School of Information Technology and Engineering at the ADA University

School of Engineering and Applied Science at the George Washington University

DEVELOPMENT OF TEXT DATA AUGMENTATION METHODS FOR AZERBAIJAN LANGUAGE

A Thesis
Presented to the Graduate Program of Computer Science and Data Analytics
of the School of Information Technology and Engineering
ADA University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science and Data Analytics
ADA University

By
Yusif Mukhtarov

Supervisor, Dr. Samir Rustamov

April, 2023

THESIS ACCEPTANCE

This Thesis by: Yusif Mukhtarov
Entitled: *Development of Text Data Augmentation methods for Azerbaijan language*

has been approved as meeting the requirement for the Degree of Master of Science in Computer Science and Data Analytics of the School of Information Technology and Engineering, ADA University.

Approved:

(Adviser)	(Date)
(Program Director)	(Date)
(Dean)	(Date)

ACADEMIC INTEGRITY STATEMENT

“I affirm that this is my own work, I attributed where I used the work of others, I did not facilitate academic dishonesty for myself or others, and I used only authorized resources for my Thesis, per the ADA University Academic Integrity requirements. If I failed to comply with this statement, I understand consequences will follow my actions. Consequences may range from failing the course to expulsion from the program/university and may include a transcript notation.”

Yusif Mukhtarov		29.04.2024
(Full Name)	(Signature)	(Date: DD.MM.YY)

ABSTRACT

Dataset or model? This question is frequently asked by newcomers to the realm of machine learning. Many would structure their thoughts as follows: There is always the option to conduct hyperparameter tuning and train the model. However, if the data is of bad quality or insufficient, it is game over for the project. Throughout the history of machine learning, numerous projects have been terminated due to data shortages, with many more being paused to collect more data. That is why addressing data related aspects of the project should be considered in the first place, and the availability of labeled data is crucial for the success of machine learning models.

We explored data augmentation techniques to enhance the accuracy of machine learning models for Azerbaijani text datasets. In this study we have tested a great number of attention-based approaches as well as new advanced substituting words with synonyms for Azerbaijani and English datasets. All the techniques have been tested using extrinsic and intrinsic methods. For extrinsic evaluation, classification tasks have been used to check the accuracy scores before and after augmentation. By this average accuracy scores of the neural network models have been improvement by **13%** in AG news classification task, **3%** in IMDB review case, and most importantly **13%** in the case of Azerbaijani dataset, OXU news. In intrinsic evaluation, an average Bert embedding cosine similarity score equal to nearly **0.95** has been obtained, while maximum average Bleu and Rouge scores were **0.43** and **0.65**. While the main purpose of the project was finding out the low resource requiring methods, attention-based approaches including transformer paraphrasers and translators still required availability of GPU.

Hence, all the techniques developed in this project can be used by anyone with a single laptop. This advantage is a very important detail of this project since in this project there were 2 main evaluation metrics: efficiency of the technique and its resource requirements. For evaluation of the effectiveness of our data augmentation approach extensive experiments were conducted using various machine learning models on different popular Azerbaijani and English datasets. The original versions of the datasets were compared to their augmented counterparts in various classification tasks, employing both traditional and gradient-based models to assess effectiveness. Regarding resource requirements, considerations were made for size and runtime on CPU. This is done to make the data augmentation process easier and faster for small research projects and startups that do not have a significant amount of budget, since in most of such projects one of the main issues has always been computation power and restrictions in usages of paid services such as Chat-GPT, Cloude, Google translator etc.

LIST OF ABBREVIATIONS

Abbreviation	Explanation
ML	Machine Learning
NLP	Natural Language Processing
MLM	Masked Language Model
GSG	Gap Sentences Generation

List of Tables

Figures

Table of Contents

2 Introduction

The quality and quantity data projects across various fields, including NLP. Data is basically the foundational building block upon which algorithms learn and make predictions. Hence, having ample and high-quality data is a prerequisite for developing robust ML models. But unfortunately, the reality is that most languages do not enjoy the luxury of abundant resources in most domains. This problem is bigger, especially for under-resourced languages, which are often sidelined in global technological advancements due to their limited data availability.

2.1 Background Information

Historically, data scarcity has been one of biggest challenges and barriers, preventing many potential innovations and applications of machine learning. Even in choosing the right topic for our master thesis most of students had to abandon many of ideas just because of unavailability of the right datasets. Various data augmentation techniques aimed at artificially expanding the datasets have been developed by researchers to tackle such issues. Techniques such as synonym replacement and back translation have been widely utilized in NLP to generate new training samples from existing data. Although those backtranslations are currently one of the most popular methods and have the potential to get improved, synonyms replacement methodology has a lot of drawbacks. For instance, they can introduce semantic inconsistencies and unnatural sentence structures, which may degrade the performance of ML models rather than enhancing it.

As highlighted in the paper "Data Augmentation for Neural NLP" (Snajder) by Domagoj Pluscec and Jan Snajder, synonym replacement often degrades the performance of models, particularly transformer-based models used in NLP. The primary reason for this is that these pre-trained models have been exposed to vast amounts of data during their training which gave the models knowledge about a wide variety of linguistic contexts and vocabularies. Consequently, simply inserting synonyms which is considered as one of the data augmentation strategies, does not significantly enhance the model's ability to generalize beyond its training data. Moreover, models like these tend to develop nearly invariant representations for synonyms in their latent spaces, making such augmentations redundant and ineffective for improving model performance. Furthermore, is leads to the inefficiency caused by the confusion that arises when replacing words with their synonyms since it can lead to inconsistencies as well as semantic and grammatical errors.

2.2 Problem Statement

Although elementary techniques such as synonym replacement offer a cost-effective approach, more sophisticated methods provide markedly superior quality at a higher expense. The power of API-based tools such as ChatGPT, Gemini, and other state-of-the-art language models is utilized to generate nuanced and contextually appropriate textual variations of the given text datasets. One such method involves leveraging advanced language models like ChatGPT and Gemini, which offer API-based platforms for generating contextually rich and syntactically diverse textual data. These tools can simulate varied linguistic styles and

complex sentence structures, thereby enriching the training corpus with high-fidelity variants of the original texts. Another potent augmentation technique is high-quality translation, particularly using back translation. This process involves translating a text to a different language and then back to the original language using sophisticated translation APIs. This method not only diversifies the syntax and diction but also helps in smoothing out cultural nuances, thus providing a dual-layer enrichment of the dataset. Such API-based tools have revolutionized data augmentation by providing scalable, efficient, and highly diverse data inputs that are crucial for training state-of-the-art language models. Those are considered to be advanced methodologies existing thanks to attention-based transformer architecture which easily solves the problem of semantic and grammatical errors caused by simplistic data augmentation methods like synonym replacement.

Hence, machine learning engineers have two primary ways to expand their dataset while preserving the quality of the original data: manually utilizing computational power or using large language models LLMs such as ChatGPT, Cloude, Gemini, etc. The first approach requires devices capable of handling significant computational loads. For example, a data augmentation research project involving the backtranslation of 200 000 text samples with Google Translator on local machines required 32 processors to run non-stop for two weeks. Not all machine learning projects and startups can provide such computational power. The second option involves financial costs; currently, people pay $1 for 1 million tokens on ChatGPT. If we assume that one text sample typically contains 300 words, and there are 5 000 000 samples in our dataset, this translates to an average of 900 tokens per sample (300 input words plus 300 output words, equating to 600 words or approximately 900 tokens with a conversion rate of 1.5). Therefore, processing 5 000 000 samples would cost 6750 $ for 4.5 billion tokens. While this might be a manageable and affordable expense for well-funded startups and companies pursuing larger goals, most early-stage startups, low-funded PhD, master’s thesis projects, and initial research projects simply cannot afford it. This is not to mention the broader commercial use of such tools by machine learning enthusiasts for educational or personal purposes.

So, the computational intensity of traditional data augmentation methods often places them beyond the reach of small-scale projects or those operating with limited budgets. This project exactly aims to solve that problem and figure out ways that are not only effective but also computationally economical. Such strategies would democratize the benefits of machine learning, allowing projects with minimal resources to leverage advanced techniques to improve data quality and model accuracy. Easily used methodologies would give the machine learning enthusiast a chance to work, improve, investigate, and achieve greater results in the realm of machine learning since the more people are working on the different tasks, the faster domain is developing. Therefore, exploring new data augmentation strategies tailored for under-resourced languages not only addresses a significant gap in the field but also expands the potential applicability of ML to a broader spectrum of languages and cultural contexts.

2.3 Objectives

This brings us to the core problem that our research seeks to address. Ideally the question would sound like: how can we develop new data augmentation techniques that are not only effective in enhancing data quality and quantity for under-resourced languages like Azerbaijani but also computationally economical enough to be implemented on standard laptops? But considering that the limited number of resources and lack of financial support in this project, the research question sounds like what open-sourced attention-based

paraphrasing and translation tools can be utilized as new data augmentation techniques that are not only effective in enhancing data quality and quantity for under-resourced languages like Azerbaijani but also computationally economical enough to be implemented on standard laptops? This question is vital as it explores the potential for small-scale projects with limited budgets to compete in the increasingly demanding arena of machine learning, without the necessity for high-end computational resources. The importance of finding computationally efficient method is also explained by the exponential growth of models' sizes, that is why our investigation aims to assess various innovative and attention-based data augmentation strategies, determining their efficacy in improving model accuracy while maintaining linguistic integrity and minimal computational demands. This research endeavors to fill a crucial gap in the literature and provide a pragmatic guide for selecting the most appropriate data augmentation methods for specific ML tasks in resource-constrained environments.

