CoCo: Coherence-Enhanced Machine-Generated Text Detection Under Low Resource With Contrastive Learning

Anonymous ACL submission

Abstract

Machine-Generated Text (MGT) detection, a task that discriminates MGT from Human-Written Text (HWT), plays a crucial role in preventing misuse of text generative models, which excel in mimicking human writing style recently. Latest proposed detectors usually take coarse text sequences as input and fine-tune pretrained models with standard cross-entropy loss. However, these methods fail to consider the linguistic structure of texts. Moreover, they lack the ability to handle the low-resource problem which could often happen in practice considering the enormous amount of textual data online. In this paper, we present a **co**herence-based **co**ntrastive learning model named CoCo to detect the possible MGT under low-resource scenario. To exploit the linguistic feature, we encode coherence information in form of graph into text representation. To tackle the challenges of low data resource, we employ a contrastive learning framework and propose an improved contrastive loss for preventing performance degradation brought by simple samples. The experiment results on two public datasets and two self-constructed datasets prove our approach outperforms the state-of-art methods significantly.

1 Introduction

Thriving progress in the field of text generative models (TGMs) (Yang et al., 2019; Kenton and Toutanova, 2019; Liu et al., 2019; Keskar et al., 2019; Lewis et al., 2020; Brown et al., 2020; Gao et al., 2021a; Madotto et al., 2021; Ouyang et al., 2022; Touvron et al., 2023; Anil et al., 2023), *e.g.*, ChatGPT[1] and GPT-4 (OpenAI, 2023), enables everyone to produce MGTs massively and rapidly. However, the accessibility to high-quality TGMs is prone to cause misuses, such as fake news generation (Zellers et al., 2019; Yanagi et al., 2020; Huang et al., 2022), product review forging (Adelani et al., 2020), and spamming (Tan et al., 2012), etc. MGTs are hard to distinguish by an untrained human for their human-like writing style (Ippolito et al., 2020) and the excessive amount (Grinberg et al., 2019), which calls for the study of reliable automatic MGT detectors.

[1]https://chat.openai.com

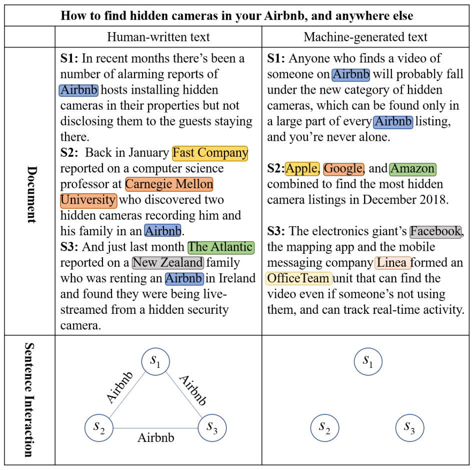

Figure 1: Illustration of sentence-level structure difference between HWT and MGT, the MGT is generated by GROVER (Zellers et al., 2019). HWT is more coherent than MGT as the sentences share more same entities with each other.

Previous works on MGTs detection mainly concentrate on sequence feature representation and classification (Gehrmann et al., 2019; Solaiman et al., 2019; Zellers et al., 2019; He et al., 2023; Mitchell et al., 2023). Recent studies have shown the good performance of automated detectors in a fine-tuning fashion (Solaiman et al., 2019; Mireshghallah et al., 2023). Although these fine-tuning-based detectors have demonstrated their effectiveness, they still suffer from two issues that limit their conversion to practical use: (1) Existing detectors treat input documents as flat sequences of tokens and use neural encoders or statistical features (*e.g.*, TF-IDF, perplexity) to represent text as the dense vector for classification. These methods rely much on the token-level distribution difference of texts in each class, which ignores high-level linguistic representation of text structure. (2) Compared with the enormous number of online texts, annotated dataset for training MGT detectors is a rather low-resource. Constrained by the amount of available annotated data, traditional detectors sustain frustrating accuracy and even collapse during the test stage.

As shown in Fig. 1, MGTs and HWTs exhibit difference in terms of coherence traced by entity consistency. Thus, we propose an entity coherence graph to model the sentence-level structure of texts based on the thoughts of Centering Theory (Grosz and Sidner, 1986), which evaluates text coherence by entity consistency. Entity coherence graph treats entities as nodes and builds edges between entities in the same sentences and same entities among different sentences to reveal the text structure. Instead of treating text as flat sequence, coherence modeling helps to introduce distinguishable linguistic feature at input stage and provides explainable difference between MGTs and HWTs.

To alleviate the low-resource problem in the second issue, inspired by the resurgence of contrastive learning (He et al., 2020; Chen et al., 2020), we utilize proper design of sample pair and contrastive process to learn fine-grained instance-level features under low resource. However, it has been proven that the easiest negative samples are unnecessary and insufficient for model training in contrastive learning (Cai et al., 2020). To circumvent the performance degradation brought by the easy samples, we propose a novel contrastive loss with capability to reweight the effect of negative samples by difficulty score to help model concentrate more on hard samples and ignore the easy samples. Extensive experiments on multiple datasets (GROVER, GPT-2, GPT-3.5) demonstrate the effectiveness and robustness of our proposed method. We also take a small step to explore why GPT-3.5 dataset is overly simple to all the detectors by token importance case study.

In summary, our contributions are summarized as follows:

- **Coherence Graph Construction:** We model the text coherence with entity consistency and sentence interaction while statistically proving its distinctiveness in MGTs detection, and further introduce this linguistic feature at input stage.

- **Improved Contrastive Loss:** We propose a novel contrastive loss in which hard negative samples are paid more attention for improving detection accuracy of challenging sample.

- **Outstanding Performance:** We achieve state-of-art performance on four MGT datasets in both low-resource and high-resource setting. Experimental results verify the effectiveness and robustness of our model.

2 Related Work

Machine-generated Text Detection. Machine-generated texts, also named deepfake or neural fake texts, are generated by language models to mimic human writing style, making them perplexing for humans to distinguish (Ippolito et al., 2020). Generative models like GROVER (Zellers et al., 2019), GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), and emerging GPT-3.5-turbo (also known as ChatGPT) have been evaluated on the MGT detection task and achieve good results (Gehrmann et al., 2019; Mireshghallah et al., 2023). Bakhtin et al. (2019) train an energy-based model by treating the output of TGMs as negative samples to demonstrate the generalization ability. Deep learning models incorporating stylometry and external knowledge are also feasible for improving the performance of MGT detectors (Uchendu et al., 2019; Zhong et al., 2020). Our method differs from the previous work by analyzing and modeling text coherence as a distinguishable feature and emphasizing performance improvement under low-resource scenarios.

Coherence Modeling. For generative models, coherence is the critical requirement and vital target (Hovy, 1988). Previous works mainly discuss two types of coherence, local coherence (Mellish et al., 1998; Althaus et al., 2004) and global coherence (Mann and Thompson, 1987). Local coherence focus on sentence-to-sentence transitions (Lapata, 2003), while global coherence tries to capture comprehensive structure (Karamanis and Manurung, 2002). Our method strives to represent both local and global coherence with inner- and inter-sentence relations between entity nodes.

Contrastive Learning. Contrastive learning in NLP demonstrates superb performance in learning token-level embeddings (Su et al., 2022) and sentence-level embeddings (Gao et al., 2021b) for natural language understanding. With in-depth study of the mechanism of contrastive learning, the hardness of samples is proved to be crucial in the training stage. Cai et al. (2020) define the

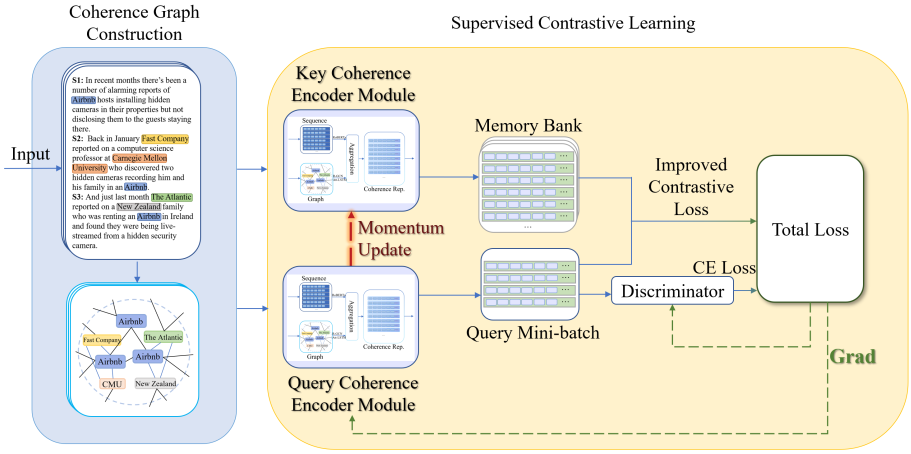

Figure 2: Overview of CoCo. Input document is parsed to construct a coherence graph (3.1), the text and graph are utilized by a supervised contrastive learning framework (3.2), in which coherence encoding module is designed to encode and aggregate to generate coherence-enhanced representation (3.2.3). After that, we employ a MoCo based contrastive learning architecture in which key encodings are stored in a dynamic memory bank (3.2.4) with improved contrastive loss to make final prediction (3.2.5).

dot product between the queries and the negatives in normalized embedding space as hardness and figured out the easiest 95% negatives are insufficient and unnecessary. Song et al. (2022) propose a difficulty measure function based on the distance between classes and apply curriculum learning to the sampling stage. Differently, our method pays more attention to hard negative samples for improving the detection accuracy of challenging samples.

