Detection of tortured phrases in scientific literature

Eléna Martel, Martin Lentschat, Cyril Labbé

Detection of tortured phrases in scientific literature

Eléna Martel, Martin Lentschat, Cyril Labbé
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{martin.lentschat,cyril.labbe}@univ-grenoble-alpes.fr

Abstract

This paper presents various automatic detection methods to extract so called *tortured phrases* from scientific papers. These tortured phrases, e.g. *flag to clamor* instead of *signal to noise*, are the results of paraphrasing tools used to escape plagiarism detection. We built a dataset and evaluated several strategies to flag previously undocumented tortured phrases. The proposed and tested methods are based on language models and either on embeddings similarities or on predictions of masked token. We found that an approach using token prediction and that propagates the scores to the chunk level gives the best results. With a recall value of .87 and a precision value of .61, it could retrieve new tortured phrases to be submitted to domain experts for validation.

1 Introduction

Over the past few years, the research community has been confronted with an emerging issue related to the use of content rewriting tools. These tools are being used to hide crude plagiarism. Some of these rewriting tools, called *spinners*[1], used to destroy the meaning of the rewritten text. In their pursuit of publication and the relentless pressure to '*publish or perish*', some researchers turn to these tools. However, these spinners, leave behind lexical traces as they transform text, replacing words with synonyms that may be less appropriate during the modification process.

For scientific text, the most brutal modifications were concerning poly-lexical sequences that carry a specific meaning as well-established scientific expressions: e.g. *Artificial intelligence, big data* or *Randomized control trial*. By performing a 'word by synonyms' replacement, the first generation of spinners would destroy the meaning conveyed by these typical collocations. For example, the previously mentioned expressions could be tortured into *man-made consciousness, enormous information* or *randomized controlled preliminary*. We define a tortured phrase as **an expression resulting from the use of a spinner on a well-established scientific expression with a specific and fixed meaning**. Its counterpart is here called expected phrase (i.e. the original scientific expression).

Cabanac et al. (2021) reveals that such meaningless expressions, referred to as *tortured phrases*, can actually be found in many scientific papers. These tortured phrases not only constitute evidences of the lack of reliability and relevance of these papers, but can also be used to quickly retrieve articles that are thus suspected of having employed spinners. A manually collected set of tortured phrases is used as *fingerprints* (Cabanac and Labbé, 2021) by the *Problematic Paper Screener*[2] to comb the scientific literature for such problematic papers. The authors are querying the academic search engine Dimensions.ai (Herzog et al., 2020) to retrieve articles with known tortured phrases.

The set of manually collected tortured phrases is limited to the expertise of its contributors. Tortured phrases from many scientific fields are still to be listed as fingerprints, so to be able to flag undetected problematic papers. To this date (13 oct. 2023), 11.945 papers containing tortured phrases have been flagged by the website *Problematic Paper Screener*, with more to come as the number of known tortured phrases increases. While it's possible that with the context of 2023, Large Language Models can perform paraphrasing of higher quality than spinners, it's crucial to note that these papers have already been published and remain accessible. Also, amongst the 12k flagged articles, 1278 have been published in 2023, as well as 2 articles to be published in 2024. Therefore, this problem still remains and it is of paramount importance to identify them for retractions.

[1] SpinBot (https://spinbot.com), SpinnerChief (https://www.spinnerchief.com)

[2] https://www.irit.fr/~Guillaume.Cabanac/problematic-paper-screener

Proceedings of the 2nd Workshop on Information Extraction from Scientific Publications, pages 43–48

This paper aims at testing automatic methods to distinguish differences between tortured phrases and expected ones. The main aims being to automatically identify tortured phrases that are yet not listed. For this purpose:

- We built a data set aiming at testing detection methods.
- We report results achieved when using different techniques that do not require massive use of labeled data, as such a large data set does not exists yet.
- We explore the use of large language model embeddings, similarity measures, masking and prediction methods to flag automatically tortured phrases not previously known.

The remainder of this paper is organized as follows: Section 2 discuss related work around spinners and the detection of tortured phrases. Section 3 describes the way we built our new data set. Section 4 presents various methods and experiments for which Section 5 provides detailed results. Finally, Section 6 concludes and gives some perspectives on the task at hands.

2 Related Work

Spinners are capable to create several versions of an original text by substituting synonyms and altering sentence structure (Shahid et al., 2017). An example can be taken from the following sentence: *'The cat is eating its food.'*, which could be transformed into: *'The feline is savoring its meal.'*.

