applied sciences

MDPI

Article

Revisiting Text Guide, a Truncation Method for Long Text Classification

Krzysztof Fiok [1], Waldemar Karwowski [1], Edgar Gutierrez [1,2], Mohammad Reza Davahli [1,*], Maciej Wilamowski [3] and Tareq Ahram [1]

[1] Department of Industrial Engineering and Management Systems, University of Central Florida, Orlando, FL 32816, USA; fiok@ucf.edu (K.F.); wkar@ucf.edu (W.K.); edgar.gutierrezfranco@ucf.edu (E.G.); tahram@ucf.edu (T.A.)
[2] Center for Latin-American Logistics Innovation, LOGyCA, Bogota 110111, Colombia
[3] Faculty of Economic Sciences, University of Warsaw, 00-927 Warsaw, Poland; m.wilamowski@delab.uw.edu.pl
* Correspondence: mohammadreza.davahli@ucf.edu

Abstract: The quality of text classification has greatly improved with the introduction of deep learning, and more recently, models using attention mechanism. However, to address the problem of classifying text instances that are longer than the length limit adopted by most of the best performing transformer models, the most common method is to naively truncate the text so that it meets the model limit. Researchers have proposed other approaches, but they do not appear to be popular, because of their high computational cost and implementation complexity. Recently, another method called Text Guide has been proposed, which allows for text truncation that outperforms the naive approach and simultaneously is less complex and costly than earlier proposed solutions. Our study revisits Text Guide by testing the influence of certain modifications on the method's performance. We found that some aspects of the method can be altered to further improve performance and confirmed several assumptions regarding the dependence of the method's quality on certain factors.

Keywords: long text; truncation; classification; language model; method; feature importance

Citation: Fiok, K.; Karwowski, W.; Gutierrez, E.; Davahli, M.R.; Wilamowski, M.; Ahram, T. Revisiting Text Guide, a Truncation Method for Long Text Classification. *Appl. Sci.* **2021**, *11*, 8554. https://doi.org/10.3390/app11188554

Academic Editors: Eui-Nam Huh and Zhengjun Liu

Received: 2 June 2021
Accepted: 8 September 2021
Published: 15 September 2021

1. Introduction

In the internet era, a vast amount of new unstructured textual data becomes available every day. Analyzing these data and dividing the types of content into categories requires an automated approach to text classification. From the point of view of the known classification methods, the general text classification task can be divided into short and long text classification. The rationale for this division is that, since deep learning methods were introduced mostly in 2010–2020, these two subtasks have often required different language modeling approaches. Before 2010, little need for this distinction existed, because for classic methods—such as bag of words (BoW), term frequency, inverse document frequency and various lexicon-based methods, such as linguistic inquiry and word count [1]—the computational cost of using each method did not substantially change with the length of the text instance (LTI), because all the methods were based on counting token occurrences (CTC).

However, when convolutional neural networks and recurrent neural networks were introduced to the domain of natural language processing (NLP), the situation changed. The computational costs of visiting each text fragment with convolutional filters in convolutional neural networks appeared to grow markedly faster with the increase in LTI, in contrast to CTC methods. Additionally, maintaining past tokens in memory of recurrent neural networks appeared to substantially increase the computational cost for longer token sequences.

Since 2017, when the attention mechanism was introduced [2] and deployed in what are now called transformer models, the limit of LTI was proposed as 512 tokens to maintain reasonable computation times. This step was necessary because, for each token in the text

instance, the original attention mechanism analyzed connections to all other tokens in the text instance, thus substantially increasing computational cost when larger LTI values were considered. To alleviate this issue, various modifications to the attention mechanism have been proposed in several studies [3–5].

Consequently, when a long text instance is to be classified, a data scientist can choose among very efficient classic CTC methods or highly inefficient deep learning methods. Even if the latter are selected for their theoretically higher prediction quality, not everyone can afford a costly computational machine capable of performing the required deep learning operations. For example, a longformer [6] model cannot always be deployed, and proper data set-dependent parameter optimization of the necessary fine-tuning procedure cannot always be performed. Without this step, the best results offered by the model cannot be acquired.

To overcome this limitation, several approaches have been proposed for decreasing the computational cost of deep learning text classification methods through introducing information loss at the beginning of the classification process. The general idea is simple: the original too long text instance must be truncated to fit the LTI limit of the language model selected for final analysis. If a model with an LTI limit of 512 tokens is selected, the most naive truncation method is to analyze only the first 512 tokens of the original text instance. Despite being naive, this truncation method is very common and is the default in, for example, the renowned NLP framework Flair [7]. Another known approach [8] has proposed to use the available analysis space of 512 tokens so that the next text instance constitutes part of the beginning and end of the original text instance. More complex deep learning approaches described by [9,10] have proposed an encoder-decoder model or another "judge" model to select relevant sentences for the final analysis. Very recently, [11] proposed a simpler method called Text Guide, which is based on a CTC model that also allows important text fragments to be selected for text classification. However, the authors of Text Guide have indicated that the performance of their method could presumably be improved through additional experiments.

