ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

School of Information Sciences and Technology
Department of Informatics
Athens, Greece

Bachelor Thesis
in
Computer Science

Data Augmentation for Legal Document Classification

Platon Karageorgis

Supervisors: Prof. Ion Androutsopoulos
Department of Informatics
Athens University of Economics and Business

Dr. Dimitris Pappas
Department of Informatics
Athens University of Economics and Business

September 2023

Platon Karageorgis
Data Augmentation for Legal Document Classification
September 2023
Supervisors: Prof. Ion Androutsopoulos, Dr. Dimitris Pappas

Athens University of Economics and Business
School of Information Sciences and Technology
Department of Informatics
Natural Language Processing Group
Athens, Greece

Abstract

Data augmentation is a technique for generating artificial instances of data. It is widely known in Computer Vision and it has been used for years in Natural Language Processing (NLP). Despite its widespread use, its application in the legal domain remains relatively unexplored. Legal texts, characterized by intricate language and lengthy documents, present unique challenges that demand specialized solutions. We implement a variety of augmentation methods aiming to demonstrate that data augmentation can enhance significantly the performance of language models in legal document classification. LEGAL-BERT, a specialized language model designed for legal text, serves as the foundation for our experiments. We demonstrate that the legal domain presents complexities beyond initial observation, and we attempt various approaches to overcome the barriers that are raised during our experiments. Finally, we provide compelling graphs and figures to support our claims.

Περίληψη

Η επαύξηση δεδομένων είναι μία τεχνική που στοχεύει στη γέννηση νέων, τεχνητών δεδομένων. Είναι ευρέως γνωστή τεχνική σε εκείνους που ανήκουν στο χώρο της μηχανικής όρασης και χρησιμοποιείται χρόνια στο χώρο της επεξεργασίας φυσικής γλώσσας. Παρά την πληθώρα των εφαρμογών της στον τομέα αυτό σε γενικότερο πλαίσιο, είναι γεγονός ότι η συνεισφορά της σε νομικά κείμενα δεν έχει διερευνηθεί σε ικανοποιητικό βαθμό. Τα νομικά κείμενα χαρακτηρίζονται από την εκτενή χρήση πολύπλοκου λεξιλογίου αλλά και από τη μεγάλη έκταση τους. Έτσι, έχουν δημιουργήσει την πεποίθηση πως απαιτούν ειδική αντιμετώπιση, όσον αφορά την επεξεργασία φυσικής γλώσσας αλλά και κατ'επέκταση την επαύξηση δεδομένων. Υλοποιούμε μία σειρά από τεχνικές επαύξησης δεδομένων, προσπαθώντας να δείξουμε τη θετική συνεισφορά που μπορεί να έχει η επαύξηση δεδομένων στα γλωσσικά μοντέλα που πραγματοποιούν ταξινόμηση νομικών κειμένων. Παράλληλα, επιδεικνύουμε και αποδεικνύουμε τη βαθύτερη πολυπλοκότητα που χαρακτηρίζει τα νομικά κείμενα και ελισσόμαστε με μία σειρά από τεχνικές προκειμένου να αντιμετωπίσουμε τα εμπόδια που προκύπτουν κατά τη διάρκεια των πειραμάτων μας. Τέλος, παρουσιάζουμε ενδιαφέροντα γραφήματα και σχήματα τα οποία υποστηρίζουν τους ισχυρισμούς μας.

Acknowledgements

I would like to thank Professor Ion Androutsopoulos for giving me the opportunity to work in an engaging field like NLP alongside a team of professionals. I am humbled by his trust and patient guidance throughout all these months and I hope he will remain an inspiration for the next NLP enthusiasts to come. Moreover, I would like to express my heartfelt thanks to Dr. Dimitris Pappas, who was there whenever I needed him to answer my questions and provide me with guidance. His contribution was instrumental to my growth as an aspiring researcher. I am also deeply appreciative of Dr. Prodromos Malakasiotis for his state-of-the-art insights, which elevated the caliber of this thesis. Last but not least, as this thesis marks the end of my Bachelor's Degree at Athens University of Economics and Business, I would like to thank the entire faculty and staff for the support they provided over the past five years.

Contents

1 Introduction

The rise of Artificial Intelligence and mainly Natural Language Processing during the last 5 years has been outstanding. It all started with the mass shift from Convolutional Neural Networks and Recurrent Neural Networks to Transformers. This led to the development of BERT ([Dev+18]) which is a type of Transformer and remains to this day a very powerful tool. The latest update is the ascension of Large Language Models, with the lead star being Chat-GPT, whose capabilities have traveled to households of every cornerstone of the planet. At the time that this thesis is written, LLMs are admittedly the hot topic that every AI researcher has to mention in any NLP-related task.

Despite the surprisingly good results of LLMs across a wide spectrum of applications, in a case where there is an option of an older more traditional non-generative model such as BERT, and the end task is specific e.g. classification, they just might be the runner-up. The challenge that arises though when BERT is fine-tuned, is a common barrier in the Deep Learning era. The amount of data needed to fine-tune BERT up to the point where the leading parameters (usually F1-score) are optimized, most of the time surpasses the realistic number of data available in a real-world task. Legal documents are a great example of this, as they have long but few texts on many occasions. Data augmentation is a technique that increases the amount of available data, by altering the existing ones or creating entirely new data rows from scratch. It is widely known in Computer Vision as it is more straightforward to implement when dealing with images. On the other hand, it is not so simple to apply it in NLP, as text documents are not as easily augmented.

For example, assume the sentence "The court decided to postpone the hearing for next week". The most intuitive manner of augmentation would be to replace several words with their synonyms to generate a different data row that also maintains the original meaning. For instance, the above sentence could be transformed to "The court chose to put off the hearing for next week". In this case, the augmentation process was effectively implemented, but is it really easy to automate this synonym replacement process using a language model? The word "decided" has multiple synonyms in general, but in this specific context, a potentially big subset of them will not be a fitting choice. This adds a whole new level of complexity in text augmentation that sources from the perplexity of human-generated languages. Ultimately, it can be observed that even when using the most intuitive augmentation method, text document augmentation is more complex than image augmentation. The topic of this thesis covers the effect of a selection of data augmentation techniques in legal document classification.

1.1 Motivation and Problem Statement

Data augmentation is likely familiar to those who specialize in Computer Vision. Image processing is a domain that people have been working on far before Transformers or Neural Networks were developed. And that is all it takes to augment data in Computer Vision, as the data rows are not texts, but images. A mere rotation, a slight alteration on the pixel settings, or an even more complicated method like image compression using Singular Value Decomposition (SVD) truncation, could be enough to generate artificial data. Conversely, the application methods of Data augmentation in Natural Language Processing vary and have quite different backgrounds due to the natural difference between text and images. The prime inspiration of this thesis is Pappas et al. ([PMA22]). The paper considered several data augmentation methods that included an artificial cloze-style machine reading comprehension dataset, back-translation, information retrieval, word substitution using word2vec and masked-lm, question generation, and context addition. The results of the experiments were very positive and showed that data augmentation can be a key contributing factor in improving the performance of a model. More details regarding the work on this paper will be provided in section 2 that will cover related work on various data augmentation techniques. The augmentation techniques that will be inherited from the mentioned paper are both word substitution implementations, meaning word2vec and masked-lm, and finally back-translation. These methods will be extensively analyzed in subsequent sections.

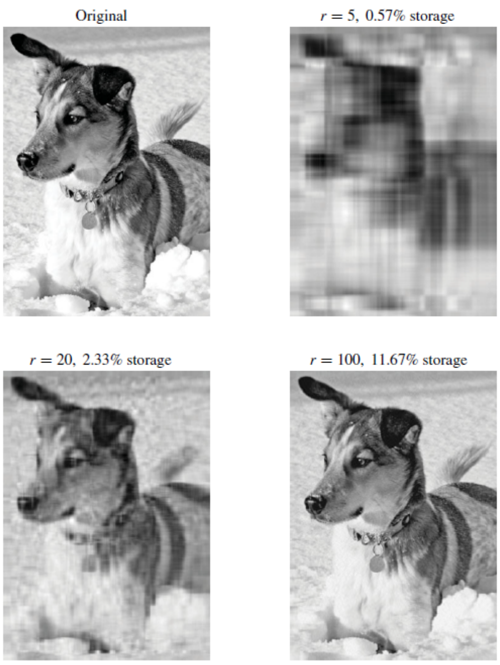

Fig. 1.1: Image compression of Mordecai the snow dog, truncating the SVD at various ranks r and therefore generating artificial image data. All rights reserved to Data-Driven Science and Engineering 2nd Edition by Steven L. Brunton and J. Nathan Kutz

As we mentioned in the introductory paragraph, the previously stated techniques will be implemented in legal document classification. The datasets will be drawn from Chalkidis et al. ([Cha+22]) and all have in common the fact that they belong in the legal domain. It should be noted that the classification categories will differ between the datasets as the tasks are not similar. LexGLUE, as in, Legal General Language Understanding Evaluation is a benchmark collection of datasets created to evaluate models in legal tasks. The files and paperwork needed for legal tasks have always been a factor that makes law a challenging sector to contribute to. AI and NLP aim to take this a step further by securing automation in tasks that have been troubling legal practitioners. LexGLUE makes a step in that direction by designing a benchmark that will benefit researchers in Natural Language Understanding (NLU) tasks by improving model performance and evolving their techniques. In this thesis, we use SCOTUS, a part of the LexGLUE dataset for multi-class classification. In addition, we use ECtHR and UNFAIR-ToS, similar parts of LexGLUE, that aim for multi-label classification. The selection was based on the amount of available data, as we expect datasets with less available data to get an additional performance boost using more data. More information regarding the work on the paper will be once again provided in section 3 that describes our experiments.

Moreover, the datasets described above will be augmented using various techniques that will be extensively analyzed in section 2. Subsequently, original and artificial data will be merged, to train the chosen language model. The model that will be used in all of the experiments in this thesis, will be LEGAL-BERT designed by Chalkidis et al. ([Cha+20]). This paper acknowledges the fantastic results that BERT achieves in NLP but also underlines that it fails to produce equally good results in specific scientific areas. Therefore, the authors attempt to manufacture a version that will be focused solely on the legal domain. Taking into account the datasets that will be used for the experiments, LEGAL-BERT seems like an excellent choice for our task. It was pre-trained on legal data and, therefore, is expected to perform exceptionally well in the legal domain. In fact, it is not only one of the models that LexGLUE used to assess the generated benchmark dataset, but it is also the best-performing one in the majority of the metrics. To conclude, we aim to replicate the results conducted in Chalkidis et al. ([Cha+22]) by applying data augmentation to the mentioned datasets and LEGAL-BERT's performance in LexGLUE constitutes the main baseline for our results.

Having analyzed briefly the necessary background, we can now state the problem that this thesis will be devoted to. Data augmentation is a technique that generates artificial data and it has been proved in past work that it can be quite beneficial in improving model results, especially in the biomedical domain. Using the results of LexGLUE, a legal benchmark dataset, we apply a variety of data augmentation methods to prove that data augmentation can be beneficial in the legal domain as well. The model used to perform the experiments is LEGAL-BERT, the best model overall in LexGLUE.

1.2 Thesis Structure

The remainder of this thesis is structured as follows:

Chapter 2

The next chapter of this thesis will cover at first the data augmentation methods applied in the biomedical domain from Pappas et al. ([PMA22]) which is the paper that we inherit our methods. Then, there will be a discussion on the techniques put into effect in this thesis, along with other implementations in the legal domain. Lastly, the final section will concern a few augmentation methods outside the spectrum of specialized scientific areas, like the biomedical and the legal ones.

