Team jelarson at SemEval 2024 Task 8: Predicting Boundary Line Between Human and Machine Generated Text

Joseph Larson
Indiana University
joelarso@iu.edu

Francis Tyers
Indiana University
ftyers@indiana.edu

Abstract

In this paper, we handle the task of building a system that, given a document written first by a human and then finished by a large-language model (LLM), the system must determine the transition word, i.e. where the machine begins to write. We built a system by examining the data for textual anomalies and combining a method of heuristic approaches with a linear regression model based on the text length of each document.

1 Introduction

Large Language Models (LLMs) have never been more available than they are today. The consequence of this is an increase in machine-generated content within various domains. While some of this content could be considered useful, concerns related to the abuse of LLMs has arisen, e.g. the generation of fake product reviews (Adelani et al., 2019), spamming/phishing schemes (Weiss, 2019) and fake news generation (Zellers et al., 2019; Brown et al., 2020; Uchendu et al., 2020). Weiss (2019) demonstrated that humans can only detect human generated text from machine generated text at chance level. This illustrates the clear need for automatic systems to detect LLM generated content.

Regarding mere impressionistic differences between the two types of text, it has been observed that LLMs tend to be more focused, i.e. never leaving the subject matter of their prompt, more objective and highly formal. Their human counterparts tend to be less formal, with more propensity to stray from the topic at hand and more emotional. In terms of linguistic differences between the two, humans use less nouns and conjugations, while employing more punctuation and adverbs. Dependency relations are shown to be shorter. Lastly, human texts have more types in texts of the same length (Guo et al., 2023).

An assumption many researchers take is that LLMs is that language models sample from the head to generate natural looking text e.g. max sampling (Gu et al., 2017) and k-max sampling (Fan et al., 2018). (Solaiman et al., 2019) use a bag-of-words approach with tf-idf feature vectors (both unigrams and bigrams) and a logistic regression model to differentiate between human-written web pages and text generated web pages from GPT2. They examine a different number of parameters of the LLM (117M, 345M, 762M and 1,542M) as well as different sampling methods (k-sampling (sampling the highest probability tokens until a threshold of specified tokens is reached), p-sampling (sampling from the smallest possible set of words until a cumulative probability is reached) and pure sampling (also known as temperature sampling, where lower 'temperatures' are associated with higher probabilities for tokens). Their findings are that the larger the LLM, the harder to detect how machine-like the generated text is and k samples are easier to detect than pure samples, probably due to the fact that k samples over-produce common words, which is easy to detect using statistical methods.

Gehrmann et al. (2019) use BERT and a group of statistical features: the probability of each word, absolute rank of each word and the entropy of the distribution and create a tool for users to see specifically what features are more likely to be machine generated over human generated. They clearly show that the model GPT-2 oversamples certain words; it is worth pointing out, however, that as LLMs grow more sophisticated, such methods might not work as well. Solaiman et al. (2019) use fine tuning on RoBERTa and finds it can detect text generated from GPT-2 with an accuracy of 95%. The most noteworthy aspect of this study is that fine-tuning on GPT-2 itself did not yield as impressive results, which contradicts Zellers et al. (2019) findings which allude to the idea that the best detector of text generated from LLMs are the LLMs themselves. The RoBERTa detector has also been used in detecting fake news articles from several LLMs (Uchendu et al., 2020), Amazon product reviews

Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024), pages 477–484

(Adelani et al., 2019) and biomedical texts (Rodriguez et al., 2022).

2 Task

This task is slightly different from the tasks described in the previous sections, since the purpose of this task is to guess the correct index at which the LLM starts writing. Since it no longer a binary classification task, i.e. given a document guess if it is a human or machine who wrote it, (which should be approached as an authorship attribution task), it was deemed helpful to examine other computational tasks whose purpose is to generate a boundary line in documents. King and Abney (2013) used four different classifiers to classify words in bilingual documents. They found that Naive-Bayes worked the best using either 1-5-grams, both character and word level. Lui et al. (2014) used a similar Bayesian model to detect language segments in multilingual documents, using byte-encoded n-grams as features and achieving the best results with higher-resource languages like English. Although the task here is profoundly different from these two previous examples, we took inspiration from these studies believing there must be linguistic differences that can be detected with statistical methods between the human generated text and the machine generated text.

