

# NEWWRITER: A Text Editor for Boosting Scientific Paper Writing

João Ribeiro Bezerra<sup>1</sup>, Luís Fabrício Wanderley Góes<sup>2</sup><sup>a</sup> and Wladimir Cardoso Brandão<sup>1</sup><sup>b</sup>

<sup>1</sup>Department of Computer Science, Pontifical Catholic University of Minas Gerais (PUC Minas), Belo Horizonte, Brazil

<sup>2</sup>Department of Informatics, University of Leicester, Leicester, England, U.K.

**Keywords:** Writing Assistance, Language Model, Transformer, BERT, Natural Language Processing.

**Abstract:** Nowadays, in the scientific field, text production is required from scientists as means of sharing their research and contribution to science. Scientific text writing is a task that demands time and formal writing skills and can be specifically slow and challenging for inexperienced researchers. In addition, scientific texts must be written in English, follow a specific style and terminology, which can be a difficult task specially for researchers that aren't native English speakers or that don't know the specific writing procedures required by some publisher. In this article, we propose NEWWRITER, a neural network based approach to address scientific text writing assistance. In particular, it enables users to feed related scientific text as an input to train a scientific-text base language model into a user-customized, specialized one. Then, the user is presented with real-time text suggestions as they write their own text. Experimental results show that our user-customized language model can be effectively used for scientific text writing assistance when compared to state-of-the-art pre-trained models.

## 1 INTRODUCTION


Text production is required from scientists as means of sharing their research and contribution to science. Scientific text writing is a task that demands time and formal writing skills. The writing process can be specifically slow and challenging for inexperienced researchers, delaying the release of their research results (Ito et al., 2019). Additionally, most scientific texts must be written in English, following a specific style, and using specific terminology to be accepted for publishing in respectable journals and conferences. This can be a difficult task specially for non native speakers or those that don't know the specific writing procedures required by some publishers, demanding extra effort and time.


Considering the importance of text production, the Natural Language Processing (NLP) area is widely studied by the scientific community in many applications to solve some of natural language problems. Recent advances in NLP come as result of the proposal of Sequence-to-Sequence (seq2seq) language models and the Transformer architecture (Vaswani et al., 2017). These model and neural network architecture address the text processing as a sequence

of words, keeping the context of other words in the text when processing each singular word. The sequences of words are transformed into embeddings that are processed by the language model (LM) by using Deep Learning (DL) algorithms. Therefore, many approaches have been proposed for problems such as machine translation (Li et al., 2019), text summarization (Kieuvongngam et al., 2020), speech recognition (Liu et al., 2020), ancient text restoration (Assael et al., 2019), and question-answering (Esteva et al., 2020). These approaches have been successful in these tasks, even being used in a production environment by huge companies, such as Google (Chen et al., 2019).

A lot of effort is being done into developing tools for writing assistance. Previous studies show positive results for this application, such as text auto-completion based on the user's writing patterns and style (Chen et al., 2019) and scientific text writing assistance for non-native English researchers (Ito et al., 2019). Furthermore, with the proposal of LMs compliant to parallel computing, NLP solutions are becoming faster and more reliable. Along with the faster processing times comes the possibility of development of real-time solutions for natural language problems, such as text writing assistance (Donahue et al., 2020).

Although a lot of works have been done in the con-

<sup>a</sup>  <https://orcid.org/0000-0003-1801-9917>

<sup>b</sup>  <https://orcid.org/0000-0002-1523-1616>

text of writing assistance (Chen et al., 2019; Donahue et al., 2020), not many of them are focused on scientific writing assistance, such as (Ito et al., 2019). Additionally, other studies in this area don't specifically focus on real-time text suggestions. Moreover, these studies don't account for user customization when it comes to sub-area specific terms and writing style. In the subject of scientific writing assistance, the user should be assisted for faster, higher quality text writing, while also having compliance with the sub-area's specific terms and writing style.