It has been shown that content rewriting tools leave behind a trail of lexical artifacts (Shahid et al., 2017). Some of these artifacts can manifest as tortured phrases, wherein the same tortured phrase might recur multiple times in place of an expected one. Furthermore, Zhang et al. (2014) highlights that approximately 94% of the vocabulary used by these tools is not regularly changed, which could explain why the same tortured phrases may reappear multiple times and thus reinforce the need for an effective detection method.

Some authors have set out with the objective of detecting spun text based on dictionaries of rewriting tools. For instance, Zhang et al. (2014) relies on tokens and phrases that remain unchanged during the content rewriting process to assess the similarity between two articles, by focusing on elements that are not found in the dictionary and therefore have not been substituted.

On the other hand, Wahle et al. (2022) attempts to identify machine-generated paraphrased plagiarism. They created a dataset of paraphrased content using commercial tools like *SpinBot* and *Spinnerchief*. This dataset will encompass paraphrased texts from arXiv, student theses, and Wikipedia articles. They employed three types of machine learning classifiers: logistic regression, support vector machines, and naive Bayes classification. Their task is a binary classification to mark the text as being spun or not.

We will be using the dataset of Wahle et al. (2022) in our study. Given its method of fabrication, it contains many undocumented tortured phrases and is thus very valuable. Nevertheless, to be usable for the evaluation of new tortured phrases detection methods, re-annotation at the token level is needed. We did perform this on a small part of the dataset.

In Cabanac et al. (2021), the authors collected data consisting of tortured expressions and their expected equivalents. This will serve as a database of known tortured phrases with their counterparts.

The usage of embeddings to detect tortured phrases was previously explored by Lay et al. (2022). They conclude that fixed embeddings (e.g. GloVe (Pennington et al., 2014)), performs better than contextual ones (e.g. BERT (Devlin et al., 2018)) when using cosine similarity measure to distinguish tortured and expected phrases. Our work goes beyond (Lay et al., 2022) as they only considered tortured phrases in bigrams. We extended this method by evaluating two additional metrics, namely Manhattan distance and Euclidean distance, while also considering trigrams, which constitute a significant part within our dataset. We also explored the usage of predictions of masked tokens to detect tortured phrases, which gave more satisfying results.

3 Dataset

Cabanac et al. (2021) collected around 3,000 distinct tortured phrases thanks to the contribution of researchers and domain experts. Then, we take advantage of the dataset provided by Wahle et al. (2022), which comprises roughly 200,000 paragraphs in both their original and paraphrased forms using spiners. We automatically extracted, from the Wahle et al. (2022) dataset, sentences con-

taining known tortured phrases. This results in around 2,000 sentences containing known tortured phrases and approximately 4,000 sentences with their expected phrases. However, it is worth noting that some of the extracted sentences may potentially contain previously *unknown/unlisted* tortured phrases from various scientific fields, for some unfamiliar to us. This presumption stems from the fact that these sentences have not undergone prior analysis by domain-specific researchers. Thanks to the contributions of other researchers, we are able to flag occurrences of known tortured phrases and their expected phrases. To ensure that our approach is not biased by the presence of unknown tortured phrases, 100 sentences were annotated using diverse sources (i.e. glossaries, scientific papers, and specialized databases). In doing so, we aimed to determine whether scientifically established expressions not present in our dataset of expected phrases would surface, and subsequently, we verified if theses expressions had been altered during the paraphrasing process.

4 Methodology

In this section, we present our methodology for the experiments involving word embedding similarity measures and the prediction of masked tokens to compare tortured phrases and expected phrases.

For the word embedding approach, cosine similarity and distance metrics were computed between the tokens of tortured phrases and the tokens of expected phrases. The two values were then compared. The aim of using the word embedding was to determine whether similarity and distance metrics could effectively distinguish the two classes of phrases. The underlying idea is that expected phrases, being conventional and legitimate, would obtain higher similarity scores and lower distance metrics scores, reflecting greater semantic coherence and regularity compared to tortured phrases.

Bigrams and trigrams were compared by, first calculating scores between constituent bigrams, then aggregating the two scores via arithmetic mean, harmonic mean or minimal value. For example, for the bigram *'big data'*, the three measures were applied between the two tokens. For a trigram like *'support vector machine'*, the measures were computed between all bigrams pairs : *'support'* & *'vector'*, *'support'* & *'machine'*, *'vector'* & *'machine'*. The resulting scores were then aggregated.Minimum takes the lowest score, mean calculates the average, and harmonic mean weights lower scores more strongly.

The chosen word embeddings are the ones from the GloVe model (Pennington et al., 2014). Specifically, we utilized the pre-trained 'glove-wiki-gigaword-100' model, which had shown good performance in previous work (Lay et al., 2022). For these experiments, we used the dataset containing around, 2763 tortured phrases and expected counterparts. The dataset is out-of-context, meaning the phrases are extracted from their original sentences. If a token within a phrase is not present in the vocabulary, no calculation is performed.