3 Methodology

The workflow of CoCo mainly contains coherence graph construction, and supervised contrastive learning discriminator, and Fig. 2 illustrates its overall architecture.

3.1 Coherence Graph Construction

In this part, we illustrate how to construct coherence graph to dig out coherence structure of text by modeling sentence interaction.

According to Centering Theory (Grosz and Sidner, 1986), coherence of texts could be modeled by sentence interaction around center entities. To better reflect text structure and avoid semantic overlap, we proposed to construct an undirected graph with entities as nodes. Specifically, we first implement the ELMo-based NER model TagLM (Peters et al., 2017) with the help of NER toolkit AllenNLP[2] to extract the entities from document. An relation $< inter >$ is constructed between same entities in different sentences and nodes within same sentences are connected by relation $< inner >$ for their natural structure relevance. Formally, the mathematical form of coherence graph's adjacent matrix is defined as follows:

$$\boldsymbol{A}_{ij} = \begin{cases} 1 & rel \ \langle \texttt{inner} \rangle \quad v_{i,a} \neq v_{j,b}, a = b \\ 1 & rel \ \langle \texttt{inter} \rangle \quad v_{i,a} = v_{j,b} \ , a \neq b \\ 0 & rel \ \text{None} \qquad\quad others \end{cases}$$

where $v_{i,a}$ represents i-th entity in sentence a, which is regarded as node in coherence graph.

[2]https://demo.allennlp.org/named-entity-recognition

3.2 Supervised Contrastive Learning

3.2.1 Model Overview

The training process is illustrated in Fig. 2. Each entry in the dataset is document with its coherence graph. The entries in training set are sampled randomly into keys and queries. Two coherence encoder modules (CEM) f_k and f_q, are initialized the same to generate coherence-enhanced representation $\boldsymbol{D}_k$ and $\boldsymbol{D}_q$ for key and query. A dynamic memory bank with the size of all training data is initialized to store all key representation and their annotations for providing enough contrastive pairs in low-resource scenario. In every training step, the newly encoded key graphs update memory bank following First In First Out (FIFO) rule to keep it updated and the training process consistent. A novel loss composed of improved contrastive loss and cross-entropy loss ensures the model's ability to achieve instance-level intra-class compactness and inter-class separability while maintaining the class-level distinguishability. A linear discriminator takes query representations as input and generates prediction results. The pseudocode of training process is shown in Appendix A.10.

3.2.2 Positive/Negative Pair Definition

In supervised setting, where we have access to label information, we define two samples with same label as positive pair and that with different labels as negative pair for incorporating label information into training process.

3.2.3 Encoder Design

In this part, we introduce how to initialize node representation and graph neural network structure which is utilized to integrate coherence information into semantic representation of text by propagating and aggregating information from different granularity with an innovated coherence encoder module.

Node Representation Initialization. We initialize the representation of entity nodes with powerful pre-trained model RoBERTa for its superior ability to encode contextual information into text representation.

Given an entity e with a span of n tokens, we utilize RoBERTa to map input document $\boldsymbol{x}$ to embeddings $\boldsymbol{h}(\boldsymbol{x})$. The contextual representation of e is calculated as follows:

$$\boldsymbol{Z}_v = \frac{1}{n}\sum_{i=0}^{n} \boldsymbol{h}(\boldsymbol{x})_{e_i}, \tag{1}$$

where e_i is the absolute position where the i-th token in e lies in the whole document.

Relation-aware GCN. Based on the vanilla Graph Convolutional Networks (Welling and Kipf, 2016), we propose a novel method to assign different weight $\boldsymbol{W}_r$ for inter and inner relation r with Relation-aware GCN. Relation-aware GCN convolute edges of each kind of relation in the coherence graph separately. The final representation is the sum of GCN outputs from all relations. We use two-layer GCN in the model because more layers will cause an overfitting problem under low resources. We define the relation set as R, and the calculation formula is as follows:

$$\begin{gathered}\boldsymbol{H}^{(i+1)} = \sum_{r\in R} \hat{\boldsymbol{A}}\mathrm{ReLU}((\hat{\boldsymbol{A}}\boldsymbol{H}^{(i)}\boldsymbol{W}_r^{(i)})\boldsymbol{W}_r^{(i+1)}), \\ \hat{\boldsymbol{A}} = \tilde{\boldsymbol{D}}^{-\frac{1}{2}}\tilde{\boldsymbol{A}}\tilde{\boldsymbol{D}}^{-\frac{1}{2}},\end{gathered} \tag{2}$$

where $\boldsymbol{H}^{(i)} \in \boldsymbol{R}^{N\times d}$ is node representation in i-th layer. $\tilde{\boldsymbol{A}} = \boldsymbol{A}+\boldsymbol{I}$, $\boldsymbol{A}$ is the adjacency matrix of the coherence graph, $\hat{\boldsymbol{A}}$ is the normalized Laplacian matrix of $\tilde{\boldsymbol{A}}$, $\boldsymbol{W}_r$ is the relation transformation matrix for relation r.

Sentence Representation. Afterward, we aggregate updated node representation from last layer of Relation-aware GCN into sentence-level representation to prepare for concatenation with sequence representation from RoBERTa. The aggregation follows the below rule:

$$\boldsymbol{Z}_{s_i} = \frac{1}{M_i}\sum_{j}^{M_i} \sigma(\boldsymbol{W}_s\boldsymbol{H}_{(i,j)} + \boldsymbol{b}_s), \tag{3}$$

where M_i represents the number of entities in i-th sentence, $\boldsymbol{H}_{(i,j)}$ represents the embedding of j-th entity in i-th sentence, $\boldsymbol{W_s}$ is weight matrix and $\boldsymbol{b}_s$ is bias. All the sentence representations within same document are concatenated as sentence matrix $\boldsymbol{Z}_s$.

Document Representation with Attention LSTM. We design a self-attention mechanism for discovering the sentence-level coherence between one sentence and other sentences, and apply LSTM with the objective to track the coherence in continuous sentences and take the last hidden state of LSTM for aggregated document representation containing comprehensive coherence information. The calculation is described as follows:

$$\boldsymbol{Z}_c = \mathrm{LSTM}(\mathrm{softmax}(\gamma\frac{\mathrm{norm}(\boldsymbol{K})\mathrm{norm}(\boldsymbol{Q})^T}{\sqrt{d_Z}})\boldsymbol{V}), \tag{4}$$

where $\boldsymbol{K}, \boldsymbol{Q}, \boldsymbol{V}$ are linear transformations of $\boldsymbol{Z}_s$ with matrix $\boldsymbol{W}_k, \boldsymbol{W}_q, \boldsymbol{W}_v$, d_Z is the dimension of representation $\boldsymbol{Z}_s$, and γ is a hypergammar-parameter for scaling.

Finally, we concatenate $\boldsymbol{Z}_c$ and the sequence representation $\boldsymbol{h}([\mathrm{CLS}])$ from the RoBERTa's last layer to generate document coherence-enhanced representation $\boldsymbol{D}$.

3.2.4 Dynamic Memory Bank

The dynamic memory bank is created to store as much as key encoding $\boldsymbol{D}_k$ to form adequate positive and negative pairs within a batch. The dynamic memory bank is maintained as a queue so that the newly encoded keys could replace the outdated ones, which keeps the consistency between the key encoding and current training step.

3.2.5 Loss Function

Following the definition of positive pairs and negative pairs above, traditional supervised contrastive

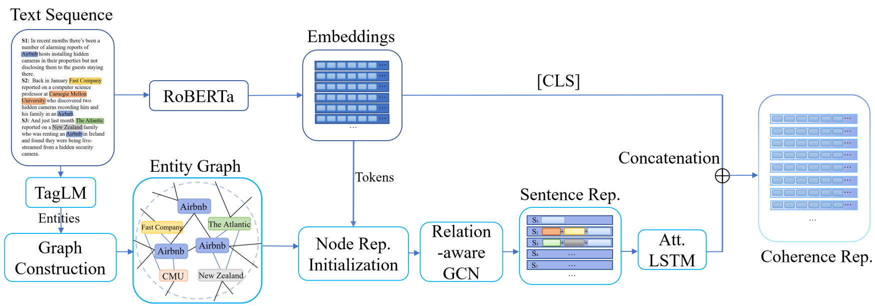

Figure 3: Illustration of CEM which encodes and fuses the coherence graph and text sequence to generate coherence-enhanced representation of document.

loss (Gunel et al., 2021) treats all positive pairs and negative pairs equally.

However, with recognition that not all negatives are created equal (Cai et al., 2020), our goal is to emphasize the informative samples for helping the model to differentiate difficult samples. Thus, we propose an improved contrastive loss which dynamically adjusts the weight of negative pair similarity according to the hardness of negative samples. To be specific, the hard negative samples should be assigned larger weight for stimulating the model to better pull same class together and push different class away. The improved contrastive loss is defined as:

$$
\begin{aligned}
\mathcal{L}_{\text{ICL}} &= \sum_{j=1}^{M} \mathbf{1}_{y_i=y_j} \log \frac{S_{ij}}{\sum_{p \in \mathcal{P}(i)} S_{ip} + \sum_{n \in \mathcal{N}(i)} rf_{in} S_{in}}, \\
rf_{ij} &= \beta \frac{\boldsymbol{D}_q^i \boldsymbol{D}_k^n}{\operatorname{avg}(\boldsymbol{D}_q^i \boldsymbol{D}_k^{1:|\mathcal{N}(i)|})}, \\
S_{ij} &= \exp(\boldsymbol{D}_q^i \boldsymbol{D}_k^j / \tau),
\end{aligned} \tag{5}
$$

where $\mathcal{P}(i)$ is the positive set in which data has the same label with q_i and $\mathcal{N}(i)$ is the negative set in which data has different label from q_i.

