written Articles

Human-written news articles in TuringBench are from The Washington Post, CNN, and a Kaggle dataset with CNN news articles from 2014-2020 and The Washington Post news articles from 2019-2020. More details on the TuringBench data are in (Uchendu et al., 2021). For the human-written articles in our **ChatGPT News** dataset, we use a random sample from the dataset of CNN and Washington Post articles as used in TuringBench.

C ChatGPT Visualizations

Here, we visually explore embeddings from the ConDA model for instances in Table 6, in order to

[10]https://www.nltk.org/



(a) CTRL → ChatGPT (b) FAIR_wmt19 → ChatGPT (c) GPT-2_xl → ChatGPT

(d) GPT-3 → ChatGPT (e) GROVER_mega → ChatGPT (f) XLM → ChatGPT

Figure 3: t-SNE plots showing text representations from our ConDA model, for each of the $S \rightarrow$ ChatGPT tasks, where $S \in$ {CTRL, FAIR_wmt19, GPT-2_xl, GPT-3, GROVER_mega, XLM}, corresponding to plots (**a**-**f**), respectively.

understand the issues surrounding the detection of ChatGPT-generated news articles. Figure 3 shows the embeddings from all 6 ConDA models. In all the plots, we see that the human-written and ChatGPT-generated news articles in our ChatGPT News dataset are very closely clustered together, and are not separable. Therefore, even though our model achieves substantially high AUROC scores, there are possibly many false positives and/or false negatives, thus providing an intuition that better feature selection methods might be necessary here.