In this study, we aimed to explore selected possibilities for improving the results obtained with the Text Guide method. In particular, we wanted to answer the following research questions (RQ):

RQ1 What is the influence of the CTC model quality on the Text Guide?
RQ2 Does the quality of the Text Guide depend on the adopted machine learning interpretability method?
RQ3 Is considering more than one occurrence of essential token found in the original text instance by Text Guide is beneficial?
RQ4 Is Text Guide useful for text instances only slightly exceeding the model limit?
RQ5 Does Text Guide obtain results with a selected transformer model transferable to other models?
RQ6 Is use of Text Guide beneficial for other text classification data sets?

The reasons for formulating and approach to answering the above research questions were the following:

Regarding RQ1, Ref. [11] has used a BoW model for obtaining token features. Here, we explore the influence of the quality of the model used for obtaining token features for final classification performance based on text instances, as created by Text Guide.

Considering RQ2, in [11], only one tool for extracting the feature importance of selected important tokens was used. In this study, we tested whether a more recent approach called Shapley Additive Explanations (SHAP) [12] might result in performance improvements.

Regarding RQ3, we observed that Text Guide searches the original text instance only for the first occurrence of a token previously identified as important; if found, Text Guide extracts the token together with its neighboring tokens and adds the resulting "pseudo sentence" to the new text instance. Here, we explored the performance of Text Guide when not only the first occurrence was considered.

Considering RQ4, based on [11], we believed Text Guide's performance for text instances exceeding the model limit to only a small extent required further exploration. For this purpose, we introduced a threshold parameter defining the instance length for which the Text Guide method should be used.

RQ5 was formulated to verify whether it can be assumed that since using Text Guide for one model is beneficial, it will also be for other models. The answer to this RQ was obtained by testing four other transformer models.

Finally, regarding RQ6, testing a method on various data sets can confirm its utility. In order to prove Text Guide can be helpful for other data sets, we apply the Text Guide to another well-known data set.

For all experiments, we used a publicly available Text Guide implementation [13]. We believe the main contribution of our research is testing and optimizing various modifications and parameter values of the very recently introduced truncation method.

2. Methods

Methods of this article can be divided into ten sections.

2.1. Revisiting Text Guide

The original Text Guide method was proposed and described in detail in [11]. Here, we provide a brief review as a context for our current experiments. Text Guide allows for non-naive truncation of text instances that exceed the length limit of a selected deep learning model, which is later used to create a vector representation of the text instance. As presented in Figure 1, Text Guide requires initial input of a list of important tokens extracted from the original text instance and sorted according to their feature importance (sITFL), the original text instance, the desired length of the new instance measured in tokens (desired LTI) and a set of additional parameters.

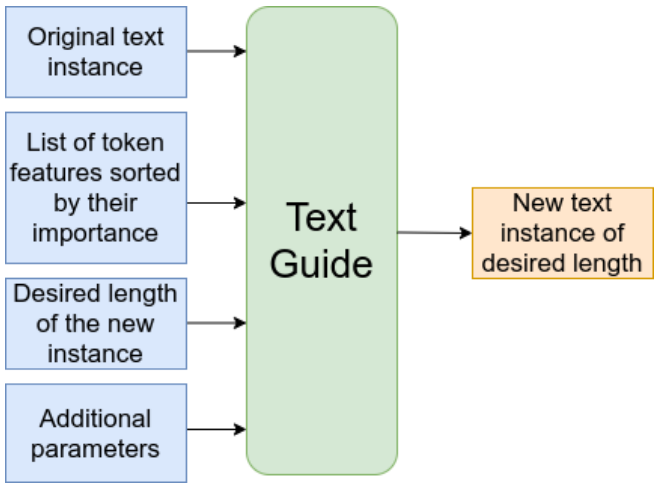

Figure 1. Inputs and output for the Text Guide method.

In a preprocessing step, to create the sITFL, a simple CTC language model operating on token features such as BoW can be used. After a machine learning classifier is trained on the extracted features, the features need to be sorted according to their importance obtained from the classifier via a method from the explainable artificial intelligence (XAI) domain.

The desired LTI can be an arbitrary number but optimally it mimics the limit of the language model later used to create vector representations of text instances. For the models from the transformer family, the value most often used is 512 tokens, but models offering a higher 4096 token limit also exist [6].

