EasyChair Preprint

№ 14130

AI Authorship Verification: an Ensembled Approach

Benjamin Ostrower, Jacob Wessell and Abhinav Bindal

July 25, 2024

AI Authorship Verification: An Ensembled Approach*

Notebook for the Voight-Kampff Generative AI Authorship Verification 2024 Lab at CLEF 2024

Benjamin Ostrower[1,*,†], Jacob Wessell[2,†] and Abhinav Bindal[3,†]

[1]Georgia Institute of Technology, 225 North Avenue, Atlanta, 30332, United States

Abstract

Our method for detecting AI written text creates an ensembled method combining 3 techniques: a Fine-tuned RoBERTa, Neural Depedency Graph, and a factual coherence graph. These methods output their logits for prediction, these are then concatenated into an xgboost classifier.

Keywords

Dependency Graph, AI Text Detection, GCN, RoBERTa

1. Introduction

With advancements in technologies, it is becoming increasingly difficult to distinguish between AI and human-authored texts. This has broad societal and ethical impacts and necessitates the creation of better techniques to determine text authorship.

To tackle this challenge, PAN 2024 lab [1] to be hosted at CLEF 2024 has posed a task for Generative AI authorship verification of text [2]. The software submissions for this task were made as easy-to-reproduce docker containers on TIRA experimental platform [3].

2. Related Work

AI text detection is a field that has grown considerably in recent years coinciding the the explosion in text generation. Other approaches have centered around teasing out patterns of probability curvature of the token outputs[4]. Training classifier heads on top of pretrained LLMs [5]. Testing the likelihood of a word relative to a ranked list of likely outputs of an LLM [6].

3. Methodology

3.1. Dataset

To train our models, we obtained the following public datasets from Hugging Face:

- GPT-wiki-intro [7]
- HF_NabeelShar
- HF_artem
- HF_dmitva

These datasets were concatenated to create the final dataset consisting of 2.7M entries. This dataset was partitioned into training, validation and test dataset using 80%/10%/10% split ratio.

CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France

*You can use this document as the template for preparing your publication. We recommend using the latest version of the ceurart style.

*Corresponding author.

†These authors contributed equally.

benjaminostrower@gmail.com (B. Ostrower); jwessell6@gatech.edu (J. Wessell); abindal6@gatech.edu (A. Bindal)

3.2. Initial Analysis

We performed the following analysis on the dataset.

- Word count distribution for texts with ≤ 500 words
- Number of unique words used in the text (total length ≤ 500 words)
- Number of non-stop words used in the text (total length ≤ 500 words)

These investigations revealed that statistically, the texts generated by human and AI were very similar to each other, requiring us to employ more complex strategies.

3.3. Baseline Model

For training purposes, we sampled 10,000 human and AI texts each from the dataset randomly. Then we did an 80-20 split between the training and the test dataset.

1. We used `TfidfVectorizer` from `sklearn.feature_extraction.text` library with `max_features = 5000` and English stop words to vectorize training and test sets.

$$\text{TF-IDF}_{td} = \text{TF}_{td} \times \log \frac{N}{\text{DF}_t}$$

   Where TF_{td} is the frequency of term t in document d, N is the total number of documents, and DF_t is the number of documents containing term t.
2. We used PCA decomposition from `sklearn.decomposition` library with 100 components to reduce the TF-IDF vectors.
3. Finally, we applied `XGBClassifier` from `xgboost` library to train our baseline model to perform classification. We evaluated the fit of the model based on accuracy metric.

We were able to achieve 87% accuracy for this baseline model.

Although we were able to achieve 87% accuracy on the reduced dataset, it was observed that the performance of this model on a completely unknown dataset was not reliable. Hence, we developed more models for robust prediction of the authorship. This could be attributed to overfitting on the test dataset in the previous model.

3.4. Ensemble Method

Our method consisted of 3 separate models ensembled together. Our three models consisted of a fine-tuned BERT, a graph neural network trained on the dependency graph representation of input documents, and another graph neural network trained on the factual coherence graph of each sentence. After pre-training each classifier, the models were ensembled together and an XGBoost model was used as the prediction head.