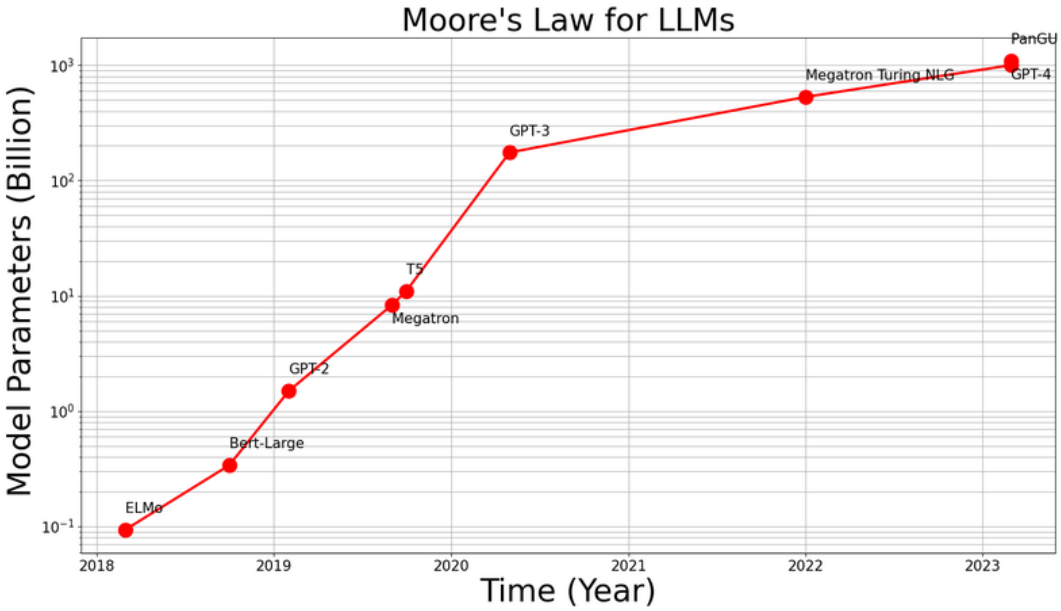


Figure 1: Exponential growth of the size of models with time.

In this research, a primary objective was to systematically evaluate the performance of attention-based paraphrasing and translation tools as methods for data augmentation in NLP for English and Azerbaijani languages. By focusing on the efficiency and effectiveness of these tools, we aim to address critical gaps in data availability that often impede the development of robust machine learning models. Our investigation is structured into several key phases, each designed to contribute to a comprehensive understanding of how these advanced computational techniques can be leveraged to improve model training processes, especially in contexts where resources are limited.

Initially, our efforts concentrated on identifying and testing a variety of open-source attention-based paraphrasers and translators available online. The purpose of this phase was to find out which of these tools could best adapt to the nuances of the Azerbaijani language, a crucial step for ensuring the relevance and applicability of our findings to real-world scenarios. All the experiments first made on English datasets. This involved rigorous testing of each tool's ability to generate linguistically and semantically accurate translations and paraphrases from large, well-established English datasets, such as those from IMDb movie reviews and AG News categorization tasks. These datasets were chosen for their high quality

and cleanliness, which are factors essential for maintaining the integrity of our experiments. Another reason was their popularity since there were a lot of projects and research experiments made on these datasets. Such popularity guaranteed that the data had been thoroughly vetted and utilized in numerous studies, providing assurance that it possesses the necessary attributes to serve as a robust testing dataset for my experiments. With its proven track record and widespread use in the research community, there is a specific level of confidence that this dataset will yield reliable and insightful results for my investigations.

Following the initial English language tests, one of the main goals of this research investigation of data augmentation ways for Azerbaijani language was achieved. For that particular dataset that has been specifically scraped from popular local news sources such as oxu.az. This allowed to directly assess the practical utility of the selected translation and paraphrasing tools when applied to content in Azerbaijani. Throughout this process, we utilized a diverse array of both traditional machine learning models such as logistic regression, Bayesian naive classifiers, and support vector machines—and more complex neural network architectures, including combinations of Long Short Term Memory networks and Convolutional Neural Networks. Our experiments were methodologically sound, employing multiple dataset sizes (small, medium, and full versions) to examine scalability and performance under varying data volumes.

To ensure thorough documentation and analysis of our experimental results, we integrated Weight & Biases (wandb) for detailed logging of model performance across all tests. Since all the models were of classification type. This setup provided an extensive array of performance metrics, including accuracy, precision, recall, and the F1-score, alongside loss charts which were pivotal for understanding model behavior over time. The culmination of this research is a rich compilation of graphical representations (graphs, charts and tables) that not only illustrate the outcomes but also guide machine learning practitioners in selecting the most appropriate data augmentation strategies for their specific needs, particularly when working with limited computational resources and Azerbaijani and English text datasets in NLP tasks. This comprehensive approach promises to significantly advance the field by providing practical, evidence-based recommendations that can democratize access to advanced machine learning capabilities.

2.4 Scope

However, it is important to understand the scope and acknowledge the limitations inherent to our study to set realistic expectations for the applicability of the findings. A significant limitation we faced was the lack of direct paraphrasing tools for Azerbaijani. As a solution, a two-step process was employed where Azerbaijani texts were first translated to English, paraphrased using English-language tools, and then translated back to Azerbaijani. This indirect method potentially introduces layers of semantic distortion and complexity, impacting the integrity of the paraphrased output. The quality of the text may drop 3 times: in the first translation to English, in the paraphrasing stage, and in the backtranslation to Azerbaijani language.

Despite the promising results of these experiments, the indirect paraphrasing process clearly shows a critical gap and pressing need in language technology for Azerbaijani, need for native-language computational tools. The positive outcomes of this study could indeed pave the way for securing funding for a dedicated research project aimed at developing bespoke attention-based paraphrasers for Azerbaijani. Such advancements could significantly

streamline data augmentation processes and enhance linguistic accuracy. After all this experiment really showed that paraphraser work for data augmentation, and this can be deciding factor for investor to invest the money into projects related to developing attention-based paraphraser and translators for Azerbaijani language.

As it was highlighted before the perfect outcome of this research would be developing Azerbaijani paraphraser. Hence another limitation and problem were the inability to train a custom paraphraser for Azerbaijani due to the absence of sufficiently large and clean datasets in the language. This scarcity of data restricts the ability to develop models that are finely tuned to the linguistic and contextual nuances of Azerbaijani, which is essential for high-quality paraphrasing. However, during this research project exact amount data and resourced needed to train Azerbaijani paraphraser was figured out, and that can be a beginning for good site research project or entire PhD project.

One more thing to take into account is specific domains. While the English paraphrasers performed adequately across general domains, their application in specialized fields like medicine, physics, astronomy, chemics etc. and other complex scientific branches posed additional challenges. These domains require not just linguistic accuracy but also a deep understanding of contextual and technical nuances, necessitating further fine-tuning of the paraphrasing models to ensure precision and reliability.

Furthermore, the study's reliance on existing datasets and models may not capture the full spectrum of linguistic diversity and complexity found in real-world applications. The translation-paraphrase-translation cycle might not only introduce errors but also flatten some of the stylistic and nuanced elements of the original text, which are crucial in high-stakes fields such as legal and healthcare communications. Lastly the computational constraints of running these models on standard laptops meant that we could not always utilize the most computationally intensive, albeit potentially more accurate, models available in the field. This compromise between computational demand and performance is a critical consideration for projects with limited resources.

2.5 Overview of the Paper

Following the introduction, the structure of our paper unfolds through several detailed sections designed to comprehensively explore the utilization of attention based paraphrasing and translation tools for data augmentation in machine learning. Initially we present a literature review that situates our research within the broader discourse on data augmentation techniques, emphasizing past methodologies and identifying gaps specific to under-resourced languages like Azerbaijani. Subsequent sections detail the methodology employed in our experiments, including the selection of datasets, the translation and paraphrasing processes used, and the models tested. This is followed by an in-depth presentation and analysis of the results, where we critically assess the performance of the paraphrasing tools across various metrics and discuss the implications of these findings. The discussion section then contextualizes our results within the existing literature, pondering the practical implications for machine learning projects limited by data scarcity and computational resources. We conclude with a summary of our key findings, a reflection on the limitations of our study, and suggestions for future research directions that could further refine data augmentation techniques for Azerbaijani and other similarly under-resourced languages. This structured approach ensures a logical flow from background to conclusion, providing a clear and comprehensive exploration of the subject matter.

3 Literature review

The rapid advancement of machine learning, particularly in NLP, has significantly benefited from the development of robust data augmentation techniques. These methods, which aim to enhance model performance through the synthetic expansion of training datasets, have become crucial in overcoming the inherent limitations posed by data scarcity in under resourced languages. It is true that in modern day the amount of unstructured data is growing, however, the key point here is “unstructured”. The problem is the fact that for most NLP projects we need labelled datasets, and as was noted in the previous section, the amount of data in ML, especially in NLP, really matters. That is why there were a lot of studies to figure out the ways to tackle the problem of data shortage, and in this section, we will discuss the most popular among them.

3.1 Data augmentation for Neural Networks

Prior studies, such as those by Snajder et al. (Snajder), have underscored the effectiveness of traditional augmentation strategies like synonym replacement and back-translation in improving data diversity and model robustness across general linguistic tasks. However, these techniques often introduce semantic inconsistencies, especially when applied to languages with limited computational resources, such as Azerbaijani.

The paper provides a comprehensive overview of state-of-the-art data augmentation methods tailored for neural and transformer-based models. Highlighting the costs associated with acquiring large, labeled datasets, the authors discuss low-cost data augmentation as a viable solution to enhance model training and performance in low-resource language settings as it has been already discussed in previous sections.

Data augmentation, as defined in the study, involves generating new data from existing datasets to improve the robustness and accuracy of machine learning models. The paper categorizes these methods into paraphrasing, noising, and sampling, each with unique strategies to diversify data while maintaining label integrity. The effectiveness of these methods is debated, particularly in the context of their application with advanced transformer-based models, which might not benefit significantly from simple transformations due to their pre-training on diverse data.

The paper also addresses the practical challenges and considerations involved in applying data augmentation. These include maintaining label accuracy during augmentation, selecting appropriate augmentation strategies based on model requirements, and the computational costs associated with these processes. The authors advocate for a strategic approach to data augmentation, suggesting the combination of multiple methods to maximize data diversity and model performance.

An in-depth discussion is provided on the specific applications of various augmentation strategies for different NLP tasks. Techniques such as synonym replacement, back-translation, and adversarial example generation are examined for their potential to contribute effectively to models trained on tasks ranging from text classification to machine translation.

The authors argue for the tailored application of these techniques based on the specific needs and constraints of the task at hand.

The importance of data augmentation is once more highlighted in overcoming the limitations caused by data scarcity in NLP. By enhancing the diversity and volume of training data, these methods help improve the generalization of machine learning models, particularly in under-resourced language scenarios. The authors conclude with a call for future research to develop more efficient and task-based augmentation techniques that could further revolutionize the field of neural NLP.