The objective of this article is to propose a neural network based approach that, given a number of input scientific texts related to a certain area of interest, generates recommendations to assist the user in the process of writing scientific text. The expected result is that the proposed approach makes the writing process faster, more intuitive and improves the quality of the written text. The text suggestions must be compliant with specific terms used in the sub-area the user is writing for, enabled by user customization. Therefore, in this article we propose an approach based on the Transformer language model for real-time scientific text writing assistance. In order to reach this objective and evaluate the proposed approach, the following specific objectives are posed:

- Train a specialized language model for scientific texts;
- Propose a learning approach that the user can fine-tune the language model training by using other scientific texts as input for customized writing assistance;
- Propose a real-time writing assistant, where the approach shows text suggestions as the user writes their text.

In NLP, writing assistance is a topic focused by many studies. The state-of-the-art techniques such as seq2seq models and the Transformer architecture are widely used since their proposal can help solve many challenges in the area. The Transformer architecture fits with the problem addressed in this research because it allows the understanding of language patterns. And the addressed problem's domain lies on scientific texts, which follow a structured pattern, built with normalized writing patterns and formal terms as a standard. Therefore, the use of NLP, specially with Transformer language models, shows great potential in addressing the challenges this article proposes to investigate. In addition, the proposed approach differs from previous works by the fact that it allows for user customization through input scientific texts, which results in more accurate assistance in terms of the user's area of study or in the specific

writing style used in a publication medium.

The remaining of this article is organized as follows. Section 2 presents theoretical foundation. Section 3 presents the related work. Section 4 presents NEWWRITER, our proposed approach. Section 5 presents the methodology, with used methods, metrics and tools. Section 6 presents the experimental setup and results. Section 7 presents the conclusion and directions for future work.

## 2 BACKGROUND

The development of novel NLP techniques is crucial for advances in its applications, such as language understanding, language modelling and machine translation. It provides researchers with tools capable of better language representations that can effectively learn language, process input faster, and reach more accurate results. Particularly, this Section is organized as follows: Section 2.1 describes Word embedding, Section 2.2 presents the seq2seq model, Section 2.3 presents the Transformer architecture, and Section 2.4 describes the BERT language model.

### 2.1 Word Embedding

NLP is related to other fields such as machine learning (ML), and broadly artificial intelligence (AI). Recently, state-of-the-art NLP techniques rely on neural networks, requiring numerical vector input. Thus, language models use word embeddings, representations of words in a vector space. These representations project the relations between words in form of their vector offsets. Figure 1 shows gender relation between words on the left panel, and singular/plural relation between words on the right panel (Mikolov et al., 2013).

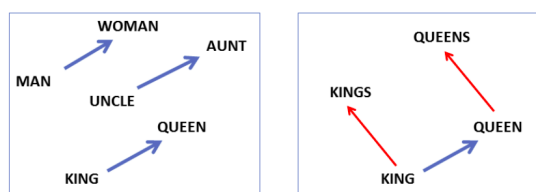


Figure 1: Vector offsets in word embeddings. Source: (Mikolov et al., 2013).

### 2.2 Sequence-to-Sequence Model

Sequence-to-sequence (seq2seq) are language models that use recurrent neural network (RNN) to convert an input sequence into an output sequence, and are

widely used in NLP since text can be represented as a sequence of words. A common example of this use is machine translation, where a sequence of words in a language is transformed into a sequence of words in another language.

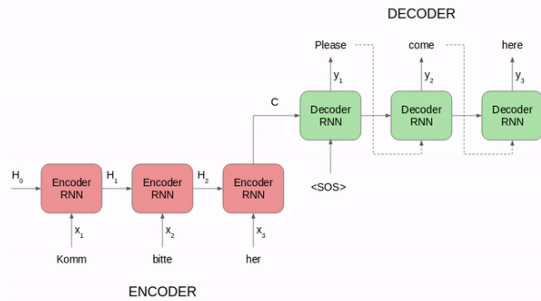


Figure 2: Seq2seq architecture. Source: (Vaswani et al., 2017).