Apart from instance-level learning mechanism, a linear classifier combined with cross entropy loss $\mathcal{L}_{\text{CE}}$ is employed to provide the model with class-level separation ability. $\mathcal{L}_{\text{CE}}$ is calculated by

$$
\mathcal{L}_{\text{CE}} = \frac{1}{N} \sum\nolimits_{i=1}^{N} -[y_i log(p_i) + (1-y_i) log(1-p_i)], \tag{6}
$$

where p_i is the prediction probability distribution of i-th sample. The final loss $\mathcal{L}_{\text{total}}$ is a weighted average of $\mathcal{L}_{\text{ICL}}$ and $\mathcal{L}_{\text{CE}}$ as:

$$
\mathcal{L}_{\text{total}} = \alpha \mathcal{L}_{\text{ICL}} + (1-\alpha) \mathcal{L}_{\text{CE}}, \tag{7}
$$

where the hyperparameter α adjusts the relative balance between instance compactness and class separability.

3.2.6 Momentum Update

The parameters of query encoder f_q and the classifier can be updated by gradient back-propagated from $\mathcal{L}_{\text{total}}$. We denote the parameters of f_q as θ_q, the parameters of f_k as θ_k, The key encoder f_k's parameters are updated by momentum update mechanism:

$$
\theta_k \leftarrow \beta \theta_k + (1-\beta)\theta_q, \tag{8}
$$

where the hyperparameter β is momentum coefficient.

4 Experiments

4.1 Datasets

We evaluate our model on the following datasets:

GROVER Dataset (Zellers et al., 2019) is a News-style dataset in which HWTs are collected from RealNews, a large corpus of news from Common Crawl, and MGTs are generated by Grover-Mega (1.5B), a transformer-based news generator.

GPT-2 Dataset is a Webtext-style dataset provided by OpenAI[3] with HWTs adopted from WebText and MGTs produced by GPT-2 XLM-1542M.

GPT-3.5 Dataset is a News-style open-source dataset constructed by us based on the text-davinci-003[4] model (175B) of OpenAI, which is one of the most capable GPT-3.5 models so far and can generate longer texts (maximum 4,097 tokens). The GPT-3.5 model refers to various latest newspapers (Dec. 2022 - Present) whose full texts act as the HWTs part, and the model generates by imitation. We design two subsets: **mixed-** and **unmixed-**provenances, whose details are explained in Appendix A.2.

The statistics of datasets is summarized in Appendix A.1. We randomly sample 500 examples

[3]https://github.com/openai/gpt-2-output-dataset
[4]https://platform.openai.com/docs/models/gpt-3-5

Dataset	GROVER				GPT-2			
Size	Limited Dataset (500 examples)		Full Dataset		Limited Dataset (500 examples)		Full Dataset	
Metric	ACC	F1	ACC	F1	ACC	F1	ACC	F1
GPT2	0.5747 ± 0.0217	0.4394 ± 0.0346	0.8274 ± 0.0091	0.8003 ± 0.0141	0.5380 ± 0.0067	0.4734 ± 0.0182	0.8913 ± 0.0066	0.8839 ± 0.0078
XLNet	0.5660 ± 0.0265	0.4707 ± 0.0402	0.8156 ± 0.0079	0.7493 ± 0.0073	0.6551 ± 0.0083	0.5715 ± 0.0095	0.9091 ± 0.0091	0.9027 ± 0.0111
RoBERTa	0.6621 ± 0.0133	0.5895 ± 0.0231	0.8772 ± 0.0029	0.8171 ± 0.0048	0.8223 ± 0.0088	0.7978 ± 0.0085	0.9402 ± 0.0039	0.9384 ± 0.0044
DualCL	*0.5835 ± 0.0857*	*0.4628 ± 0.1076*	*0.7574 ± 0.0855*	*0.6388 ± 0.1300*	*0.6039 ± 0.1367*	*0.5435 ± 0.0903*	*0.8023 ± 0.1120*	*0.8046 ± 0.1530*
CE+SCL	0.6870 ± 0.0142	0.5961 ± 0.0197	0.8782 ± 0.0044	0.8202 ± 0.0057	0.8355 ± 0.0046	0.8127 ± 0.0067	0.9408 ± 0.0006	0.9390 ± 0.0009
GLTR	0.3370	0.4935	0.6040	0.5182	0.7755	0.7639	0.7784	0.7691
DetectGPT	0.5910	0.4258	0.6142	0.5018	0.7941	0.6982	0.7939	0.7002
CoCo	**0.6993 ± 0.0119**	**0.6125 ± 0.0159**	**0.8826 ± 0.0018**	**0.8265 ± 0.0036**	**0.8530 ± 0.0019**	**0.8410 ± 0.0018**	**0.9457 ± 0.0004**	**0.9452 ± 0.0004**
Dataset	GPT-3.5 Unmixed				GPT-3.5 Mixed			
Size	Limited Dataset (500 examples)		Full Dataset		Limited Dataset (500 examples)		Full Dataset	
Metric	ACC	F1	ACC	F1	ACC	F1	ACC	F1
GPT2	0.9023 ± 0.0095	0.8920 ± 0.0073	0.9917 ± 0.0056	0.9905 ± 0.0042	0.8898 ± 0.0094	0.8914 ± 0.0084	0.9910 ± 0.0046	0.9910 ± 0.0033
XLNet	0.9107 ± 0.0068	0.9037 ± 0.0064	0.9620 ± 0.0043	0.9634 ± 0.0068	0.8925 ± 0.0106	0.8922 ± 0.0089	0.9513 ± 0.0052	0.9505 ± 0.0039
RoBERTa	0.9670 ± 0.0084	0.9681 ± 0.0077	0.9928 ± 0.0035	0.9913 ± 0.0040	0.9565 ± 0.0103	0.9583 ± 0.0092	0.9923 ± 0.0017	0.9901 ± 0.0024
CE+SCL	0.9823 ± 0.0053	0.9703 ± 0.0070	0.9944 ± 0.0023	0.9943 ± 0.0031	0.9628 ± 0.0077	0.9686 ± 0.0062	0.9932 ± 0.0017	0.9905 ± 0.0038
GLTR	0.9255	0.9287	0.9350	0.9358	0.9175	0.9181	0.9210	0.9212
DetectGPT	0.9220	0.8744	0.9245	0.8991	0.8980	0.8814	0.9113	0.9041
CoCo	**0.9889 ± 0.0044**	**0.9791 ± 0.0062**	**0.9972 ± 0.0015**	**0.9957 ± 0.0020**	**0.9701 ± 0.0069**	**0.9735 ± 0.0086**	**0.9932 ± 0.0019**	**0.9937 ± 0.0028**

Table 1: Results of the model comparison. It should be noticed that DualCL is easily affected by random seed, which may be caused by its weakness in understanding long texts. We do not present the experiment results for DualCL on GPT-3.5 dataset because the documents in GPT-3.5 dataset is so long that DualCL completely fails.

as training data for low-resource setting. As for full dataset setting, we utilize all training data. The implementation details are in Appendix A.4.

4.2 Comparison Models

We compare CoCo to state-of-art detection methods to reveal the effectiveness. We mainly divide comparison methods into two categories, **model-based** and **metric-based** methods. The metrics-based methods detect based on specific statistical text-evaluation metrics and logistic regression while the model-based methods learn features via fine-tuning a model.

The **model-based** baselines are as follows.

GPT-2 (Radford et al., 2019), **RoBERTa** (Liu et al., 2019), **XLNet** (Yang et al., 2019) are powerful transformers-based models fine-tuned on the binary classification task, implementing GPT-2 small(124M), RoBERTa-base(110M) and XLNet-base(110M).

CE+SCL (Gunel et al., 2021), a state-of-the-art supervised contrastive learning method in various downstream task. We train the detector with Cross-Entropy loss (CE) and supervised contrastive loss (SCL) calculated within a mini-batch.

DualCL (Chen et al., 2022), a contrastive learning method with the addition of label representations for data augmentation.

The **metric-based** baselines are as follows.

GLTR (Gehrmann et al., 2019), a supporting tool for facilitating humans to recognize MGTs with visual hints. We follow the settings of (Guo et al., 2023) and select the Test-2 feature, which counts the top-k tokens ranking from GPT-2 medium (355M) predicted probability distributions as features for training a logistic regression classifier.

DetectGPT (Mitchell et al., 2023), a contemporaneous metric-based method utilizing the difference of model's log probability after text perturbations. We use T5-3B to perturb texts, and Pythia-12B (Biderman et al., 2023) for scoring in the model. A logistic regression classifier is trained to make predictions.

4.3 Performance Comparison

As shown in Table 1, CoCo surpasses the state-of-the-art methods in MGT detection task by **at least 1.23%** and **1.64%**, **1.75%** and **2.83%** on the GROVER, GPT-2 limited datasets in terms of Accuracy and F1-Score, respectively. And CoCo achieves comparable performance with the most capable detectors in the complete dataset setting. The result indicates the utility of contrastive learning and the rationality of coherence representation.

