Stacking the Odds: Transformer-Based Ensemble for AI-Generated Text Detection

Duke Nguyen **Khaing Myat Noe Naing** **Aditya Joshi**
University of New South Wales, Sydney, Australia
{duke.nguyen, khaingmyatnoenaing}@student.unsw.edu.au, aditya.joshi@unsw.edu.au

Abstract

This paper reports our submission under the team name 'SynthDetectives' to the ALTA 2023 Shared Task. We use a stacking ensemble of Transformers for the task of AI-generated text detection. Our approach is novel in terms of its choice of models in that we use accessible and lightweight models in the ensemble. We show that ensembling the models results in an improved accuracy in comparison with using them individually. Our approach achieves an accuracy score of 0.9555 on the official test data provided by the shared task organisers.

1 Introduction

Transformer (Vaswani et al., 2017) is a sequence-to-sequence model that has enabled the training of large language models (LLMs). LLMs such as GPT enable text generation in response to user-defined prompts, allowing for wide applicability. As a result, they have proliferated into several aspects of society, both for good and for bad. Text generated from LLMs, when used unethically, can have several detrimental implications: they can cause widespread fake news, dispense away with all notions of academic honesty and authorship, and threaten to replace human-generated information with AI-generated data at large.

Motivated by these existential concerns, many models have been developed to distinguish AI-generated content from human's. ALTA is participatory in tackling this issue by announcing the ALTA 2023 Shared Task, whose goal is to build 'automatic detection systems that can discriminate between human-authored and synthetic text generated by Large Language Models (LLMs)' (Molla et al., 2013). The text comes from a variety of sources in terms of domains (e.g. medical, law), and source model (e.g. GPT-X, T5). Technically, participating teams are required to build an automated system to solve a binary classification task, distinguishing between human and AI-generated text. Models are evaluated based on robustness and accuracy. There is no requirement on the efficiency and run-time performance. Our team participated in the said shared task. The code is available here[1]. We stack multiple Transformer-based models in an ensemble and show that the ensemble performs better than the individual models. In this paper, we will discuss existing works in the domain, our analysis of the original training data, our proposed pipeline and architecture, our experimental results, and suggested future work.

2 Related Work

AI-generated text detection has a long history. The sources of our AI-generated text are LLMs, which constrain our task to 'authorship attribution (AA) for neural texts', also known as Neural Text Detection (NTD). It is a subclass of the task of binary classification (and sometimes multi-class, when we are detecting the source model). We will summarise briefly the current literature in this domain. Our main source comes from two major surveys by Jawahar et al. (2020) and Uchendu et al. (2023). The latter classifies automated NTD as follows:

Stylometric attribution detects Neural Text Generator (NTG) using ensembles of classical machine learning (ML) models trained on stylometric features such as LIWC (Linguistic Inquiry & Word Count), POS tags, n-grams, Readability score, WritePrints, Empath. These models work best on a small dataset. However, as we increase the data size, they are outperformed by deep learning models rapidly.

Deep learning: GLoVe-based attribution: GLoVe (Pennington et al., 2014) is an unsupervised learning algorithm that aggregates global word-word co-occurrence statistics from text to

[1] https://github.com/dukeraphaelng/synth_detectives

build word representation. GLoVe-based models use these embeddings with RNN and LSTM, which was considered SOTA before BERT (Devlin et al., 2018).

Deep learning: Energy-based attribution: Energy-based models (EBMs) (LeCun et al., 2007) are 'un-normalized generative models' using some energy function to generate high-quality data by modelling the probability distribution of the training data. Adapted for NTD (Bakhtin et al., 2019), they perform well on unseen data, however, they do not scale as well, and are very expensive to train.

Deep learning: Transformer-based attribution: is Transformer-based models fine tuned to perform NTD. These models surpass stylometric and GLoVe-based models and are cheaper than EBMs. RoBERTa and BERT are two models that frequently achieve high performance on NTD benchmarks (Uchendu et al., 2023). Other Transformer-based models that are used in NTD include ELECTRA, XLNet, and DeBERTa. These inspire our choice of weak learners.

Statistical attribution: was developed to combat top-p and top-k decoding strategies which Transformers are not well-equipped against. It has been shown that 'human language is stationary and ergodic as opposed to neural language' (Varshney et al., 2020) suggesting the validity of this approach. Four different algorithms have been proposed which detect AI-generated text through statistical distributions. These are: GLTR (Gehrmann et al., 2019), MAUVE (Pillutla et al., 2021), Distribution detector (Gallé et al., 2021), and DetectGPT (Mitchell et al., 2023), the last three of which perform competitively.