Since the semantic of a tortured phrase is destroyed during spinning (i.e. compared to the semantic of a expected phrase), we though of using language models to try to predict tokens in the text. For this masking approach, the SciBERT (Beltagy et al., 2019) pretrained language model was used to predict masked words based on surrounding context. The masking approach was inspired by the methodology used in Gehrmann et al. (2019). Specifically, we adopted their use of three metrics: probability of the original word, rank of the original word in the predicted distribution and entropy over the predicted token distribution. Our goal was to analyze whether there were significant differences in probability, ranking, and entropy between expected and tortured phrases

Two evaluations were performed, token-level and noun chunk-level, to thoroughly analyze approach performance on detecting tortured phrases. Tokens were labeled as 0 or 1 for classification. O when the token is not part of a tortured phrases and 1 when the token is part of a tortured phrase. An optimal threshold was determined to best separate the two classes based on the predicted scores. For the token-level evaluation, we compared the true and predicted categories matched for each token.

In contrast, when using noun chunk for classification, the approach propagates the detection of a tortured token to its chunk.The intuition being that a noun chunk containing one tortured token can be considered in full as a tortured phrase.

Measures	Aggregation functions	Tortured phrases	Expected phrases
Cosine similarity	Arithmetic mean	0.136 (± 0.157)	0.289 (± 0.201)
	Harmonic mean	0.134 (± *1.856*)	0.284 (± *0.581*)
	Minimum	0.088 (± 0.153)	0.254 (± 0.205)
Manhattan distance	Arithmetic mean	42.901 (± 21.922)	41.100 (± 20.233)
	Harmonic mean	42.714 (± 21.852)	40.936 (± 20.171)
	Minimum	40.898 (± 21.408)	39.427 (± 19.759)
Euclidean distance	Arithmetic mean	5.416 (± 2.765)	5.184 (± 2.554)
	Harmonic mean	5.391 (± 2.756)	5.16 (± 2.546)
	Minimum	5.159 (± 2.700)	4.973 (± 2.494)

Table 1: Average similarity and distance measures depending on the aggregation function

In details, results were analyzed at the noun chunk level using the following rules:

- A true positive (TP) is a TP if at least one token of the chunk is labeled as tortured in both the true and predicted categories.
- A false positive (FP) is a FP if no tokens are tortured, but at least one is predicted as tortured.
- A true negative (TN) is a TN if no tokens are labeled as tortured in the chunk in either true or predicted categories.
- A false negative (FN) is a FN if at least one token is tortured in the chunk, but no token in the chunk is predicted as tortured.

This accounts for phrases as a single unit rather than independent tokens. Case examples can be found in Appendix A, Table 4.

5 Results

Here, we present the results of our experiments.

The word embedding experiments analyzed similarity and distance metrics on bigrams and trigrams to compare tortured and expected phrases. The hypothesis was that conventional phrases exhibit greater semantic regularity in their vector representations. The outcomes are depicted in Table 1, which showcases the cosine similarity and distance results for the various aggregations.

While Manhattan and Euclidean distances are generally greater for tortured phrases than for expected phrases, the gaps are marginal compared to cosine similarity. It exhibited the clearest differentiation between tortured and expected phrases based on word embeddings (cf. Appendix A, Figure 1). Additionally, harmonic mean revealed to be a poor aggregation function due to its higher variability. However, this approach has a long computation time which reduces its usage. In addition, while this approach shows a distinction in the overall values between tortured and expected phrases, it is not readily applicable to individual cases (i.e. standard deviation values show a clear overlap).

The masking approach leveraged language models to predict masked words in context, assessing probability, rank, and entropy differences between phrases types. Two levels of evaluation were conducted: token-level and noun chunk-level. To analyze the impact of punctuation, we first generated predictions with and without punctuation marks. We compared the results for the three metrics probability, rank and entropy.

Table 2 shows the precision, recall and F1 scores for the two categories with and without punctuation. For the expected tokens (category 0), we observe high precision and recall score both with and without punctuation. For the tortured tokens (category 1), the precision and recall scores are lower, especially without punctuation. This suggests that the model struggles more to correctly predict the tortured tokens. This is in part due to a class distribution imbalance in the data (i.e. the amount of legitimate tokens far exceeds the tortured tokens), which is hard to correct as this distribution is inherent to the problem at hand. However, the scores for class 1 improve when punctuation is present.