3.2 Data augmentation techniques for Azerbaijani language

Another recent new research that aimed to augment the dataset for Azerbaijani language is the research project by Ziyaden (Atabay Ziyaden, 2023) which explores the enhancement of text classification capabilities in low-resource languages, focusing specifically on Azerbaijani. Central to their methodology is the innovative use of the Facebook mBart50 model and the Google Translate API for text augmentation. These tools were used both independently and in conjunction to expand the text data available for training models in Azerbaijani. The study's results demonstrated a significant improvement in classification performance when models were trained on augmented datasets compared to those trained on original, non-augmented datasets. This underscores the potential of data augmentation strategies to enhance NLP capabilities for underrepresented languages. Furthermore, the research contributes to the field by publishing a labeled text classification dataset and a pre-trained RoBERTa model specifically for Azerbaijani which offers valuable resources for future studies.

This research project is very similar to our project with the only difference that we have chosen a little bit different path and decided to test small attention-based models and figure out low resource (financial and computational) requiring models. Usage of mBar50 model was our common intersection point since it is the best translation model that is available and does not require high computational power at inference time. We have also utilized same Azerbaijani dataset scrapped from oxu.az website since this dataset is popular with its cleanness and high quality.

The research once more shows the disparity in the development of language models between well-resourced languages and those considered low resource. Although Azerbaijani is spoken by millions, typifies the latter category due to the dearth of advanced, transformer-based language models adapted to its linguistic nuances. The study’s motivation is rooted in this gap, aiming to leverage limited resources to enhance text classification in Azerbaijani through augmentation and fine-tuning techniques. This approach not only seeks to advance NLP research within the context of Azerbaijani but also to stimulate similar efforts for other low-resource languages.

The findings of the study reveal that text augmentation, particularly through translations using the Facebook mBart50 model and the Google Translate API, significantly boosts the performance of news text classification models. This performance uptick is evident when comparing models trained on augmented datasets to those trained on original data, showcasing the effectiveness of combining different data augmentation strategies. There is an interesting fact – we also utilized mBart50 model and got similar results that will be shown in next sections. So, considering their results it can be concluded that this study highlights the

feasibility and benefits of these augmentation techniques, especially for languages like Azerbaijani that face challenges due to limited linguistic resources.

3.3 Data Augmentation for Sentiment analysis

As has been already noted in this project only classification datasets have been considered and tested, including the sentiment analyses. That is why the particular and recent research project, Toward Text Data Augmentation for Sentiment Analysis, (Hugo Queiroz Abonizio, 2022) was extremely useful in the earlier stages of our project. This study explores a variety of augmentation methods such as Easy Data Augmentation, back-translation, and advanced transformer-based methods like BART and PREDATOR, paired with diverse classifiers including LSTM, CNN, BERT, and others across different sentiment analysis tasks and datasets. As has been earlier discussed, we have also utilized LSTM and CNN architecture type of models in a pair with transformer-based paraphrasers.

These methods were evaluated in contexts of imbalanced datasets and limited data availability, which are common challenges in natural language processing (NLP). The paper provides a comprehensive taxonomy of NLP augmentation methods, highlighting the effectiveness of these strategies across various scenarios. For instance, it was noted that augmenting data using BT and PREDATOR significantly enhanced the performance of traditional machine learning models on imbalanced datasets by balancing class distribution and enriching the feature space with syntactically varied examples.

The research also delves into the impact of augmentation on resource usage, a particularly relevant aspect given our project's focus on low-resource settings. The authors of the paper managed to demonstrate that text data augmentation could be effectively achieved without substantial computational resources, thus aligning well with our project’s aim to develop models that are feasible on non-specialized hardware like standard laptops. The existence of such papers and projects once more outlines the importance of our project.

The results indicate substantial improvements in model performance, especially in scenarios involving reduced datasets or those balanced by augmenting the minority class, which notably enhances the quality of the datasets and thereby the robustness of the classifiers. The authors came to the conclusion that while all the augmentation techniques generally improve classifier performance, the degree of enhancement varies depending on the classifier and the specific characteristics of the data scenario. Techniques like PREDATOR and back-translation were particularly effective, suggesting that complex augmentation strategies that go beyond simple transformations (such as synonym replacement or random insertion) can provide more substantial benefits. That is why in out project we have decided to include back translation for Azerbaijani language as one of the main data augmentation tools. Additionally, the study provides a critical discussion on the implications of augmented datasets in real-world applications, offering insights into how different augmentation techniques can mitigate common data issues in sentiment analysis such as class imbalance and limited labeled data. This exploration helps pave the way for more sophisticated and contextually aware augmentation strategies in improving the performance and applicability of NLP models in sentiment analysis and beyond.

3.4 LM-based Text Augmentation

Finally, the most important and useful paper for our project was Neural Data-to-Text Generation with LM-based Text Augmentation (Ernie Chang). The paper "Neural Data-to-Text Generation with LM-based Text Augmentation" by Ernie Chang et al. proposes a novel few-shot learning approach to improve neural data-to-text generation systems. The primary challenge in training these models, as highlighted in the paper, is the scarcity of paired text samples for supervised learning. To address this, the authors introduce a method to augment the available text data by using a pretrained language model (GPT-2), which can generate new text samples from existing ones by altering specific values or through generative techniques. This augmented dataset is then paired with existing data samples using a cycle consistency technique to ensure the fidelity of the generated text to the original data intentions.

The experimental results presented in the paper indicate that this new approach significantly enhances the performance of sequence-to-sequence models on the E2E and WebNLG benchmarks. In particular even with less than 10 percentages of the data annotations typically required for training, the augmented model achieves better performance than fully supervised models, increasing BLEU scores by over 5 points and setting new state-of-the-art results on these datasets. This showcases the potential of combining few-shot learning with generative text augmentation to reduce the need for extensive annotated datasets in data-to-text applications.

A critical aspect of the methodology is the cycle consistency approach, which underpins the model's ability to learn accurate pairings between data samples and augmented text. This process helps mitigate the risk of incorporating noisy data into the training set, which is a common issue with automatically generated text. By ensuring that data samples can be accurately reconstructed from their textual descriptions, and vice versa, the model maintains high fidelity to the original data meanings, which is essential for reliable text generation.

The paper also delves into representation matching and the alignment of augmented text samples with similar data samples in vector space, which is crucial for the model to generalize well from limited examples. This approach not only improves the quality of the training pairs but also enhances the model's performance over iterations as the matching becomes increasingly precise. The use of a shared encoder for both data and text inputs is another key feature that facilitates this process by fostering a unified representation space.So by leveraging few-shot learning and text augmentation techniques, the model significantly reduces the dependency on large annotated datasets, which are costly and labor-intensive to produce. This approach not only improves efficiency and scalability but also opens up new possibilities for deploying advanced neural generation models in resource-constrained settings.

The extensive review of literature provided in this paper has been instrumental in framing and informing the scope of our research on data augmentation techniques for machine learning in under-resourced languages like Azerbaijani. The studies discussed, ranging from general NLP applications to specific issues with data sparsity and computational limitations, have highlighted the critical role that sophisticated data augmentation methods can play in enhancing model robustness and performance. By examining various augmentation strategies, including synonym replacement, back-translation, and more advanced transformer-

based methods, we have gleaned insights into their effectiveness and limitations across different linguistic and domain-specific contexts. This understanding has been pivotal in choosing the right augmentation techniques for our project, ensuring that our methods are not only theoretically sound but also practically viable in addressing the nuances of Azerbaijani language processing, and by all of these we could at the end achieve the results and improve the performance of the tested models.

Furthermore, the literature review has underscored the feasibility of using such augmentation strategies in low-resource settings, a key consideration given our project's constraints on computational resources and dataset availability. The insights from these papers have directed our experimental design, particularly in our approach to using translation-based augmentation methods to circumvent the lack of direct Azerbaijani paraphrasing tools. This strategy, inspired by the successful applications discussed in the reviewed literature, has allowed us to adapt and innovate within our resource limits, thereby contributing to the broader field of NLP. Only because of these papers we have decide to use small open-sourced models that are available in internet since most papers outlined many times the problem of easy augmentation ways like synonym replacement and proved the importance and need in attention based small models. By drawing on these foundational studies, we have not only advanced our understanding of data augmentation's potential but also positioned our research to potentially stimulate further developments in the augmentation of other under-resourced languages, reflecting a significant step forward in the global pursuit of equitable technological advancements in NLP.

4 Methodology and acknowledgment

In the initial phase of our methodology the main focus was put on data augmentation for the English language due to the availability of robust datasets and already existing sequence-to-sequence attention-based paraphrasing models. The primary sources for these models were various academic papers that had such goals as enhancing model capabilities in tasks such as summarization and paraphrasing. Among the models that have been tested, one of the best models was Pegasus which stood out due to its effectiveness in generating paraphrased content while maintaining semantic integrity. This mode was rigorously tested to determine its utility in paraphrasing English language datasets, which served as the foundational step in our data augmentation process.

4.1 Pegasus

Pegasus was released to the world not as a paraphraser but more as a summarizer. While the model itself was released by Google in 2019, "PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization" (Jingqing Zhang, 2020) paper was released in the middle of 2020. It focused on an approach to pre-train large Transformer-based models for abstractive text summarization using a self-supervised objective. The authors showed a method called Gap Sentences Generation (GSG), where important sentences are removed from an input document and are then generated as a summary from the remaining content. One of the main aims of this approach is to mimic the extractive summarization process and is hypothesized to improve summarization performance by fostering better document understanding and summary generation capabilities.

The authors pre-trained the PEGASUS model on a large corpus that included web and news articles and used the GSG objective. They evaluated the model on a diverse set of 12 summarization tasks involving different types of text such as news, scientific articles, emails, patents, and legislative bills. Their model demonstrated state-of-the-art performance across all 12 datasets it was tested on, and achieved a high ROUGE score which is one of the standard metrics for evaluating text summarization models in machine learning domain. One of the key features of the model became robustness that was showed by the model in low-resource settings, surpassing previous best results on several datasets with limited training examples. The nuance which is worthy of attention is fact that model's summaries were comparable to human-generated summaries in terms of quality, as validated through human evaluation. The research members came to conclusion that the GSG pre-training objective is effective for abstractive text summarization tasks. The method not only performs well across various domains but also excels in low-resource scenarios. This suggests that the GSG objective, which focuses on generating gap sentences from texts, is a viable strategy for preparing models to generate coherent and contextually accurate summaries. They also showed that this approach allows for rapid adaptation to new summarization and paraphrasing tasks with minimal fine-tuning. The overall findings support the use of targeted pre-training strategies to improve and enhance the capabilities of language generation models in specialized tasks like paraphrasing.