Hybrid attribution: is ensembles using several previously described detectors. These include TDA-based detector (Kushnareva et al., 2021), which extracts attention matrices of BERT's word representations and process them through TDA-based methods as features for a logistic regression model, Fingerprint detector (Diwan et al., 2021), which ensembles fine-tuned RoBERTa embeddings and CNN classifier), FAST (Zhong et al., 2020), which uses RoBERTa with a Graph Neural Network), and CoCo (Liu et al., 2022), a coherence-based contrastive learning model. Our work is an ensemble-based approach to the task. However, we use an ensemble of Transformer models.

id	text	label
0	'Have you ever heard of the Crusades? A time in which Christians went on a 200 year rampage throughout Europe and on their path to Isreal in which they slaughtered innocent people in the name of your God?'	1
4	'The Circuit Court of Appeals of New Jersey had jurisdiction of the controversy between these parties, and its decree was affirmed. But as the court had jurisdiction under the original act of Congress, the jurisdiction in this case was also, under the act of Congress, a bar to the suit.'	0

Table 1: Samples from the training set.

3 Dataset

Three subsets of the dataset are presented: training, validation, and testing. The training set contains 18,000 entries, and the validation and testing each contains 2,000 entries. Evaluation is based on the testing set which was not released until the testing phase of the competition. The training set contains three columns 'id', 'text', and 'label' (1 if human-generated, 0 if AI-generated). The validation and testing set each contains two columns 'id', and 'text'. Samples of the training set are shown in Table 1.

When analyzing the dataset, we find that the AI-generated and human-generated text is evenly split into 9000-9000 entries respectively in the training set. We also find that the average word count per text is relatively low. The mean length is in the 34-35 range in the three subsets, with a standard deviation in the 26.7-27.9 range, a maximum of 172-193, and a minimum of 1, making this a short sequence task.

To find the main domains of the text, we remove all stop words from each set and find the frequency of n-gram phrases from the cleaned corpus, and pick the top-k elements from each set. We look at the n-gram range of $(3, 4)$, with $k = 10$. We find that overwhelmingly all the phrases are in the domain of law across the three sets. The following list is the union of the three sets with the above configuration: {'court of appeals', 'of the court', 'of the united', 'of the united states', 'opinion of the', 'the court of', 'the court of appeals', 'the district court', 'the opinion of', 'the united states'}.

4 Approach

Our approach uses a stacking ensemble of classifiers (as shown in Figure 1) to perform our training, validation, and testing. A stacking ensemble of classifiers acts similarly to a weighted voting classifier. Our choice of architecture is inspired by Maloyan et al. (2022), which achieved high performance in the RuATD Shared Task 2022 on Artificial Text Detection in Russian (Shamardina et al., 2022).

We train each weak classifier using the above dataset split, and then we concatenate the raw predictions on the training set together and feed them to the meta-learner. We use a simple Logistic Regressor as our meta-learner. Our criteria for picking models are ease of use, short-sequence-task-based models, and variety in model architecture. We also choose only encoder-only models, since they are built for regression/ classification tasks, and we can conveniently extract the [CLS] token from their last hidden state to perform Logistic Regression. As a result, we use ALBERT, ELECTRA, RoBERTa and XLNet as the Transformer-based models.

To optimise the training cycle, we tokenise the entire dataset (with the respective model's tokeniser), and pass them through their respective pre-trained model to obtain the [CLS] token from the last hidden state. We consider this to be our dataset and do our splitting, training, and testing on this processed dataset. To train, we pass the [CLS] token through a single fully connected layer, with the input dimension equivalent to the model's [CLS]'s dimension, and the output dimension of 2, then we softmax the output. After fine-tuning the weak models, we perform inference on the training split and concatenate the predictions which are fed for the meta-learner to train.

5 Experiment Setup

5.1 Setup

We do not perform any data preprocessing on the dataset. We have a train-validation-test split of 0.8, 0.1, 0.1. All training was done on Google Cloud Platform's Vertex Colab GPU for GCE usage on NVIDIA A100 (40 GB).

5.2 Pipeline

For both our weak learners and our meta-model, we use the AdamW optimiser with the default settings, i.e. $lr = 0.001, \beta = (0.9, 0.999), \epsilon = 1e - 08, weight\_decay = 0.01$. All models are trained with $epochs = 300$ and $batch\_size = 128$.