Table 3 shows the precision, recall and F1 scores at the noun-chunk level. We observed improved scores to token-level masking without noun chunks. We obtained an interesting recall of 0.873, showing a good capability to detect new tortured phrases, but a precision of 0.615 implying that domain experts should still filter the phrases identified.

With punctuation				Without punctuation			
Class	Precision	Recall	F1 score	Class	Precision	Recall	F1 score
Probability				**Probability**			
0	0.96	0.70	0.81	0	0.98	0.73	0.83
1	0.32	0.81	0.46	1	0.22	0.81	0.35
Entropy				**Entropy**			
0	0.93	0.63	0.75	0	0.96	0.64	0.77
1	0.25	0.72	0.37	1	0.16	0.73	0.26
Rank				**Rank**			
0	0.96	0.73	0.83	0	0.98	0.74	0.84
1	0.34	0.80	0.48	1	0.23	0.81	0.36

Table 2: Results summary of token classification with and without punctuation.

Precision	Recall	F1-score
Probability		
0.614	**0.873**	0.716
Entropy		
0.589	**0.873**	0.706
Rank		
0.615	0.867	**0.718**

Table 3: Results for noun chunks

6 Conclusion

This paper presents different methods to extract *tortured phrases* from scientific papers. These tortured phrases can then be used to query academics search engine in search for problematic scientific papers. The aim is to apply this identification method of tortured phrases to increase the existing database.

The most promising method is based on large language model token predictions propagate to their noun chunks. It achieves a good recall (0.87) but the precision still needs to be improved (0.61). This means that the detection of tortured phrases still requires some sort of manual checking by domain experts. We also noticed that distinguishing tortured phrases from their legit counterpart can be highly contextual. Future work could try to be more context aware and explore the use of more specific language models.

Acknowledgements

The NanoBubbles project has received Synergy grant funding from the European Research Council (ERC), within the European Union's Horizon 2020 program, grant agreement no. 951393.

References

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Guillaume Cabanac, Cyril Labbé, and Alexander Magazinov. 2021. Tortured phrases: A dubious writing style emerging in science. evidence of critical issues affecting established journals. *CoRR*, abs/2107.06751.

Guillaume Cabanac and Cyril Labbé. 2021. Prevalence of nonsensical algorithmically generated papers in the scientific literature. *Journal of the Association for Information Science and Technology*, 72(12):1461–1476.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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

Christian Herzog, Daniel Hook, and Stacy Konkiel. 2020. Dimensions: Bringing down barriers between scientometricians and data. *Quantitative Science Studies*, 1(1):387–395.

Puthineath Lay, Martin Lentschat, and Cyril Labbé. 2022. Investigating the detection of tortured phrases in scientific literature. In *Proceedings of the Third Workshop on Scholarly Document Processing, SDP@COLING 2022, Gyeongju, Republic of Korea, October 12 - 17, 2022*, pages 32–36. Association for Computational Linguistics.

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.

Usman Shahid, Shehroze Farooqi, Raza Ahmad, Zubair Shafiq, Padmini Srinivasan, and Fareed Zaffar. 2017. Accurate detection of automatically spun content via stylometric analysis. In *2017 ieee international conference on data mining (icdm)*, pages 425–434. IEEE.

Jan Philip Wahle, Terry Ruas, Tomáš Foltýnek, Norman Meuschke, and Bela Gipp. 2022. Identifying machine-paraphrased plagiarism. In *International Conference on Information*, pages 393–413. Springer.

Qing Zhang, David Y Wang, and Geoffrey M Voelker. 2014. Dspin: Detecting automatically spun content on the web. In *NDSS*.

A Example of tortured phrases

Figure 1 shows results using cosine similarity and minimum as the aggregation function. Table 4 shows True Positive (TP) tortured phrases detected by chunk method as well as False Positive (FP), True Negative (TN), False Negative (FN).

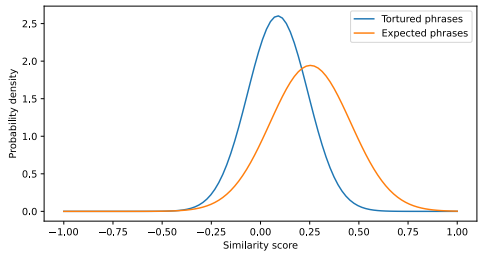

Figure 1: Cosine similarity using minimum aggregation

Case				Decision
	width and profundity			
value	1	1	1	True
predict.	0	0	1	Positive
	convoluted neural system			
value	1	1	1	False
predict.	0	0	0	Negative
	breast cancer			
value	0	0		True
predict.	0	0		Negative
	brain tumor			
value	0	0		False
predict.	1	0		Positive

Table 4: Example of TP, FP, FN, TN with the chunk method