3 Data Examination

Anomaly	Frequency	Location
word..word	875	Transition
^␣.Word	252	Transition
single line break	2,334	Human
double spacing	599	Human
gratuitous spacing	65	Human
2× 5-gram	1558	Machine*
2× 10-gram	382	Machine*
2× 15-gram	160	Machine*
2× 20-gram	96	Machine*
3× 5-gram	115	Machine*

Table 1: Anomalies found in training data, where 2× indicates that a particular n-gram appears twice in sequence. Machine* denotes that these occurred overwhelmingly in the machine text (with exceptions being under 1%).

The dataset for this task is the same from (Wang et al., 2024). We created a script to manually examine the transition words for all documents. One striking feature of the data is that only the human generated text featured single line breaks. Another was that in many cases the transition word occurred after tokens which had a word, followed by two full stops and another word. Table 1 provides a full list of anomalies found in the training and development set with their their respective frequencies. Some of the anomalies were present only in the human written text or occurring at the transition word token. Others were found mostly in the machine generated text. The anomalies that were featured near or around the transition word resulted as the most predictable for the creation of our model.

We also examined how frequent each transition token was in the corpus and how often it occurred as a transition word. Since the final evaluation was in terms of the distance from the actual index where the transition word occurred using the formula $\text{text} = \text{document.split(' ')}$, we decided to include tokens with different case and punctuation as separate tokens. Table 7 in the appendix contains the most frequent transition words, all appearing as transition words at least 30 times in the data set, as well as their relative frequency in the training data overall.

4 Experiments

For all experiments, scores are reported as a mean of the results of three runs ± the standard deviation, assuming random choice was involved somehow.

4.1 Random Choice

To establish our own baseline, we decided to create a system that randomly chose an index between 0 and the length of the split text. We then chose various coefficients to multiply the text length by. Table 2 gives a complete list of these experiments. Dividing the length by half or around half e.g. 0.4 gave the lowest MAE, suggesting that the majority of indexes are towards the beginning halves of the texts, not towards the end.

4.2 Heuristics

Based upon the anomalies we found in the data, we decided to incorporate explicit rules in our system of random choice. The first rule that that we experimented with was having the system guess the position of the transition word as the the one proceeding any token that had the pattern Word..Word. While experimenting with the exact regular expression pattern to use, we found the one that accurately guessed the correct index every time was $\backslash\text{w}\backslash.\backslash.\backslash\text{w}$. After this, we established a similar pattern found in the dataset is that when the document's first token was a space, full stop and then a word, then the

Upper bound	MAE
1 len(t)	75.4 ± 0.342
0.5 len(t)	41.6 ± 0.247
0.33 len(t)	43.0 ± 0.287
0.25 len(t)	47.2±0.111
0.66 len(t)	48.2±0.265
0.75 len(t)	53.9±0.246
0.4 len(t)	41.5±0.661
0.6 len(t)	44.6±0.788

Table 2: Random Choice Experiments for training data. The upper bound column indicates the upper bound of the random choice from $0...n$. Margin of error is given for the mean of three trials.

document was entirely machine generated, meaning the correct index was 0. After incorporating these two rules into our system, we left them in all subsequently tested systems, since their predicative power was completely accurate. After establishing these two baseline rules, we investigated having the system guess a random position after the last single line break in the document, guessing the transition word as being one of the frequent transition words under a certain threshold of relative frequency (< 0.01, 0.005, 0.001) and guessing the position as starting with the second repeated n-gram.

Table 3 provides a summary of all heuristic experiments. In the case that a certain rule did not apply to a given document, a random position between 0 and half the length of the text was guessed. The first two rules mentioned reduced the best score from the previous experiments by over 10 MAE, demonstrating they were by far the most robust. Guessing the index as being after the last line break improved the MAE by over 2, indicating it also had a slight effect on overall accuracy.