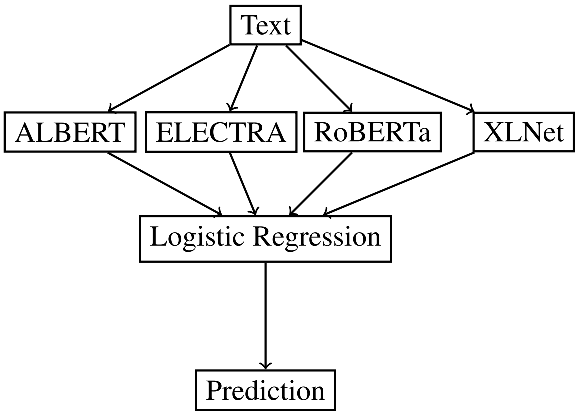

Figure 1: Our Stack Ensembling Architecture for AI-generated text detection.

All models in the ensemble are pre-trained models available on HuggingFace (as of 25th October 2023). For each model, we include their architecture name and the unique HuggingFace model identifier associated with their pre-trained weights.

ALBERT (albert-base-v2) (Lan et al., 2020) is a modification of BERT (Devlin et al., 2018) which reduces its memory consumption and increases the training speed by repeating layers split among groups and splitting the embedding matrix into smaller matrices, whilst being more performative than BERT in GLUE, RACE, and SQuAD.

ELECTRA (google/electra-small-discriminator) (Clark et al., 2020) is another modification of BERT that changes the pretraining objective, as inspired by GAN where ELECTRA acts as the discriminator which predicts whether a token in a randomly masked text is original or generated by the generator (which we train simultaneously). This approach makes ELECTRA perform comparably to larger models whilst using a lot less compute.

RoBERTa (roberta-base) (Liu et al., 2019) optimises BERT in four aspects of training: using full-sentences without Next Sentence Prediction (NSP) loss, with dynamic masking, with larger mini-batches, and with a larger byte-level Byte-Pair Encoding (BPE).

XLNet (xlnet-base-cased) (Yang et al., 2020) uses a generalised autoregressive pretraining method that maximises the 'expected likelihood over all permutations of the input sequence factorisation order' enabling bidirectional contexts and overcoming BERT's pretrain-finetune discrepancy due to neglecting masked positions dependency. XLNet also builds on Transformer-XL (Dai et al., 2019), outperforming BERT on 20 tasks.

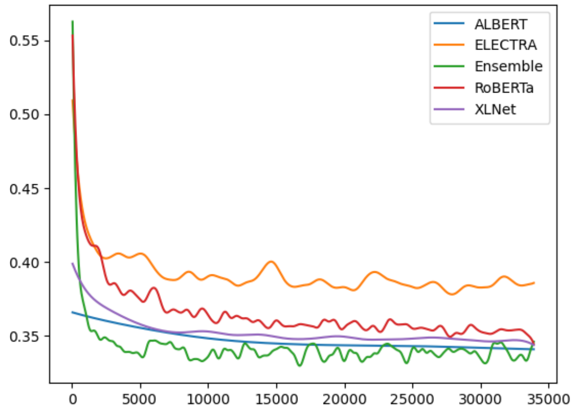

Figure 2: Training Loss.

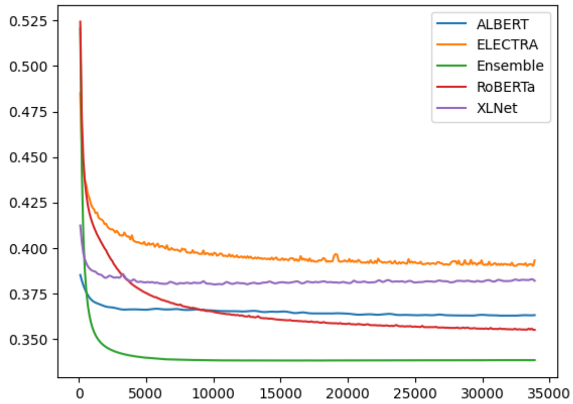

Figure 3: Validation Loss.

Model	Accuracy
ELECTRA	0.9311
XLNet	0.9361
ALBERT	0.9567
RoBERTa	0.9572
Ensemble	0.9694

Table 2: Model Accuracy on the Test Set.