Figure 2 shows that in seq2seq models' architecture, multiple RNNs are created for each word in the input text (both for encoders and decoders). In Figure 2,  $x_i$  we observe the elements of the word vector and  $H_i$  are the hidden states that work as a context vector. Words are processed in order, and the sum of  $H_i$  is passed from each RNN to the next, in a process known as the attention mechanism. So the context from previous words is used in the processing of the current word in the vector.

Although very robust, the sequential processing nature of this architecture prevents parallel processing, which limits its use for longer sequences of text. Another challenge for the use of seq2seq models is the difficulty for the RNNs to learn from the dependencies originated by long-ranged sequences of words on the input vector (Vaswani et al., 2017).

### 2.3 The Transformer Architecture

The Transformer architecture was proposed to solve the challenges faced by seq2seq architectures, such as dealing with long-range dependencies. This architecture can handle dependencies between input and output using only recurrence and the attention mechanism. Figure 3 presents the Transformer architecture, with the encoder on its left half and the decoder on its right half.

From Figure 3 we observe that both the encoder and the decoder are composed of  $N = 6$  respectively identical layers. Each of the encoder's layers has two sub-layers: a multi-head self-attention mechanism, and a position-wise fully-connected feed-forward network. Each of the decoder's layers has, in addition to the two layers in each encoder layer, a multi-head at-

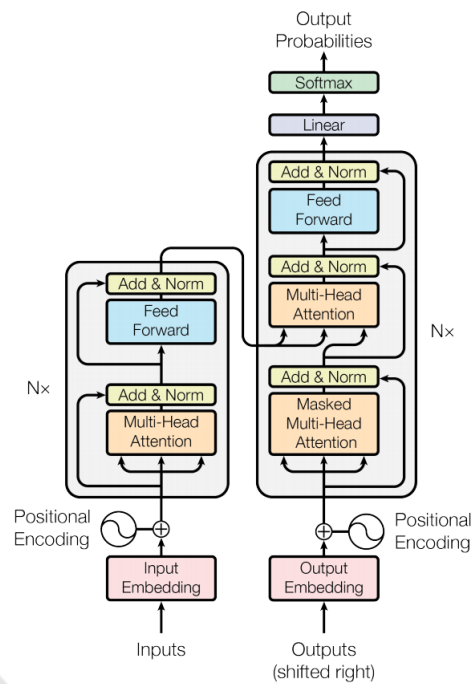


Figure 3: The Transformer architecture. Source: (Vaswani et al., 2017).

tention module over the output of the encoder stack. The multi-head attention modules are formed by a stack of self-attention modules, an attention mechanism that relates different positions of a sentence in order to compute a representation of the sequence. Multi-head attention modules consist of multiple instances of self-attention modules. The self-attention modules enable the architecture to consider other words of the input sequence to better understand each individual word in the sequence. The self-attention module estimates the importance of each word in the sequence to the current word. These values are estimated multiple times in the Transformer architecture in parallel and independently, hence the name multi-head attention. Figure 4 shows the data flow between encoders and decoders in the Transformer architecture.

From Figure 4 we observe that first, the word embeddings from the input sequence are passed into the first encoder block. The embeddings are transformed and passed on to the next encoder, and this process repeats until the last encoder is reached. The last encoder outputs the result to all the decoders in the stack. However, attention modules can only work with fixed-length strings, meaning the input text must be subdivided before used as input. This causes context fragmentation, which limits the Transformer architecture effectiveness. To overcome this challenge, the Transformer-XL architecture was proposed (Dai