After successful release of the paper PEGASUS was fine-tuned from a pre-trained checkpoint for specific tasks that was paraphrasing. When adapted for paraphrasing, it uses the powerful capabilities of its seq2seq framework, originally designed for summarization, to generate diverse paraphrases from a given text. The paraphraser model has around 568 million parameters, and while there is no official information about the data that was fed to the model, it is assumed that it was trained on massive text corpus that includes C4 corpus which has 350M Webpages and HugeNews dataset which as 1.5B news documents. The number of parameters and the huge amount of the dataset that it was trained on looked promising in terms of improving the results of the tested models.

Figure 2 illustrates the architecture of the model which utilizes 2 tasks Gap Sentences Generation and Masked Language Model. GSG task is basically the selection and masking of whole sentences in a given document containing multiple sentences, which are then used as a summary target that the model should give as an output. This is designed to encourage the model to learn summary capabilities since the masked sentences often carry salient information about the document's content. Meanwhile, MLM masks the token from the remaining sentences and tries to generate them which gives model ability to understand the context better. Hence, MLM and GCG operate concurrently in the real process in which Encoder is forming rich representation of masked input sequence of tokens to pass it to Decoder which predicts the target text that is basically previously removed sentence. This helps model to better understand the context of the text and accurately summarize and paraphrase token sequences at inference time during which it just predicts the next token given the input text plus already generated tokens.

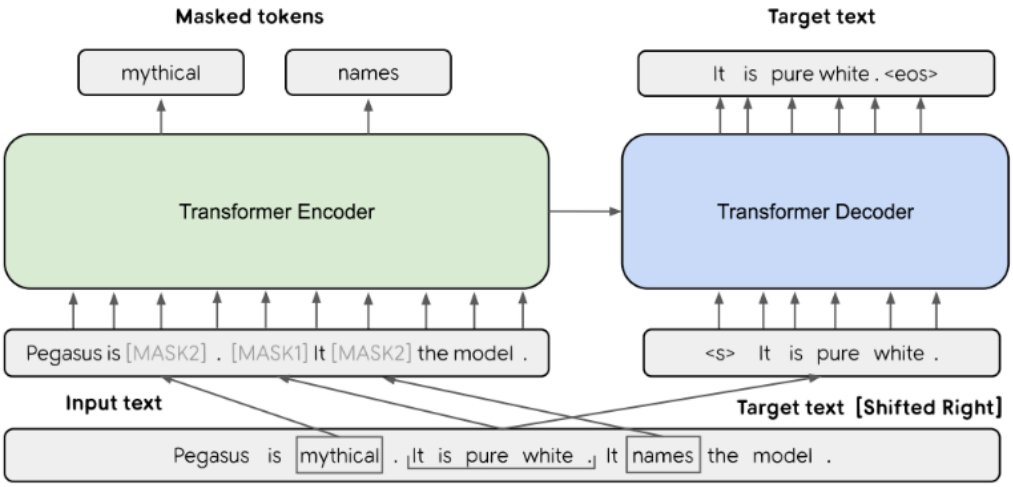


Figure 2: Pegasus model architecture,[Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2019). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization]

The main challenge with working PEGASUS paraphraser was the limited number of tokens that we could take as an input. It could process only 65 tokens, and if it is converted with 1.5 rate of conversion, 43-44 words will be gotten. The problem is that most of the NLP samples in datasets, even in classification problems, consist of an average of 200 words. This meant that the whole text should be divided into several parts pairing N number of sentences. And here we have started our first experiment. The samples were divided into N parts (from 7-10), and the results were checked to decide how to smartly divide the text samples before feeding into the model. All of those results will be shown in the following section. Detailed analyses of the dataset will be made as well to give a better understanding to the readers, and in this section, we will focus on the details of the experiments as well as the theoretical background of the models and different implementation techniques.

4.2 Optimization techniques

By way of anticipation, it turned out that the smartest way to divide the text was taking the each sentence as 1 part, and giving the whole text to the model as a batch. Here we came to one of the most important parts of our project – oprimization which was the key factor that will reduce the time of data augmentation process several times:

Firstly, the utilization of data parallel technique, specifically through PyTorch's "nn.DataParallel", played a pivotal role. This module facilitates the distribution of data across multiple processing units, such as GPUs, which gives a chance for simultaneous model training on different batches of data. After segmenting the text into sentences and distributing these across GPUs, each unit is able to process individual sentences in parallel, dramatically accelerating the throughput and efficiency of the model. This approach not only optimizes the use of available hardware but also minimizes the computational overhead, leading to a significant reduction in training time without sacrificing the accuracy of the model outputs. How well it fastened the model at the inference will be shown in the result part of the paper.

Secondly, parallel batch processing within Transformer models like PEGASUS further enhanced performance. Transformers inherently benefit from batch processing thanks to their ability to handle sequences of data in parallel, while, for example, recurrent neural networks do that process sequentially. After implementing parallel batch processing, the PEGASUS model is able to leverage its attention mechanisms more efficiently through the entire dataset. Each batch of sentences have been processed concurrently which allowed the model to maintain high throughput rates and ensure the context coherence and accuracy in the generated summaries. This method maximizes the utilization of the Transformer architecture's parallelization capabilities, hence, optimizing both resource usage and processing speed.

Thirdly, introduced in 2022 Better Transformer has been instrumental in enhancing the operational speed of Transformer models. This function optimizes Transformer models, and enables a more efficient execution path that is fully compatible with existing model structures, and hence not requiring any changes to the models themselves. When this method is applied, it aligns the Transformer model with optimized inference paths that integrate improvements similar to fused kernels for multi-operation efficiency and sparsity exploitation. These enhancements are very crucial in environments where computational resources are at a premium, and they help reduce latency significantly.

We have used those techniques in our pipeline which made us able to harness these optimizations seamlessly. The enhanced model leveraged advanced kernel fusion techniques that aggregate multiple operations into fewer more efficient steps The cumulative effect of these optimizations was a substantial boost in processing speed and made our use of the PEGASUS model notably more efficient in handling large datasets for text paraphrasing tasks. This method proved essential for maintaining high performance while scaling our applications, highlighting its value in practical, high-demand settings.

4.3 T5

The second paraphraser that showcased good results was the model that was a T5 paraphraser. The model has been finetuned on a dataset of text paraphrases, including questions from Quora, text from SQUAD 2.0, and CNN news articles. Before starting the discussion of the implementation details, the initial acknowledgement amount of the base model should be provided to understand the paraphrasing capabilities of the model.

The most interesting part is that it appeared to the world approximately at the same time as Pegasus model - T5 model, or Text-to-Text Transfer Transformer, introduced by researchers from Google in 2020, represents a significant advancement in the field of natural language processing. Central to its design is the unifying principle that all NLP tasks can be framed as text-to-text problems, meaning any task—be it translation, summarization, classification, and even paraphrasing, literature anything can be approached by converting input text into output text. This approach simplifies the NLP pipeline, eliminating the need for task-specific architectures. T5 was pre-trained on a large corpus derived from the Colossal Clean Crawled Corpus, employing a denoising objective similar to the one used in another Google's creation, BERT, which involves predicting masked or scrambled parts of sentences. It has around only 220 million parameters compared with Pegasus which has 568 million parameters. One of the main questions in our research was about the ability of Pegasus model to outperform T5 models since it has more parameters. However, T5 extends this concept by

training not only to predict missing tokens but also to reformat disordered sentences, thereby integrating aspects of understanding and generation more seamlessly than its predecessors. Its architecture is based on the Transformer model, which is favored for its ability to handle long-range dependencies and its efficiency in parallel computation.

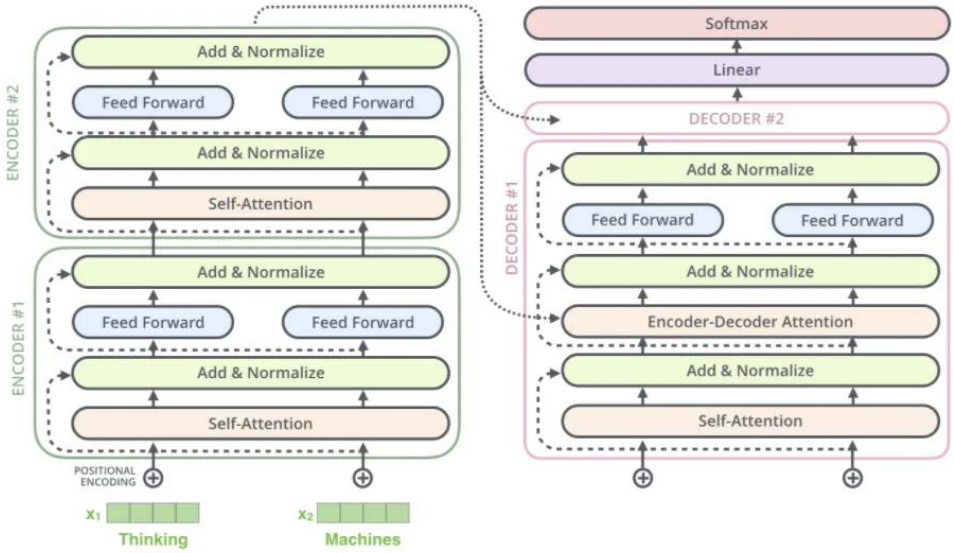


Figure 3: T5 model architecture, [Jay Alammar]

T5 has typical attention based Encoder-Decoder architecture described in Figure 3, in which the encoder maps an input sequence of words, which has been already passed through positional encoder to retain sequence order, into a latent space representation, capturing the context and semantics of the text through self-attention mechanisms. This representation is then taken by Decoder through a series of self-attention and encoder-decoder attention layers, generates the target sequence word by word.

T5 has already demonstrated state-of-the-art performance across multiple benchmarks, making it a versatile and powerful tool in modern AI applications, notably improving efficiency and reducing complexity in deploying models for diverse NLP tasks. Exactly because of these achievements it was decided to test it for data augmentation.

4.4 mBart50

In our project a great number of paraphrasing tools have been tested, and those 2 were selected due to their ability to preserve the quality and initial semantic meaning of the words. All the examples of paraphrasing as well as real results on machine learning datasets will be illustrated in the next sections. And although the results of those 2 models were incredible, we had a biggest challenge in our entire research project – those models did not work in Azerbaijani language since there were trained and fine-tuned in English language. Although there are a lot of translating tools on the internet, we could not use any of them since the main goal of our project is to achieve data augmentation without any API tools and find out ways to do it locally on simple laptops with minimal GPU power.