To create a new truncated text instance from the original one, Text Guide extracts fragments of text located around important tokens found in original text instances. Consequently, when a token from sITFL is found in the text instance, Text Guide extracts

it together with its neighbors, and such "pseudo sentences" are used to create the new text instance. Additional parameters define, for example, the number of neighbor tokens to be extracted or whether an arbitrary part from the beginning and/or end of a text instance is to be extracted. In [11], these parameters were subject to model and data set-specific optimization.

2.2. Experimenting with the Quality of the Model Responsible for Extracting Token Features

The token features extracted from the text instance in [11] were obtained by training a simple BoW model on the text corpus. Here, we decided to assess the extent to which modifying the BoW model affected the overall text classification achieved in instances prepared by Text Guide. All BoW experiments were performed with the python scikit-learn CountVectorizer function [14], the number of features was reduced according to the computed mutual information [15] score, and 200, 4000 and 10,000 features output from the model were tested. The rationale for these choices was that when the lowest value of 200 was considered, the BoW model would presumably perform poorly, owing to the excessively limited number of features, whereas in the other two cases, this limitation would be alleviated. We expected to observe poorer Text Guide performance in the first test case and improved performance in the other two cases. In the experiments throughout this study, unless stated otherwise, BoW with 4000 features was used.

2.3. Testing Output from Other Model Interpretability Tools

In [11], the sITFL was obtained by training a gradient boosting machine learning classifier (XGBoost) on the token features obtained from the BoW model and then extracting the feature importance directly from XGBoost. We sought to test whether using a more recent SHAP method from the XAI domain might improve the order of features in sIFTL as well as the overall classification performance based on text instances created by Text Guide. We used both the Kernel SHAP [16] and Tree SHAP [17] methods. Additionally, we computed results for randomly sorted sITFL for baseline comparison. In later experiments throughout this study, unless stated otherwise, XGBoost was used.

2.4. Using More than the First Token Occurrence

In the basic approach, when Text Guide is provided with the sITFL and the original text instance, it begins operation by searching in the original text instance for the first important token taken from the sITFL. If the important token is found, then a "pseudo sentence"—consisting of the important token and its neighbors—is extracted from the original text instance and used to create a fragment of the new text instance. Later, the next important token from the sITFL is analyzed. Consequently, if more occurrences of the first important token exist in the original text instance, they are neglected. In this study, we explored how the Text Guide performance is influenced by considering more occurrences. For our experiments, we selected 1, 2, 3, 4, 5 and all occurrences for testing.

2.5. Experiments with Text Instances That Are Not Extremely Long

The initial results obtained in [11] with a fine-tuned Albert base v2 (AlBERT) model [18] on the text instances truncated by Text Guide were slightly inferior to those obtained via the naive truncation method. An additional experiment was performed in which Text Guide was used only for instances that exceeded the model limit by a factor above 1.5. In this case, the fine-tuned AlBERT model yielded slightly improved results relative to those of the naive truncation method. However, the authors of the original Text Guide have stated that the threshold value of 1.5 was selected without any proper investigation and that this value should be explored in future research. Here, to respond to that call, we performed a more detailed analysis of the influence of this parameter, which we denote the over length threshold (OLT). We investigated OLT values of 1, 1.1, 1.2, 1.3, 1.4 and 1.5. For instances shorter than the Text Guide application threshold, we used a naive truncation

method, which resulted in analysis of only the first tokens that fit the model limit. In later experiments throughout this study, unless stated otherwise, an OLT equal to 1 was used.

2.6. Data Sets

The choice of data sets used in experiments with Text Guide followed [11], which used DMOZ [19], a well-known data set also used by other researchers for benchmarking [20] or web page classification with long text instances [21,22]. As in [11], our study focused on the 30 most common classes in the DMOZ data set and divided the original data set into three data sets on the basis of text instance length. Additionally, in order to demonstrate the Text Guide performance on a different well-known data set, we utilized the Internet Movie Data Base Large Movie Review data set [23], which defines a binary sentiment classification task. Specifically, this data set was analyzed in two settings: (1) 'IMDB FULL', i.e., the whole data set consisting of 49,582 unique data instances, and (2) 'IMDB 510 1000', i.e., the data set limited to 3609 unique instances with the length of over 510 tokens and below 1000 tokens. All the used data sets are described in Table 1.

Table 1. Data sets used in this study.

Name of the Data Set	Number of Instances	Length of Instances [Tokens]
DMOZ 510–1500	18,732	510–1500
DMOZ 1500+	8379	>1500
DMOZ 4090+	2737	>4090
IMDB FULL	49,582	all
IMDB 510–1000	3609	510–1000

2.7. Models, Training Procedure and Metrics

To create vector representations of the text instances prepared by Text Guide for most of our experiments, we used AlBERT, a small transformer model enabling less computationally expensive experiments. To test the extent to which the obtained conclusions were transferable to other pre-trained models, for final comparison of Text Guide with naive truncation, we also used the (1) RoBERTa base (RoBERTa) model [24], (2) Squeezebert uncased (SqueezeBERT) model [25], (3) Distilbert base uncased (DistilBERT) model [26] and (4) BERT base uncased (BERT) model [27]. All models were downloaded from the Huggingface Transformers server [28]. The rationale for model selection was to demonstrate the performance of models representing the state of the research transformer model architecture with a length limit of 512 tokens. Some selected models, such as AlBERT, DistilBERT or SqueezeBERT, have been demonstrated to provide high performance with limited computational cost. BERT was selected because it is the most renowned transformer baseline model, whereas RoBERTa was tested because it usually has the best possible prediction quality.