Moreover, it should be noticed that compared with metric-based methods, model-based methods usually tend to achieve better results. This can be explained because metric-based methods can only concern and regress on a few features, which are over-compressed and under-represented for the detection task. Also, metric-based methods mainly use the pre-trained model for token probability instead of fine-tuning the whole model. And with more training samples involved, the performance of model-based methods improves drastically, while metric-based methods do not benefit much from more training examples. It reveals that logistic regression is not strong enough to take in many texts with diverse semantics. Meanwhile, CoCo outperforms CE+SCL and DualCL regardless of the size of the training set, which suggests the success of improved contrastive loss to solve the performance degradation problem brought by simple negative samples.

We also find GROVER Dataset is the hardest to detect. It is because the GROVER generator is trained in an adversarial heuristic with the objective of deceiving the verifier, which endows the generator with deceptive nature. To our surprise, the GPT-3.5 dataset is overly simple for all detectors. The result is also in accord with conclusions in recent works (Mireshghallah et al., 2023; Chen et al., 2023). We conduct extensive experiments on different self-constructed and published GPT-3.5 datasets generated by a series of prompts, validating this thundering conclusion. The experiment details and results are in Appendix A.3. We also implement experiments and discussions to explore further explanations in Section 4.5.2.

4.4 Ablation Study

To illustrate the necessity of components of CoCo, we conduct ablation experiments on the unbalanced 1,000-example GROVER dataset. The ablation models' structure are as follows:

Model	ACC	F1
CoCo (Plain)	0.7697	0.6428
CoCo (Sentence nodes)	0.7733	0.6379
CoCo (Coherence)	0.7777	0.6463
CoCo (Coherence + LSTM)	0.7787	0.6471
CoCo (Coherence + LSTM + SCL)	0.7827	0.6609
CoCo	0.7843	0.6684

Table 2: Results of ablation study.

CoCo (Plain) removes graph information and encodes only by RoBERTa parts. The model removes contrastive learning and only uses CE loss.

CoCo (Sentence Nodes) treats sentences (instead of entities) as nodes and establishes edges between sentences that share same entities. Node representation is initialized by RoBERTa embedding and mean-pooling operation. Document representation is obtained by one CEM discarding sentence representation and attention LSTM part in Section 3.2.3. Document representation is calculated by mean-pooling operation on sentence node representations. A linear classification head with cross-entropy loss is used for detection.

CoCo (Coherence) incorporates the coherence graph into the representation of document and deploys sentence representation part in Section 3.2.3. The rest are the same with CoCo (Sentence Nodes).

CoCo (Coherence + LSTM) uses attention LSTM for document-level aggregation, and the rest is the same as CoCo (Coherence).

CoCo (Coherence + LSTM + SCL) utilizes the contrastive learning framework, but the loss function is traditional supervised contrastive loss instead of the improved contrastive loss.

As shown in Table 2, coherence information and the contrastive learning framework greatly contribute to the development of model performance, especially in F1-Score. Replacing entity nodes in coherence graph with sentences impairs the detector, which could be caused by semantic overlap between graph representation and text sequence representation. The attention LSTM also plays an important role in preserving coherence information during sentence aggregation. Lastly, the results also shows the advantage of improved contrastive loss over standard supervised contrastive loss.

4.5 Discussion

4.5.1 Model Robustness to Perturbation

To validate the robustness of CoCo to various perturbations, we train CoCo on the GROVER dataset in the low-resource setting and perturb the test set with four different operations: **Delete** (randomly delete tokens in each entry), **Repeat** (randomly select tokens and repeat them twice in the text), **Insert** (add random tokens from the vocabulary of the pre-trained model into random positions in the text), **Replace** (randomly replace tokens with randomly selected tokens from the vocabulary). The perturbation scale is set to 15%. The experiment result is shown in Table 3.

Despite the structural complexity, CoCo keeps

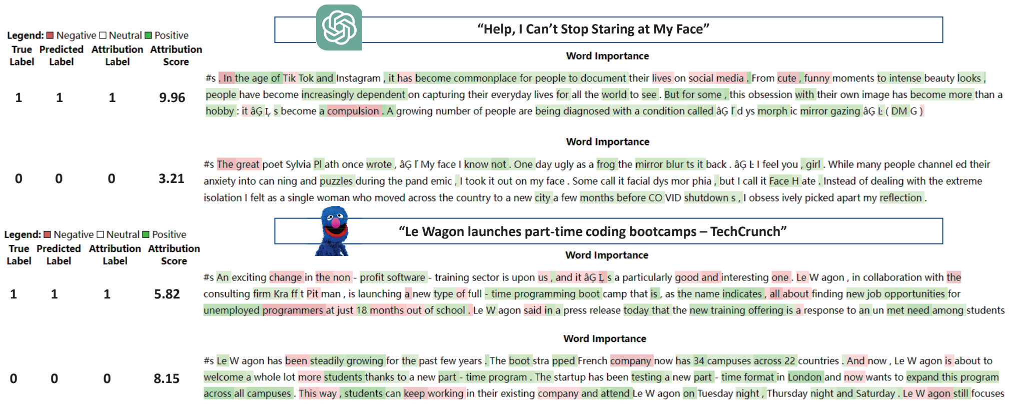

Figure 4: Visualization of token attributions. The first text pair is sampled from GPT-3.5 mixed dataset and the second text pair is from GROVER dataset. The tokens in green represent contributing positively to the predicted label, while those in red contribute negatively. Label "0" represents HWT, and Label "1" represents MGT.

Model	RoBERTa		CoCo	
Metric	Acc	F1	Acc	F1
Original	0.6635	0.5901	**0.6993**	**0.6125**
Delete	0.5736 (-0.0899)	0.5545 (**-0.0356**)	**0.6363** (**-0.0630**)	**0.5703** (-0.0422)
Repeat	0.6320 (-0.0315)	0.5743 (-0.0158)	**0.6732** (**-0.0261**)	**0.6004** (**-0.0121**)
Insert	**0.6325** (**-0.0310**)	0.4881 (**-0.1020**)	0.6286 (-0.0707)	**0.4970** (-0.1155)
Replace	0.5554 (-0.1081)	0.4814 (**-0.1087**)	**0.6367** (**-0.0626**)	**0.5023** (-0.1102)
Average	0.5984 (-0.0651)	0.5246 (**-0.0655**)	**0.6437** (**-0.0556**)	**0.5425** (-0.0700)

Table 3: Model robustness to different perturbations.

outperforming the baseline during perturbations. CoCo's performance fluctuations are as minor as the baseline. And CoCo maintains **4.53%** better in accuracy and **1.79%** better in F1-score on average, which stands for its robustness.

4.5.2 Token Importance in GPT-3.5 Detection

To further investigate the rationale behind the easy-to-detect nature of GPT-3.5 generated texts, we utilize Transformers-Interpret[5], a tool for evaluating feature attribution in predictions based on Integrated Gradients(Sundararajan et al., 2017), for discovering the important tokens in decision-making stage. We fine-tune RoBERTa-base model with a classification head on GPT-3.5 mixed dataset and visualize how tokens in GPT-3.5 mixed test data affect the model predictions. As shown in Fig. 4, we take segments from two text pairs consisting of HWT and its corresponding MGT in GPT-3.5 mixed and GROVER dataset. It could be noticed that consecutive spans in text generated by GPT-3.5 tend to contribute more to the model decision. However, in HWTs, model pays more attention to individual tokens. Following this observation, we infer that with the improvement of model scale, LLMs fit extremely well to the corpus so that it generates more general expressions compared with HWTs, which follows certain patterns (always demonstrated by a span of tokens) that could be expected by fine-tuned models. Thus, barely all the methods show nearly perfect performance on GPT-3.5 dataset.

As for GROVER dataset, more tokens contribute negatively to the model prediction, even if the prediction is correct. This reflects the deceptive nature of GROVER and explains the reason why it is the hardest dataset in our experiment to some extent.

We discuss more topics in Appendix, *e.g.*, the effect of hyper-parameters (A.5), case study (A.6), static geometric analysis on coherence graph (A.7), and exploration on imbalanced data (A.8).

5 Conclusion

In this paper, we propose CoCo, a coherence-enhanced contrastive learning model for MGT detection. We construct a novel coherence graph from document and implement a MoCo-based contrastive learning framework to improve model performance in low-resource setting. An innovative encoder composed of relation-aware GCN and attention LSTM is designed to learn the coherence representation from coherence graph which is further incorparated with sequence representation of document. To alleviate the effect of unnecessary easy samples, we propose an improved contrastive learning loss to force the model to pay more attention to hard negative samples. CoCo outperforms all detection tasks generated by GROVER, GPT-2, and GPT-3.5, respectively, in both low-resource and high-resource settings.