For that reason, a great number of open-source translating tools have been tested and one particularly useful and of good quality has been chosen to be used as our translatior - mBART-50, which was introduced by Facebook AI, and is a multilingual sequence-to-

sequence model designed for machine translation, and it represents a significant extension of the original mBART model. Standing for multilingual denoising pre-training for machine translation, mBART-50 is pre-trained on a large corpus covering 50 different languages. This broad linguistic coverage makes it especially valuable for translation tasks involving low-resource languages. The model architecture is based on the Transformer model, which is known for its effectiveness in handling sequence-to-sequence tasks. Such architecture makes it very easy to finetune it for different languages as the figure below describes.

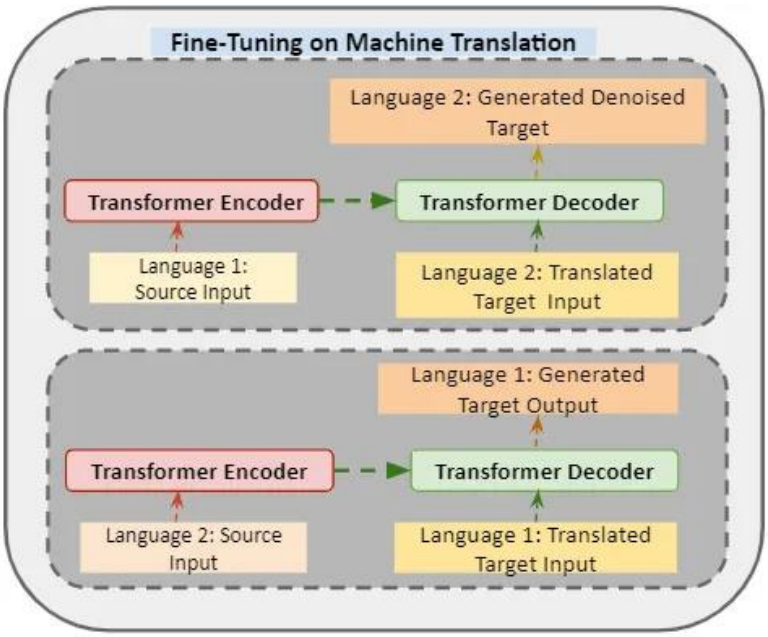


Figure 4: mBart50 fine-tuning process, [The NLP Cookbook: Modern Recipes for Transformer based Deep Learning Architectures - Scientific Figure on ResearchGate]

mBART-50 is particularly noteworthy for its large number of parameters, totaling approximately 680 million. One of the interesting moments in out experiment was the fact that it took much less time to translate our datasets with the mBart model than to paraphrase the same dataset with our paraphraser models, although mBart has more parameters than Pegasus and T5 models. Nevertheless, his extensive parameter set enables the model to capture complex patterns and nuances across different languages. The languages supported include a diverse range from various linguistic families, such as Romance, Germanic, Slavic, and Indo-Aryan languages, as well as East Asian and African languages, and the most importantly it supported Azerbaijani language, this wide-ranging support makes mBART-50 a versatile tool in the field of machine translation, capable of handling direct translations between many language pairs without pivoting through English, thus preserving the original context, and meaning more effectively. Another interesting nuance was the fact that research that recently released the paper, (Atabay Ziyaden, 2023) in which they have tried to augment Azerbaijani datasets, also utilized mBart as one of their paraphrasing tools.

Once all the major attention transformer tools have been introduced, it is time to give information about the datasets that were used in our experiment. Overall, there were 2 English datasets, and 1 Azerbaijani dataset. There were several criteria that was considered in choosing them, the first one was the number of classes, we have binary classification datasets, a dataset with 4 classes and a multiclass dataset, to consider as many possible scenarios as possible and see how different augmentation ways work for different cases. Secondly, we paid attention to the length distribution of the text samples in datasets, so we have a specific variability in terms of having datasets with short text samples as well as with

long text samples. The third factor was the distribution of the classes to avoid imbalance situations, although it might be a good idea for paper, to augment samples of classes which have fewer examples in dataset. Since our main goal was to test the paraphrasers itself, we decided to choose datasets with evenly distributed classes, and remove some of the imbalanced classes in one of our datasets.

4.5 IMDB reviews dataset

The first dataset is the IMDb review dataset, commonly known as the IMDb Movie Reviews Dataset, a widely utilized dataset in NLP and sentiment analysis research. There are 50k movie reviews sourced from the IMDb website, which are evenly split between positive and negative sentiments. This balanced division makes it particularly valuable for binary classification tasks, which is explained by each review being labeled as either 'positive' or 'negative', which in its tern, help to establish a clear framework for supervised learning models aimed at sentiment classification. Those 50 000 samples have been divided into 40 000 and 10 000 train and test data points.

The dataset's importance level can be understood by its application in numerous research papers, where it has often served as a benchmark to evaluate and compare the performance of various text processing and machine learning algorithms. Notably, it was featured in the work by Andrew L. Maas et al., in 2011 paper "Learning Word Vectors for Sentiment Analysis" (Andrew L. Maas, 2011) where the dataset's utility in training word vectors for sentiment analysis was demonstrated. Figure 5 shows the token distribution per sample in train dataset, and from this figure it is clear that each sample had in average 250 tokens which is equivalent to 1 paragraph of text making it easy in applications. Another advantage of this dataset is perfect balance in classes since there exactly 25 000 positive and 25 000 negative reviews. This dataset is also remarkable for its exclusion of any reviews with neutral sentiments, since it solely focuses on clearly polarized opinions, and such phenomenon enhances its effectiveness for training models to recognize extreme sentiments. Its use in academic and industrial research continues to provide insights into the advancements in machine learning techniques and their application to real-world text analysis tasks.

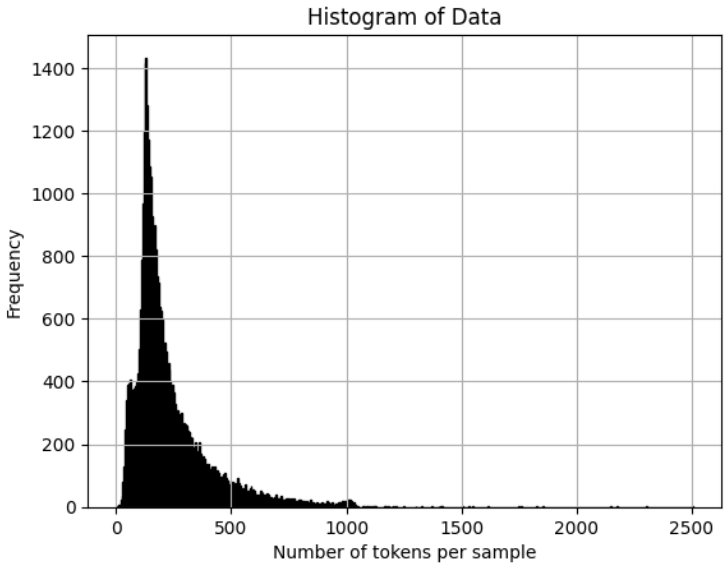


Figure 5: Distribution of the tokens in IMDB dataset.

The tables below present the augmentation results with 2 paraphrasers. The overall pipeline of the experiments will be discussed in detail in Methodology section, while in this section more attention is paid to the data analytics. As Table 1 shows, to imitate the data shortage 80% of the train dataset was taken to augment it with different paraphrasers, hence, 32 000 samples were augmented, and 64000 samples were obtained. In terms of vocabulary richness, we see an elevation of 20 000 and 60 000 tokens after augmentation with T5 and Pegasus, while the number of unique common tokens from the test dataset increased by approximately 2000 and 3000 tokens respectively. In terms of preserving the semantic structure and grammatical correctness, Table 2 shows excellent paraphrasing capabilities of both models.

Model	Data	Data type	Number of samples	Vocabulary length	Number of common tokens in test dataset
-	IMDB	original	32000	203946	77749
Pegasus	IMDB	augmented	64000	223664	79170
T5	IMDB	augmented	64000	263907	80700

Table 1: IMDB dataset augmentation results

-	Text
original	As a big fan of Tiny Toon Adventures, I loved this movie!!! It was so funny!!! It really captured how cartoons spent their summers.
T5	I loved the movie, Tiny Toon Adventures, because I'm a huge fan of comic book series. It was funny and captured the essence of how cartoons spent their summers.
Pegasus	I loved this movie because I am a big fan of Tiny Toon adventures. It was hilarious. It showed how cartoons spent their summers.

Table 2: IMDB augmentation example

4.6 AG news dataset

The second dataset is the AG News dataset is a widely utilized collection in the field of text classification. It encompasses a large-scale assembly of news articles divided into four primary classes, which are World, Sports, Business, and Science/Technology. These categories facilitate a comprehensive evaluation framework for various machine learning models aimed at handling multi-class classification tasks. Each class in the dataset contains approximately 30000 training samples and 1900 test samples, summing up to around 120,000 training samples and 7600 test samples in total. However, considering the limitations of the models, samples with two long text samples have been removed since some sentences in those samples were too long. As T5 can handle longer sequences, fewer samples were drop in its experiments, hence, in T5 case 114407 out of 120 000 were left, while for Pegasus this number was 91700. But even after reduction the significant volume of data still preserves and together with diverse content spectrum make it a perfect benchmark in evaluating and comparing the performance of text classification algorithms. AG News has been prominently featured in several influential papers, including "Character-level Convolutional Networks for Text Classification" (Xiang Zhang) by Xiang Zhang et al., where it was used to demonstrate the efficacy of deep learning models in text understanding and classification without the need for word-level preprocessing. Its straightforward format and well-defined class structure have established it as a fundamental resource in NLP studies, enabling researchers to test the robustness and accuracy of various computational models in a controlled environment.