All compared models were used in their pre-trained versions to avoid bias from unoptimized fine-tuning procedures.

The instance embeddings created by the models were fed to the XGBoost classifier [29]. All machine learning classification experiments were performed with a five-fold stratified cross validation procedure. In each fold, the training, validation, and testing sets differed, and classification metrics were computed after gathering the ground truth and predicted labels for testing instances from all five folds.

Unless stated otherwise, experiments were performed with Text Guide parameters optimized for the pre-trained AlBERT model adopted from [11], as follows: (1) for all data sets $0.1 \times 510 = 51$ tokens were selected from the very beginning of the original text instance to form the beginning of the new text instance, (2) TN was set to two for the DMOZ 510–1500+ data set and to three for the other data sets, and (3) the end of the original text instance was not specifically considered.

For all experiments, we used the Matthews correlation coefficient introduced in [30] as the decisive quality measure of predictions. As demonstrated in [31], the Matthews correlation coefficient is superior to other metrics when imbalanced data sets with numer-

ous classes are analyzed, as in our DMOZ derived 30-class data sets. However, for the experiments carried out on IMDB data sets where a binary sentiment classification task was addressed, we also include more popular metrics such as F1 micro score and receiver operating characteristic area under curve (ROC AUC) score [32].

2.8. Computing Machine and Software

The in silico experiments were conducted on a single computing machine equipped with a 16-core CPU, 64 GB RAM, and Titan X 24 GB RAM GPU. The software for the experiments was implemented in Python3 with publicly available packages including Flair [7], XGBoost [29], Transformers [28], Scikit-learn [33].

2.9. Experimental Procedure

To answer the research questions described in the introduction, the necessary experiments were carried out according to the order demonstrated in Figure 2. Following the presented procedure firstly allowed the parameters of the Text Guide to be defined, and secondly allowed them to be used for final experiments with various models and data sets.

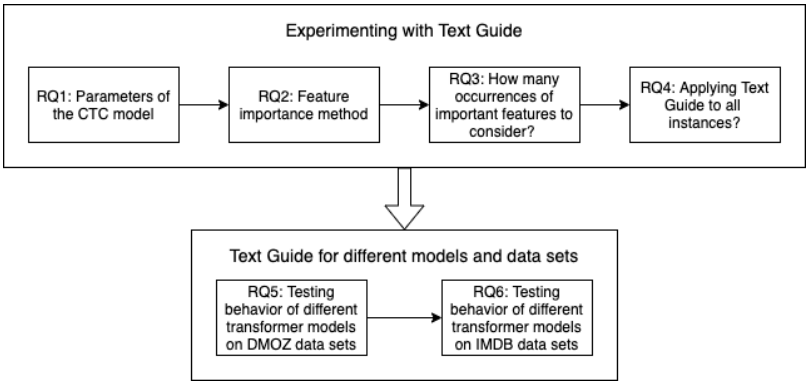

Figure 2. The order in which experiments were carried out and research questions answered.

2.10. Statistical Analysis of Results

The change in truncation method does not necessarily provide large differences in data set-wide results, for instance, due to a small fraction of text instances exceeding the adopted length limit in the data set. Therefore, we conducted the statistical bootstrap analysis to resolve uncertainties regarding the significance of differences stemming from this fact in the final comparison with the IMDB data sets. The analysis focused on the decisive MCC metric. This analysis was conducted for each transformer model separately to compare the significance of differences between naive truncation and Text Guide for each model for each IMDB data set. The following procedure was adopted: (1) we hypothesized that the model results obtained with the Text Guide would not be significantly better than those of naive truncation, (2) we bootstrapped the distribution of the metric value for the model trained with naive truncation method. In all cases, we resampled and computed the metric value 10,000 times, and (3) we computed the statistical significance of differences from the metric value of the model utilizing the Text Guide truncation method. The resulting p-values were used to accept or reject the stated hypothesis as follows: (1) p-value greater than 0.1 indicates no statistically significant differences between models, and only in this case the hypothesis was accepted, (2) p-value between 0.1 and 0.01 (marked with '*' sign) indicates a weak significant difference, (3) p-value between 0.01 and 0.001 ('**') indicates a significant difference, and (4) p-value lower than 0.001 ('***') denotes highly significant difference between models.