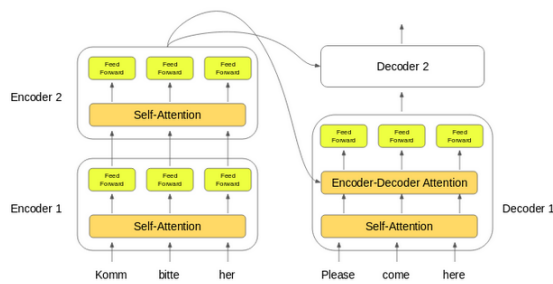


Figure 4: Transformer's encoder and decoder stacks. Source: (Vaswani et al., 2017).

et al., 2019). In the Transformer-XL architecture, the hidden states obtained from the previous input fragment from the original text are used in the processing of the next fragment, thereby causing no context fragmentation.

## 2.4 BERT

The bidirectional encoder representations from Transformers (BERT) is a state-of-the-art language model proposed by Google AI team that uses pre-training and fine-tuning for several NLP tasks. It uses a multi-layer, bidirectional transfer encoder, in which the self-attention layer works on the input sequence in both directions.

The BERT model is pre-trained using two different strategies: masked language modeling and next sentence prediction. In masked language modeling, 15% of an original text words are replaced by either a token "[MASK]" or a random word and are feeded for the language model as input. The objective is that these masked tokens should be predicted by the language model. For next sentence prediction, pairs of sentences are feeded for the language model as input, where in 50% of pairs the second sentence is the subsequent sequence in the original text, whereas in the other 50% the second sentence is another random sentence in the text. The objective is to learn and predict if the second sentence fits as subsequent to the first sentence in the original text.

In order to gather huge language understanding, BERT is trained on huge datasets, a process that demands a lot of time and processing. However, once a language model is pre-trained, it can be used for further training in the fine-tuning process. Fine-tuning enables training a language model for specific tasks, while also being able to skip the initial training with the use of knowledge transference (Devlin et al., 2019).

## 3 RELATED WORK

Several work were reported in literature for writing assistance using NLP. Most of them are related to scientific text writing. The approach proposed by (Ito et al., 2019) is the most similar to NEWWRITER proposal and domain, focusing on scientific writing assistance. However, SmartCompose (Chen et al., 2019), that aims to assist writing of emails, proposes a learnig mechanism very similar to NEWWRITER.

In (Ito et al., 2019) the authors propose sentence-level revision (SentRev), a writing assistant focused on surface-level issues such as typographical, spelling and grammatical errors. Their system focused on helping inexperienced authors by producing fluent, complete sentences given their incomplete, rough text drafts. The authors also released a evaluation dataset containing incomplete sentences authored by non-native writers along with their final versions in published academic papers that can be used for further research in this area.

In (Chen et al., 2019) the authors go over details on the implementation of Gmail's SmartCompose functionality, that generates interactive, real-time suggestions to assist users in writing mails. For the language model selection, they compared state-of-the-art models such as the Transformer architecture and LSTM-based seq2seq models for efficiency and latency. Upon testing, they chose a LSTM-based seq2seq model, since even though the Transformer architecture had more accurate results, its extra computing time wasn't ideal for their application as the model had more latency as it became more complex. SmartCompose showed high-quality suggestion prediction, enough to be adopted to Google's platform in production.

In a similar vein, (Donahue et al., 2020) present an approach for text infilling for prediction of missing spans of text at any position in a document. Although the authors cite the potential of masked language infilling for writing assistance tools, their research focuses on language modeling, in the form of infilling text at the end of a document. In this work, the authors fine-tune language models capable of infilling entire sequences on short stories, scientific abstracts and lyrics. A survey showed that humans had difficulty identifying which sentences were machine-generated or original sentences in the short stories domain.

In (Aye and Kaiser, 2020), the authors propose a novel design for predicting tokens in real time for source code auto-completion, combining static analysis for in-scope identifiers with the use of a language model, in a system that produces valid code with type-

checking. The developed solution achieves state-of-art accuracy while also fitting the constraints of real-world completion implementations in modern IDEs.