[5] https://github.com/cdpierse/transformers-interpret

Limitations

In this work, we step forward to better distinguishing MGTs under the low-resource setting. However, several limitations still exist for the broader applications of this detector. Firstly, MGTs are easier to generate and collect than HWTs, which may cause an imbalanced label distribution in the dataset. And CoCo literally corrupts in extremely imbalanced data distribution condition, as shown in A.8. Future work could build upon the contrastive learning method of CoCo with innovation on sampling strategy for harsh low-resource and imbalanced data settings. Secondly, our method artificially generates a coherence graph for every entry, which is not efficient for larger datasets. What's more, short text, codes, and mathematical proofs, which are hard to generate coherence graphs, are also limitedly detected by CoCo. More distinctive and easy-to-calculate features are worth exploring for generating distinguishable representations for texts with efficiency while better understanding the essence of TGMs. Thirdly, with instruct-based generation and human-in-loop fine-tuning models prevailing, the strategy and defect of TGMs change slightly but constantly. The entity relation with the same semantic granularity and concretization in this paper would not be enough to detect the high-quality content by TGMs in the future. More generative and adaptive detection models should be considered.

Ethical Considerations

We provide insight into the potential weakness of TGMs and publish GPT-3.5 news dataset. We understand that the discovery of our work can be viciously used to confront detectors. And we understand that malicious users can copy the contents of our GPT-3.5 news dataset to disguise real news and publish them. However, with the purpose of calling for attention to detecting and controlling possible misuse of TGMs, we believe our work will inspire the advance of the stronger detector of MGTs and prevent all potential negative uses of language models.

Our work complies with sharing & publication policy of OpenAI[6] and all data we collect is in public domain and licensed for research purposes.

[6]https://openai.com/api/policies/sharing-publication/

Acknowledgements

We thank all the anonymous reviewers and the area chair for their helpful feedback, which aided us in greatly improving the paper.

References

David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H Nguyen, Junichi Yamagishi, and Isao Echizen. 2020. Generating sentiment-preserving fake online reviews using neural language models and their human-and machine-based detection. In *International Conference on Advanced Information Networking and Applications*, pages 1341–1354. Springer.

Ernst Althaus, Nikiforos Karamanis, and Alexander Koller. 2004. Computing locally coherent discourses. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 399–406.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *arXiv preprint arXiv:1906.03351*.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*.

Roi Blanco and Christina Lioma. 2011. Graph-based term weighting for information retrieval. *Information Retrieval*, 15:54–92.

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.

Tiffany Tianhui Cai, Jonathan Frankle, David J Schwab, and Ari S Morcos. 2020. Are all negatives created equal in contrastive instance discrimination? *arXiv preprint arXiv:2010.06682*.

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

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. 2020. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Ramakrishnan. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *arXiv preprint arXiv:2305.07969*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021a. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116.

Nir Grinberg, Kenneth Joseph, Lisa Friedland, Briony Swire-Thompson, and David Lazer. 2019. Fake news on twitter during the 2016 us presidential election. *Science*, 363(6425):374–378.

Barbara J Grosz and Candace L Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational linguistics*, 12(3):175–204.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2021. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2023. Mgtbench: Benchmarking machine-generated text detection. *arXiv preprint arXiv:2303.14822*.

Xiaochen Hou, Peng Qi, Guangtao Wang, Rex Ying, Jing Huang, Xiaodong He, and Bowen Zhou. 2021. Graph ensemble learning over multiple dependency trees for aspect-level sentiment classification. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2884–2894.

Eduard H Hovy. 1988. Planning coherent multisentential text. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pages 163–169.

Binxuan Huang and Kathleen M Carley. 2019. Syntax-aware aspect level sentiment classification with graph attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5469–5477.

Kung-Hsiang Huang, Preslav Nakov, Yejin Choi, and Heng Ji. 2022. Faking fake news for real fake news detection: Propaganda-loaded training data generation. *ArXiv*, abs/2203.05386.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1808–1822.

Nikiforos Karamanis and Hisar Maruli Manurung. 2002. Stochastic text structuring using the principle of continuity. In *Proceedings of the International Natural Language Generation Conference*, pages 81–88.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

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*.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *ACL*, volume 3, pages 545–552. Citeseer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Xien Liu, Xinxin You, Xiao Zhang, Ji Wu, and Ping Lv. 2020. Tensor graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8409–8416.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *ArXiv*, abs/2110.08118.

Fragkiskos D Malliaros and Konstantinos Skianis. 2015. Graph-based term weighting for text categorization. In *Proceedings of the 2015 IEEE/ACM international conference on advances in social networks analysis and mining 2015*, pages 1473–1479.

William C Mann and Sandra A Thompson. 1987. *Rhetorical structure theory: A theory of text organization*. University of Southern California, Information Sciences Institute Los Angeles.

Chris Mellish, Alistair Knott, Jon Oberlander, and Mick O'Donnell. 1998. Experiments using stochastic search for text planning. In *Proceedings of the 9th International General Workshop*, pages 98–107. ACL Anthology.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.

Fatemehsadat Mireshghallah, Justus Mattern, Sicun Gao, Reza Shokri, and Taylor Berg-Kirkpatrick. 2023. Smaller language models are better black-box machine-generated text detectors. *arXiv preprint arXiv:2305.09859*.

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*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*.

Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1756–1765.

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.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Xiaohui Song, Longtao Huang, Hui Xue, and Songlin Hu. 2022. Supervised prototypical contrastive learning for emotion recognition in conversation. *arXiv preprint arXiv:2210.08713*.

Yixuan Su, Fangyu Liu, Zaiqiao Meng, Tian Lan, Lei Shu, Ehsan Shareghi, and Nigel Collier. 2022. Tacl: Improving bert pre-training with token-aware contrastive learning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2497–2507.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.

Enhua Tan, Lei Guo, Songqing Chen, Xiaodong Zhang, and Yihong Zhao. 2012. Spammer behavior analysis and detection in user generated content on social networks. In *2012 IEEE 32nd International Conference on Distributed Computing Systems*, pages 305–314. IEEE.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Peter D Turney. 2002. Learning to extract keyphrases from text. *arXiv preprint cs/0212013*.

Adaku Uchendu, Jeffrey Cao, Qiaozhi Wang, Bo Luo, and Dongwon Lee. 2019. Characterizing man-made vs. machine-made chatbot dialogs. In *TTO*.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*.

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. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2001–2016.

Feng Wang and Huaping Liu. 2021. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2495–2504.

Max Welling and Thomas N Kipf. 2016. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*.

Yuta Yanagi, Ryohei Orihara, Yuichi Sei, Yasuyuki Tahara, and Akihiko Ohsuga. 2020. Fake news detection with generated comments for news articles. In *2020 IEEE 24th International Conference on Intelligent Engineering Systems (INES)*, pages 85–90. IEEE.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 7370–7377.

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.

Wanjun Zhong, Duyu Tang, Zenan Xu, Ruize Wang, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Neural deepfake detection with factual structure of text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2461–2470.

A Appendix

A.1 Basic Statistics of Datasets

Dataset		Train	Valid	Test
GROVER	HWT	5,000	2,000	8,000
	MGT	5,000	1,000	4,000
GPT-2	HWT	25,000	5,000	5,000
	MGT	25,000	5,000	5,000
GPT-3.5 Unmixed	HWT	3,454	1,000	1,000
	MGT	3,454	1,000	1,000
GPT-3.5 Mixed	HWT	3,032	1,000	1,000
	MGT	3,032	1,000	1,000

Table 4: Basic statistics of datasets.

A.2 Details of GPT-3.5 Dataset

GPT-3.5 Dataset for CoCo is our latest dataset for the MGT detection task. There are two subsets in the self-made dataset for easy analysis of the impact of provenance and writing styles: unmixed- and mixed provinces. We use the text-davinci-003 model of OpenAI to generate MGT examples. The maximum length of HWTs is 1024 tokens, and the target generation length is set as 1024 tokens. Here is an example of the MGT data.

"title": "On Eve of World Cup, FIFA Chief Says, ‘Don’t Criticize Qatar; Criticize Me.’",
"text": "DOHA, Qatar. The president of world soccer’s governing body on Saturday sought to blunt mounting concerns about the World Cup in Qatar with a strident defense of both the host country’s reputation and FIFA’s authority over its showpiece championship....... Citing statistics, history and even childhood to bolster his case, he at one point likened his own experience as a redheaded child of immigrants to Switzerland to the assimilation problems of gays in the Middle East, and defended the laws, customs and honor of the host country.",
"authors": ["Tariq Panja"],
"publish_date": "2022-11-19 00:00:00",
"source": "The New York Times",
"url": "https://www.nytimes.com/2022/11/19/sports/soccer/world-cup-gianni-infantino-fifa.html"

And the following data shows the corresponding MGT in the dataset.

"title": "On Eve of World Cup, FIFA Chief Says, ‘Don’t Criticize Qatar; Criticize Me.’",
"text": "The 2022 FIFA World Cup in Qatar is fast approaching, and its organizing committee’s president, Gianni Infantino, is speaking out about the lingering criticism of the country hosting the event. ...... he said. “It is a once-in-a-lifetime opportunity for the region to show the world its values and aspirations, and it is vital that this event is seen as a celebration of football and a celebration of the region.””,
"authors": "machine",
"source": "The New York Times",
"matched_hwt_id": 202,
"label": "machine""

A.2.1 Human Written Texts

Unmixed Subset. The HWTs of the unmixed subset are all from The New York Times[7] to exclude the impact of writing style. The time span of our data is Nov 1, 2022 - Dec 25, 2022, making sure that no pre-trained model has learned them. We develop the crawler based on news-crawler[8].

Mixed Subset. The HWTs of the mixed subset come from various sources, listed as Table 5. The time span of the data is Jan 1, 2022 - Jan 7, 2023. We develop the crawler based on Newspaper3k[9].