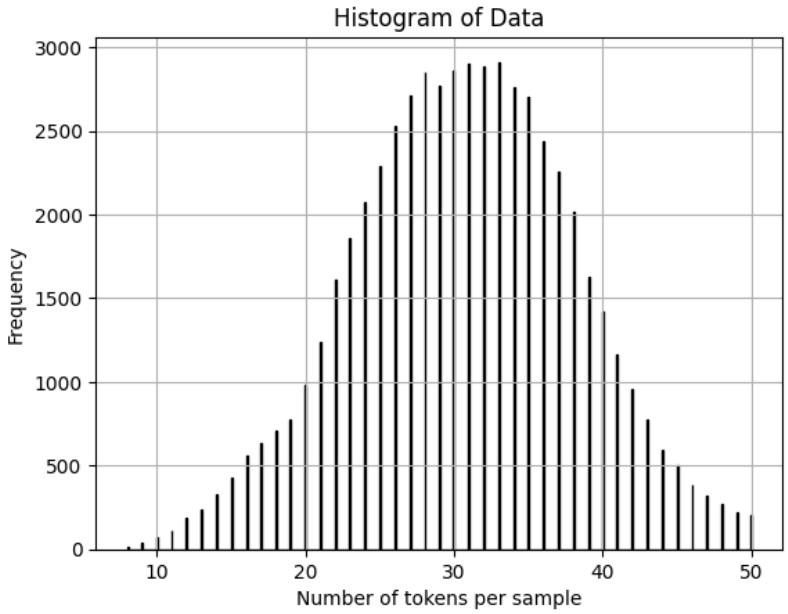


Table 3: Distribution of the tokens in AG news dataset

Model	Data	Data type	Number of samples	Vocabulary length	Number of common tokens in test dataset
Pegasus	AG	original	45850	63950	22263
Pegasus	AG	augmented	91700	71426	23145
T5	AG	original	57204	76734	23340
T5	AG	augmented	114408	100478	25254

Table 4: AG dataset augmentation results

-	Text
original	A monthly deficit on its oil balance saw the UK s worldwide trade gap widen to its highest level since January.
T5	The UK's oil balance experienced a monthly deficit, leading to the largest trade gap since January.
Pegasus	The UK's trade gap widened to its highest level since January due to a monthly oil deficit.

Table 5: AG dataset augmentation example

The table above shows that on average there were 30 tokens per sample, making it a lot shorter than samples of IMDB dataset. This dataset was specifically picked for preserving variety in our experiments. Another difference from IMDB dataset is its multiclass nature. There are 4 classes equally distributed over the dataset, which again adds variability to our data pool. In terms of vocabulary richness again we observe superiority of T5, it could increase the vocabulary size by 24k compared to 8k tokens elevated by Pegasus, and the main reason behind such phenomenon is large corpus data that was fed into T5.

4.7 OXU dataset

Finally, the third data is Azerbaijani dataset scrapped from oxu.az website, which is essentially news categorization dataset. The dataset has 15571 samples and 16 classes of news categories. Among them there are some very imbalanced categories that were decided to drop, about which the detailed information will be provided in next sections. Generally, dataset is one of the cleanest and high-quality datasets in Azerbaijani language that guarantees high accuracy and consistency in machine learning models. Because one of our

main goals was to develop paraphraser for Azerbaijani language, this dataset is very valuable for developing algorithms that require understanding and processing Azerbaijani text since there is not that many good Azerbaijani classification datasets. There are a very wide range of topics from politics and economics to technology and health which gives a comprehensive overview of affairs which have been reported in local media. I next sections we will discuss in detail the preprocessing techniques used to handle class imbalances and optimize the dataset for more effective training and validation phases.

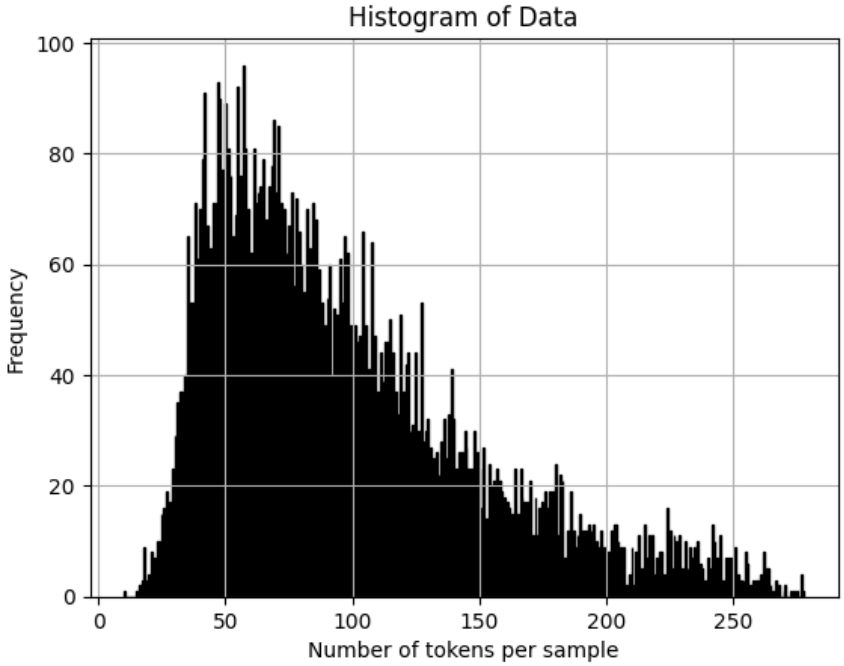


Table 6: Distribution of the tokens in OXU news dataset

Model	Data	Data type	Number of samples	Vocabulary length	Number of common tokens in test dataset
Pegasus	Oxu	original	5296	79833	25603
Pegasus	Oxu	augmented	10592	116796	26545
T5	Oxu	original	7964	106360	29099
T5	Oxu	augmented	15928	165382	29937
mBart	Oxu	augmented	15928	168439	30024

Table 7: AG dataset augmentation example

-	Text
original	Azərbaycan Respublikasının Prezidenti, Müzəffər Ali Baş Komandan İlham Əliyev xalqa müraciət edəcək. Müraciət televiziya kanalları ilə yayımlanacaq.
T5	Azərbaycan Respublikasının Prezidenti və Müzəffər Ali Baş Komandan cənab İlham Əliyev ölkə xalqına müraciət etməyi planlaşdırır. Bu müraciət yerli televiziya kanalları vasitəsilə yayımlanacaq.
Pegasus	Azərbaycanın Prezidenti və Müzəffər Ali Baş Komandan İlham Əliyev televiziya vasitəsilə xalqa müraciət edəcək.
mBart	Azərbaycan Respublikasının Prezidenti və Qələbəli Ali Baş Komandan İlham Əliyev xalqa müraciət edəcək. Müraciət televiziya kanalları vasitəsilə nümayiş etdiriləcək.

Table 8: OXU dataset augmentation example

The tables above showcase the results of augmentation, and surprisingly back translation brought more new tokens than T5 paraphraser, although backtranslation is a stricter data

augmentation technique in term of sentence modifications. In examples from Table 8 we observe that T5 as usually increases the number of tokens while Pegasus is doing the opposite. Strict augmentation of backtranslation is also observed in the case with mBart. It almost didn't change the sentence structure.

4.8 Evaluation

Now it is time to discuss the final piece of our project - the models that were trained on those augmented datasets. There were basically two types of models that were considered in our project. The first one is traditional supervised machine learning models such as Linear Regression, Naïve Bayse, SVM and Random Forest classifier. Those models guarantee simplicity and interpretability, which are crucial for understanding the underlying mechanics of the model and how it makes predictions. There are even cases where in official domains like banks use of highly complex and not easily explainable AI is prohibited since clarity is very crucial in scenarios where trust and transparency are paramount. Moreover, simpler models often require less computational resources, which is another goal of our .Their ability to work effectively on smaller or less complex datasets after enhancing by our data augmentation techniques ensures that they remain a viable option for many practical applications.

The second type of model was Neural Networks, we have tried several model architectures and decided to utilize Bi-Directional LSTM-CNN model (Zhou & Long, 2015) since it combines the temporal dynamic modeling capabilities of LSTM with the spatial feature extraction abilities of CNNs, which is a perfect fit for our case since it very well-suited for tasks that has sequence and context recognition such as NLP or time series analysis. This hybrid architecture allows for both forward and backward information flow and improves the model's ability to capture context from both directions, which has already shown better results in improving performance on complex datasets. The integration of CNN layers, in its turn, improves the model's detecting patterns or features at various scales or positions within the data, thereby enriching the model's learning capacity. So, our model utilizes the strengths of both LSTM and CNN, and combined Bi-Directional LSTM-CNN model offers a robust solution for tackling challenging problems that require an understanding of both long-term dependencies and spatial hierarchies in data.

4.9 Methodology

After introduction of all the details and parts of our project, it is finally possible to easily explain the whole pipeline of the experiment.

1. **Data processing**. Due to some limitations of generative models described above some samples with sentences that contain too many tokens have been dropped. Too long samples have been dropped as well in case of T5 since they did not fit into one batch and dividing it into several parts would lead to some semantic errors.
2. **Translation to Englis**h. This step was applied to only Azerbaijani dataset, Oxu. Hence, 7964 samples were translated to English language.
3. **Backtranslation to Azerbaijani language**. This step was again only applied to Azerbaijani sentence to get the first augmented dataset.
4. **Dividing the datasets into smaller parts**. This step has been applied only to English datasets to imitate data shortage cases. Azerbaijani dataset did not need such

procedure since it already has only around 15 000 samples in original version. In the case with English datasets 80% of IMDB dataset and 50% of AG news were taken for training dataset, although other percentages were also tested.

5. **Paraphrasing with tools**. English datasets as well as translated to English Azerbaijani dataset have been paraphrased with T5 and Pegasus to get news samples of text. At this stage we got 2 more data augmented datasets.
6. **Again backtranslation**. Backtranslation of paraphrased OXU dataset withT5 and Pegasus to Azerbaijani language. At this step we finally get our last 2 augmented datasets.
7. **Evaluation using traditional and neural network models**. Different machine learning models have been trained to get the accuracy of the model trained on original dataset versus on augmented dataset. For evaluation Precision, Recall, Accuracy, F1 score as well as their macro and weighted averages have been utilized.

5 Results

As it has been already discussed in Methodology part, overall, we had several types of augmentation:

1. English dataset, AG news, has been augmented utilizing different augmentation techniques:
    a. With Pegasus paraphraser in 10, 20, and 30 percentages. As was already discussed, those percentages show the smaller fractions of original dataset that was taken to simulate shortage of data.
    b. With T5 paraphraser in 10, 20, and 30 percentages.
2. Azerbaijani dataset, OXU news, has been augmented utilizing different augmentation techniques:
    a. With Pegasus paraphraser in full size. As was already discussed, since its size is small there is no need to further reduce it.
    b. With Pegasus paraphraser in full size.
    c. With mBart translator in full size.
3. English dataset, IMDB news, has been augmented utilizing different augmentation techniques:
    a. With Pegasus paraphraser in 80%. As was already discussed, those percentages show the smaller fractions of original dataset that was taken to simulate shortage of data.
    b. With T5 paraphraser in 80%.