Chapter 3

The third chapter will first present the background of the datasets and the model used for the experiments, as well as the metrics and the baselines that were adopted to support them. Subsequently, there will be a detailed presentation of the experiments that were carried out, along with their corresponding results. These experiments aim to compare the effects of our techniques with the benchmark results produced by the original, non-augmented datasets of LexGLUE. Moreover, there will be a discussion on the quality of the results, accompanied by graphs, that will enlighten the total effect of our work.

Chapter 4

The fourth and final chapter will primarily sum up the goals of this thesis and comment on the results of the experiments. Last but not least, there will be suggestions for future work in data augmentation in the legal domain, but also in NLP in general.

Data Augmentation Methods 2

This section is separated into three parts. The first part explores the seven augmentation methods implemented in the biomedical domain proposed by Pappas et al. ([PMA22]). The second part tackles data augmentation in the legal domain and is further segmented into two components. The initial one covers our work and the latter discusses similar augmentation methods in a legal context, implemented by the research community. Finally, the third section introduces a couple of extra augmentation methods generally in NLP, not necessarily concerning the legal or biomedical domain.

2.1 Data Augmentation in Biomedical Domain

In 2022, Pappas et al. ([PMA22]) investigated the impact of seven data augmentation techniques in the context of factoid question answering, with a specific emphasis on the biomedical domain. A selection of models in the BERT family was recruited for this cause, and the employed data were drawn from the BIOASQ challenge ([Tsa+15]). It's pertinent to note that the data have a specific shape, each data row is a question-snippet-answer triple. As mentioned in the introduction we have inherited a subset of these augmentation techniques and we will assess their effect on the legal domain. Before continuing with this assessment though, we will focus on the work done in a biomedical context, and we will describe the aspects of each augmentation technique, as some of these are implemented in this thesis.

2.1.1 Models

Before delving into the analysis of augmentation methods, let's introduce the models that will be used throughout the paper. The authors use DISTILBERT ([San+20]), BIOBERT ([Lee+19]) and ALBERT ([Lan+20]) which are BERT-like pre-trained models. DISTILBERT is a lighter and faster version of the basic version of BERT, bert-base-uncased. BIOBERT is a version of BERT pre-trained on biomedical documents, and ALBERT is like DISTILBERT, a lighter version of BERT as it attempts to limit BERT's computational demands. They are fine-tuned on SQUAD ([Raj+16]) or SQUAD-v2 ([RJL18]) which are Stanford's question-answering datasets, and ALBERT is further fine-tuned on BIOASQ data aiming to be a strong baseline. There are further modifications to the models that will not be mentioned, as we aim to focus on the augmentation techniques.

2.1.2 Word2vec

To begin with, the first augmentation method is implemented using word2vec, a well-known term to those that specialize in NLP, initially proposed by Mikolov et al. ([Mik+13]). Word2vec represents words in a vector space where each word is depicted by a unique vector known as a word embedding.

The research team of Pappas et al. ([PMA22]) analyzed question-snippet-answer training instances and began by examining the word tokens in the snippet, excluding stop-words. For each token w_i within the snippet, ranging from 1 to n, the authors selected up to k_i most similar words w_j from the vocabulary based on cosine similarity between word embeddings $\vec{w}i$ and $\vec{w}j$ (with cos($\vec{w}i$, $\vec{w}j$) $\geq$ C, where C is the similarity threshold). This process resulted in generating (k1+1)(k2+1)...(kn+1)-1 artificial training instances, with at least one token changed in each instance. It's worth noting that the total number of neighboring words k_i (which is set at 10) for a given word may not always reach the specified value. This is due to the authors' choice of a stringent 0.95 similarity threshold, which results in a smaller pool of words exceeding this threshold compared to k_i.

Furthermore, the authors sampled training instances ranging from 10,000 to 100,000 and integrated them into the training dataset, contributing to a substantial increase in the F1-score by up to six percentage points. Finally, the decision to set the values for the parameters K (maximum similar words to search) and C (similarity threshold) to 10 and 0.95, respectively, was informed by preliminary experiments conducted on development data.

2.1.3 Masked-LM

The second augmentation method is masked-lm. This method is similar to word2vec a word substitution technique, but operates using a different scheme. It takes advantage that BERT-like models are trained for Masked Language Modelling. These models are exposed to millions of sentences with hidden words and are tasked with identifying the missing token. The authors employ BIOLM ([Lew+20]), specifically a ROBERTA-LARGE model pre-trained on PUBMED Baseline Repository[1], PMC, and MIMIC-III ([ZPT18]), enriched with a new vocabulary sourced from PUBMED.

The difference in this approach in comparison with word2vec is that each potential replacement word w_j for an original word w_i in the snippet must meet the condition p(w_j)$\geq$P (instead of cos(wi,wj) $\geq$ C enforced in word2vec). The term p(w_j) represents the probability assigned to w_j by the pre-trained model. Additionally, they arrange the candidate

[1]See https://pubmed.ncbi.nlm.nih.gov

replacements w_j for each w_i based on their p(w_j) values. They establish P=0.95, grounded in preliminary experiments conducted on development data.

It's noteworthy that in their experiments, models trained using BIOLM for data augmentation performed similarly to models trained using word2vec for data augmentation, but with an important difference. For BIOLM, the optimal performance is achieved when incorporating 50k artificial examples, in contrast to 10k for word2vec. The authors hypothesize that this happens due to BIOLM suggesting words that align closely with the specific context of the word being replaced, whereas word2vec takes a more direct approach by comparing each original word with candidate replacements.

2.1.4 Back-Translation

The third augmentation technique is back-translation, a technique widely applied in NLP tasks ([SKF21];[Fen+21]). This method involves translating training examples from a source language to a pivot language and then back to the source language, thus generating paraphrased versions of the original text. Initially, the authors utilized French as the pivot language and later added Spanish and German.

For each question-snippet-answer training triplet within BIOASQ, there are two additional triplets created by performing back translation on either the question or the snippet. If any of the new triplets happen to match the original, they are discarded. This process yields 4,901 new training examples when pivoting solely to French, and 15,593 when extending the pivot languages to include Spanish and German.

Adding the generated data into the BIOASQ training dataset results in a nearly 2% enhancement in the development data F1-score, regardless of using one or three pivot languages. Also, the authors attempted to run the model only with artificial data and the results were almost identical with the original ones, proving therefore the quality of artificial data. Finally, the authors mention that simpler methods like word2vec, can yield more substantial gains with fewer artificial training instances, and perhaps are a better choice regarding augmentation methods.

2.1.5 Additional Augmentation Methods

Up to this point, we have discussed techniques that will be further explored in this thesis, to evaluate their performance in a legal context. The following four methods will not be the focus of our work. Firstly, the authors implement an artificial cloze-style Machine Reading Comprehension (MRC) augmentation method that leverages biomedical articles to generate question-snippet-answer triples, significantly expanding the dataset. Secondly, the authors use information retrieval systems to extract relevant content from sources like PUBMED. The third technique employs T5 ([Raf+20]) for question generation, creating new question-snippet-answer triples from BIOASQ training data, with modifications to highlighted areas. Lastly, context addition adds neighboring sentences to snippets, enhancing context awareness. It uses variables to determine the extent of sentence addition, with a focus on maintaining character limits.

All of these approaches achieved results surpassing the established baseline and were deemed successful by their respective authors. However, it's worth noting that the information retrieval method yielded less data than initially expected, which influenced our decision to exclude it from our thesis. Question generation, too, does not align with the nature of our task, which primarily focuses on classification rather than question generation. Additionally, the artificial cloze-style Machine Reading Comprehension (MRC) augmentation and context-aware methods are not suitable for our dataset, which is centered around LexGLUE benchmark datasets, and these techniques typically involve snippets extracted from articles.

2.2 Data Augmentation In Legal Domain

Data augmentation in legal texts is not widely recognized as a prominent domain within the field of Natural Language Processing (NLP). Nevertheless, in recent years researchers have demonstrated an increased interest in the field as Csányi et al. ([CO21]) described. In this section, we refer to relevant work in the field, but before that, we focus on the methods implemented in this thesis.

2.2.1 Our Work

Our work consists of three main methods that perform data augmentation. They all have been inherited from Pappas et al. ([PMA22]) and we shortly discussed them above in a biomedical context. In this part of the thesis, we will analyze word2vec, masked-lm, and back-translation once more, but from our scope this time, adding more details and examples.

2.2.1.1 Word2vec

Text augmentation using word2vec is essentially an advanced word substitution. Each word in the text is examined using pre-trained legal embeddings called Law2Vec ([CK19]). Law2Vec's extensive corpus comprises over 123,000 documents with 492 million tokens. Texts were preprocessed by removing non-UTF8 characters and tokenizing using NLTK.

Additionally, all words were converted to lowercase, and numbers were replaced with 'D' ([CA17]) to maximize matches during the embedding search.

When searching for word embeddings, there are two outcomes. If a corresponding embedding is not found, the word remains unchanged in the augmented text. However, if the embedding is found, the method retrieves its vector representation and searches for its n most similar embeddings using cosine similarity. The formula is the same as in the biomedical implementation, $\cos(\vec{w}_i, \vec{w}_j) \geq \mathrm{C}$ (where C is the similarity threshold). These embeddings constitute potential candidates to replace the initial word, forming a 'bag of words' for each word in the text. It's important to note that the initial word is also included in its bag, which means it may still appear in the augmented text, even if other candidates are selected, as seen in figure 2.1.

As we mentioned in the previous description of word2vec in a biomedical context, it's not certain whether the output will contain in total n neighboring words for a given word. This occurs because for the method to be precise, the resulting words must truly be "close" to the target word to preserve the overall sentence meaning. This is achieved by setting a threshold that will set how different are the words allowed to be compared to the initial one. We set the threshold C to be 0.95 after tuning it using the development dataset. This means we allow the word to deviate from the original by up to 5%. For a random word, the bag of words created via this process can have n words, but it can also have less or even none. In practice, due to the high threshold, the latter case will likely be more common.

Finally, after processing the entire text and creating the bags of words for each word, the augmentation process begins. For each word in the text, a replacement word is randomly selected from its corresponding bag. This process can be repeated indefinitely, potentially generating a large number of text variations. However, it's essential to note that after a certain point, the generated texts may become identical to those previously created. The maximum number of possible text variations can vary and depends on several factors, including the similarity threshold, text length, and randomness. Also, some texts may contain words more likely to have suitable replacement candidates due to their inherent characteristics.

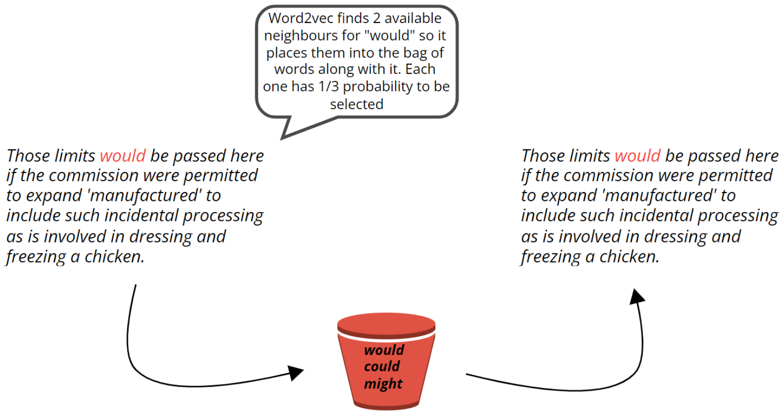

Fig. 2.1: Word2vec example in a chunk of our SCOTUS legal texts.