In (Shih et al., 2019) the authors propose XL-Editor, a training framework for state-of-the-art generalized auto-regressive pre-training methods to revise a given sentence using variable-length insertion probability in order to reflect the nature of how a human revisits a sentence to obtain a refined result. The XL-Editor is able to estimate the probability of inserting a sequence into a specific position of a sentence, execute post-editing operations and complement existing sequence-to-sequence models to refine generated sequences. The authors demonstrated improved post-editing capabilities from XLNet to XL-Editor. Additionally, XL-Editor is extended to address post-editing style transfer tasks and achieves significant improvement in accuracy, while also maintaining coherent semantic.

In (Grangier and Auli, 2018), the authors propose a framework for computer-assisted text editing, applied to translation post-editing and to paraphrasing. In the proposed framework, a human editor modifies a sentence, marking tokens they would like the system to change. Their model then generates a new sentence that reformulates the original sentence by avoiding the marked words. The model was developed using sequence-to-sequence modeling along with a neural network which takes a sentence with change markers as input. Their model is shown to be advantageous for translation post-editing through simulated post-edits and also for paraphrasing through a user study.

In this section one observe that a lot of progress is being made in the field of NLP with the use of state-of-the-art language models. These works show that modern language models are fit for dealing with tasks related to text writing, writing style learning and reproduction, and text correction. Related work show that language models are able to work with scientific text (Ito et al., 2019) and can be used for real-time writing assistance applications (Chen et al., 2019). Therefore, the related work also show that the current existing approaches can be used in order to address the problem on scientific paper writing. However, none of the related works implement scientific text language modeling to a real-time writing assistant application, such as NEWWRITER.

## 4 NEWWRITER

Upon research reported in the literature, the assistance in scientific text writing is a widely studied problem. However, the state-of-the-art NLP approaches can be

used to address this problem. In addition, the availability of free usage of computing power to accelerate machine learning applications leads to fast and accessible language model training for user-customized solutions. In this article, we propose NEWWRITER, a neural network based approach to address scientific text writing assistance. Figure 5 presents the NEWWRITER architecture.

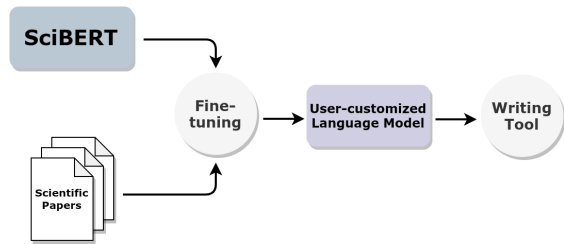


Figure 5: NEWWRITER's architecture.

As shown in Figure 5, initially, the user gathers a number of scientific articles from their area of study. These articles are then used in the process of fine-tuning a base language model in order to customize it to the user's needs. The language model used as a base was SciBERT, as it is a BERT model fine-tuned using scientific text and accomplishes better results in this domain than the base BERT model. Using SciBERT as a base language model saves the time needed to fine-tune a specialized model for scientific text, which would require gathering a number of scientific articles and many time for the fine-tuning process. For the fine-tuning process, the Huggingface's Transformers library is used, as it implements the needed methods in an intuitive and easy to use API, and is widely supported by the community.

In order to make the fine-tuning process faster and accessible for the end-user, this process is made using Google Colab. A notebook was created for the user to be able to input scientific texts in their area of interest. By running the available scripts, the language model is fine-tuned using the provided texts and the user can download the resultant customized language model for use in the writing process. For the writing assistance, an additional module was developed. In order to develop a quick, multi-platform interface, the module was developed using Python 3 and the *prompt\_toolkit* python library. The *prompt\_toolkit* library is used for interfacing with the user via terminal, displaying an input text area, while processing the text recommendations in a separate thread to not interrupt the user in the writing process.