The dataset is specifically designed for MGTs detection and improving generation models. The contents of dataset are obtained from official news websites and the names of indicidual people are not mentioned maliciously. And we strongly reject using our dataset to create offensive content or peek at private information.

[7] https://www.nytimes.com/
[8] https://github.com/LuChang-CS/news-crawler
[9] https://github.com/codelucas/newspaper

Name	Website
Kotaku	https://kotaku.com
The Daily World	https://www.thedailyworld.com
CNN	https://edition.cnn.com
BBC	https://www.bbc.com
NBC News	https://www.nbcnews.com
Reuters	https://www.reuters.com
Huffpost	https://www.huffpost.com
Pando	http://pandodaily.com
Yahoo	https://news.yahoo.com
Sun Times	https://chicago.suntimes.com/news
Sfgate	https://www.sfgate.com
New Republic	https://newrepublic.com
Time	https://time.com
Pcmag	http://www.pcmag.com
CNBC	https://www.cnbc.com/world/
News	https://www.news.com.au/
The Atlantic	https://www.theatlantic.com/latest/

Table 5: Data sources for the mixed subset.

A.2.2 Machine Generated Texts

As the GPT-3.5 and ChatGPT model need prompts to generate, we write hints for the generation models to generate texts that meet our news-style long text generation. The hints format is as follows, and the content is related to HWTs.

```
Write a news more than 1000 words.
The news is written by {Authors} from {Source}
in {date}. Title is {title}.
```

A.3 GPT-3.5 Dataset Generated by Different Prompts and Experiment Results

To further validate the conclusion that GPT-3.5 generated texts are easier to detect, we utilize CNN news as reference and design different prompts for GPT-3.5 generation. The principle is to provide as more information as possible to GPT-3.5 for alleviating the possible gap in semantics and in length.

Keywords as Prompt (KP). We extract the keywords and entities with GPT-3.5-turbo and provide examples in original news to form the prompt for generation. The prompt format is as follows.

Example prompt for generation.

```
"role": "system", "content": "Extract all
the keywords, entities, and examples in the
following passage:"
"role": "user", "content": {text}
```

Example prompt for generation.

```
Generate a news passage.
The news is written by {Authors} from {Source}
in {date}.
Title: Lionel Messi isn't expected to be back
with PSG until early January after World Cup
success
Keywords: exploring, mountains, space, Poorna
Malavath, Kavya Manyapu, NASA, Mount Everest,
Project Shakthi, girls' education, Ladakh,
India, virgin peak, climbing, altitude sickness,
safety, motivation, empowerment, education,
gender gap, Mount Aconcagua, sponsorship.
Entities: CNN, Poorna Malavath, Kavya Manyapu,
NASA, Mount Everest, Project Shakthi, Ladakh,
India, Mount Aconcagua, South America, World
Bank.
Examples: designing space suits, youngest
ever woman to summit Mount Everest, climbed a
6,012m virgin peak, raise money to fund girls'
education, difficulties of climbing a virgin
peak, experiences of altitude sickness, purpose
of Project Shakthi, India's Right to Education
Act, sponsorship for underprivileged school
children, scaling Mount Aconcagua, expanding
sponsorship globally.
The target length for generation is 731 tokens.
Add as much details and examples as you can.
News:
```

Summary as Prompt (SP). We employ GPT-3.5-turbo to summarize the original texts. The compression ratio is set to $[0.3, 1.0]$, which means the summary is required to be longer than 0.3 of the length of original text and shorter than whole original text. The generated summary is used as prompt and the format is as follows:

```
Generate a news based on the following
abstract:
Paris Saint-Germain's coach Christophe Galtier
has stated that Lionel Messi is not expected
to join the team until early January as he is
spending time in Argentina following the World
Cup. Kylian Mbappé, Neymar Jr. and Achraf
Hakimi, who played for their respective national
teams at Qatar 2022, could return to the team as
long as they are physically and mentally fit...
The news is written by Matias Grez from CNN in
2022-12-28 00:00:00.
Title: Lionel Messi isn't expected to be back
with PSG until early January after World Cup
success
News:
```

Outline as Prompt (OP). We also outline the skeleton of original texts by GPT-3.5-turbo and feed the outline into GPT-3.5 text-davinci-003. The prompt format is as follows:

Prompt for extraction.

```
"role": "system", "content": "Write a
hierarchical multi-point outline for the
paragraph."
"role": "user", "content": {text}
```

Example prompt for generation.

```
News Title: There's a shortage of truckers, but
TuSimple thinks it has a solution: no driver
needed
The news is written by Jacopo Prisco, CNN from
CNN in 2021-07-15 02:46:59.
Outline:
I. TuSimple's plan for fully autonomous truck
tests
A. Reliability of software and hardware needs
to improve
B. Fully autonomous tests without human safety
driver planned by end of year
C. Results will determine if company can launch
trucks by 2024
D. 7,000 trucks reserved in US alone
II. TuSimple's competition
A. ...
Add more details and examples.
News:
```

We first remove the HWTs that do not have desired length (i.e., 200-1024 tokens). And we take half of the selected HWTs as references to formulate different prompts mentioned above and feed it into GPT-3.5 to get MGTs. The MGTs are sampled by Gaussion Distribution of their lengths. To avoid the possible label leakage brought by text length, we directly filter the no-reference HWTs according to the Gaussion Distribution of MGT lengths.

Besides the self-constructed datasets, we also utilize the published GPT-3.5 dataset TuringBench benchmark (abbraviate as GPT-3.5 (TB)) (Uchendu et al., 2020) to validate the deceptiveness of GPT-3.5. The statistics of datasets we use is in Table 6.

Dataset		Train	Valid	Test	# of tokens
GPT-3.5(KP)	HWT	446	148	148	427.96 ±45.49
	MGT	446	148	148	403.88 ±75.63
GPT-3.5(SP)	HWT	446	148	148	427.96 ±45.49
	MGT	446	148	148	415.72 ±66.54
GPT-3.5(OP)	HWT	446	148	148	427.96 ±45.49
	MGT	446	148	148	429.34 ±78.62
GPT-3.5(TB)	HWT	5,964	975	1915	236.17 ±72.96
	MGT	5,507	894	1763	147.29 ±70.15

Table 6: Statistics of GPT-3.5 datasets.

We conduct experiments with 3 random seeds and the average results are shown in Table 7. Counterintuitively, even if we elaborate the prompts and eliminate the length difference between MGTs and HWTs, the detection results are still superior, even on outdated baselines like GPT-2. The conclusion might be counterintuitive, but texts generated by the most advanced and popular GPT-3.5 model are the easiest to detect.

A.4 Implementation Details

This part mentions the implementation details and hyper-parameter settings of all the methods in the experiment. To imitate the situation of low data-resources, we randomly sample 500 entries from the datasets as limited dataset (positive:negative=1:1), which will test models together with the complete datasets. And we conduct experiments on 10 different seeds and report the average test accuracy, F1-Score, and standard deviation only for model-based methods because metric-based methods would not be affected by random seeds.

We use RoBERTa base model to initialize the embedding of our representation and optimize the model using AdamW (Loshchilov and Hutter, 2018) optimizer with a 0.01 weight decay. We set the initial learning rate to 10^{-5} and the batch size to 8 for all datasets based on experiences.

We utilize packages, namely transformers, pytorch, and allennlp to implement CoCo. And the GPT-3.5 datasets and ChatGPT case is generated by OpenAI API and websites. We spend $300 for API costs, including development and final generation costs. We train and do experiments on 8 NVIDIA A100 GPUs on 2 Ubuntu-based servers. The total budget for training 20 epochs, dev, and testing on the GROVER dataset is 2.5 hours. On GPT-2 dataset is 12 hours, and on GPT-3.5 dataset is 1.5 hours. We will publish our code and dataset recently.

A.5 Effect of Hyper-Parameters

A.5.1 Contrastive Learning Parameters

We evaluate the influence of contrastive learning hyper-parameters α and τ with experiments on different combinations of them. The result is shown in Fig. 5. Considering the discovering that smaller τ leads to better hard negative mining ability (Wang and Liu, 2021), we select α from $\{0.1, 0.2, ..., 0.9\}$ and τ from $\{0.1, 0.2, 0.3\}$. We find that the extreme α value causes the performance degradation and the best hyper-parameter combination is $\alpha, \tau = 0.6, 0.2$. Our analysis is that large α forces the model to concentrate on the instance-level contrast and small α lets class separation objective take control. Both will reduce the generalization performance of the detector on test set.

Dataset	GPT-3.5 (KP)		GPT-3.5 (SP)		GPT-3.5 (OP)		GPT-3.5 (TB)	
Metric	ACC(val/test)	F1(val/test)	ACC (val/test)	F1 (val/test)	ACC (val/test)	F1 (val/test)	ACC (val/test)	F1 (val/test)
GPT2	0.9914/0.9916	0.9916/0.9918	0.9890/0.9893	0.9885/0.9889	0.9925/0.9928	0.9923/0.9924	0.9884/0.5422*	0.9880/0.6335*
RoBERTa	0.9946/0.9950	0.9950/0.9952	0.9935/0.9941	0.9933/0.9937	0.9946/0.9943	0.9942/0.9940	0.9962/0.6406*	0.9960/0.7273*
CoCo	0.9955/0.9950	0.9942/0.9945	0.9938/0.9941	0.9936/0.9940	0.9942/0.9943	0.9942/0.9943	0.9966*	0.9970*

Table 7: Experiment of different detectors on different GPT-3.5 Dataset. * : The great performance difference between validation set and test set on GPT-3.5 (TB) are because the test set randomly sample 50% of the words of each article in the dataset (Uchendu et al., 2021). We do not test CoCo on GPT-3.5 (TB) for the reason that such operation greatly influences the coherence in texts. We provide an example of this in Table 8.