2.2.1.2 Masked-LM

The implementation of masking using a language model is relatively straightforward, with one exception. Due to our documents being way too long for the model to process (some of the biggest documents have over 180,000 characters), we have to make repeated segmentations before feeding the text to the model. To enhance the effectiveness of our code, we utilize a tailored text-breaking technique, which will be described in detail below.

Our approach to text segmentation is as follows. Knowing that LEGAL-BERT has a token limit of 512, we set the maximum number of BPEs, which represent sub-word units, to be 500. In a few words, Byte-Pair Encodings (BPEs) is a sub-word tokenization technique used to break down words into smaller units for natural language processing tasks, such as machine translation and text generation. They were initially proposed by Sennrich et al. ([SHB16]). This choice accounts for potential BPE tokenization that may split a token into two, causing the total token count to exceed 512. Therefore, we set the threshold slightly lower to avoid exceeding the limit. As we process the document, the reading process pauses when the token limit, based on BPEs, is reached. We then backtrack through the text to identify the last paragraph change before the point where the limit was exceeded. If a paragraph change is found, we split the text at that point, and this chunk becomes a paragraph. If no paragraph change is detected, we repeat the process by looking for full stops. In rare cases, such as a document with no full stops or paragraph changes within a 500-token span, we break the text at the last line change. This process is repeated for the next chunk, and we meticulously track the type of segmentation that occurred between each pair of chunks. The goal is to assemble them at the end of the iteration, once the entire document has been processed. This careful process ensures that the final augmented data row maintains up to one difference in each chunk from the original text, which corresponds to the predicted word. This approach guarantees that the rest of the text remains identical to the original.

Once we have a valid chunk in terms of length, the subsequent part is masked-lm prediction. A token contained in the chunk is randomly chosen and replaced by a [MASK] token. Then, the chunk is tokenized using LEGAL-BERT and afterward, we ask the model to predict the word behind [MASK]. LEGAL-BERT replaces [MASK] with its corresponding prediction and the process is completed. Notably, BERT-like models such as LEGAL-BERT can yield impressive results even without fine-tuning, which means that the prediction might be accurate. This implies that occasionally, the generated chunks will match the original ones. Even though it's unlikely that this will occur for every chunk, to avoid extreme cases like this one, we handle things a bit differently.

Instead of replacing the [MASK] token with LEGAL-BERT's prediction we fetch the top 2 predictions. If the second prediction has a probability above 30%, we use the second prediction instead. If it doesn't, then we get the first one despite it having more chances of being the correct one (and hence making no changes in that chunk). A relative example can be seen on figure 2.2. The percentage that we use has been fine-tuned up to a point, but in general, the logic behind this choice is that we don't want to select the second-best option if it has a great chance of being entirely wrong in terms of context and flow. Setting aside the fine-tuning, looking at the results of the predictions, the model makes correct predictions quite frequently, so we consider the approach we follow reasonable. We tested it in several instances and concluded that the changes in the resulting text were satisfactory. At last, after every mask token is replaced by the model's predictions, the sentences are put back together and the final assembled text is an augmented version of the original one.

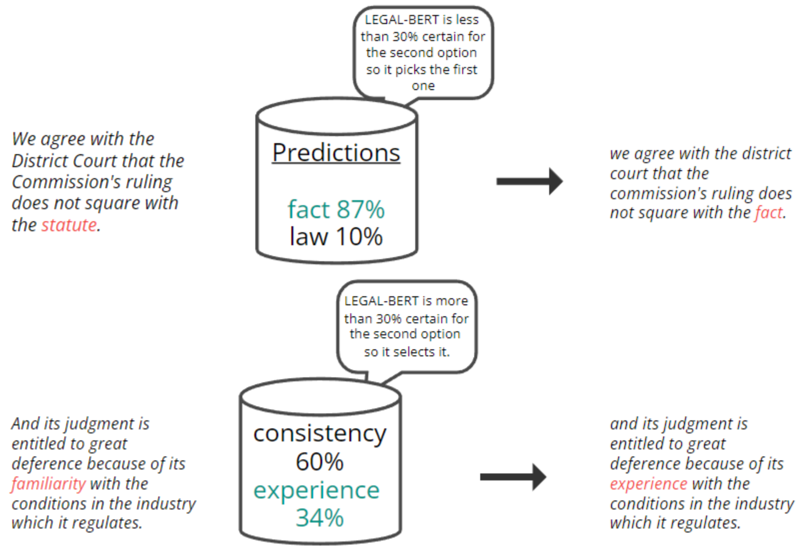

Fig. 2.2: Two different cases of masked-lm predictions applied in a chunk of our SCOTUS legal texts.

2.2.1.3 Back-Translation

The third augmentation technique is back-translation. The idea in this case is that Google Translate will translate the text from the source language which is English, to another language, and subsequently translate back to English. Google Translate is known to provide translations that preserve the overall meaning of the input most of the time, but tend to slightly change some parts. Hence, we employ Google's machine to do back-to-back translations and we expect in return altered chunks that have preserved the overall meaning.

Similar to the previous technique we described, masked-lm, we use a shifting window and break the text into chunks. This is necessary because Google Translator does not allow the input texts to be more than 5,000 characters. We will not explain the process as it is identical. The only difference is the limitation, as LEGAL-BERT required at most 512 tokens while now the issue involves the number of characters. Therefore, instead of using a shifting window of BPEs, we will implement a shifting window based on character count.

An interesting note is that we set a limit of 4,000 characters instead of 5,000. Translating from the source language to the pivot language may significantly increase the character count, potentially leading to errors in the subsequent translation from the pivot language to the destination language. This occurs because some languages demand more words than others to articulate the same information. Hence, we err on the side of caution and set the limit at 4,000.

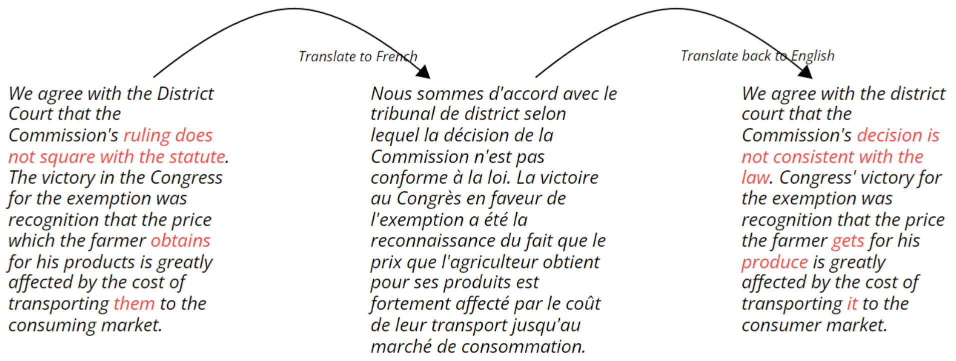

Fig. 2.3: Application of Back-Translation employing French, using a chunk of our SCOTUS legal texts. It is apparent that after back-translation has finished, a new artificial chunk has been generated. Also, the simplicity of the new chunk's vocabulary in comparison to the original chunk is noteworthy.

2.2.2 Data Augmentation in Legal Texts

As mentioned in the previous section, prior research has laid the groundwork for data augmentation in legal texts. Since this section is aimed specifically at data augmentation methods, we will enumerate and briefly describe various augmentation techniques from recent years.

2.2.2.1 Random Deletion/Insertion/Scrambling/Swapping

These data augmentation techniques represent a fundamental approach to enhancing textual data and they are very similar in a way, so we present them unified. Random deletion involves removing random words from the text, which causes variations compared to the original text. This proves that random deletion is a valid augmentation technique. Wei et al. ([WZ19]) applied this method, resulting in slight enhancements, particularly when employing a low word removal parameter. This complexity arises because too many deletions can eventually lead to text lacking coherence or meaning.

Similarly, random deletion was also applied by Santosh et al. ([SBG23]) as they claim that it helps the model gain a deeper contextual comprehension of the sentence, rather than depending solely on superficial word-level characteristics. The paper tackles the Rhetorical Role Labeling of legal documents, which concerns dividing a document into semantically coherent chunks and setting a label to the chunk that depicts its corresponding role in the legal discourse. They seem to agree that increasing the amount of deletions would lead to unstable results and therefore set the max deletions to be 20%. The outcome of their experiments is positive but marginal.

Furthermore, another technique is random insertion. In this approach, the authors of Wei et al. ([WZ19]) search for synonyms for every word in the text. A random word with at least one synonym is then chosen and inserted randomly into the sentence. It's essential to note that this approach may face greater challenges with more complex legal data due to their inherent complexity, which can make it difficult to find suitable synonyms that fit the context.

The authors of Yan et al. ([Yan+19]) developed three data augmentation methods, to perform classification for crime prediction based on case descriptions. Despite not applying directly the above two methods, their philosophy is quite similar. The first method they introduce is random deletion of entire sentences. They claim that the data have many statements that are not mandatory in the fast description text, and choose to delete one sentence randomly. They repeat this process for every sample and generate new instances previously unseen in the original dataset.

Moreover, the second method they present is random insertion, again for sentences. They observe that cases sharing the same accusation label often contain multiple similar sentences within the case description, leading them to adopt a distinct approach. To tackle this issue and enhance dataset diversity, they divide the data based on accusation labels. In simpler terms, they group cases that have identical accusations or charges. Within each cluster of cases sharing the same accusation label, they randomly select a sentence from any of the cases within that cluster. This selected sentence is then added to the original dataset. By doing that they essentially combined original data with a new addition, but if they do this repeatedly they will end up with a new dataset of the same scale, as they do not process data that do not share the same label.

The third method they use is random scrambling. The authors observe that the order the sentences are fed to the model does not matter in their case. So, to introduce variety and randomness into the dataset without changing the essential content, they perform a random scrambling operation. This shuffles the sentences within each case description text randomly, while maintaining the labels unchanged.

Finally, random swapping is a technique implemented by Santosh et al. ([SBG23]) which aims to swap sentences contained in a text but does that in a manner that maintains the readability for humans. They do this by limiting the exchanges of sentences with others in the same section, maintaining this way the total flow of the document. Although this might introduce some noise, the content remains intact, and the roles of sentences don't shift. This enables the model to understand how the discussion flows throughout the document and helps it handle the challenge mentioned earlier, where changes only occur within the same sections.

2.2.2.2 Data Augmentation using Semantic Relations

On top of that, Aoki et al. ([AYS22]) is a paper that tackles legal textual entailment and the authors state their need for artificial training data. Their task is essentially locating questions in law articles for the Japanese bar (each data row is a question-article pair). This causes several semantic and logical mismatches due to the intricacies of legal language and the specific nature of legal content, and that could constitute a barrier to data augmentation. The authors reveal a method to use these mismatches for the benefit of the augmentation. They select a similar approach to Min et al. ([Min+20]) and Evans et al. ([Eva+18]) and generate positive and negative data using semantic relations inside the articles. The process is the following.

Every article has a great chance to contain either juridical decisions, a list of juridical conditions, or both. The authors collect the chunks that have at least one of these and split them into sentences. The sentences that fall into the first category and contain juridical decisions are augmented with the assistance of information from the previous sentence. The corresponding information is located using semantic relations between the sentences. On the other hand, if the sentences have a list of condition parts, they utilize it to augment a sentence that alludes to this list. This way, they make augmented question-article pairs. Finally, for the positive data, which concerns data that represent valid or correct textual entailment relationships, the augmented sentences are employed for questions as well as article parts. On the other hand, for the negative data, which refers to data that represent invalid or incorrect textual entailment relationships, they select sentences that flip the decision for the question parts.