As the user writes their text, the module gathers all text currently in the text area and adds the special token "[MASK]" after the last word. The result-

ing string is processed using the user-customized language model in a fill-mask task, where the language model returns the top most-appropriate tokens to be placed over the "[MASK]" token. For each one of the five recommendations, a thread is created. Each thread repeats the same procedure for the language model input, but only gets the best rated token repeatedly until the generated string reaches a specific length or is a punctuation mark. Finally, it picks the five best rated next tokens as a 5-way recommendation route and extends each route to generate a better context for the user to pick from. This process is done in order to provide the user with multiple possible routes to continue writing the current sentence.

## 5 METHODOLOGY

In this section, one present experimental setup and tools used to develop and evaluate a proper scientific text writing assistant. Therefore, Section 5.1 presents the tools that were used for the development of the scientific text writing assistant and Section 5.2 presents the methods and metrics used in this article.

### 5.1 Tools

This study's objective is to generate recommendations to assist a user in the process of writing scientific text. For this objective's accomplishment, a base pre-trained language model was selected, a module was developed for the model to be further fine-tuned, and another module was developed for the user to be able to write with the assistance provided by the language model. In the following sections, one present the tools used in this article. Section 5.1.1 presents the SciBERT language model, Section 5.1.2 presents HuggingFace's Transformers library, and Section 5.1.3 presents Google Colaboratory.

#### 5.1.1 SciBERT Language Model

In order do address the lack of high-quality, large-scale labeled scientific language models, SciBERT was proposed by (Beltagy et al., 2019). SciBERT is a BERT-based model trained on scientific text on a large multi-domain corpus of scientific publications, aimed to improve the usage of NLP in the scientific text domain. SciBERT is trained on 1.14M papers from Semantic Scholar<sup>1</sup> and contains its own vocabulary (scivocab) with 3.1B tokens. Compared to BERT, SciBERT shows better results on scientific domain NLP tasks.

<sup>1</sup><http://www.semanticscholar.org>

#### 5.1.2 HuggingFace's Transformers Library

HuggingFace's Transformers (Wolf et al., 2019) is a library created to gather several state-of-the-art language models and architectures into an unified API along with examples, tutorials and scripts, for use by the community.

The library contains implementations of state-of-the-art architectures such as BERT, GPT-2, RoBERTa, XLM, DistilBert, among others. There are thousands of pre-trained models available in more than 100 languages with community-contributed models available online. It also has interoperability between PyTorch and TensorFlow 2.0, tools widely used for NLP tasks. The library is highly adopted among both the researcher and practitioner communities.

#### 5.1.3 Google Colaboratory

Google Colaboratory, also referred simply as Colab, is a cloud service provided by Google based on Jupyter Notebooks, used for education and research on machine learning. It provides free access to a GPU suitable for deep learning. The research from (Pessoa et al., 2018) shows the service can be used to successfully accelerate not only deep learning applications but also other classes of GPU-centric applications. Experimental results show faster training of a convolutional neural network on Colaboratory's runtime than using 20 physical cores on a Linux server.

## 5.2 Methods and Metrics

The perplexity metric is commonly used to evaluate the quality of a trained language model. It measures the effectiveness of a probability model, such as a language model, in predicting a given sample. It allows the comparison of language models, with a low perplexity value indicating that a language model is better suited at predicting the given sample. Equation 1 shows how perplexity  $p$  is estimated given a probability model  $q$ , for  $N$  test samples and  $b$  is a constant, usually set to 2 (Brown et al., 1992).

$$p = b^{-\frac{1}{N} \sum_{i=1}^N \log_b q(x_i)} \quad (1)$$

A comparison between SciBERT and a customized language model was made to evaluate the quality of the user-customized language model's recommendations. A synthetic use-case was used for comparison of the language models, using a 10-fold cross-validation based on their perplexity metric to an input sample. In a k-fold cross-validation, the input sample

is equally partitioned into  $k$  sub-samples. For  $k$  iterations,  $k_i$  is used as a test sample while the other  $k - 1$  parts are used as training samples. Then, the results of the  $k$  tests are averaged for a single estimation result. The most used value for  $k$  is 10, also referred as a 10-fold cross-validation. The selected area of interest for the test was NLP, particularly scientific text writing assistance.