4.3 Linear Regression

We investigated a heuristic model based upon the length of the text. We were able to combine the first two selected rules with other rules based on text length of each document. The best MAE we obtained from doing this was 25.045±0.231. We decided to investigate using a linear regression model based upon text length, since $R^2 = 0.659$. Figure 1 shows the distribution of the index positions in the training set based on text length. The first experiment combined the first two rules of the previous section and predictions based on a linear regression model for all documents that these rules did not apply to. We obtained a baseline MAE of 20.464 for this. Figure 2 shows the distribution of our guesses using this baseline. We created several different linear models, based upon text length and well as excluding non-heuristic data points and found that our baseline performed the best on the training data. We also combined some heuristic methods from the previous section with this model and found they mostly performed worst, with the exception of slightly modifying the predictions for the texts with a length over 975 (since these are mostly outliers), in which case this method performed slightly better than baseline. Initially, we trained our model on the train data and tested on the dev set provided by the organizers. We obtained slightly better this way, with our baseline model obtaining 18.9 MAE on the dev set. Figure 3 shows the distribution of the dev set by text length and index. Figure 4 shows our predictions for the dev set. They are much more linearly distributed than the overall train set, which

i	Else?	MAE
Last \r\n+1	2nd 10 gram	50.0±0.123
Last \r\n+1	2nd 15 gram	49.2±0.321
2nd 10 gram	Last \r\n+1	47.4±0.412
2nd 15 gram	Last \r\n+1	46.2±0.374
2nd 5 gram	Random Choice	39.1±0.212
$f < 0.001$	Random Choice	32.5±0.232
Word..Word	Random Choice	32.2±0.542
␣.Word	Random Choice	30.8±0.214
Last \r\n+1	$f < 0.005$	30.1 ±0.401
2nd 10 gram	Random Choice	30.1±0.341
2nd 15 gram	Random Choice	29.8±0.021
\r\n+1	Random Choice	28.2±0.439
Last \r\n+1	$f < 0.001$	28.2±0.436
Last \r\n+1	$f < 0.01$	28.2±0.303

Table 3: Heuristic Experiments. i stands for index and f stands for relative frequency i.e. the proportion a token appeared as a transition word to how often it appeared in the data overall. The i column refers to what was guessed as the index first. If this feature was not present in the document, the Else? column indicates what was guessed for that document instead. $f < n$ refers to a transition word with absolute frequency less than n that was guessed as the index. We first tested the Word..Word rule then the ␣.Word rule. We found they always yielded the correct index so we included them in all subsequent experiments. Results are still given in descending order of MAE. Beyond this, all experiments were independent, not cumulative.

i	Else?	MAE
Baseline	N/A	20.5
Non-heuristic data	N/A	28.7
len(t) $<$ 650	N/A	20.4
len(t) $<$ 1000	N/A	20.5
len(t) $<$ 800	N/A	20.4
Random Choice*	Baseline	20.4$\pm$0.009
Last \r\n	Baseline	43.3
2nd 10-gram	Baseline	21.6

Table 4: Linear Regression Experiments. The baseline model refers to a linear model for all data. In some cases, a linear model was created for only some data. The i column refers to index that was chosen first e.g. in the case of the baseline model it was always the either the two previously mentioned heuristics or the index predicted by the linear model. *Applied a random index between 600 and 700 for this experiment for len(t) $<$ 950

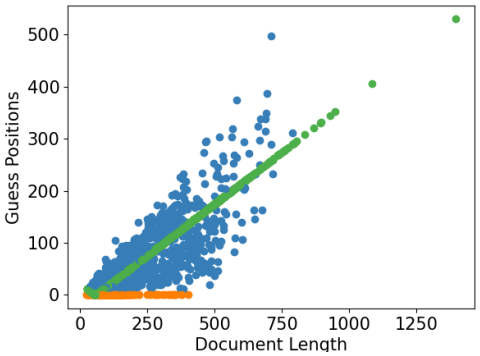

Figure 2: Distribution of Training Data by Text Length (X-Axis) and Our Predicted Index of Transition Word Using Linear Regression and Heuristics. (Y-Axis) Blue corresponds to guesses based on the rule Word..Word, orange corresponds to guesses based on the ^␣.Word rule and green corresponds to guessed made with linear regression.