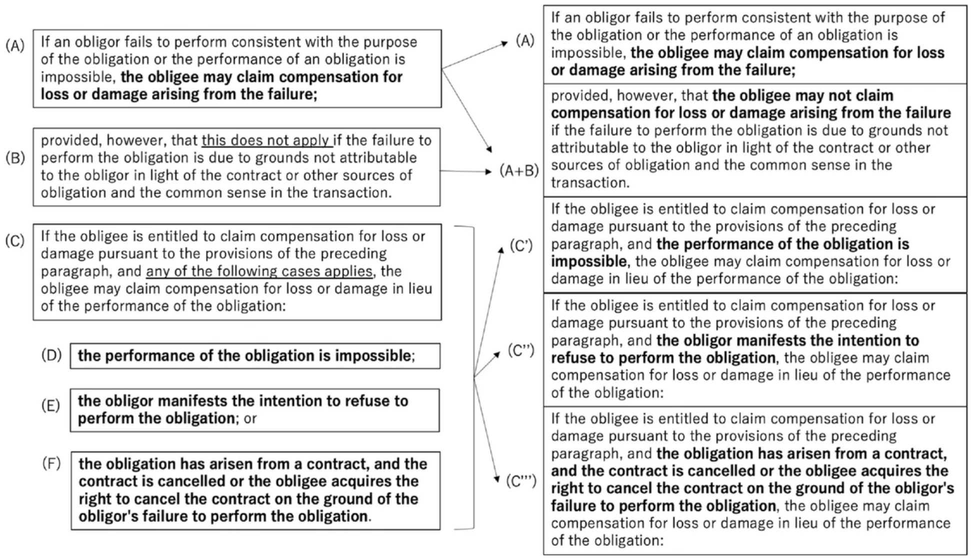

Fig. 2.4: Examples of generating sentences for data augmentation. All rights reserved to Aoki et al. ([AYS22])

2.2.3 Additional Data Augmentation Methods in NLP

When it comes to augmentation methods in the broader area of NLP, there are so many implementations that it is not possible to mention every one of them. Feng et al. ([Fen+21]) kept track of a respected amount of work in this field and is a good source to get a generic perspective of data augmentation techniques in NLP. In conclusion, in the discussion on data augmentation methods, we will further mention a few additional methods. These haven't necessarily been applied in the legal or biomedical domains but are considered important for various reasons.

2.2.3.1 Synonym Replacement

Synonym replacement is an early technique that has been discussed by several research papers over the years. The idea proposed by Zhang et al. ([ZZL16]) was to choose random words and then retrieve their synonyms, to replace them and produce artificial data. They used an English thesaurus from WordNet ([Fel05]), where synonyms for a word or phrase are organized based on their semantic similarity to the most commonly used meaning. To determine the number of word replacements, they identified all replaceable words in the provided text and selected a random subset of size r for replacement, where r followed the geometric distribution.

This technique was some years later suggested as an augmentation method in legal document classification by Csányi et al. ([CO21]), but was not implemented on their behalf. It is discussed in several papers like Yan et al. ([Yan+19]) as well but again they chose not to put it into effect. The consensus of the research community seems to support that even though it is an intuitive method for data augmentation, it has several formidable barriers with the most significant being the fact that it is difficult to find a manner to substitute words by their synonyms repeatedly, without altering the total meaning of the text. An example of this was presented in the introduction with the sentence "The court decided to postpone the hearing for next week", which demonstrated that finding synonyms in a specific context is not an easy task.

It should be noted that Kobayashi et al. ([Kob18]) discussed the effect of synonym replacement and proposed a different idea called contextual augmentation. In this scheme, words are augmented with more varied words by using instead of synonyms, words that are predicted by a language model, given the context that surrounds the original - to be augmented - words. The authors conducted a variety of experiments and the positive results were proof that this method can be beneficial as a data augmentation method.

2.2.3.2 Data Augmentation using LLMs

In the introduction, we underlined the rising attention that LLMs have been enjoying for the past two years. It is natural, therefore, to reflect on the potential benefit of LLMs in data augmentation. This was considered by Dai et al. ([Dai+23]) who attempted to feed datasets to ChatGPT, the most famous LLM these days, to augment them. They do this using Amazon's dataset, PUBMED, and a Symptoms dataset from Kaggle[2]. Their method initially included prompting ChatGPT for data augmentation. They delivered samples of all classes into ChatGPT and requested that it produce samples that maintain semantic consistency with the existing labeled examples. Subsequently, they trained a BERT-based sentence classifier on the total data, merging this way the initial and the generated from ChatGPT data, and evaluated the model. They also added numerous baselines, including word substitution, word embeddings, 2-shot ChatGPT, and many others.

To conclude, the results the authors present have both upsides and downsides. The positive part of the results is that AugGPT was the best-performing model on every dataset beating all the baselines. The downside is that they consider AugGPT to have limitations when it comes to difficult datasets like the biomedical or the legal ones in our case, due to how LLMs are trained. It is known that LLMs are adapted in more generic areas and that's why ChatGPT for example, produces these incredible results worldwide on routine tasks.

In a nutshell, in this section, we attempted to present various augmentation methods put into effect in distinct domains. These methods were developed by the research community, but we also attach our scope to some of them. We first discussed a variety of augmentation techniques applied in a question-answering context using biomedical data. Then, we investigated methods applied specifically to the legal domain, starting with our implementations that will be evaluated in section 3. Lastly, we referred to a couple of augmentation techniques that concern the broader area of NLP.

[2]See www.kaggle.com for more details

3 Experiments

The majority of the time spent on this thesis during the past months was devoted to conducting experiments. In this section, we will explain in depth the whole process behind these experiments, providing all of the details in case someone wants to reproduce our results. In addition, we will include the background of the datasets and the models we employ in these experiments.

3.1 Experimental Setup

The initial objective of our experiments is clear. We aim to apply a subset of the augmentation methods proposed by Pappas et al. ([PMA22]) to the legal domain, utilizing datasets from Chalkidis et al. ([Cha+22]). The goal is to assess if some or all of our augmentation methods increase the F1-score that was produced by the model using the original not-augmented dataset.

3.1.1 Hardware Setup

Before delving into the details of our implementation, we provide the specifications of the hardware supported on the servers used for our experiments.

Server 1

- GPU: GeForce GTX TITAN X
- CPU: 16 CPUS -> Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz
- RAM: 32 GB
- OS: Ubuntu 22.04.1 LTS

Server 2

- GPU: 2 GPUS -> GeForce GTX 1080

- CPU: 12 CPUS -> Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz
- RAM: 64 GB
- OS: Ubuntu 22.04.1 LTS

3.1.2 Datasets

As we mentioned throughout this thesis so far, the evaluation of our augmentation techniques in the legal domain took with the help of LexGLUE ([Cha+22]), a collection of legal datasets. This collection was initially created to evaluate the performance of various legal NLU tasks. The complete collection, as detailed in the paper, comprises seven datasets. However, our primary focus was on three datasets with a significantly smaller number of training instances. This choice was driven by two principal factors. Firstly, we anticipated that these datasets may yield more notable improvements. Secondly, the fact that data augmentation is particularly effective when applied to datasets that are inherently limited in size, as we expect that a dataset with a sufficient amount of data would gain less if augmentation was applied.

3.1.2.1 SCOTUS

SCOTUS stands for US Supreme Court, which is the highest federal court in the United States. It is responsible for interpreting federal law as well as the Constitution. It typically hears cases that are highly controversial or involve complex legal issues that have not been adequately resolved by lower courts. The authors designed it to be a combination of Supreme Court DataBase ([Spa+20]) along with SCOTUS opinions generated by CourtListener a free and open-source legal research website. In a few words, CourtListener provides access to a vast collection of legal opinions, court cases, and legal documents from courts across the United States. Furthermore, SCDB provides metadata about Supreme Court cases, including details like decisions, issues, and decision directions. The SCDB covers cases from 1946 up to 2020 and it must be underlined that the data are chronologically split into training (5k, 1946–1982), development (1.4k, 1982–1991), test (1.4k, 1991–2016) sets. The task is to classify every court opinion of the SCOTUS cases in the corresponding issue area it belongs. The total issue areas are 14 but in the total data we inherit (train, development, and test) there are 13 assigned labels, as there are no instances classified as label 14.

The 14 issue areas, accompanied by their assigned integer label are the following:

0. Criminal Procedure

1. Civil Rights
2. First Amendment
3. Due Process
4. Privacy
5. Attorneys
6. Unions
7. Economic Activity
8. Judicial Power
9. Federalism
10. Interstate Relations
11. Federal Taxation
12. Miscellaneous
13. Private Action (does not exist in our data)

3.1.3 LEGAL-BERT

Researchers from LexGLUE ([Cha+22]) and their team, employed the benchmark dataset they generated, to evaluate a variety of models. The best-performing model overall was LEGAL-BERT suggested by Chalkidis et al. ([Cha+20]). Hence, it is a natural choice to select this model to run our experiments.

LEGAL-BERT is a BERT-like model designed to support legal NLP research, computational law, as well as legal technologies. The authors produced a family of models, each one with different characteristics, and the model we are using is one of these variants. In a few words, the authors underline the mediocre results of BERT in scientific areas like the biomedical one or the legal domain, despite the fine-tuning of relevant datasets. Therefore they conduct several experiments to find a fitting coping mechanism and they propose two main methods, further pre-training a BERT on domain-specific corpora and pre-training a BERT from scratch. They employed three datasets, which we will briefly discuss below, and concluded that the best-performing model was the pre-trained from scratch BERT on all three datasets combined, outperforming the pre-trained from scratch BERT on each dataset separately.

The datasets that were employed for the training of LEGAL-BERT were EURLEX57K, ECHR-CASES, and CONTRACTS-NER. EURLEX57K suggested by Chalkidis et al. ([Cha+19]) is a multi-label classification dataset that encompasses European Union laws. In addition, ECHR-CASES is a multi-label classification dataset (also suitable for binary classification) that involves cases from the European Court of Human Rights. Lastly, CONTRACTS-NER referring to US contracts is a dataset that was developed for named entity recognition and consists of three separate parts that center around contract headers, dispute resolution, and lease details.

Subsequently, we will compare LEGAL-BERT and BERT to assess the impact of the modifications introduced by the authors, that resulted in significant improvements. Both LEGAL-BERT and BERT share the same architecture, featuring 12 layers, 768 hidden units, and 12 attention heads, and have been trained with 110 million parameters. This entails the creation of a new vocabulary of the same size as BERT's vocabulary.

LEGAL-BERT underwent training for 1 million steps, roughly equivalent to 40 epochs, using all three datasets mentioned earlier. The training was conducted in batches of 256 samples, with sequences of up to 512 sentence-piece tokens. Additionally, the authors utilized the Adam optimizer with a learning rate set at 1e-4, mirroring BERT's configuration. Hardware-wise, the authors performed training using the official BERT code on v3 TPUs equipped with 8 cores from Google Cloud Compute Services. In terms of tuning, they experimented with batch sizes from the set 4, 8, 16, 32. Furthermore, they made adjustments, such as lowering the learning rate to 1e-5 to prevent overshooting local minima and increasing the dropout rate to 0.2 to enhance regularization.

The results achieved by the LEGAL-BERT variant, which we adopted in our experiments, consistently outperformed fine-tuned BERT models. Notably, the most significant improvements were observed in multi-label tasks, where the LEGAL-BERT variations showcased remarkable enhancements.