## 6 EXPERIMENTAL RESULTS

Table 1 presents the 10-fold cross-validation test. Even with the small provided input, the user-customized language model shows to be more than 3% better suited for the task, as shown in the improvement column. The training process took, in average, 25 seconds, which is compliant for a one-time pre-processing for using the writing software.

Table 1: Perplexity comparison between the user-customized language model and SciBERT for specific domain scientific text.

Fold	User LM	SciBERT	Improvement
Fold 1	5.10	5.19	0.02
Fold 2	5.73	5.80	0.01
Fold 3	4.33	4.42	0.02
Fold 4	5.81	5.89	0.01
Fold 5	4.54	4.60	0.01
Fold 6	9.11	9.55	0.05
Fold 7	13.37	13.80	0.03
Fold 8	8.85	9.11	0.03
Fold 9	12.68	13.32	0.05
Fold 10	12.82	13.26	0.03
Average	8.23	8.49	0.03

Figure 6 shows an example of the writing approach's usage in three sequential moments. As the user writes their text, NEWWRITER displays possible writing routes, highly related to the current context. Additionally, the recommended string is highly related to the domain addressed in the scientific texts used for the language model fine-tuning.

Figure 7 presents two cases where the language model was capable of memorizing and displaying acronyms for terms used in the natural language area, along with correct parenthesis and punctuation usage.

The displayed recommendations show multiple options for the next word in real-time, and progressively extend upon the recommended writing routes in the following few seconds, without interrupting the user interaction for the text area. In cases where the recommendations might not be accurate, which can happen specially for recommendation paths starting

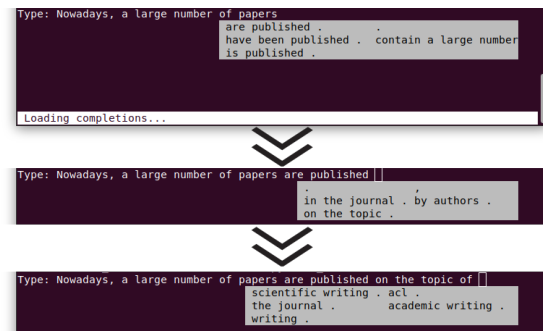


Figure 6: NEWWRITER's writing software example 1. Source: Author.

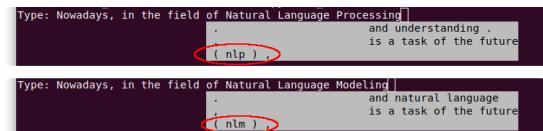


Figure 7: NEWWRITER's writing software example 2. Source: Author.

from a lower score, it does not affect the final user experience, as the final text is curated by the user. Even if the recommendation itself makes sense only to a certain point, it still helps users with idiomatic expressions or connectives to link words together in the text and make the writing process flow better.

## 7 CONCLUSION

The scientific text domain can be more welcoming for beginner researchers and for non-native English speakers, and the accomplishments in this article show that the existing NLP approaches can be used to reach this objective. Experimental results show that user-customized language models can be used to improve the effectiveness of scientific text writing assistance compared to state-of-the-art pre-trained models. NEWWRITER presents a proof of concept for real-time writing assistance using customized, user-trained language models. However, there are still more parameters to explore for the language model fine-tuning and other ways the user could be assisted in the writing process.