3.5. Fine-Tuned RoBERTa Model

In this approach, the text was initially tokenized using `RobertaTokenizerFast` function in the `transformer` library. The text was then classified using `RobertaForSequenceClassification` function using weights from the pre-trained RoBERTa model. For this phase of the training, the text was fed into the model one by one.

The weights of the base RoBERTa model were fine-tuned using Low Rank Adaptation (LoRA) technique [8]. This technique reduced the number of trainable parameters to about 1% of the total parameters thus greatly reducing the computation resources required for training this model.

Tuning was performed on the following hyperparameters of the model:

- LoRA Rank (r): {1, 2, 10, 20}
- LoRA α: {0.5, 1, 5, 10}
- Learning Rate: {1e-2, 1e-3, 1e-4}

The model was trained for 10 epochs. We achieved an accuracy of 95.66% on the validation dataset.

3.5.1. Dependency Parsing

In natural language processing, the dependency parse of a sentence aims to visually represent the syntactic and grammatical structure of a sentence by mapping the dependencies between words. Such dependencies include categories such as direct objects or the subject of a verb. A directed graph naturally represents this structure, and we felt that converting our documents into a collection of such graphs would make structural features about the document readily available to the neural network.

The first step in this pipeline was to convert raw text documents into dependency graphs. In this step, we limited the length of the documents to 500 words and skipped any sentence with less than 3 words. The NLP library SpaCy provides functionalities for generating the dependencies for each word in a sentence. So we simply looped over each sentence in each document and made note of the dependencies between each word in the sentence as well as the type of dependency. These represented the edges and edge attributes in our graph respectively. For the node features, we used Wikipedia2Vec's 100 dimensional pre-trained embeddings to convert words into meaningful representations. Once these graphs were created, they could be converted into the necessary format required by PyTorch-Geometric to create our dataset. One small addition we made was to add a special edge type: root. Root connected the root of each sentence in a document (the root is typically the main verb although not always). This is because the dependency parse is a sentence-level structure while we are concerned with classifying documents. Thus without our special root connection, the message-passing operations employed by the downstream graph neural network would be unable to propagate across sentences in the same document.

With our dataset in hand, we were quickly able to train a GNN model to distinguish AI and human-written documents. We experimented with standard Graph convolutional networks as well as the Graph Attention Network and GIN models provided by PyTorch-Geometric. The best iterations of these models achieved accuracy of .95 on a dataset consisting of examples from Wiki Intros and the Nabeel Shar dataset. However, this performance did not translate well to the data provided for the competition. We believe this can be attributed to our models overfitting to the training data. For example, our model used Wikipedia2Vec as its embeddings so this naturally lends itself to performing well on the Wiki Intros dataset. Additionally, many of the human-written examples in Nabeel Shar are very poorly written, making distinguishing between the AI and Human generated texts very easy (when trained on only Nabeel Shar, the model very quickly achieved nearly 100% accuracy).

3.5.2. Factual Coherence

This implementation was based off the paper by [9]. The method implements a graph convolutional neural network based off the named entities in a text with a classifier head to discriminate between AI and human text.

There are 3 main extractions to be made from each document.The texts are first processed by utilizing Spacy and NLTK to tokenize a document into sentences and parse out any Named Entities. Secondly, each sentence is passed through a BertforNextSentencePredictionModel to obtain NextSentencePrediction scores for each sentence in a document (this helps to model the coherencecy from one sentence to the next and serves as a weight for how much credence to place on entities from that sentence). thirdly the entire text (up to 400 tokens) is also passed through a RoBERTa Model to obtain the Classification token (CLS) as a sort of semantic representation of the text itself. The entities obtained from the first step are embedded using RoBERTa and concatenated with their corresponding wiki2vec embedding (all 0's if that entity is not found).

To create the graph of entities two conditions for creating an edge are followed: firstly if the entities appear together in the same sentence (intra-sentence). Secondly if when creating an inter-sentence edge their cosine similiarity between embeddings is greater than 0.9. Some named entities would have different embedding lengths than others - to remedy this we take the average so that each entity embedding is a 1x886 vector (786 for the RoBERTa, 100 for the wiki2vec).