3.1.4 Hierarchical Model

LEGAL-BERT is a model that is expected to perform particularly well in a classification task where the corresponding dataset is relevant to legal issues. And yet, LEGAL-BERT according to its development that was analyzed in the previous section, does not include any implementation that would make it competent to read and classify long texts, like the ones our datasets include. For this reason, when the authors of LexGLUE evaluated a variety of models using these datasets, they created a script that "hacks" the BERT-like model's implementation to add a hierarchical variant introduced in Chalkidis et al. ([Cha+21]).

The hierarchical variant addresses the challenge of processing long texts efficiently. It leverages pre-trained Transformer-based models like BERT in our case, to independently encode each paragraph in the input text. This process results in paragraph representations, that hold essential information. To make these paragraph representations context-aware, the authors employ a second-level shallow Transformer encoder. This encoder maintains consistent specifications like hidden units and number of attention heads, across different pre-trained models and ensures that the paragraph representations are aware of the surrounding paragraphs. The hierarchical variant combines these context-aware paragraph representations through max-pooling and yields a document representation, which is then passed through a classification layer for downstream tasks like text classification in our case.

The process described above is the exact implementation that takes place in the code we inherited from Chalkidis et al. ([Cha+22]). We used LEGAL-BERT as our base encoder and then we called the hierarchical encoder which comprises two Transformer layers. The first one encodes paragraphs independently, and the second ensures context awareness. During the forward pass (the process where input data is fed into LEGAL-BERT), the input is split into paragraphs before it's encoded and transformed into a document representation. We then used this representation as a foundation for the classification that happens in the next step.

At this step, we should mention that from the datasets we employed, which are SCOTUS, ECtHR, and UNFAIR-ToS, only SCOTUS and ECtHR have long enough texts to need the hierarchical variants. This means that when the classification is bound to happen, the process is slightly different for UNFAIR-ToS. In this case, the authors utilize LEGAL-BERT to process the data and extract the top-level representation $h_{[CLS]}$ from a special [CLS] token (see figure 3.1). This representation, as per Devlin et al. ([Dev+18]) serves as the document representation. It undergoes further processing through a dense layer with L output units, where each unit corresponds to a label. After this, a sigmoid function is applied. Conversely, for datasets that implement the hierarchical variant, the authors naturally follow a different approach. Here, they take the document representation obtained by max-pooling across paragraphs. This representation is then passed through a linear layer for classification. Depending on the dataset, this linear layer is followed by either a sigmoid activation (for ECtHR) or a softmax activation (for SCOTUS). The discrepancy in the choice of activation functions between these two datasets can be attributed to their fundamental differences. SCOTUS is designed as a multi-class classification dataset, where each input belongs to one and only one class, making Softmax a standard choice for this scenario. In contrast, ECtHR is structured as a multi-label classification dataset, allowing each input to be associated with multiple labels. In such cases, a sigmoid activation function is the preferred choice because it independently assigns a probability value between 0 and 1 to each label, aligning with the multi-label classification problem's requirements.

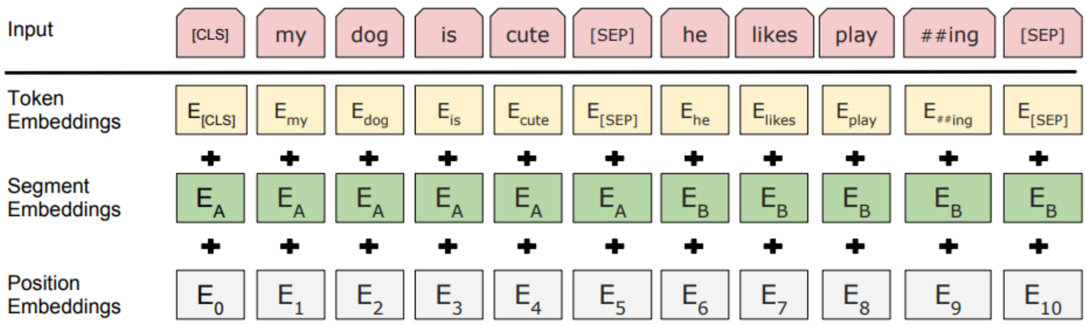

Fig. 3.1: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings, and the position embeddings. In the final hidden state (when the second [CLS] is processed, the model captures the overall information from the entire sequence. All rights reserved to Devlin et al. ([Dev+18])

In addition, at this point, we should state that all of the experiments that we conducted have one important thing in common. Due to limited computing power, during the training, we froze all of the layers except the last one which handles the long training instances and classifies the text. This holds for all the datasets (even ECtHR despite not employing the hierarchical variant). This means that before we started developing the augmentation methods, we needed to run the frozen model and feed the non-augmented data to get a fair baseline.

Also, due to the results being unstable and occasionally causing the model to halt training, we enforced gradient clipping, which is a technique that prevents exploding gradients ([PMB13]). Exploding gradients occur when the gradients of the model's parameters become extremely large, causing training instability and slow convergence. Gradient clipping, is a technique that limits the size of gradients during training to a specified threshold, preventing them from becoming too large. In our case we set the maximum norm to be equal to 1, hoping this way to restrict the unstable nature of the results. In general, this value should have been fine-tuned or at least have been selected with a sufficient explanation regarding the choice. We set the maximum norm to be equal to 1 without any fine-tuning or sufficient arguments to justify it. The reason is that we aim to repeat our experiments without layer freezing or gradient clipping, using exactly the settings of LexGLUE, when there is enough time and computing power available. As far as this thesis is concerned, we focus on the comparison between augmentation methods and a selection of baselines and ignore the bigger picture, which is to find the best methods and optimize their hyper-parameters, achieving this way the best possible results.

3.1.5 Evaluation Metrics

The F1-score ([SJS06]) is a popular evaluation metric in artificial intelligence nowadays and is particularly useful when dealing with imbalanced datasets. Accuracy, which is the most common metric, cannot be trusted for imbalanced datasets, since a baseline model that classifies all instances to the over-represented class, achieves high accuracy scores. F1-score avoids that, as it has two other terms that handle things differently as its foundation, precision and recall. Before continuing our analysis regarding the F1-score, let's quickly define its building blocks: Precision and Recall.

$$Precision = \frac{TP}{TP + FP}$$

TP stands for *True Positive*, which pinpoints that this category contains the correctly classified positive examples. Similarly, FP stands for *False Positive*, which demonstrates that this category contains misclassified positive examples.

$$Recall = \frac{TP}{TP + FN}$$

FN stands for *False Negative*, which shows that this category contains misclassified negative examples.

Having covered the foundation of the F1-score, we can now define the F1-score to be the harmonic mean of precision and recall. It balances the trade-off between precision and recall and provides a single score that reflects a model's overall performance. Its formula is the following:

$$F1 = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

Moreover, in an attempt to understand intuitively the F1-score's functionality, a high F1-score means that precision and recall both have high values. This shows that the model is making accurate positive predictions and it shifts away from generating many false negatives. On the other hand, a low F1-score is an indication of an imbalance between precision and recall, meaning that conversely to the previous case, the model is either making too many false positive errors or missing many positive instances. This proves that the F1-score is a quality metric when we seek a balance of precision in recall, and we almost always do. By letting it steer our evaluation measures, we gain a solid understanding of our model's performance and that is a crucial reason why it has dethroned accuracy, which has the advantage that it is very intuitive, but it is also prone to be mislead.

Having settled on the aspects of F1-score it's high time we take it a step further. The main metric that we consult to monitor the progress of our model is indeed the F1-score but with a slightly altered scheme. In our cases in SCOTUS, ECtHR, and UNFAIR-ToS we have multi-class and multi-label classification, not binary classification like the basic example we provided when classifying if a text belongs in the legal domain or not. In these cases, the F1-score has to adapt its calculations due to the greatest amount of classes and there are two different ways to handle that.

The first case is to consider each class independently and then take the average of these F1-scores. This is done by initially calculating the F1-score for each class separately using the standard formula for binary classification and then calculating the average F1-score. In this case, F1-score is called "Macro F1-score" and is defined by the formula:

Macro F1-Score:

$$\text{Macro-F1} = \frac{1}{N} \sum_{i=1}^{N} \text{F1}_i$$

The second case is to consider all instances across all classes as a single large binary classification problem. This happens by first computing the total number of True Positives (TP), False Positives (FP), and False Negatives (FN) over all classes and subsequently calculating micro-precision and micro-recall using the aggregated values. In this case, the F1-score is called the "Micro F1-score" and its formula is:

Micro F1-Score:

$$\text{Micro-F1} = \frac{2 \cdot \text{Micro-Precision} \cdot \text{Micro-Recall}}{\text{Micro-Precision} + \text{Micro-Recall}}$$

3.2 Experimental Results

How we assess our model is similar to the one that Chalkidis et al.([Cha+22]) did, to compare fairly the results. For each dataset and each augmentation method, we run the model with 5 distinct seeds. The final results are generated by calculating the mean of the 3 best-performing seeds based on the macro F1-score of the development dataset, which is exactly how Chalkidis et al. produce their results. Before we start the presentation of our results, we first define two baselines.

3.2.1 Baselines

3.2.1.1 Naive Oversampling Baseline

In this baseline, under-represented classes are augmented until each class has the same number of instances as the class with the most instances (N instances). This means that the majority class remains unaltered. Specifically, for each data row under consideration, we check its label and determine if it requires augmentation. If the class has fewer than N instances, we augment it. We aim to perform two augmentations for each data row unless the class reaches the desired number of instances (N) after the first augmentation.

To set a vivid example, in SCOTUS we have 13 classes where '0': 1011, '1': 811, '2': 423, '3': 193, '4': 45, '5': 35, '6': 255, '7': 1043, '8': 717, '9': 191, '10': 53, '11': 220, '12': 3 and in total 5000 instances. Applying this augmentation method to SCOTUS would result in a total of 8,200 instances. This is because we set N=1043 and enforce each data row to be augmented up to two times, as long as the threshold N is not surpassed. The classes '0', '1', '5', and '2' will at some point reach the threshold and halt the production of augmented data rows, class '7' as we mentioned will not be augmented, and the rest of the classes will triple in size as they will not reach N. After this process is finished SCOTUS has 8,200 instances where 3,200 are artificially generated.

3.2.1.2 Masked-LM without Substitution

Masked-lm without substitution baseline does exactly what the title infers. Using LEGAL-BERT in the same manner we did for the advanced masked-lm method (where we substitute the given token) described in 2.2.1.2, we replace a random token with [MASK], but we never predict to take its place. So, we create one artificial instance for each data row, duplicating this way the total data rows. Ultimately, the only difference between augmented and original instances, is that the augmented ones have one [MASK] per chunk in random positions in the text, replacing the corresponding words.

3.2.2 SCOTUS Results

SCOTUS includes 5,000 training instances that are dated 1946-1982, 1,400 evaluation instances from the following nine years (1982-1991), and finally 1,400 test instances that are placed in 1991-2016. We will first provide the results of the original data (without augmentation) in our configuration. Before proceeding with that, we present a specific trait that this dataset has, class imbalance.