GPT-3.5 (TB)	**GPT-3.5 (OP)**
'.video : morne morkel press conference * cricbuzz.video : england cricbuzz.bevan leads scotland 's 21-man squad for their first ever test match against pakistan in edinburgh icc.chris rogers retires after champions trophy defeat : australian cricketer announces international retirement the sun.icc super eight teams : odi ranking results.bahrain host oman on sunday kitply hans vohra gold cup gulf today.icc results.new zealand series history : india v new zealandyazan mohsen qawasma : how bahrain caught	Recent changes to key international indexes have resulted in the unprecedented exclusion of Russian stocks at a "zero" price, causing further losses in Moscow's already-dismal stock exchange. This exclusion has made Russia no longer an option for investors, prompting a shift to other emerging markets.\n\nThe dramatic shift was made in early March, when FTSE Russell and MSCI announced the removal of Russian stocks from their indexes due to the country's escalating economic and geopolitical problems. Shortly after, the Moscow Exchange suspended trading, sending ripples through the market.\n\nThe possible default on Russian debt has Western investors further reconsidering their investments in Russia...

Table 8: A comparison example between texts in test set of GPT-3.5 (TB) and GPT-3.5 (OP). The GPT-3.5 (TB) text shows great disorder while GPT-3.5 (OP) text is neat.

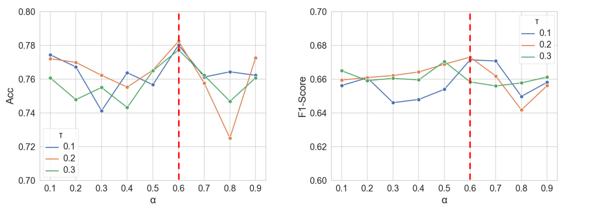

Figure 5: Effect of parameters α and τ on model performance.

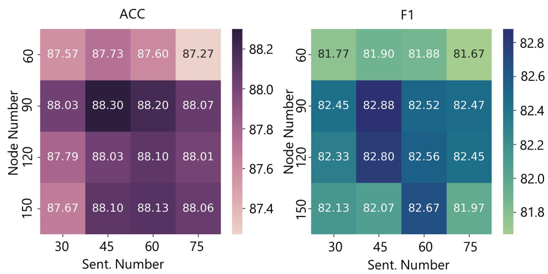

Figure 6: Performance of CoCo with different graph parameters.

A.5.2 Graph Parameters

We further investigate the effect of max node number and max sentence number on model performance. The result is shown in Fig. 6. We select max node number from $\{60, 90, 120, 150\}$ and max sentence number from $\{30, 45, 60, 75\}$. The detector performs best when max node number is 90 and max sentence number is 45. The experiment results prove that the large node and sentence number are not necessary for the improvement of detection accuracy. We infer that even though setting large node and sentence number includes more entity information, excessive nodes bring noise to the model and impair the distinguishability of coherence feature.

A.6 Case Study

In this subsection, we conduct a case study with HWT and MGT produced by sensational ChatGPT with the same metadata. As illustrated in Fig. 7, we parse two news as coherence graphs. And we observe that although ChatGPT expresses fluently, it is not coherent from the perspective of coherence graph. Hence, CoCo utilizes the distinctive coherence feature and makes correct predictions. However, RoBERTa fails to discriminate the MGT without noticing the coherence difference. This reflects even the most popular and advanced language model could suffer from weak coherence and be detected by CoCo.

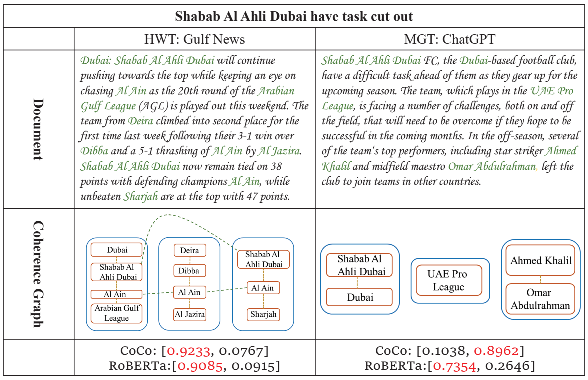

Figure 7: An illustration for case study of our method. Entities in documents are colored green. The blue solid box indicates the sentence. The orange dashed lines are inner edges and green dashed lines are inter edges. Numbers in red indicate the probability of predicted label.

A.7 Static Geometric Analysis on Coherence Graph

We have witnessed performance enhancement by applying the graph-based coherence model to the detection model, but how does the coherence graph help detection? In this subsection, we apply static geometric features analysis to coherence graph we construct to evaluate the distinguishable difference between HWTs and MGTs with explanation. In the following discussion, we take the dataset of GROVER into the analysis. Some basic metrics of data and the corresponding graph are shown in Table 9.

Metric	HWT	MGT
Sample Num.	4994	4991
Avg. Num. of Token	463.2	456.0
Avg. Num. of Vertex	43.60	32.37
Avg. Num. of Edge	107.4	65.44

Table 9: Basic metrics of texts and corresponding graphs.

Though HWTs and MGTs have approximately the same number of tokens in every text, coherence graph for HWTs has larger scale than MGTs' with **34.7%** more vertexes and **64.1%** more edges, which shows that HWTs have more complex semantic relation structures than MGTs.

A.7.1 Degree Distribution

Semantically, degree of coherence graph measures the co-occurrence and TF-IDF feature of keywords. Moreover, degree distribution shows global coherence because high-degree nodes devote to the main topic and low-degree nodes are the extension.

Metric	Avg. Degree
HWT	2.980
MGT	2.591

Table 10: Average of degree (whole dataset).

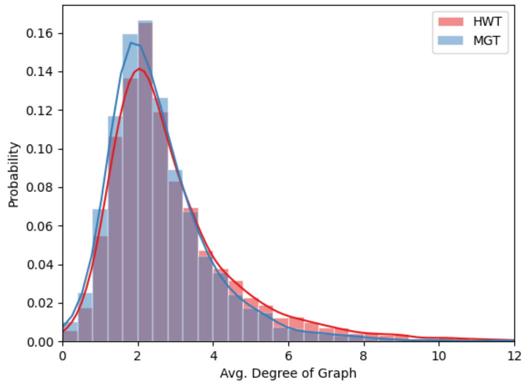

Figure 8: Distribution of average degree of graphs.

As shown in Table 10, The degree of the graph representation of HWTs is **2.980**, which is **15.0%** larger than MGTs (**2.591**), which shows disparities of MGTs to form coherent interaction between sentences. Fig. 8 measures the distribution of each graph's average nodes' degree, showing that the distribution of HWTs has a longer tail than MGTs.

Furthermore, we analyze the distinguishability of degree features when impacted by other factors. One most considerable influences is the style and genre of different provenance. We chose around 60 articles from The Sun[10] and Boston[11]. Then we use GROVER to mimic their style to generate similar topic news. Fig. 9 shows the degree distribution of HWTs and MGTs of both provenances.

We use the Jensen–Shannon divergence to evaluate the similarity of the degree distribution. The JS-divergence of MGTs mimicking The Sun and Boston is **0.029**, while the JS-divergence of MGTs and HWTs in Boston is **0.050**, in The Sun is **0.061**. The apparent gap shows that degree distribution can robustly detect MGTs and HWTs when impacted by provenance differences.

[10] https://www.thesun.co.uk/
[11] https://www.boston.com/

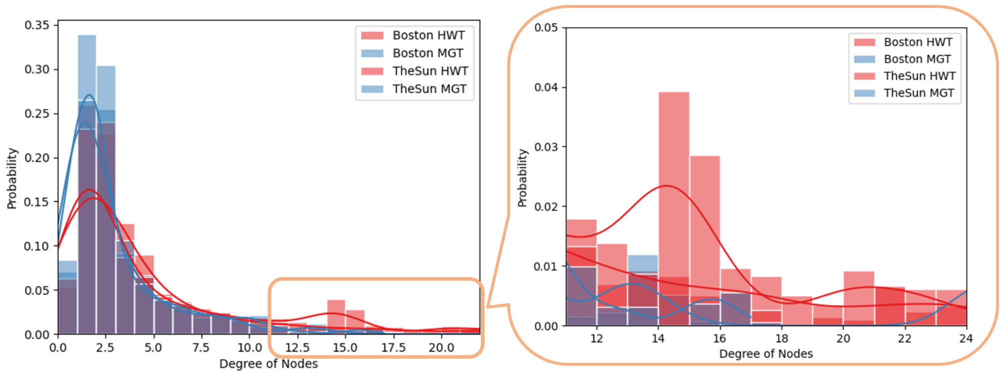

Figure 9: Distribution of degree with different provenance.

A.7.2 Aggregation

Aggregation is a shared metric for complex networks and linguistics, depicting how closely the whole is organized around its core. We propose two metrics to evaluate the aggregation of graph-based text representation in our coherence model, the size of the largest connected subgraph and the clustering coefficient.