The model first uses a GCN to create graph enhanced representations of these nodes. This adjacency

matrix is first normalized by the degree matrix then passed through a variable number of convolutions to create these new representations:

$$H_e^{(i+1)} = \sigma(AH_e^{(i)}W_i)$$

These enhanced entities are then substituted into the original list they were tracked in. Each sentence may have a variable number of entities so the entities are averaged through several linear and ReLu layers to create an averaged entity representation in each sentence.

$$y_i = \frac{1}{N_i}\sum_{j=0}^{N_i}\sigma(W_s H(i,j) + b_s)$$

Each document may have a dynamic number of sentences so these averaged sentence entities are then passed through an LSTM. The LSTM output then creates a sumproduct with the NSP scores:

$$D = \sum_{j=1}^{s} S(j-1,j) * [y_{e_{j-1}}, y_{e_j}]$$

Finally this output,D, is concatenated with the CLS token to create the final representation before being passed through to the classifier head. Several different ablations were run with and without the wikipedia2vec embeddings,number of convolutions. Optimal results were with 10 convolutions and the wiki2vec embeddings achieving a best validation accuracy (on a combined set of equal parts human/ai across the aforementioned datasets) of 65

4. Results

While our method did not obtain a submission, we did achieve some preliminary results on test datasets provided by the competition. Our ensembled method was able to reach a 61% accuracy comprised of a bootstrapped competition supplied dataset of PANs Human and Bard-generated text.

Acknowledgments

Thank you to the DS@GT CLEF team for their support. Special thanks to Anthony Miyaguchi for putting together this team for the competition.

References

[1] A. A. Ayele, N. Babakov, J. Bevendorff, X. B. Casals, B. Chulvi, D. Dementieva, A. Elnagar, D. Freitag, M. Fröbe, D. Korenčić, M. Mayerl, D. Moskovskiy, A. Mukherjee, A. Panchenko, M. Potthast, F. Rangel, N. Rizwan, P. Rosso, F. Schneider, A. Smirnova, E. Stamatatos, E. Stakovskii, B. Stein, M. Taulé, D. Ustalov, X. Wang, M. Wiegmann, S. M. Yimam, E. Zangerle, Overview of PAN 2024: Multi-Author Writing Style Analysis, Multilingual Text Detoxification, Oppositional Thinking Analysis, and Generative AI Authorship Verification, in: L. Goeuriot, P. Mulhem, G. Quénot, D. Schwab, L. Soulier, G. M. D. Nunzio, P. Galuščáková, A. G. S. de Herrera, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Fifteenth International Conference of the CLEF Association (CLEF 2024), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2024.

[2] J. Bevendorff, M. Wiegmann, J. Karlgren, L. Dürlich, E. Gogoulou, A. Talman, E. Stamatatos, M. Potthast, B. Stein, Overview of the "Voight-Kampff" Generative AI Authorship Verification Task at PAN and ELOQUENT 2024, in: G. Faggioli, N. Ferro, P. Galuščáková, A. G. S. de Herrera (Eds.), Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, CEUR Workshop Proceedings, CEUR-WS.org, 2024.

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

[4] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, C. Finn, Detectgpt: Zero-shot machine-generated text detection using probability curvature, in: International Conference on Machine Learning, PMLR, 2023, pp. 24950–24962.

[5] I. Solaiman, M. Brundage, J. Clark, A. Askell, A. Herbert-Voss, J. Wu, A. Radford, G. Krueger, J. W. Kim, S. Kreps, et al., Release strategies and the social impacts of language models, arXiv preprint arXiv:1908.09203 (2019).

[6] S. Gehrmann, H. Strobelt, A. M. Rush, Gltr: Statistical detection and visualization of generated text, arXiv preprint arXiv:1906.04043 (2019).

[7] Aaditya Bhat, Gpt-wiki-intro (revision 0e458f5), 2023. URL: https://huggingface.co/datasets/aadityaubhat/GPT-wiki-intro. doi:10.57967/hf/0326.

[8] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, LoRA: Low-rank adaptation of large language models, in: International Conference on Learning Representations, 2022. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

[9] W. Zhong, D. Tang, Z. Xu, R. Wang, N. Duan, M. Zhou, J. Wang, J. Yin, Neural deepfake detection with factual structure of text, arXiv preprint arXiv:2010.07475 (2020).