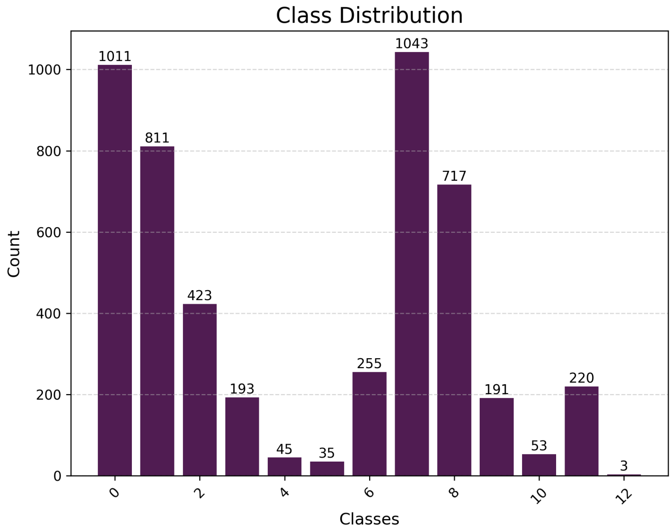

Fig. 3.2: A bar chart of the class distribution in SCOTUS

It is apparent that the classes are far from equally distributed. As we mentioned when introducing SCOTUS, there is a class with only 3 instances and that is class 12 (see section 3.2), which corresponds to the issue area called "Miscellaneous". Moreover, there is a class that corresponds to the Private Action issue area and is not represented at all in our data (it does not exist in development or test data as well).

Moving on, we present the results of the inherited data where augmentation has no involvement, in our configuration setting.

	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5	**Results**
eval_loss	0.796	0.856	0.899	0.778	0.767	**0.841**
eval_macro_f1	0.715	**0.737**	**0.74**	0.722	**0.728**	**0.735**
eval_micro_f1	0.797	0.809	0.812	0.805	0.806	**0.809**
test_loss	0.952	1.081	1.257	0.976	1.031	**1.123**
test_macro_f1	0.665	0.694	0.645	0.684	0.669	**0.669**
test_micro_f1	0.759	0.773	0.749	0.764	0.762	**0.761**

Tab. 3.1: LEGAL-BERT's performance using the original data for 5 different seeds. The last column "Results" takes the average of the 3 best-performing seeds. The best-performing seeds are determined by their results in the category of evaluation macro-F1. The bold cells indicate the best-performing seeds and the total results.

The last column depicts the final results of LEGAL-BERT using the original data. It shows the average of seeds 2, 3, and 5 that had the best macro-F1 performance on the evaluation dataset. These results will be our main baseline since this whole initiative aims to prove that augmentation will improve these numbers.

We continue with the next two baselines that we defined a few lines above. We will not present again the results of each seed from this moment forward, every result that we introduce has been generated with the same methodology.

	Original Data	Naive Oversampling	Masked-LM w/out Subst.
eval_loss	**0.841**	0.879	0.856
eval_macro_f1	**0.735**	0.725	0.73
eval_micro_f1	**0.809**	**0.809**	0.803
test_loss	**1.123**	1.281	1.152
test_macro_f1	**0.669**	0.647	0.659
test_micro_f1	**0.761**	0.748	0.755

Tab. 3.2: A comparison of the 3 baselines. The bold cells indicate the best-performing method in each category. It can be seen that the two baselines that were supposed to overcome the results of the original data on paper, Naive Oversampling and Masked-LM without substitution, failed to surpass the original results in every metric.

We notice that the two baselines that we defined are weaker than the data without augmentation. For the latter method, we ran the model with 5,000 training instances, whereas for the two baseline methods, we had more due to augmentation. To figure out the number of training instances we need to go back a few pages where we described how they work. Starting from masked-lm without substitution, the total number of instances is 10,000, as for every data row we created a new one. Furthermore, naive oversampling contains 8,200 examples and we can compute that following the calculations we provided when defining this baseline.

Continuing our result presentation, we introduce the first augmentation technique, word2vec. The first experiment we conducted had a similarity equal to 0.95 and the available augmentations per row were set to 1. This essentially means that this run fed LEGAL-BERT with at most 10,000 instances. We underline "at most" as it is possible that some data rows were not available for augmentation, due to the embedding search failing to find neighbors that fulfill the similarity threshold.

	Original Data	Naive Oversampling	Masked-LM w/out Subst.	Word2vec 0.95
eval_loss	0.841	0.879	0.856	**0.77**
eval_macro_f1	**0.735**	0.725	0.73	0.733
eval_micro_f1	**0.809**	**0.809**	0.803	0.808
test_loss	1.123	1.281	1.152	**1.01**
test_macro_f1	0.669	0.647	0.659	**0.674**
test_micro_f1	0.761	0.748	0.755	**0.763**

Tab. 3.3: Data Augmentation using Word2vec compared to the 3 baselines. The bold cells indicate the best-performing method in each category. Word2vec with the current settings (0.95 similarity threshold), appears to surpass the baselines in the test set but fails to do so in the development set.

After this run, we question whether increasing the number of augmented instances (for example we could generate up to 2 from each data row), or lowering the similarity threshold would produce better results. Due to time and computing power restrictions, instead of conducting similar experiments with 5 seeds for each hyperparameter tweak, we selected the best-performing seed for word2vec (which was seed 3) and tried different settings running only this seed. The results of this process are portrayed below.

	Word2vec 0.95	Word2vec 0.95, x3	Word2vec 0.9	Word2vec 0.85
eval_loss	**0.726**	0.896	0.791	0.945
eval_macro_f1	0.723	0.709	**0.733**	0.72
eval_micro_f1	0.808	0.795	**0.81**	0.803
test_loss	**0.88**	1.184	1.07	1.296
test_macro_f1	0.67	0.668	**0.672**	0.615
test_micro_f1	0.762	0.75	**0.764**	0.716

Tab. 3.4: Word2vec hyper-parameter tuning. The bold cells indicate the best-performing method in each category. All of these results were produced by seed #3 which was the best-performing seed. In this table, we compare different similarity thresholds as well as different amounts of generated artificial instances. Word2vec with 0.9 and 0.95 similarity thresholds appear to be the best options.

These results infer that tripling the data is not beneficial, so from this point forward, we chose to augment each data row at most 1 time when applying word2vec. Additionally, even though the performance of word2vec using 0.85 as a threshold was much worse than the baselines, word2vec with 0.9 as a limit was inspiring enough to motivate us to run all of the seeds. We present the comparison of word2vec with a 0.95 threshold versus 0.9 below.

Word2vec 0.95	Word2vec 0.9
0.77	0.816
0.733	**0.74**
0.808	**0.808**
1.01	1.054
0.674	0.666
0.763	**0.76**

Tab. 3.5: A head-to-head comparison of Word2vec with different similarities. The generated results depict the average performance of all the 5 seeds. Word2vec with 0.9 similarity is the best method for the development data. The bold cells indicate the best-performing method in each category.

The head-to-head comparison shows that word2vec with 0.9 similarity is slightly better on the development data, but worse on the test data. To avoid overfitting we chose word2vec with 0.9 similarity as our main word2vec implementation.

We proceed to the subsequent augmentation method, back-translation. We conformed with the implementation of Pappas et al. ([PMA22]) that initially tried back-translation with only French as the pivot language. Afterwards, we conducted the same experiment using French, German, and Spanish as pivot languages which means that this implementation has 20,000 instances in total. We reveal the head-to-head comparison between these back-translation implementations.

Back-Translation /w French	Back-Translation /w all Languages
0.959	1.361
0.734	0.73
0.806	**0.806**
1.294	1.83
0.668	0.656
0.761	0.749

Tab. 3.6: A head-to-head comparison of Back-Translation with different languages. The bold cells indicate the best-performing method in each category. Back-translation using only French as a pivot language beats back-translation that employed 3 languages (French included) and becomes the default back-translation for the remainder of our experiments.

We conclude that back-translation with only 1 pivot language is the best option as it beats back-translation with 4 languages in almost every metric. Hence, after reviewing these results we held this as our back-translation method.

Finally, we put into effect our final augmentation method, masked-LM. We evaluated our model with 10,000 instances, where in each augmented row one word has been predicted by LEGAL-BERT. The table below compares masked-lm with the previous two augmentation methods, along with the three baselines we defined. It is a total comparison between all of the augmentation methods we applied and the three baselines we defined.

	Original Data	Naive Oversampling	Masked-LM w/out Sub.	Word2vec	Back-Translation	Masked-LM
eval_loss	0.841	0.856	0.879	**0.816**	0.959	**0.78**
eval_macro_f1	0.735	0.73	0.725	**0.74**	0.734	0.732
eval_micro_f1	**0.809**	0.803	**0.809**	0.808	0.806	0.808
test_loss	1.123	1.152	1.281	1.054	1.294	**1.018**
test_macro_f1	**0.669**	0.659	0.647	0.666	0.668	0.666
test_micro_f1	0.761	0.755	0.748	0.76	0.761	**0.764**

Tab. 3.7: A full comparison between all methods and baselines across all seeds. The bold cells indicate the best-performing method in each category.

Data augmentation was successful regardless of the method used in Pappas et al. ([PMA22]) where we inherited the methods from. The differences between the results of the methods concerned the scale of the benefit each method offered. The results generated on the legal domain, for the SCOTUS dataset, which are depicted in the above table, do not point in the same direction. None of the augmentation methods managed to surpass the performance of the initial data. In the subsequent part of this section, we seek potential reasons that might have caused this.

The unexpected results show that augmentation not only does not benefit, but it corrupts the final metrics, leading to a different idea. If the class imbalance is indeed affecting negatively the model's performance, then by augmenting all of the data rows the results may be also affected negatively. For instance, the majority label that has 1,043 instances will have 2,086 in the augmented dataset, but the minority class with initially 3 instances will only have 6.

To address this we conduct all of our experiments again (for the 3 augmentation methods) using a different technique. Instead of doing what we did so far, which was essentially generating one artificial data row without accounting for its label, we augment rows, in the same manner we augmented duplicate data in naive oversampling. Therefore, our datasets will now contain at most 8,200 instances. Below, we can see the difference in the class distribution between the two augmentation techniques. From this point forward we will call the augmentation method that we applied so far as basic or standard augmentation.

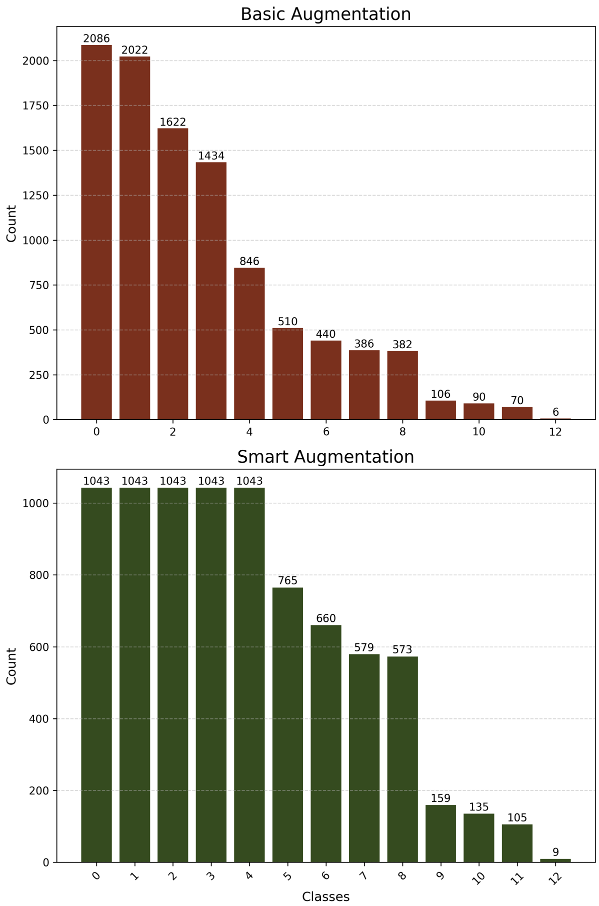

Fig. 3.3: A bar chart of the class distribution in SCOTUS

Ultimately, we present the results of this alternative augmentation approach. The generated results have been produced with the same circumstances used previously for the standard augmentation method, in order to make a valid comparison.