3. Results and Discussion

3.1. Influence of CTC Model Quality on Text Guide

The results of experiments regarding the influence of the quality of the CTC model used for the extraction of token features from the original text instances on the final classification performance with Text Guide are presented in Table 2. As predicted, the deliberately hindered BoW model, which was allowed to extract only a very limited number of token features, yielded visibly poorer results than the standard quality BoW models. In addition, the differences between superior BoW versions were small and did not allow for straightforward conclusions: for two data sets, the BoW model with 4000 token features provided better results, whereas for the data set with the longest text instances, the 10,000 token feature option appeared to be the best. Thus, we concluded that the quality of the CTC model affects the final classification performance with Text Guide, that is, the better the CTC model used to extract token features from the original text instance, the better the final classification performance.

Table 2. Influence of BoW model quality on final classification performance of the AlBERT model on data instances obtained with Text Guide. Experiments were performed with Text Guide by using five occurrences of important token features found in the original text instance, and XGBoost was used to obtain feature importance.

	Data Set		
Number of Token Features Analyzed by the BoW Model	**DMOZ 510–1500**	**DMOZ 1500+**	**DMOZ 4090+**
200	0.6099	0.5600	0.4935
4000	0.6617	0.5989	0.5137
10,000	0.6602	0.5948	0.5208

3.2. Which of the Tested Model Interpretability Tools Performed Best?

The influence of the three tools used to extract feature importance from the machine learning classifier was tested in experiments (Table 3). The random sorting of important features, which served as the baseline, was significantly outperformed by all three methods from the XAI domain when longer text instances from the DMOZ 1500+ and DMOZ 4090+ data sets were analyzed. For the DMOZ 510–1500 and DMOZ 4090+ data sets, the best results were obtained with the XGBoost feature importance method, whereas for the DMOZ 1500+ data set, Tree SHAP provided superior results. On the basis of these experiments, we believe that extracting feature importance via the XGBoost method is advisable as the default method. However, in some cases, Tree SHAP can also be considered.

Table 3. Influence of the XAI method for acquiring the feature importance used to sort the list of important token features (sITFL). Experiments were performed with Text Guide by using all occurrences of important token features found in the original text instance.

	Data Set		
Model Interpretability Tool	**DMOZ 510–1500**	**DMOZ 1500+**	**DMOZ 4090+**
Random	0.6598	0.5831	0.4937
XGBoost feature importance	0.6640	0.5958	0.5256
Kernel SHAP	0.6596	0.5943	0.5159
Tree SHAP	0.6585	0.5997	0.5121

3.3. The Benefits of Considering More Occurrences of Important Tokens in the Original Text Instance

Next, we investigated the influence of the number of important token occurrences in the original text instance considered in the Text Guide method (Table 4). As demonstrated,

in all cases, using only the first occurrence as in the original Text Guide study was inferior to using five or all occurrences, and using all occurrences was better than using five occurrences in two of the three tested data sets. Therefore, using all occurrences is advisable as the default parameter choice. However, because in some cases the best performing number of token occurrences differed, thus enabling slightly superior results to be obtained, the best results can be obtained if data set-dependent optimization is performed.

Table 4. Results obtained with a pre-trained AlBERT model and Text Guide in experiments considering various numbers of important token occurrences.

	Data Set		
Number of Occurrences of Important Tokens Considered by Text Guide	**DMOZ 510–1500**	**DMOZ 1500+**	**DMOZ 4090+**
1	0.6612	0.5895	0.5116
2	0.6606	0.5880	0.5054
3	0.6640	0.5886	0.5050
4	0.6606	0.5979	0.5165
5	0.6617	0.5989	0.5137
All	0.6628	0.5958	0.5168

3.4. Is Text Guide Useful for Text Instances Only Slightly Exceeding the Model Limit?

We performed experiments in which Text Guide was used for a text corpus with text instances only slightly exceeding the LTI model limit (Table 5). The experiments allowed us to assess whether using Text Guide for all text instances is better than using this method for only text instances substantially exceeding the LTI model limit. The results obtained for the DMOZ 510–1500 data set demonstrated that the best performing option was using Text Guide for all text instances. Our findings additionally indicated that Text Guide is superior to the most naive truncation method, which uses only the initial tokens of the original text instance.

Table 5. Experiments on the DMOZ 510–1500 data set regarding the over length threshold (OLT), i.e., the threshold length of the text instance from which Text Guide is to be used. For shorter text instances, a naive truncation method was used. Experiments were performed with Text Guide by using all occurrences of important token features found in the original text instance.