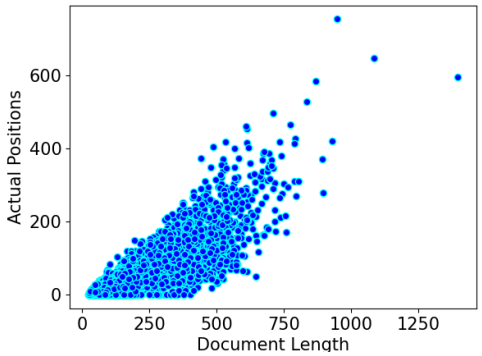

Figure 1: Distribution of the Training Data by Text Length (X-Axis) and Index of Transition Word (Y-Axis)

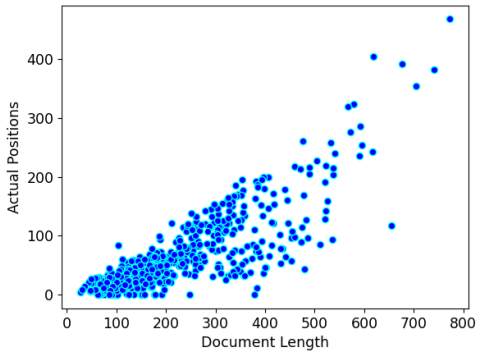

Figure 3: Distribution of the Dev Data by Text Length (X-Axis) and Index of Transition Word (Y-Axis)

explains why the model performed slightly better. Table 4 shows the results for all experiments using linear regression.

5 Results

The solution we submitted to the contest was our baseline linear regression model, since it performed the best, with the exception of the one model with the outlier rule. We chose this over the latter since we assumed the test data would have less outliers in terms of text length, so the baseline linear regression model might perform the best. Our final score for our submission was an MAE of 48.139.

We first examined the test data for the same anomalies found in the training data. Table 8 in the appendix gives a summary of these anomalies. There are far fewer transition word anomalies than

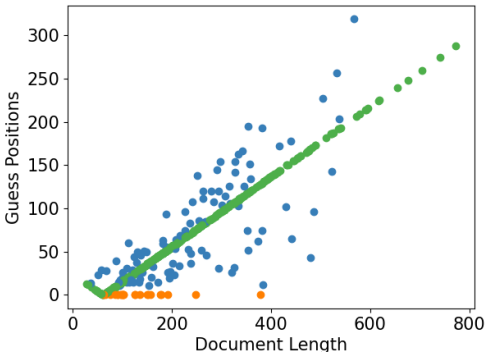

Figure 4: Distribution of Dev Data by Text Length (X-Axis) and Our Predicted Index of Transition Word Using Linear Regression and Heuristics (Y-Axis) Blue corresponds to guesses based on the rule Word..Word, orange corresponds to guesses based on the rule ^␣Word and green corresponds to guessed made with linear regression.

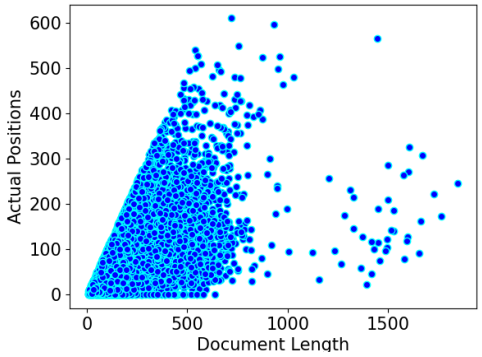

Figure 5: Distribution of Data by Text Length (X-Axis) and Actual Position (Y-Axis) in Test Data

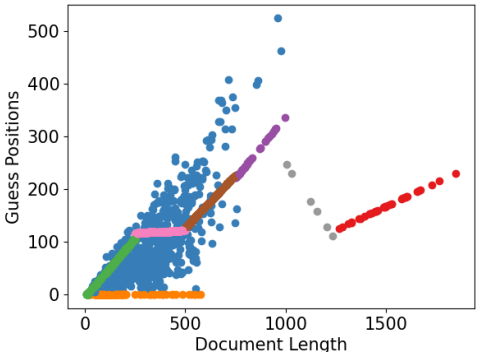

Figure 6: Distribution of Data by Text Length (X-Axis) and Predicted Position of Baseline Model (Y-Axis). Blue corresponds to guesses made with Word..Word rule, orange corresponds to guesses based on our ^␣Word rule, green corresponds to the first linear regression model, pink to the second, brown to the third, purple to the fourth, grey to the fifth and red to the sixth.

in the training data, while the number of repeat n-grams is much higher than in the training data. We also looked at the most frequent transition words in test data. Table 9 in the appendix shows a complete list of all frequent transition words that occurred more than 50 times in the training data, along with their relative frequencies.

