TueSents at SemEval-2024 Task 8: Predicting the Shift from Human Authorship to Machine-generated Output in a Mixed Text

Valentin Pickard and Hoa Do
Seminar für Sprachwissenschaft
Eberhard Karls Universität Tübingen, Germany
{valentin.pickard, hoa.do}@student.uni-tuebingen.de

Abstract

This paper describes our approach and results for the SemEval 2024 task of identifying the token index in a mixed text where a switch from human authorship to machine-generated text occurs. We explore two BiLSTMs, one over sentence feature vectors to predict the index of the sentence containing such a change and another over character embeddings of the text. As sentence features, we compute token count, mean token length, standard deviation of token length, counts for punctuation and space characters, various readability scores, word frequency class and word part-of-speech class counts for each sentence. class counts. The evaluation is performed on mean absolute error (MAE) between predicted and actual boundary word index. While our competition results were notably below the baseline, there may still be useful aspects to our approach.

1 Introduction

With the rapid proliferation of Large Language Models (LLMs) that are able to produce fluent texts in response to user queries across a wide range of domains and topics, concerns are raised about the potential misuses of such powerful tools. In spite of their fluency, LLM-generated texts may contain factual errors, inadvertently spreading misinformation. Another common issue occurs in the education system, where students may attempt to pass off the responses of such an LLM as their own work, evading commonly used safeguards against plagiarism. Given the overwhelming volume of potentially machine-generated content, it is desirable to have automated means of detecting such texts to address the above-mentioned issues. In this task (Wang et al., 2024), we examine exclusively English mixed texts, where a switch from human authorship to LLM output occurs at most once in a text sample (some samples are entirely machine-generated). To us, this models a plausible use case, where a human user employs an LLM to finish their work for them. For each sample, the task is to predict the token index at which the authorship change occurs. We observe, that due to the structure of the samples, we can reformulate the task more generally as trying to detect an authorship change and its location within the sample texts, without explicitly trying to detect the presence of LLM-generated text. This allows us to adapt more traditional, computationally relatively inexpensive approaches to stylometry and authorship identification/attribution. While the task is formulated as prediction of a boundary word, we begin by identifying the boundary sentence in which the authorship change occurs. For each sentence, textual feature vectors are extracted and combined with character n-gram information, those sentence vectors are then fed into a Bidirectional LSTM network (Hochreiter and Schmidhuber, 1997) which is trained to predict the boundary sentence. We found that our approach performed reasonably well in-domain on the development set, in spite of inevitably introducing some token offset error by only making sentence level predictions and choosing the middle tokens, but failed out-of-domain on the test set, ranking at 26 out of 30 in the competition on subtask C.

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 829–832

2 Background

In accordance with the task guidelines we do not use external data, but use the English subsets of larger data sets from subtask A and B to extract a character vocabulary. The subtask C dataset comprises a bit over 4000 texts with 505 pre-split into a dev set by the task authors, each text labeled with the index of the boundary word. The following table shows token and sentence counts for train and dev set, when tokenized by splitting on whitespace (U+0020) as in the task baseline model. Sentence splits are determined subsequently on the token lists by identifying sentence-final tokens using our detection regex, this is done to ensure matching the given boundary word labels.

	train	dev
texts	3,649	505
sentences	41,570	5,628
tokens (types)	864,153 (29,593)	116,221 (8,641)
chars	5,933,701	803,771
avg. sentences	11.4	11.1
avg. boundary	3.4	3.4

Table 1: Task data statistics

We observe that about half of the samples contain 4 - 11 sentences and sentence count per sample ranges from 1 to 76, with 24 samples containing just one sentence, like e.g. *"We have added a 2+ page* ***discussion*** *on the experimental results, highlighting the superiority of the ARC-based models and their impact on the field of deep learning."* (boundary word 'discussion' in bold). While on average the author switch occurs in the fourth sentence, in about 15-20% of the samples the switch occurs in the first sentence. Examining the boundary word position within their respective sentences we found an average offset of -1.6 (train set) or -1.8 (dev set) from the middle of the sentence, i.e. the switch occurs slightly before mid sentence on average.

An example text, split in sentences and tokens can be observed below, with the boundary word **"baseline"** at index 20 highlighted:

- Format: *label*: (*start token index*) [*tokens*]
- 0: (0) ['The', 'paper', 'proposes', 'a', 'method', 'to', 'recognize', 'time', 'expressions', 'from', 'text.']
- 1: (11) ['It', 'is', 'a\r\nsimple', 'rule-based', 'method,', 'which', 'is', 'a', 'strong', '**baseline**', 'for', 'time', 'expression', 'recognition.']
- 0: (25) ['The', 'authors', 'analyze', 'different', 'datasets', 'and', 'discover', 'that', 'only', 'a', 'small', 'set', 'of', 'words', 'are', 'consistently', 'used', 'to', 'convey', 'time', 'information.']
- remaining sentences omitted for brevity

Interestingly, by using the given tokenization method on whitespace only, we preserve linebreak characters such as in the third token of the second sentence, and obtain empty string tokens in between multiple whitespaces. We choose to include both this 'raw' text data as well as a normalized version in our system, since on the one hand, such typographic choices are indicative of authorship changes but on the other hand, we may not be able to rely on their presence in unseen data.