OLT [Threshold Value]	**Resulting Threshold Number of Tokens for The Instance to Be Truncated with Text Guide**	**Results for The DMOZ 510–1500 Data Set**
1	510	0.6640
1.1	561	0.6544
1.2	612	0.6492
1.3	663	0.6443
1.4	714	0.6433
1.5	765	0.6439

3.5. Are the Results of Text Guide and AlBERT Transferable to Other Models?

To test the extent to which experiments with the AlBERT model might be transferable to other models, wRoe performed test runs with the RoBERTa, Squeezebert, BERT and DistilBERT models. The results (Table 6) indicated that adopting Text Guide with parameters optimized for the AlBERT model, in comparison to the naive truncation method (1) allowed improvement of performance of RoBERTA and SqueezeBERT, (2) hindered performance of BERT and (3) improved performance of DistilBERT in two out of three tested data sets. These observations suggest that before applying Text Guide, model and data set-specific parameter optimization should be conducted. In this regard, detailed analysis of the results

in Table 6 additionally indicated that for some models and data sets, Text Guide leads to marginal improvement, whereas in other cases, such as SqueezeBERT and the DMOZ 4090+ data set, the benefits of applying Text Guide can be excellent.

Table 6. Experiments performed with selected pre-trained transformer models. Text Guide parameters were adopted (1) from the optimization procedure for the AlBERT model from the original Text Guide study and (2) according to suggestions regarding other parameter values, as presented in this study.

		Data Set		
Model	**Truncation Method**	**DMOZ 510–1500**	**DMOZ 1500+**	**DMOZ 4090+**
RoBERTa	Naive	0.8350	0.7466	0.6793
	Text Guide	0.8358	0.7767	0.7158
BERT	Naive	0.7952	0.7168	0.6427
	Text Guide	0.7673	0.7047	0.6353
DistilBERT	Naive	0.7997	0.7188	0.6440
	Text Guide	0.7973	0.7412	0.6794
SqueezeBERT	Naive	0.8107	0.7273	0.6485
	Text Guide	0.8235	0.7720	0.7295

3.6. Can Text Guide Be Used Successfully for Other Data Sets?

To demonstrate that Text Guide can be successfully deployed on another data set, for the IMDB FULL and IMDB 510-1000 data sets, we repeated the final experiments carried out earlier for DMOZ data sets with various models. The experiments on DMOZ data sets were conducted without any data set-specific Text Guide optimization. Instead, we adopted parameter values that performed well for the AlBERT model and DMOZ data sets in previously described experiments. The results presented in Table 7 allow us to conclude that: (1) For three out of four tested models, the utility of the Text Guide was evident when considering the IMDB 510–1000 data set in which all data instances could be transformed with Text Guide. Specifically, for the RoBERTa model, the positive impact of the Text Guide was highly significant ($p = 0.0001$). However, there was no significant ($p = 0.7118$) improvement provided by the Text Guide for BERT. Text Guide used with DistilBERT significantly ($p = 0.01$) improved performance over naive truncation. The application of the Text Guide for the SqueezeBERT model was significant to a minor extent ($p = 0.0135$). (2) In the case of IMDB FULL data set, given the great dominance of text instances shorter than 510 tokens, the obtained differences in results were mostly inconclusive. The RoBERTa model trained with the Text Guide was the only one with weak significance ($p = 0.0873$) in improvement over naive truncation. Finally, there were no significant ($p = 0.9078$) improvements for the BERT model, as was also the case for the DistilBERT ($p = 0.6238$) and SqueezeBERT ($p = 0.1106$) models.

Table 7. Experiments performed with selected pre-trained transformer models on IMDB data sets. For both data sets and all models, the adopted parameters for Text Guide were: XGBoost feature importance as the feature importance method, OLT = 1, all occurrences of important tokens were taken into consideration, the part of the beginning of the original text instance used by Text Guide = 0.1, the part of the ending of the original text instance used by Text Guide = 0, number of tokens surrounding the important token to be used for creating the pseudo sentence = 2.

		Data Set					
		IMDB FULL			IMDB 510–1000		
Model	**Truncation Method/Metric**	**ROC AUC Score**	**F1 Micro**	**MCC**	**ROC AUC Score**	**F1 Micro**	**MCC**
RoBERTa	Naive	0.9483	0.8767	0.7536	0.8617	0.7886	0.5738
	Text Guide	0.9492	0.8787	0.7576 *	0.8942	0.8132	0.6234 ***
BERT	Naive	0.9282	0.8493	0.6986	0.8652	0.7817	0.5595
	Text Guide	0.9265	0.8472	0.6944	0.8545	0.7778	0.5516
DistilBERT	Naive	0.924	0.8447	0.6894	0.8619	0.7869	0.5702
	Text Guide	0.9237	0.8442	0.6884	0.8758	0.8033	0.6032 **
SqueezeBERT	Naive	0.9343	0.8559	0.7119	0.8856	0.806	0.6089
	Text Guide	0.9353	0.8579	0.7158	0.902	0.8204	0.638 *

'*' sign indicates a weakly significant difference, '**' indicates a significant difference, '***' denotes highly significant difference between model versions deployed on text instances truncated by Text Guide and in the naive manner.