All the results have been tested on different machine learning models to get the comprehensive understanding about the capabilities of listed augmentation techniques.

5.1 AG news

Initially, 10, 20, and 30 % of AG news dataset was paraphrased to increase the data size twice. We will start with the results of augmentation of 20% dataset (18368 in original dataset, and 36704 samples in augmented dataset). The figures below represent the

classification report after training with and without augmented samples utilizing traditional models as well as neural networks:

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.86	0.69	0.76	0.65	0.79	0.71
1	0.74	0.81	0.77	0.95	0.64	0.77
2	0.73	0.58	0.65	0.78	0.68	0.73
3	0.52	0.68	0.59	0.64	0.79	0.71
Accuracy	0.69			0.73		
Macro average	0.71	0.69	0.69	0.76	0.73	0.73
Weighted average	0.71	0.69	0.69	0.76	0.73	0.73

Table 9: LSTM model validation accuracy results trained on augmented AG news dataset with Pegasus, utilizing 10% of the original dataset, vs 10% of the original dataset.

The tables above clearly depict superiority of the models that were trained with increased dataset. We observe an increase of 4% in almost all the average measurement metrics, and even 5% in average Precision.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.62	0.8	0.7	0.85	0.82	0.83
1	0.95	0.53	0.68	0.93	0.89	0.91
2	0.72	0.62	0.67	0.81	0.75	0.78
3	0.62	0.8	0.7	0.71	0.83	0.76
Accuracy	0.69			0.82		
Macro average	0.73	0.69	0.69	0.83	0.82	0.82
Weighted average	0.73	0.69	0.69	0.83	0.82	0.82

Table 10: LSTM model validation accuracy result trained on augmented AG news dataset with Pegasus, utilizing 20% of the original dataset, vs 20% of the original dataset.

As it is clear from the tables above, after increasing the size of the dataset twice, the accuracy increased from 0.69 to 0.82, while average precision showed an increase from 0.73 to 0.82. We also see that precision, recall and F1 scores of all classes were increased and evenly distributed, while in the original dataset's case the values vary between 0.53 and 0.95.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.69	0.9	0.78	0.94	0.68	0.79
1	0.95	0.82	0.88	0.86	0.95	0.9
2	0.85	0.67	0.75	0.73	0.81	0.77
3	0.73	0.76	0.74	0.76	0.81	0.78
Accuracy	0.79			0.81		
Macro average	0.8	0.79	0.79	0.82	0.81	0.81
Weighted average	0.8	0.79	0.79	0.82	0.81	0.81

Table 11: LSTM model validation accuracy result trained on augmented AG news dataset with Pegasus, utilizing 30% of the original dataset, vs 30% of the original dataset.

After analyzing all the tables above together with Table 11 it is obvious that when the dataset is too small (10%) the improvement in accuracies is not too significant (4%). The gap between models increases to 13% when 20% of the dataset is taken, and then this difference drops to 3% when 30% of the datasets is considered.

Next the same dataset was augmented with T5 paraphraser, which was able to work with longer sequences, hence, there was no need to drop any samples. For variety it was decided to illustrate results of 30%, 40%, and 50% of the dataset.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.67	0.88	0.76	0.91	0.77	0.83
1	0.87	0.91	0.89	0.85	0.95	0.9
2	0.82	0.71	0.76	0.85	0.66	0.74
3	0.86	0.67	0.75	0.71	0.88	0.79
Accuracy	0.79			0.82		
Macro average	0.81	0.79	0.79	0.83	0.82	0.82
Weighted average	0.81	0.79	0.79	0.83	0.82	0.82

Table 12: Bi-LSTM-CNN model validation accuracy result trained on augmented AG news dataset with T5, utilizing 30% of the original dataset, vs 30% of the original dataset.

After careful analysis of all results from Ag news augmented with T5 data, it again showed that data augmentation is not as affective when the data is too small (30% in this case, in the case with Pegasus it was 20%) as it is when the dataset is bigger (40 and 50%). Tables above still shows improvement in measurement metrics, while the same superiority can be observed in validation and training loss and accuracy results shown in Appendix, Figure 20 and Figure 21.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.66	0.89	0.76	0.66	0.89	0.76
1	0.88	0.87	0.87	0.94	0.85	0.89
2	0.71	0.8	0.75	0.82	0.78	0.8
3	0.89	0.47	0.62	0.85	0.68	0.75
Accuracy	0.76			0.8		
Macro average	0.78	0.76	0.75	0.82	0.8	0.8
Weighted average	0.78	0.76	0.75	0.82	0.8	0.8

Table 13: Bi-LSTM-CNN model validation accuracy result trained on augmented AG news dataset with T5, utilizing 40% of the original dataset, vs 40% of the original dataset.

After augmentation of 40% of AG news dataset (45792 samples), the tables above clearly showcase the improvement of 4-5% in average accuracy metrics, while values of some classes an escalation from 0.47 to 0.68, although in some cases the accuracy of the particular classes demonstrated a decline, for example, precision of the 3rd element class from 0.89 to 0.85. But generally, we see the improvement of all the metrics and more balanced distribution of class accuracies. Figure 16, Figure 17 and Figure 18 from Appendix confirm the

superiority of augmented dataset in the validation & training and accuracy & loss plot comparisons.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.95	0.61	0.74	0.93	78	0.85
1	0.93	0.89	0.91	0.86	0.97	0.91
2	0.69	0.83	0.76	0.74	0.87	0.8
3	0.68	0.82	0.74	0.86	0.73	0.79
Accuracy	0.79			0.84		
Macro average	0.81	0.79	0.79	0.85	0.84	0.84
Weighted average	0.81	0.79	0.79	0.85	0.84	0.84

Table 14: Bi-LSTM-CNN model validation accuracy result trained on augmented AG news dataset with T5, utilizing 40% of the original dataset, vs 50% of the original dataset.

50% of AG news, 57216 text samples, were augmented with T5 dataset, and then tested with Bi-LSTM-CNN model. The training process is illustrated in Appendix, in

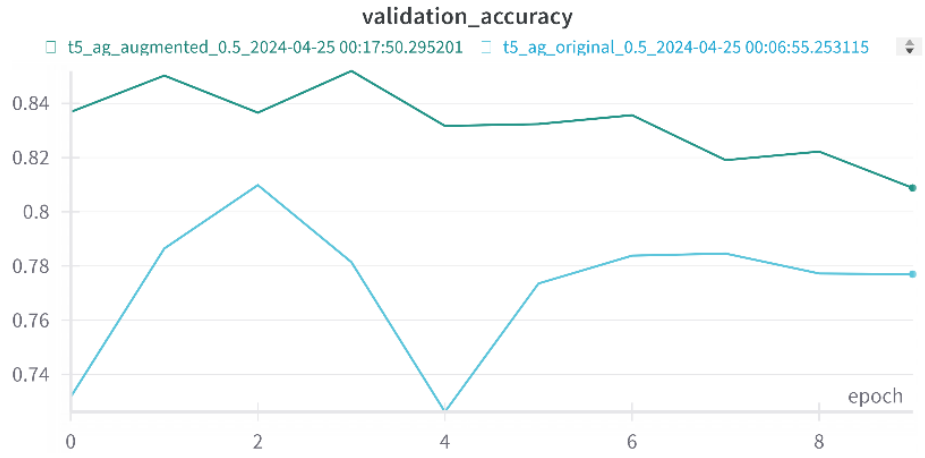


Figure 19 and Figure 20 in which augmented dataset clearly outperformed original dataset. Tabular results from and

Table 14 also demonstrate the upward trend in accuracy measurements, and again classes with the low precision and recall scores illustrate an increase by 0.8-0.10 points which makes accuracy distribution across classes even.

model_name	data type	accuracy	f1_score	precision	recall
MultinomialNB	augmented	0.877	0.877	0.877	0.877
MultinomialNB	original	0.877	0.876	0.876	0.877
LogisticRegression	original	0.864	0.864	0.863	0.864
LogisticRegression	augmented	0.859	0.859	0.859	0.859
SVC	original	0.843	0.842	0.842	0.843
RandomForestClassifier	augmented	0.841	0.841	0.840	0.841
SVC	augmented	0.836	0.836	0.836	0.836
RandomForestClassifier	original	0.834	0.833	0.833	0.834

Table 15: Validation Accuracy results of traditional machine learning models trained on augmented AG news dataset with Pegasus, utilizing 30% of the original dataset, vs 30% of the original dataset.

Although, paraphraser augmentations showed perfect results in the case of Neurol Network, Bi-LSTM-CNN model, when it came to traditional machine learning models, the improvements in the accuracies were not significant, augmentation almost didn't change the model's performance, as it is described in Table 15. The value of some average scores are only increased by 1%, which is an insignificant improvement.

After testing paraphrasers, it was decided to compare the results of backtranslation using mBart50 translator. As was discussed earlier, the text samples were translated from English to Germany language, and backward to get new samples. The tables below show that accuracies of some classes increased in their metrics, while others reduced. It is obvious that the model did not experience a huge improvement, while showing a slight rise in average precision and accuracy, which is very insignificant and cannot be considered as a big improvement.

5.2 OXU

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.67	0.88	0.76	0.9	0.71	0.8
1	0.87	0.91	0.89	0.72	0.98	0.83
2	0.82	0.71	0.76	0.85	0.7	0.77
3	0.86	0.67	0.67	0.77	0.79	0.78
Accuracy	0.79			0.8		
Macro average	0.81	0.79	0.79	0.81	0.8	0.79
Weighted average	0.81	0.79	0.79	0.81	0.8	0.79

Table 16: Bi-LSTM-CNN model validation accuracy result trained on augmented AG news dataset with Mbart50, utilizing 50% of the original dataset, vs 50% of the original dataset.