6 Results

Figure 2 shows our smoothed training loss using the Gaussian kernel (since the original training loss displays too wide short-cycle variation, obfuscating the overall trend), Figure 3 shows our validation loss which follows a similar pattern. Table 2 shows our accuracy on the test set as described in the experiment setup. Among our weak learners, RoBERTa performs the best, followed by ALBERT, XLNet, and finally ELECTRA. As expected, our meta-model (Ensemble) outperforms even RoBERTa by more than 0.012. The final testing accuracy model ranking is reflected in the validation loss, and to a lesser extent in the training loss. This agrees with much of the literature indicating that RoBERTa is the best learner in AI-generated text prediction (Jawahar et al., 2020). We also note that XLNet and ALBERT start with extremely low loss, suggesting their pre-training procedure might be conducive to AI-generated text detection.

Finally, the ALTA shared task organisers provided us with a shared task test set *i.e.*, the official test set. We achieve an accuracy of 0.9555 with our stacking ensemble on the official test set.

7 Conclusion & Future Work

In this paper, we describe our system for the ALTA Shared Task 2023. We show how an ensemble of Transformer-based models can be combined using a logistic regression classifier to predict if a text was generated by AI. We achieved an accuracy of 0.9555 using a stacking ensemble of basic encoder-only Transformer models.

Our work presents a novel approach to ensemble Transformer-based models to approach the ALTA shared task. However, this work identifies several potential directions for future work. Ensembling models usually benefit from a variety of learners specialised in different types of inputs. We only implemented an ensemble of Transformer classifiers, but it would be beneficial to integrate other non-Transformer-based weak learners as detailed in Section 2. Especially useful would be to integrate contrastive learning in our training procedure. In addition, it would also be useful to perform data augmentation which can help generalise the model. One suggested technique is ‘text continuation’, where given a human-generated text, we slice the first n words and have an LLM finish the sentence. Furthermore, the scope of the shared task does not imply the possibility of an adversarial attack. It has been shown that the RoBERTa detector can be attacked easily through misspelling (Wolff and Wolff, 2022). It would also be helpful to build detectors that are resilient in this regard.

Acknowledgements

We thank Jake Renzella and the School of Computer Science and Engineering at the University of New South Wales, Sydney for providing us with GPU cloud credits, without which we would not be able to run our experiments.

References

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*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Nirav Diwan, Tanmoy Chakravorty, and Zubair Shafiq. 2021. Fingerprinting fine-tuned language models in the wild. *arXiv preprint arXiv:2106.01703*.

Matthias Gallé, Jos Rozen, Germán Kruszewski, and Hady Elsahar. 2021. Unsupervised and distributional detection of machine-generated text. *arXiv preprint arXiv:2111.02878*.

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

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: A critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Laida Kushnareva, Daniil Cherniavskii, Vladislav Mikhailov, Ekaterina Artemova, Serguei Barannikov, Alexander Bernstein, Irina Piontkovskaya, Dmitri Piontkovski, and Evgeny Burnaev. 2021. Artificial text detection via examining the topology of attention maps. *arXiv preprint arXiv:2109.04825*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and Fujie Huang. 2007. Energy-based models. *Predicting structured data*, 1(0).

Xiaoming Liu, Zhaohan Zhang, Yichen Wang, Yu Lan, and Chao Shen. 2022. Coco: Coherence-enhanced machine-generated text detection under data limitation with contrastive learning. *arXiv preprint arXiv:2212.10341*.

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.

Narek Maloyan, , Bulat Nutfullin, Eugene Ilyshin, and and. 2022. DIALOG-22 RuATD generated text detection. In *Computational Linguistics and Intellectual Technologies*. RSUH.

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*.

Diego Molla, Haolan Zhan, Xuanli He, and Qiongkai Xu. 2013. Overview of the 2023 alta shared task: Discriminate between human-written and machine-generated text. In *Proceedings of the 21st Annual Workshop of the Australasian Language Technology Association (ALTA 2023)*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

Tatiana Shamardina, Vladislav Mikhailov, Daniil Cherniankii, Alena Fenogenova, Marat Saidov, Anastasiya Valeeva, Tatiana Shavrina, Ivan Smurov, Elena Tutubalina, and Ekaterina Artemova. 2022. Findings of the the RuATD shared task 2022 on artificial text detection in russian. In *Computational Linguistics and Intellectual Technologies*. RSUH.

Adaku Uchendu, Thai Le, and Dongwon Lee. 2023. Attribution and obfuscation of neural text authorship: A data mining perspective. *SIGKDD Explor. Newsl.*, 25(1):1–18.

Lav R Varshney, Nitish Shirish Keskar, and Richard Socher. 2020. Limits of detecting text generated by large-scale language models. In *2020 Information Theory and Applications Workshop (ITA)*, pages 1–5. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Max Wolff and Stuart Wolff. 2022. Attacking neural text detectors.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

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. *arXiv preprint arXiv:2010.07475*.