3 System overview

In our system, we at first sought to compare and combine more traditional textual features with task-specific learned character embeddings, as the former offer the benefit of cheap computation and greater transparency, whereas the latter should allow the system to capture more subtle patterns at the expense of transparency and at a higher computational cost. We choose a relatively straightforward basic architecture for our models, using a BiLSTM over sentence vectors to predict the boundary sentence at which the authorship change occurs. With regards to textual features, we compute for each sentence: token count, mean token length, standard deviation of token length, counts for punctuation and

space characters, various readability scores, word frequency class and word part-of-speech class counts. For our character model, we used another BiLSTM with embedding layer over the text characters, adjusting the token labels to the character level. As the character level model did not perform to our expectations on both normalized and raw text, we did not combine it with the textual feature model and decided to use the latter as a stand-alone model.

4 Experimental setup

We used the provided train/dev split to tune our models. We compared performances of a purely textual feature based model and a character-based model. We extracted the textual features offline, using the *spacy* (Honnibal and Montani, 2017) library for Python and its *textdescriptives* (Hansen et al., 2023) extension library. We used the *en_core_web_sm (v3.7.1)* pipeline for spacy. Hyperparameters were tuned manually, we settled on single-layer networks of hidden size 16, using Adam optimizer (Diederik, 2014) with learning rate 1e-5, training on batches of size 8 over 100 epochs. For the character-based network we choose an embedding size of 8. The task is evaluated on mean absolute error (MAE) between predicted and actual boundary word index. To translate our boundary sentence prediction into a token index, we selected the middle token index as default, rounding it down for sentences with an even token count. For the character model, we chose the token containing the predicted character index.

5 Results

On the development set our textual feature model showed somewhat promising results with regards to predicting the sentence containing the boundary word. It predicted the correct sentence in 69.9% of cases, the adjacent sentence in a further 21.4% with a sentence index MAE of 0.47. Translating these predictions into token level predictions using the sentence mid-point yielded a token MAE of 13.5, notably worse than the baseline model's. Our character embedding model did perform notably worse on the development set, with a token MAE of 48.9. We therefore did not pursue it further and abandoned our initial idea of combining it with the textual feature model. On the test set, for the competition, we submitted the predictions of our textual feature model, unfortunately not matching the performance on the development set. We only managed to predict 30% of boundary sentences correctly, with another 22.6% predictions of the adjacent sentence, resulting in a sentence MAE of 3.2 and a disappointing token MAE of 59, ranking 26 out of 30 among the participant models in subtask C. We suspect this drop in performance mainly be caused by introduction of a new text domain in the test set, while development and training samples are exclusively drawn from PeerRead (Kang et al., 2018) i.e. academic peer reviews, the test set introduces student essays from OUTFOX (Koike et al., 2023). This degradation of performance for stylistic textual features is in line with other findings, e.g. the comparisons performed on the M4 dataset (Wang et al., 2023) of which this competition's dataset is an extension.

6 Conclusion

While our model's performance leaves plenty of room for improvement, we can envision the use of simple textual features in a lightweight model, similar to ours as a basic tool in contexts where more powerful models are either unavailable or too expensive to run and the text domain is known in advance. Focusing on sentence level instead of token level predictions allows us to reduce computational effort and we consider it sufficient for many practical applications, where automated detection of LLM generated text is only a first step, such as e.g. examining student essays, where we would expect a teacher to follow up with affected students regarding suspicious spans of text individually.

References

P. K. Diederik. 2014. Adam: a method for stochastic optimization.

Lasse Hansen, Ludvig Renbo Olsen, and Kenneth Enevoldsen. 2023. Textdescriptives: A Python package for calculating a large variety of metrics from text. *Journal of Open Source Software*, 8(84):5153.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018. A dataset of peer reviews (PeerRead): Collection, insights and NLP applications. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1647–1661, New Orleans, Louisiana. Association for Computational Linguistics.

Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2023. Outfox: LLM-generated essay detection through in-context learning with adversarially generated examples.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Alham Fikri Aji, and Preslav Nakov. 2023. M4: Multi-generator, multi-domain, and multi-lingual black-box machine-generated text detection. *arXiv:2305.14902*.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Giovanni Puccetti, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. SemEval-2024 task 8: Multigenerator, multidomain, and multilingual black-box machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation*, SemEval 2024, Mexico City, Mexico.