	Original Data	Naive Oversampling	Masked-LM w/out Sub.	Word2vec	Back-Translation	Masked-LM
eval_loss	**0.841**	0.856	0.879	0.935	1.008	0.859
eval_macro_f1	0.735	0.73	0.725	**0.741**	0.733	0.73
eval_micro_f1	**0.809**	0.803	**0.809**	0.807	0.8	0.805
test_loss	**1.123**	1.152	1.281	1.284	1.329	1.13
test_macro_f1	**0.669**	0.659	0.647	0.659	0.66	0.654
test_micro_f1	**0.761**	0.755	0.748	0.75	0.748	0.752

Tab. 3.8: A full comparison between all methods and baselines, across all seeds, with an alternative augmentation approach. The bold cells indicate the best-performing method in each category. We observe that the results of the original data remain the best overall, whilst the margin between them and the 3 augmentation methods increases despite the initial predictions.

Analyzing the data presented in the table above, it becomes evident that none of the employed augmentation methods managed to improve the results achieved by Chalkidis et al. ([Cha+22]). Word2vec surpassed the original data in evaluation macro-F1, but regarding the test results, each augmentation method failed to score a better result. Surprisingly, the more sophisticated augmentation approach that we expected to improve the results, performed even worse than the standard one. To shed light on these perplexing metrics, we will employ data visualization in the form of figures. These visual aids will enable us to assess the model's learning progress across different augmentation techniques and will hopefully explain the background of the generated results. Furthermore, we will shift our focus to the confusion matrices, along with the F1-scores for each class, to get a better understanding of the performance of each distinct augmentation method.

To begin, we will examine the training and evaluation losses associated with each augmentation method. Our approach involves selecting the best and worst-performing seeds for each method and plotting their results on a shared graph or sub-graph, as appropriate. In total, we will generate four plots, one for the original non-augmented data, one for back-translation, one for masked-lm, and one for word2vec. Each of these plots will contain four distinct curves: the training and evaluation curves for both the best seed and the worst seed.

This method of analysis allows us to compare the performance of the best and worst seeds and gain valuable insights into the learning process for all the augmentation techniques. It's important to note that the curves for different seeds will conclude at varying epochs, as they correspond to a different run.

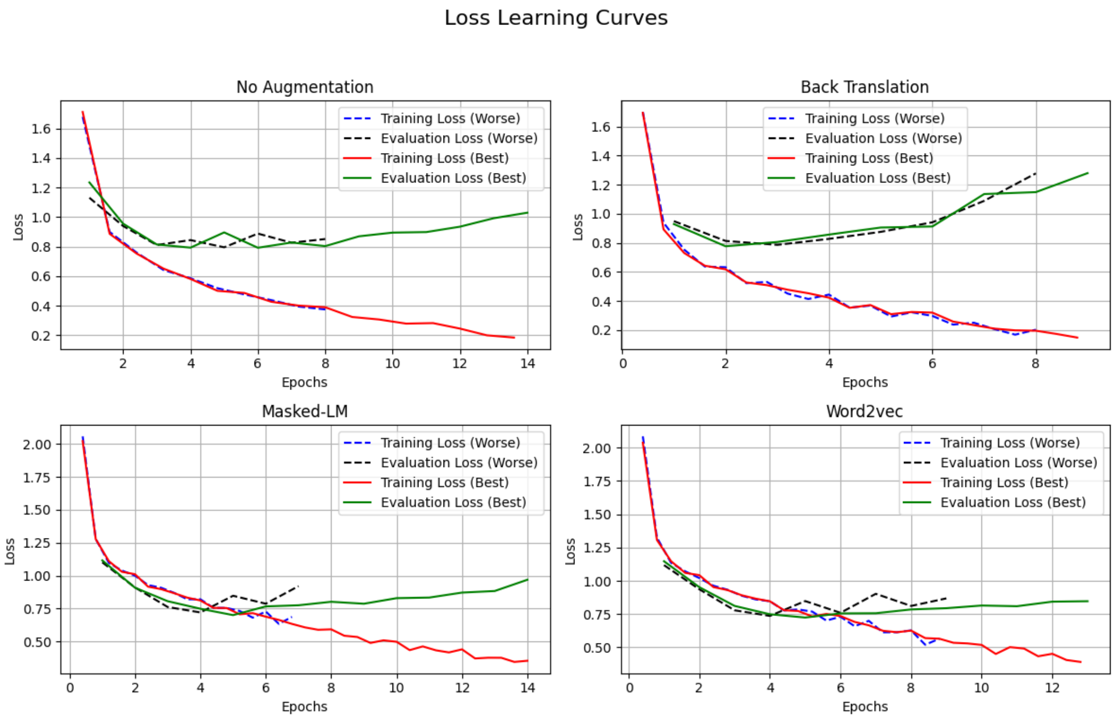

Fig. 3.4: Loss Learning Curves for the best and worse seed of each method. We can see that seeds within each method display similar trends, with word2vec having the largest margins. The values of the train and evaluation losses across all the methods demonstrate that the learning process has not performed optimally.

The plots show that despite having diverse results, the seeds perform similarly. We observe that word2vec has the biggest difference between the seeds while back-translation's seeds are the most alike. To provide a clearer view, the difference between the best seed and the worst seed of back-translation was roughly 1.1% whilst for word2vec it was 2.5%.

In general, the sub-graphs return a uniform result which warns that the model is not behaving properly. The evaluation loss is always more than 0.7 which demonstrates that the learning process is far from ideal. SCOTUS is not an easy dataset as it contains huge texts in length, and this could just be the reason. On the other hand, these results could be a sign that something else is off, so we err on the side of caution and investigate further the behavior of the model by generating the confusion matrices.

We begin by calculating the confusion matrix of the original data which was the run that performed best overall. Then, we do the same for the best-performing standard augmentation method and the best smart augmentation method. Back-translation had the highest macro F1-score in both cases which is our main point of comparison, so we selected it. Lastly, we need to select a seed as we have 5 seeds per method, so we choose the best-performing seed for each method.

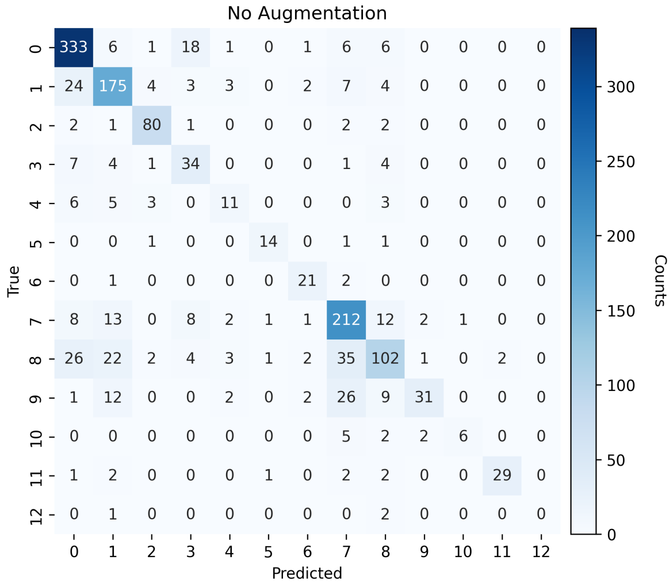

Fig. 3.5: Confusion Matrix # No Augmentation

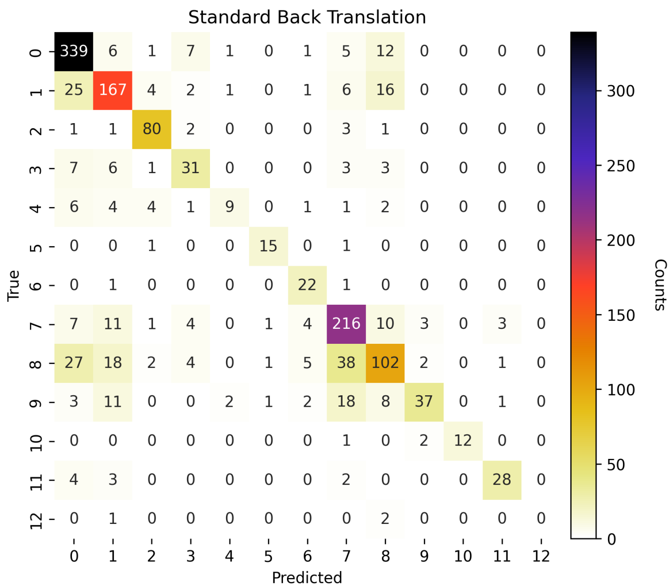

Fig. 3.6: Confusion Matrix # Standard Augmentation

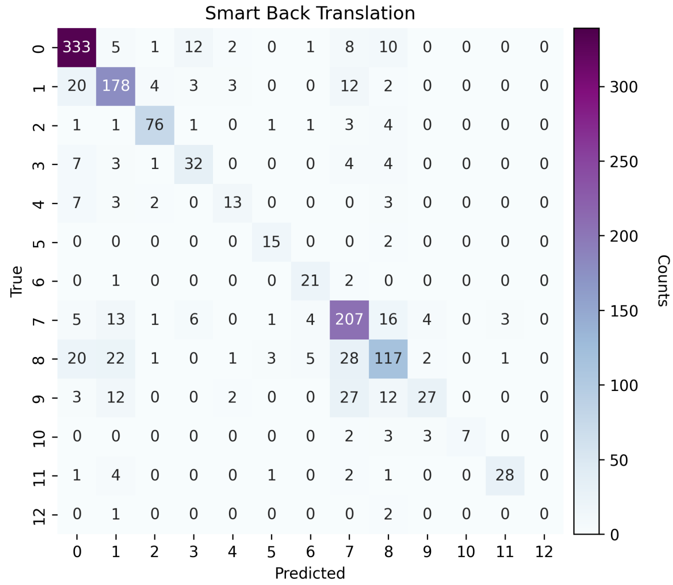

Fig. 3.7: Confusion Matrix # Smart Augmentation

What we can derive from the matrices is that class '12' has 3 instances on the test set and the model fails in every method to get one correct. This is not a surprising result as the training instances are far too low, so low that augmentation can't change the result. In addition, the differences are not significant between each augmentation technique, prompting us to question whether the model can perceive the complexity of the dataset. For example label '4' has 45 instances in the first matrix and the sum of the correctly classified instances is 11 out of 28. With the standard augmentation it has 90 training instances doubling the initial amount, and yet the result is even worse with 9 correctly classified instances out of 28.

Subsequently, we calculate the F1-score per class.