Figure 4 shows the distribution of the indexes based on text length for test data. It is much less linearly distributed than the training data (R^2 = 0.26471) and contains a lot more outliers, which explains why our linear model performed much more poorly on it. The next sections explain experiments we did with the training data to improve the linear and heuristic model.

6 Post Hoc Data Analysis

Since the test data is less linearly distributed than the training data, we decided to try different linear models for different lengths of text. Nonetheless, we still trained a linear model on the test data to get a baseline for subsequent experiments. We obtained a baseline of 44.6 MAE. After, we tried different fitting the data to a different number of linear models based upon different text lengths. We used up to six different models in each experiment, adjusting bin sizes. Ultimately, we fit each bin to correspond to the number of quintiles for the model e.g. for final bimodal model, we used the threshold as the median. Our best result ended up being a sixmodal model with the bins 250, 500, 750, 1000 and 1250. We decided to use this linear model when applying subsequent heuristic methods. Table 5 provides the results of our linear experiments. Figure 6 shows a scatter plot of our guesses.

Regarding heuristics, we found that of the frequent transition words in the test data, the ones that almost always occurred as the transition word in a document containing it were those with a relative frequency in the corpus under 0.0001, with the exception of commas and empty strings. Combining this rule with the baseline linear model reduced MAE by ~2. We then looked at repeat higher order n-grams with worse performance. Even though the machine generally generated repeat higher order n-grams, it was still not predictive when determining the boundary line. Lastly, we looked at frequent transition bigrams and frequent transition trigrams. Setting these as the index when they occurred in a document only improved our score slightly. More than anything, they were more accurate in picking the index if they occurred at the beginning of the document, in which case the index for that document was 0. Table 6 provides a summary of all heuristic modifications we made to our system.

7 Conclusion and Considerations for Future Tasks

For our system, due to time constraints, we did not perform any state of the art techniques and of course did not obtain any state of the art results. However, what our paper demonstrates above all is the need for both training data and test data to be better processed with as few textual anomalies as possible. For those teams who trained a neural model for this task, it would be interesting to see what the model learns if these anomalies are removed from the test and training data. It is our hypothesis that

Model	Bins	MAE
Baseline	N/A	44.6
Bimodal	200	42.7
Bimodal	212	42.8
Bimodal	250	42.8
Bimodal	500	44.0
Bimodal	750	44.0
Trimodal	750, 1000	44.0
Trimodal	148, 301	42.7
Fourmodal	125, 212, 358	42.7
Fivemodal	111, 171, 261, 395,	42.6
Sixmodal	103, 149, 212, 299, 423	42.6
Sixmodal	250, 500, 750, 1000, 1250	42.5

Table 5: Linear Regression Experiments, Training and Testing on Test Data. Bins indicate cut off points for models within that particular text length.

Model	*i*	*f*	MAE
Baseline	FTW	< 0.0001	42.6
Baseline	FTW	< 0.001	42.6
Baseline	FTW	< 0.005	71.6
Baseline	FTW	< 0.002	53.0
Sixmodal	FTW	< 0.0001	41.3
Sixmodal	FTB	< 0.0001	41.2
Sixmodal	0 if FTB	< 0.0001	41.1
Sixmodal	FTT	< 0.0001	41.1
Sixmodal	0 if FTT	< 0.0001	41.0

Table 6: Heuristic Adjustments to Linear Models. i indicates index, f indicates relative frequency, FTW indicates Frequent Transition Word, FTB indicates Frequent Transition Bigram and FTT indicates Frequent Transition Trigram.

removing these anomalies would worsen the performance of neural models. Since the best models in this shared task received a score of around 16.0, it would also be interesting to see what kinds of texts they scored better on. Our hypothesis is that texts with the mentioned anomalies were easier to detect for neural networks and that they had more difficulty with longer texts, since longer texts were not featured in the training set. Not knowing the exact LLM used to generate the machine generated text for this dataset it is difficult to say with certainty, but it also our hypothesis that a more sophisticated statistical model could potentially detect more differences between the human and machine text by examining the sampling frequency of words to determine a more probable boundary line.