In our representation, not all sentences have entities related to others. Hence the graph is an unconnected one. The average number of nodes in subgraphs of MGTs is **4.49** and of HWTs is **4.84**. We propose that the size of the largest connected subgraph shows the contents which are closely organized around the topic. Moreover, the size of graphs may be an unfair factor, so we use the portion of nodes in the largest connected subgraph to reflect its size. The average portion in HWTs is **0.6725** and in MGTs is **0.6458**. Fig. 10 shows the distribution of the portion of graphs, and HWTs distribute more high-portion ones than MGTs.

The clustering coefficient represents how nodes tend to cluster. For the entities of texts, clustering evaluates how the author narrates around the central theme. The larger the clustering coefficient is, the tighter the semantic structure is. The average cluster coefficient of the graphs of HWTs is **0.2213** and of MGTs is **0.1983**, HWTs is **11.6%** better than MGTs. Fig. 11 shows the distribution.

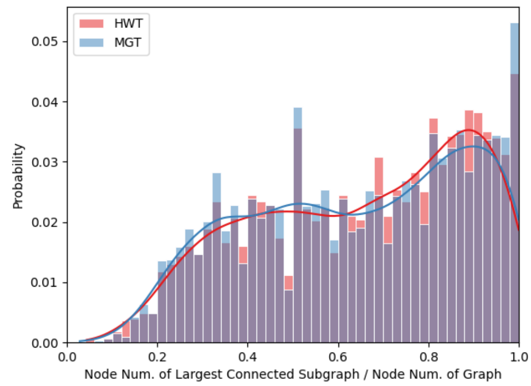

Figure 10: Portion of the largest connected subgraph.

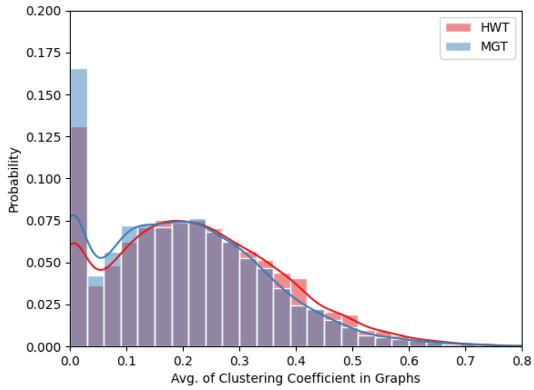

Figure 11: Distribution of clustering coefficient.

A.7.3 Core & Degeneracy

The degeneracy of a graph is a measure of how sparse it is, and the k-core is the subgraph corresponding to its significance in the graph. We propose that, in our graph representation, the degeneracy process of graphs equals summarizing texts semantically. The maximum of core-number shows the complexity of hierarchical structure in texts. Furthermore, the distribution of the core-number reflects the overall sparse and is a graph-perspective N-gram module. Based on experiments, the average core-number of HWTs is **5.772** while MGTs with **4.458**. HWTs are **29.5%** ahead. Fig. 12 is the distribution of the core-number.

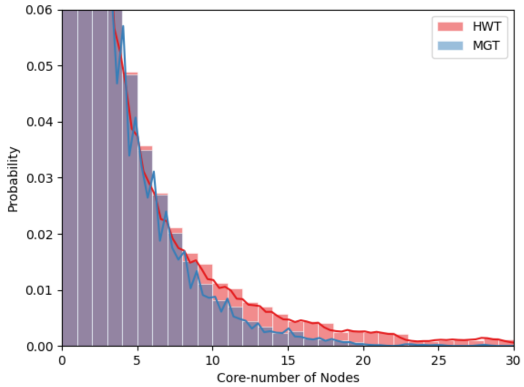

Figure 12: Core-number of nodes in graphs

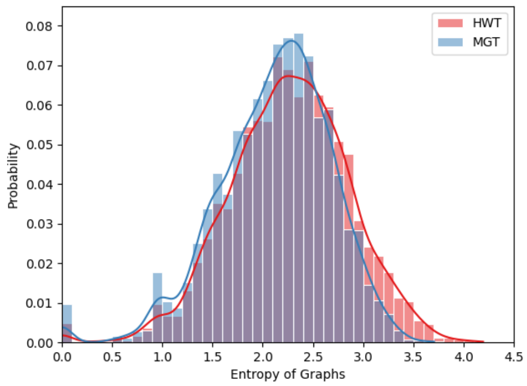

Figure 13: Structure entropy of graphs

A.7.4 Entropy

Entropy is a scientific concept to measure a state of disorder, randomness, or uncertainty. The well-known Shannon entropy is the core of the information theory, measuring the self-information content. For the graph data, network structure entropy defined as the following can examine the information amount of the graph structure.

$$Entropy = -\sum_{i=1}^{N} I_i \ln I_i = -\sum_{i=1}^{N} \frac{k_i}{\sum_{j=1}^{N} k_j} \ln(\frac{k_i}{\sum_{j=1}^{N} k_j}), \tag{9}$$

where I_i is the information content represented by the degree distribution, N is the number of nodes, and k_i is the degree of the i-th node.

Global coherence, from our perspective, equals refining more information inside the semantic structure of the whole text, which matches to structure entropy of our graph representation. From our experiments, the structure entropy of HWTs (2.263) is **6.80%** larger than MGTs (2.119), which means HWTs obtain more structured information because their semantic information is globally organized.

We show the network structure entropy distribution in Fig. 13.

A.8 Exploration on Imbalanced Data

Imbalanced distribution in data is another crucial limitation in the task of MGTs detection, which is similar to the low resource limitation. It is imaginable that, with the development of generation technology, MGTs will overwhelmingly dominate low-quality articles since they are easier and faster to generate than human writing. The detection model will face training resources with MGTs as the main part and HWTs as the small part. We test the current models in the imbalanced limitation and find the dramatic decline in accuracy when the ratio of HWTs is less than 30%, as shown in the Fig. 14. The test is based on the 10% GROVER dataset.

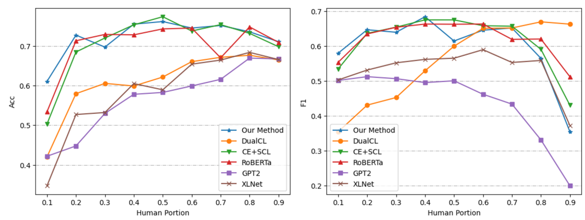

Figure 14: Model comparison results on DL dataset with 9 different human-generated text portions.

All models show poor performance at low HWTs ratios. With a percentage of HWTs of 0.1 (only 100 HWTs in the training set in this case), most of the models have an accuracy below 50%, which performance is close to random and reflects intolerance for extreme cases. Besides, we find that a high proportion of HWTs also cause a decrease in F1 score to some extent.

A.9 Related Work: Graph-based Text Representation

Graph-of Words (GoW) Model (Turney, 2002; Mihalcea and Tarau, 2004) is a type graph representation method in which each document is represented by a graph, whose nodes correspond to terms and edges capture co-occurrence relationships between terms. Using GoW, keywords can be extracted by retaining the document graph (Turney, 2002). Thus, graph representation is sensible to apply in tasks like information retrieval (Blanco and Lioma, 2011), categorization (Malliaros and Skianis, 2015) and sentiment classification tasks (Huang and Carley, 2019; Hou et al., 2021).

Most models enhance classification or detection performance by combining graph representation with neural networks. Text-GCN (Yao et al., 2019) first builds a single large graph for the whole corpus, followed by Tensor-GCN (Liu et al., 2020) with tensor representation. Also, the relation between words varies, and should be treated as different edges. CoCo matches keywords PLM embedding to nodes and sentence representation, considers dealing inner- and inter-sentence relation differently in GCN, and merges the structure graph and flat sequence representation to predict accurately.

A.10 Pseudocode of CoCo

Algorithm 1 Algorithm of CoCo

Input: Input X, consisting of documents D and corresponding coherence graph G, hyper-parameters such as the size of dynamic memory bank M and batch size S, labels Y

Output: A learned model CoCo, consisting of key encoder f_k with parameters θ_k, query encoder f_q with parameters θ_q, classifier f_c with parameters θ_c

Initialize $\theta_k = \theta_q$, θ_c
Initialize dynamic memory bank with $f_k(x_1, x_2 ... x_M)$, where x_i is randomly sampled from X.
Freeze θ_k
$epoch \leftarrow 0$
while $epoch \leq epoch_{\text{max}}$ **do**
  $n \leftarrow 0$
  while $n \leq n_{\text{max}}$ **do**
    Randomly select batch $\boldsymbol{b}_k$, $\boldsymbol{b}_q$
    $\boldsymbol{D}_q = f_q(\boldsymbol{b}_q)$, $\boldsymbol{D}_k = f_k(\boldsymbol{b}_k)$
    $\widehat{p} = f_c(\boldsymbol{D}_q)$
    Calculate $\mathcal{L}_{ICL}$ with equation 5, calculate $\mathcal{L}_{CE}$ with equation 6, calculate $\mathcal{L}_{total}$ with equation 7
    Backward on $\mathcal{L}_{total}$ and update θ_q, θ_c based on AdamW gradient descent with an adjustable learning rate
    Momentum update θ_k with equation 8
    Update dynamic memory bank $queue$ with $enqueue(queue, \boldsymbol{D}_k)$, $dequeue(queue)$
    $k \leftarrow k + 1$
  end while
  if Early stopping **then**
    break
  else
    $epoch \leftarrow epoch + 1$
  end if
end while
return A trained model CoCo