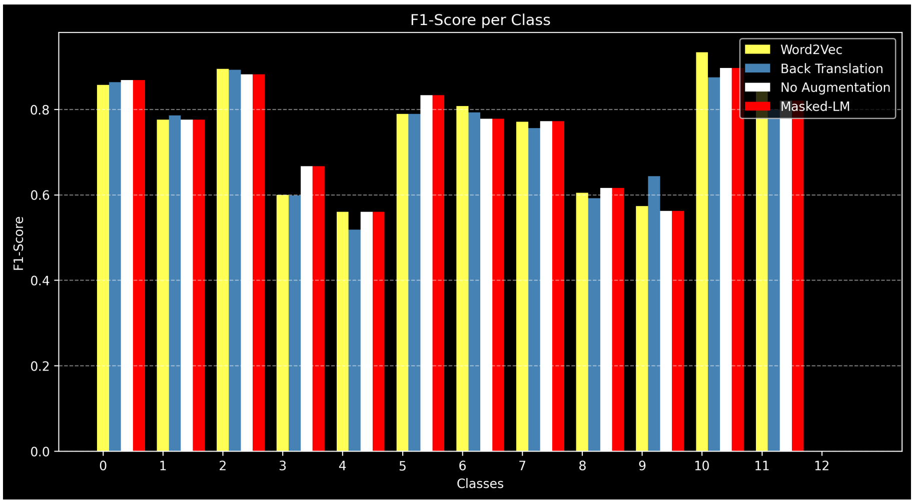

Fig. 3.8: F1 Per Class Standard Augmentation

By examining the graph we see that there are shifts for each label. These shifts are sourced from the differences we discussed previously in the confusion matrices. The results between each method are very close, so we can't be confident that there is a reason behind this, as it could be random. Assuming that this is not occurring randomly, it is an interesting result as it shows that each augmentation method captures different kinds of details. For example, back-translation fails to increase the performance in label '3' whatsoever, but outputs great results in comparison to the other methods for label '9'.

The final thoughts behind LEGAL-BERT's performance in SCOTUS with the addition of artificial data are mixed. None of the techniques achieved a rise in macro-F1 for the test data, showing that changing the dataset only leads to worse results. The tables and graphs we included were also pointing in the same direction. The model's performance demonstrates it that hasn't been able to fully comprehend the complexity that describes the dataset.

An interesting observation is that the more we interfere with the initial data using word2vec, the worse the results are. We might have used word2vec with a 90% similarity threshold after all, but as we presented in our results, on the test set the best method was the one with 95% as a lower bound, and even that did not manage to beat the test macro F1-score. It is natural that the lower the threshold is, the bigger the amount of alterations in the initial text, so this could potentially be a reason. For instance, the experiment with similarity set to 85%, which is far from low, resulted in a 5% macro F1-score dive. It is unlikely that the words replaced are so off context to explain this dive, as 85% is a considerate threshold.

Similarly, back-translation as we explain in figure 2.3, tends to simplify complicated words during the pivot translation process, and it is a fact that specialized words in legal texts are in affluence. This could be a potential reason why this method fails to deliver the expected results. It should be mentioned though that this method changes significantly the initial text (definitely a lot more than the other two methods), and yet produced the best macro-F1 results even if it was a marginal result. This clashes with the assumption we made regarding the negative relation between good performance and the number of shifts in the initial text.

Moreover, masked-lm is the method that is expected in theory to make the smallest amount of shifts in comparison with the other two methods. This is because it changes only 1 word per chunk, while the other two methods don't have a limit. The fact that masked-lm managed to surpass the original data baseline in three categories (evaluation loss, test loss, and micro F1-score) supports our claim that perhaps changes are affecting negatively the results.

4 Conclusions and Future Work

4.1 Conclusions

In this thesis, we augmented the training instances of three LexGLUE datasets, investigating the impact of augmentation in a legal context. Having performed and visualized a number of experiments, we conclude that augmentation did not manage to succeed in our case. None of the word2vec, back translation, and masked-lm that reported optimistic results in previous work, achieved to surpass the benchmark. Even when we tried to adapt to SCOTUS's peculiarities, which had a substantial class imbalance, the outcome was the same.

The various methods we tried did not result in an improvement in the macro-F1 score for the test data. The first results we presented in table 3.2 showing the performance of the two "strong" baselines, were an early indication of that, as none of them managed to surpass the original results. Word2vec was the only method that managed to beat the best baseline in one category involving F1-score and that was the macro-F1 of the evaluation dataset. At the same time though, it performs quite worse on the test data. Back-translation was the best technique in comparison with the other augmentation techniques, as it showed 66% macro F1-score beating word2vec that output almost identical results (65.9%). Also, masked-lm had much better evaluation and prediction loss than every other method but had poor performance in terms of F1-scores, as it was 1.5% below the benchmark in macro-F1 for the test data. Finally, the additional experiments we conducted applying a different style of augmentation by prioritizing the minority labels, despite the optimistic predictions proved to be even worse than our standard augmentation plan.

Altering the dataset ultimately led to poorer performance in every method, and the results of word2vec might be the most clear indication of that. By lowering the similarity threshold from 95% to 85% we observe an inclination of the performance, which is noteworthy. Meanwhile, another interesting remark, is that some methods seem to perform better in separate labels, potentially demonstrating that some techniques might fit better to some classes. It has to be mentioned though that these discrepancies are quite low and they do not constitute a concrete argument. Overall, the model's performance suggests that it has not fully grasped the complexity of the dataset.

4.2 Future Work

The direction that Dai et al. ([Dai+23]) selected is interesting (and obvious considering the hype behind LLMs nowadays) and it just might be the method that will unlock the potential of data augmentation in legal texts. Therefore, using LLMs to generate artificial data is a very intriguing idea for the legal domain, and for data augmentation methods in NLP in general. Moreover, reflecting on the results our experiments generated where there seemed to be a universal problem regardless of the augmentation technique, an interesting approach would be one suggested by Csányi et al. ([CO21]). Specifically, to create a list of protected words that would be guaranteed to remain intact, maintaining this way the core parts of the legal document that need to be static according to the authors.

Bibliography

[AYS22] Yasuhiro Aoki, Masaharu Yoshioka, and Youta Suzuki. "Data-Augmentation Method for BERT-based Legal Textual Entailment Systems in COLIEE Statute Law Task". In: *The Review of Socionetwork Strategies* 16.1 (2022), pp. 175–196.

[CA17] Ilias Chalkidis and Ion Androutsopoulos. "A deep learning approach to contract element extraction". In: *Proceedings of the 30th International Conference on Legal Knowledge and Information Systems (JURIX 2017)*. Luxembourg City, Luxembourg, 2017.

[Cha+19] Ilias Chalkidis, Emmanouil Fergadiotis, Prodromos Malakasiotis, and Ion Androutsopoulos. "Large-Scale Multi-Label Text Classification on EU Legislation". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 6314–6322.

[Cha+20] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. "LEGAL-BERT: The Muppets straight out of Law School". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online, 2020, pp. 2898–2904.

[Cha+21] Ilias Chalkidis, Manos Fergadiotis, Dimitrios Tsarapatsanis, et al. "Paragraph-level Rationale Extraction through Regularization: A case study on European Court of Human Rights Cases". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, pp. 226–241.

[Cha+22] Ilias Chalkidis, Abhik Jana, Dirk Hartung, et al. *LexGLUE: A Benchmark Dataset for Legal Language Understanding in English*. 2022. CoRR: `abs/2110.00976`.

[CK19] Ilias Chalkidis and Dimitrios Kampas. "Deep learning in law: early adaptation and legal word embeddings trained on large corpora". In: *Artificial Intelligence and Law* 27.2 (2019), pp. 171–198.

[CO21] Gergely Csányi and Tamás Orosz. "Comparison of data augmentation methods for legal document classification". In: *Acta Technica Jaurinensis* 15.1 (2021), pp. 15–21.

[Dai+23] Haixing Dai, Zhengliang Liu, Wenxiong Liao, et al. *AugGPT: Leveraging ChatGPT for Text Data Augmentation*. 2023. CoRR: `abs/2302.13007`.

[Dev+18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers).* Minneapolis, Minnesota, 2018, pp. 4171–4186.

[Eva+18] Richard Evans, David Saxton, David Amos, Pushmeet Kohli, and Edward Grefenstette. *Can Neural Networks Understand Logical Entailment?* 2018. CoRR: `abs/1802.08535`.

[Fel05] Christiane Fellbaum. "Wordnet and Wordnets". In: *Encyclopedia of Language and Linguistics.* Elsevier, 2005, pp. 2–665.

[Fen+21] Steven Y. Feng, Varun Gangal, Jason Wei, et al. "A Survey of Data Augmentation Approaches for NLP". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021.* Online, 2021, pp. 968–988.

[Kob18] Sosuke Kobayashi. "Contextual Augmentation: Data Augmentation by Words with Paradigmatic Relations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers).* New Orleans, Louisiana, 2018, pp. 452–457.

[Lan+20] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, et al. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations.* 2020. CoRR: `abs/1909.11942`.

[Lee+19] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, et al. "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4 (2019), pp. 1234–1240.

[Lew+20] Patrick Lewis, Myle Ott, Jingfei Du, and Veselin Stoyanov. "Pretrained Language Models for Biomedical and Clinical Tasks: Understanding and Extending the State-of-the-Art". In: *Proceedings of the 3rd Clinical Natural Language Processing Workshop.* Online, 2020, pp. 146–157.

[Mik+13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space.* Scottsdale, Arizona, USA, 2013.

[Min+20] Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. "Syntactic Data Augmentation Increases Robustness to Inference Heuristics". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.* 2020, pp. 2339–2352.

[PMA22] Dimitris Pappas, Prodromos Malakasiotis, and Ion Androutsopoulos. "Data Augmentation for Biomedical Factoid Question Answering". In: *Proceedings of the 21st Workshop on Biomedical Language Processing.* Dublin, Ireland, 2022, pp. 63–81.

[PMB13] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. *On the difficulty of training Recurrent Neural Networks.* 2013. CoRR: `abs/1211.5063`.

[Raf+20] Colin Raffel, Noam Shazeer, Adam Roberts, et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: 21.1 (2020).

[Raj+16] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, 2016, pp. 2383–2392.

[RJL18] Pranav Rajpurkar, Robin Jia, and Percy Liang. "Know What You Don't Know: Unanswerable Questions for SQuAD". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia, 2018, pp. 784–789.

[San+20] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. 2020. CoRR: `abs/1910.01108`.

[SBG23] T. Y. S. S. Santosh, Philipp Bock, and Matthias Grabmair. *Joint Span Segmentation and Rhetorical Role Labeling with Data Augmentation for Legal Documents*. 2023. CoRR: `abs/2302.06448`.

[SHB16] Rico Sennrich, Barry Haddow, and Alexandra Birch. "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 2016, pp. 1715–1725.

[SJS06] Marina Sokolova, Nathalie Japkowicz, and Stan Szpakowicz. "Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation". In: vol. Vol. 4304. 2006, pp. 1015–1021.

[SKF21] Connor Shorten, Taghi M. Khoshgoftaar, and Borko Furht. "Text Data Augmentation for Deep Learning". In: *Journal of Big Data* 8.1 (2021), p. 101.

[Spa+20] Harold J. Spaeth, Lee Epstein, Jeffrey A. Segal, et al. *Supreme Court Database, Version 2020 Release 01*. Washington University Law. 2020.

[Tsa+15] George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, et al. "An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition". In: *BMC Bioinformatics* 16.1 (2015), p. 138.

[WZ19] Jason W. Wei and Kai Zou. "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks". In: (2019). eprint: `abs/1901.11196`.

[Yan+19] Ge Yan, Yu Li, Shu Zhang, and Zhenyu Chen. "Data Augmentation for Deep Learning of Judgment Documents". In: *International Conference on Intelligent Science and Big Data Engineering*. Nanjing, China, 2019, pp. 232–242.

[ZPT18] Henghui Zhu, Ioannis Ch. Paschalidis, and Amir Tahmasebi. *Clinical Concept Extraction with Contextual Word Embedding*. 2018. CoRR: `abs/1810.10566`.

[ZZL16] Xiang Zhang, Junbo Zhao, and Yann LeCun. *Character-level Convolutional Networks for Text Classification*. 2016. CoRR: `abs/1509.01626`.

List of Figures

List of Tables