3.7. Gains over the Original Text Guide Method

In order to facilitate comparison of results obtained in the original Text Guide paper [11] with the best results of here conducted experiments, we gather the results obtained by comparable pre-trained AlBERT models in Table 8. Based on this comparison, it seems evident that gains from the introduced modifications are visible in all cases and become more evident along with the increasing length of text instances.

Table 8. Comparison of results achieved by the original Text Guide method [11] and best results from our study. The presented results were obtained by pre-trained AlBERT models. Only the varying parameters of the Text Guide were mentioned.

Data Set	**Text Guide Version**	**MCC**	**Gain Over Study [1]**	**XAI Method**	**Important Token Occurrences**
DMOZ 510–1500	From study [11]	0.659	0.005	XGBoost feature importance	1
	Best from this study	0.6640		XGBoost feature importance	3
DMOZ 1500+	From study [11]	0.5913	0.0084	XGBoost feature importance	1
	Best from this study	0.5997		Tree SHAP	All
DMOZ 4090+	From study [11]	0.5138	-	XGBoost feature importance	1
	Best from this study	0.5256	0.0118	XGBoost feature importance	All

3.8. The Added Benefit of Increased Interpretability

In the final part of the discussion, we note an added benefit of utilizing the Text Guide: the increased interpretability of the whole classification process owed to the highly interpretable language model used to construct the sIFTL. When the naive truncation method is concerned, it is difficult to interpret which text fragments were precisely the focus of the later-used deep learning language model without any dedicated methods from

the explainable AI domain. However, when the Text Guide is concerned, inspecting the obtained text instance allows us to explicitly observe which important tokens from the sIFTL were used to form the text instance of the desired length. This fact increases the interpretability of the classification process, as it allows for interpretations on a model level, such as "the overall model performance was achieved by pointing the attention towards a known list of important tokens" and instance level, for example, "for the given instance, the model decision was based on analysis of text fragments surrounding the following key tokens". Of course, using the Text Guide does not indicate precisely which fragments of the text instance were considered more and less important by the deep learning model. Still, the evident benefit is narrowing the area of possible speculation.

4. Conclusions

In this study, we explored selected means of optimizing and testing the Text Guide method introduced in earlier research. Six research questions were defined and answered, which allowed us to propose recommendations regarding improved default parameters for Text Guide and confirmed the dependency of the final classification performance from selected methods used by Text Guide. Our research formulated clear and practical contributions for future users of Text Guide. In particular, we found that the quality of the CTC model and XAI tool, both of which are responsible for extracting relevant tokens and their neighbors from the original text instance in the correct order, influence Text Guide performance. Our findings also demonstrated that Text Guide could provide superior performance to that of the naive truncation method for several recent transformer models and for well-known DMOZ and IMDB large movie review data sets.

However, we also note a limitation of our study and Text Guide method, i.e., that model and data set-specific parameter optimization is recommended before Text Guide is applied. This may be important and challenging, especially from the computational point of view, with future research focusing on using Text Guide together with fine-tuned transformer models.

Author Contributions: Conceptualization, methodology, writing—original draft, software: K.F.; writing—review and editing, supervision, funding acquisition, project administration: W.K.; writing—review and editing, investigation: E.G., T.A. and M.W.; methodology, data curation, M.R.D. All authors have read and agreed to the published version of the manuscript.

Funding: This study was supported in part by a research grant from the Office of Naval Research (N000141812559) and was performed at the University of Central Florida, Orlando, FL, USA.

Institutional Review Board Statement: Not applied to this study.

Informed Consent Statement: Not applied to this study.

Data Availability Statement: The study did not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pennebaker, J.W.; Francis, M.E.; Booth, R.J. Linguistic inquiry and word count: LIWC 2001. *Mahway Lawrence Erlbaum Assoc.* **2001**, *71*, 2001.
2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Polosukhin, I. Attention is all you need. *arXiv* **2017**, arXiv:1706.03762.
3. Kitaev, N.; Kaiser, Ł.; Levskaya, A. Reformer: The efficient transformer. *arXiv* **2020**, arXiv:2001.04451.
4. Choromanski, K.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlos, T.; Weller, A. Rethinking attention with performers. *arXiv* **2020**, arXiv:2009.14794.
5. Zaheer, M.; Guruganesh, G.; Dubey, A.; Ainslie, J.; Alberti, C.; Ontanon, S.; Ahmed, A. Big bird: Transformers for longer sequences. *arXiv* **2020**, arXiv:2007.14062.
6. Beltagy, I.; Peters, M.E.; Cohan, A. Longformer: The long-document transformer. *arXiv* **2020**, arXiv:2004.05150.
7. Akbik, A.; Bergmann, T.; Blythe, D.; Rasul, K.; Schweter, S.; Vollgraf, R. FLAIR: An easy-to-use framework for state-of-the-art NLP. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), Minneapolis, MN, USA, 2–7 June 2019; pp. 54–59.