References

David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. 2019. Generating sentiment-preserving fake online reviews using neural language models and their human- and machine-based detection.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander 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, Florence, Italy. Association for Computational Linguistics.

Jiatao Gu, Kyunghyun Cho, and Victor O.K. Li. 2017. Trainable greedy decoding for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1978, Copenhagen, Denmark. Association for Computational Linguistics.

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.

Ben King and Steven Abney. 2013. Labeling the languages of words in mixed-language documents using weakly supervised methods. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1110–1119, Atlanta, Georgia. Association for Computational Linguistics.

Marco Lui, Jey Han Lau, and Timothy Baldwin. 2014. Automatic detection and language identification of multilingual documents. *Transactions of the Association for Computational Linguistics*, 2:27–40.

Juan Diego Rodriguez, Todd Hay, David Gros, Zain Shamsi, and Ravi Srinivasan. 2022. Cross-domain detection of GPT-2-generated technical text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1213–1233, Seattle, United States. Association for Computational Linguistics.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

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

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, Mexico.

Max Weiss. 2019. Deepfake bot submissions to federal public comment websites cannot be distinguished from human submissions. *Technology Science*, 2019121801.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. *Defending against Neural Fake News*. Curran Associates Inc., Red Hook, NY, USA.

8 Appendix

Token	TW Frequency	Rel. Frequency	Token	TW Frequency	Rel. Frequency
the	375	0.07279	The	261	0.01178
paper	238	0.01176	ϵ	119	0.00405
proposed	47	0.00568	authors	131	0.00605
This	30	0.00443	in	32	0.01612
is	58	0.01310	a	44	0.02188
of	71	0.03655	this	40	0.00713
and	57	0.03038	it	43	0.00596
to	69	0.02741	␣.The	197	0.00020
However,	54	0.00167	I	44	0.00241
␣.In	34	0.00004			

Table 7: Frequency of Transition Words in Training Data. TW indicates Transition Word, i.e. how often a particular word appeared as a transition word. Relative Frequency refers to the ratio of how many times a word appeared in the training data over how many tokens in training data n = 987,374).

Anomaly	Location					Frequency
word..word	Transition					626
^␣.Word	Transition					97
single line break	N/A					3,555
double spacing	Human					981
gratuitous spacing	Human					92
n-**grams:**		$2\times$	$3\times$	$4\times$	$5\times$	$6\times$
5-gram	Machine*	4,180	1,447	609	282	167
10-gram	Machine*	1,444	270	100	65	—
15-gram	Machine*	674	93	48	42	—
20-gram	Machine*	352	46	38	30	—

Table 8: Anomalies found in test data, where $2\times$ indicates that a particular n-gram appears at least twice. Machine* denotes that these occurred overwhelmingly in the machine text (with exceptions being under 1%).

Token	Tw Frequency	Rel. Frequency	Token	Tw Frequency	Rel. Frequency
I	167	0.00433	The	257	0.00637
""""\nThe	100	<0.00001	the	485	0.05645
authors	203	0.00354	is	94	0.01476
would	74	0.00422	have	77	0.00563
.The	68	< 0.00001	However,	204	0.00139
\n\nPlease	141	< 0.00001	they	53	0.00515
this	73	0.00474	a	93	0.02312
In	77	0.00132	of	191	0.02941
there	83	0.00226	that	66	0.01293
in	65	0.01422	to	187	0.03286
be	60	0.00867	,	57	<0.00001
It	114	0.00197	For	81	0.00144
paper	186	0.00482	ϵ	270	0.00162
This	129	0.00415	\n\nThe	53	0.00021
and	87	0.02577	They	62	0.00124
\nthe	69	< 0.00001	for	64	0.00974

Table 9: Frequency of Transition Words in Test Data. TW indicates Transition Word, i.e. how often a particular word appeared as a transition word. Relative Frequency refers to the ratio of how many times a word appeared in the training data over how many tokens in training data n = 2,838,565).