For future work, the language model fine-tuning process can be further tested and asserted, and the tool's interface can be improved. The recommendations could be made to select better words and expressions in the middle of the text, as it would explore the bidirectionality of the BERT model. Additionally, as the perplexity tends to reach lower values, texts written using NEWWRITER will tend to become repetitive and the tool could also promote what could be consid-

ered plagiarism. Therefore, NEWWRITER's architecture could be extended to implement a Computational Creativity module. Finally, the tool should be tested with real users - scientists, writing scientific text - to answer a survey to assert the utility of the developed tool.

## ACKNOWLEDGEMENTS

The present work was carried out with the support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) - Financing Code 001. The authors thank the partial support of the CNPq (Brazilian National Council for Scientific and Technological Development), FAPEMIG (Foundation for Research and Scientific and Technological Development of Minas Gerais), and PUC Minas.

## REFERENCES

- Assael, Y., Sommerschild, T., and Prag, J. (2019). Restoring ancient text using deep learning: a case study on greek epigraphy. In *Proceedings of the 9th International Joint Conference on Natural Language Processing, IJCNLP'19*, pages 6368–6375.
- Aye, G. A. and Kaiser, G. E. (2020). Sequence model design for code completion in the modern IDE. *CoRR*, abs/2004.05249.
- Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: Pre-trained language model for scientific text. In *Proceedings of the 9th International Joint Conference on Natural Language Processing, IJCNLP'19*, pages 3615–3620.
- Brown, P., Della Pietra, S., Pietra, V., Lai, J., and Mercer, R. (1992). An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18:31–40.
- Chen, M. X., Lee, B. N., Bansal, G., Cao, Y., Zhang, S., Lu, J., Tsay, J., Wang, Y., Dai, A. M., Chen, Z., Sohn, T., and Wu, Y. (2019). Gmail smart compose: Real-time assisted writing. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19*, page 2287–2295.
- Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., and Salakhutdinov, R. (2019). Transformer-XL: Attentive language models beyond a fixed-length context. *CoRR*, abs/1901.02860.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional Transformers for language understanding. *CoRR*, abs/1810.04805.
- Donahue, C., Lee, M., and Liang, P. (2020). Enabling language models to fill in the blanks. *ArXiv*, abs/2005.05339.
- Esteva, A., Kale, A., Paulus, R., Hashimoto, K., Yin, W., Radev, D., and Socher, R. (2020). CO-Search: COVID-19 information retrieval with semantic search, question answering, and abstractive summarization. *CoRR*, abs/2006.09595.
- Grangier, D. and Auli, M. (2018). QuickEdit: Editing text & translations by crossing words out. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT'18*, pages 272–282.
- Ito, T., Kuribayashi, T., Kobayashi, H., Brassard, A., Hagiwara, M., Suzuki, J., and Inui, K. (2019). Diamonds in the rough: Generating fluent sentences from early-stage drafts for academic writing assistance. In *Proceedings of the 12th International Conference on Natural Language Generation, INLG'19*, pages 40–53.
- Kieuvongngam, V., Tan, B., and Niu, Y. (2020). Automatic text summarization of COVID-19 medical research articles using BERT and GPT-2. *CoRR*, abs/2006.01997.
- Li, L., Jiang, X., and Liu, Q. (2019). Pretrained language models for document-level neural machine translation. *CoRR*, abs/1911.03110.
- Liu, C., Zhu, S., Zhao, Z., Cao, R., Chen, L., and Yu, K. (2020). Jointly encoding word confusion network and dialogue context with BERT for spoken language understanding. *CoRR*, abs/2005.11640.
- Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT'13*, pages 746–751.
- Pessoa, T., Medeiros, R., Nepomuceno, T., Bian, G.-B., Albuquerque, V., and Filho, P. P. (2018). Performance analysis of Google Colaboratory as a tool for accelerating deep learning applications. *IEEE Access*, PP:1–1.
- Shih, Y.-S., Chang, W.-C., and Yang, Y. (2019). XL-Editor: Post-editing sentences with xlnet. *CoRR*, abs/1910.10479.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). HuggingFace's Transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.