8. Sun, C.; Qiu, X.; Xu, Y.; Huang, X. How to Fine-Tune BERT for Text Classification? In *China National Conference on Chinese Computational Linguistics*; Springer: Cham, Switzerland, 2019; pp. 194–206.
9. Min, S.; Zhong, V.; Socher, R.; Xiong, C. Efficient and robust question answering from minimalcontext over documents. *arXiv* **2018**, arXiv:1805.08092.
10. Ding, M.; Zhou, C.; Yang, H.; Tang, J. CogLTX: Applying BERT to Long Texts. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 12792–12804.
11. Fiok, K.; Karwowski, W.; Gutierrez, E.; Davahli, M.R.; Wilamowski, M.; Ahram, T.; Al-Juaid, A.; Zurada, J. Text Guide: Improving the Quality of Long Text Classification by a Text Selection Method Based on Feature Importance. *IEEE Access* **2021**, *9*, 105439–105450. [CrossRef]
12. Lundberg, S.; Lee, S.I. A unified approach to interpreting model predictions. *arXiv* **2017**, arXiv:1705.07874.
13. krzysztoffiok TextGuide. Available online: https://github.com/krzysztoffiok/TextGuide (accessed on 29 July 2021).
14. Pedregosa, F. Sklearn. Feature_Extraction. Text. CountVectorizer. 2013. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html (accessed on 7 September 2021).
15. Latham, P.E.; Roudi, Y. Mutual Information. *Scholarpedia* **2009**, *4*, 1658. [CrossRef]
16. Lundberg, S.; Erion, G.; Lee, S.I. Consistent individualized feature attribution for tree ensembles. *arXiv* **2018**, arXiv:1802.03888.
17. Lundberg, S.M.; Erion, G.; Chen, H.; DeGrave, A.; Prutkin, J.M.; Nair, B.; Lee, S.I. From local explanations to global understanding with explainable AI for trees. *Nat. Mach. Intell.* **2020**, *2*, 56–67. [CrossRef]
18. Lan, Z.; Chen, M.; Goodman, S.; Gimpel, K.; Sharma, P.; Soricut, R. Albert: A lite bert for self-supervised learning of language representations. *arXiv* **2019**, arXiv:1909.11942.
19. Sood, Gaurav, Parsed DMOZ Data 2016. Available online: https://doi.org/10.7910/DVN/OMV93V (accessed on 7 September 2021).
20. Partalas, I.; Kosmopoulos, A.; Baskiotis, N.; Artieres, T.; Paliouras, G.; Gaussier, E.; Galinari, P. Lshtc: A benchmark for large-scale text classification. *arXiv* **2015**, arXiv:1503.08581.
21. Lee, J.-H.; Yeh, W.-C.; Chuang, M.-C. Web page classification based on a simplified swarm optimization. *Appl. Math. Comput.* **2015**, *270*, 13–24. [CrossRef]
22. Özel, S.A. A web page classification system based on a genetic algorithm using tagged-terms as features. *Expert. Syst. Appl.* **2011**, *38*, 3407–3415. [CrossRef]
23. Maas, A.; Daly, R.E.; Pham, P.T.; Huang, D.; Ng, A.Y.; Potts, C. Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 142–150.
24. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
25. Iandola, F.N.; Shaw, A.E.; Krishna, R.; Keutzer, K.W. SqueezeBERT: What can computer vision teach NLP about efficient neural networks? *arXiv* **2020**, arXiv:2006.11316.
26. Sanh, V.; Debut, L.; Chaumond, J.; Wolf, T. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv* **2019**, arXiv:1910.01108.
27. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
28. Wolf, T.; Chaumond, J.; Debut, L.; Sanh, V.; Delangue, C.; Moi, A.; Cistac, P.; Funtowicz, M.; Davison, J.; Shleifer, S. Transformers: State-of-the-Art Natural Language Processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Stroudsburg, PA, USA, 16–20 November 2020; pp. 38–45.
29. Available online: https://xgboost.readthedocs.io/en/latest/python/python_intro.html (accessed on 14 May 2021).
30. Matthews, B.W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochim. Et Biophys. Acta* **1975**, *405*, 442–451. [CrossRef]
31. Chicco, D.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genom.* **2020**, *21*, 1–13. [CrossRef]
32. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
33. Available online: https://scikit-learn.org/stable/ (accessed on 14 May 2021).