Initially, both paraphrasers and translation tools were tested on English dataset, AG news, results of which were shown above. Then the same tools were used to augment Azerbaijani dataset scrapped from oxu.az website. Since Oxu dataset perfectly represents a typical small dataset with limited number of samples of particular classes, there was no need for further reduction of the dataset. As was noted before there were approximately 9984 results in Oxu dataset: 7968 in train, 2016 in test dataset. With T5 paraphraser train dataset was augmented to get 15936 augmented samples. Tables below illustrate the results:

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.84	0.69	0.76	0.89	0.71	0.79
1	0.89	0.72	0.8	0.72	0.85	0.78
6	0.55	0.81	0.65	0.73	0.79	0.76
7	0.76	0.88	0.81	0.94	0.91	0.92
8	0.38	0.21	0.27	0.52	0.58	0.55
Accuracy	0.72			0.78		
Macro average	0.68	0.66	0.66	0.76	0.77	0.76
Weighted average	0.75	0.72	0.72	0.79	0.78	0.78

Table 17: LSTM model validation accuracy result trained on augmented OXU news dataset with T5, utilizing 100% of the original dataset, vs original dataset.

The improvement of 0.06 points is observed in the model's average accuracy trained on augmented dataset with T5, while some other average scores like Recall and F1 increased by 0.1 and even 0.11 points (Table 17). The 8th class, which model trained on the original dataset struggled to predict, has a rise of F1 score after augmentation from 0.27 to 0.55, and its Recall elevated from 0.21 to 0.58 points.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.84	0.69	0.76	0.78	0.86	0.82
1	0.89	0.72	0.8	0.83	0.85	0.84
6	0.55	0.81	0.65	0.77	0.73	0.75
7	0.76	0.88	0.81	0.94	0.83	0.88
8	0.38	0.21	0.27	0.44	0.34	0.38
Accuracy	0.72			0.79		
Macro average	0.68	0.66	0.66	0.75	0.72	0.73
Weighted average	0.75	0.72	0.72	0.79	0.79	0.79

Table 18: LSTM model validation accuracy result trained on augmented OXU news dataset with mBart50, utilizing 100% of the original dataset, vs the original dataset.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.84	0.69	0.76	0.86	0.81	0.83
1	0.89	0.72	0.8	0.87	0.8	0.83
6	0.55	0.81	0.65	0.74	0.82	0.78
7	0.76	0.88	0.81	0.93	0.91	0.92
8	0.38	0.21	0.27	0.52	0.7	0.6
Accuracy	0.72			0.81		
Macro average	0.68	0.66	0.66	0.78	0.81	0.79
Weighted average	0.75	0.72	0.72	0.82	0.81	0.82

Table 19: LSTM model validation accuracy result trained on augmented OXU news dataset with combined mBart50 back translator and T5 paraphraser, utilizing 100% of the original dataset, vs the original dataset.

In Table 18 the illustrated results of backtranslation again showcase the improvement of all the average parameters compared to the model trained on original dataset. Since the original training dataset for the T5 augmentation and mBart backtranslation was the same, it is possible to compare the results of those two to find out which augmentation methos is better. However, it is not that obvious as detailed analyses of Table 17 and Table 18 show that while mBart augmentation could increase the average accuracy, average weighted F1 score and Recall higher than T5, the micro average values of Recall and Precision are higher in the case of T5. Although based on the main measurement metrices, average F1 score and accuracy, mBart backtranslation showed better results, both of those models could increase the accuracy of the base model trained on original dataset to a significant value with an improvement factor of 10% in average.

But that is not even our best result. Utilizing the same train dataset did not give us only possibility to compare the results, but also to combine the augmented samples from both techniques and see the improvement in accuracy. Hence, 7968 samples augmented with T5 were added to 7968 samples augmented with mBart and 7968 samples of original dataset to get the total of 23904 samples. The model was again trained with that combined dataset results of which are shown in Table 19, and compared with the results of model trained on original dataset. As it is obvious even better accuracies were obtained as a result. The average accuracy is increased from 0.72 to 0.81, average F1 scores are elevated from 0.66 and 0.72 to 0.82, Recall also experienced a rise from 0.66 and 0.72 to 0.81, and finally the precisions are boosted from 0.68 and 0.75 to 0.78 and 0.82. The problem with imbalanced 8th class is also solved, which is shown by a rise of a class accuracy from 0.27 to 0.6.

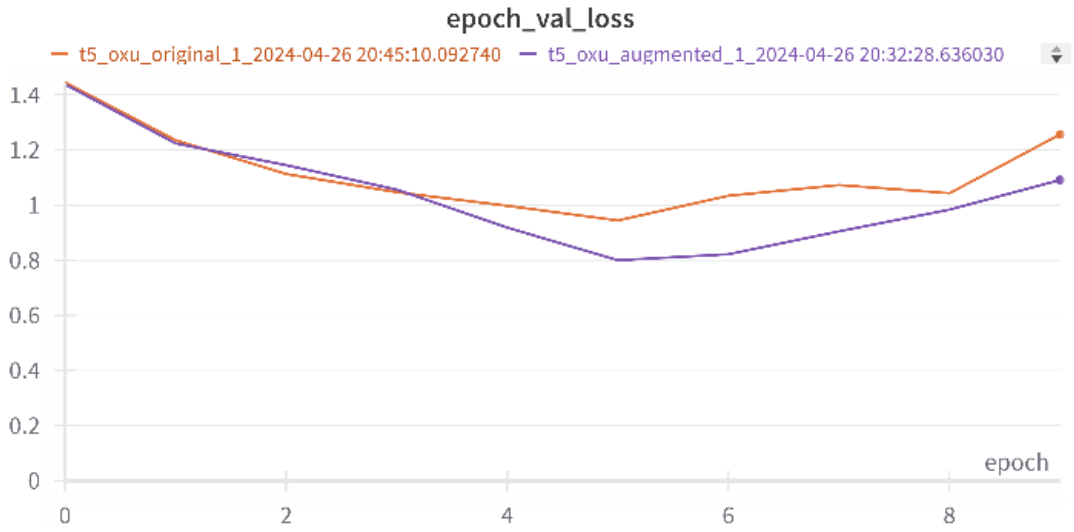


Figure 6: Validation loss Comparison. The graph showcases the results of an LSTM model trained with 100% of OXU, augmented with T5 versus the original version.

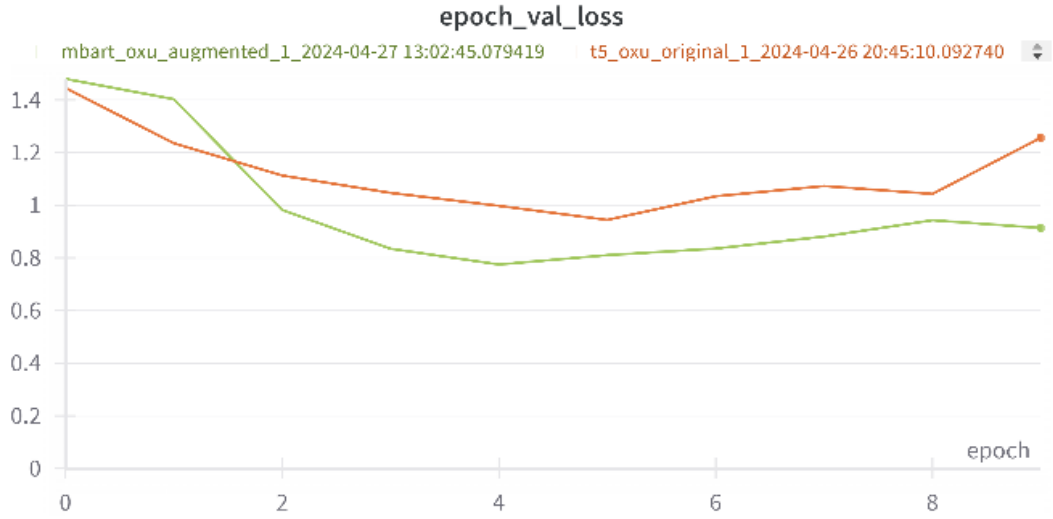


Figure 7: Validation loss Comparison. The graph showcases the results of an LSTM model trained with 100% of OXU, augmented with mBart50 versus the original version.

There can be a lot of questions regarding the number of epochs, model architecture and parameters, how all of these affect the results. Figure 6 and Figure 7 explicitly show that after 4th-5th epoch both models' loss curves:trained on augmented and original datasets, go upward, which means that models have already converged and further improvement in results will not be achieved. In terms of architecture, generally there was a specific pattern and correlation between the models' parameters, architecture, type and the improvement difference after augmentation: if the accuracy of the model trained on original dataset increased after finetuning and model improvement, then the accuracy of the model trained on augmented dataset also increased.

Although, Pegasus paraphraser did not show any significant results, after augmentation the model experienced slight improvement in some accuracy scores as it is illustrated in the tables below:

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.76	0.71	0.73	0.82	0.62	0.71
1	0.55	0.9	0.68	0.78	0.49	0.6
6	0.72	0.44	0.55	0.48	0.76	0.59
7	0.7	0.76	0.73	0.65	0.86	0.74
8	0	0	0	0.22	0.2	0.21
Accuracy	0.66			0.63		
Macro average	0.55	0.56	0.54	0.59	0.59	0.57
Weighted average	0.65	0.66	0.63	0.67	0.63	0.63

Table 20: LSTM model validation accuracy result trained on augmented OXU news dataset with Pegasus, utilizing 100% of the original dataset, vs the original dataset.

Since Pegasus cannot handle long sequences, some of the samples were dropped and in the original train dataset there were 5312 samples, and twice as many in augmented. Although the overall accuracy dropped from 0.66 to 0.63, the accuracy of the imbalanced 8th class increased from 0 to 0.21, and precision elevated from 0.55 and 0.65 to 0.59 and 0.67. But

generally, that cannot be taken as an improvement, that is why this augmentation method cannot be considered as a good technique for this particular Azerbaijani dataset.

With regards to traditional machine learning models, as was observed in the case with AG news, traditional machine learning models did not show any improvement. The table below describes all results for traditional machine learning model:

model_name	data type	accuracy	f1_score	precision	recall
LogisticRegression	original	0.895	0.885	0.898	0.874
LogisticRegression	augmented	0.887	0.877	0.883	0.871
SVC	original	0.885	0.872	0.885	0.861
SVC	augmented	0.875	0.864	0.870	0.859
MultinomialNB	original	0.873	0.855	0.875	0.843
MultinomialNB	augmented	0.859	0.842	0.868	0.826
RandomForestClassifier	augmented	0.877	0.841	0.888	0.817
RandomForestClassifier	original	0.881	0.839	0.899	0.814

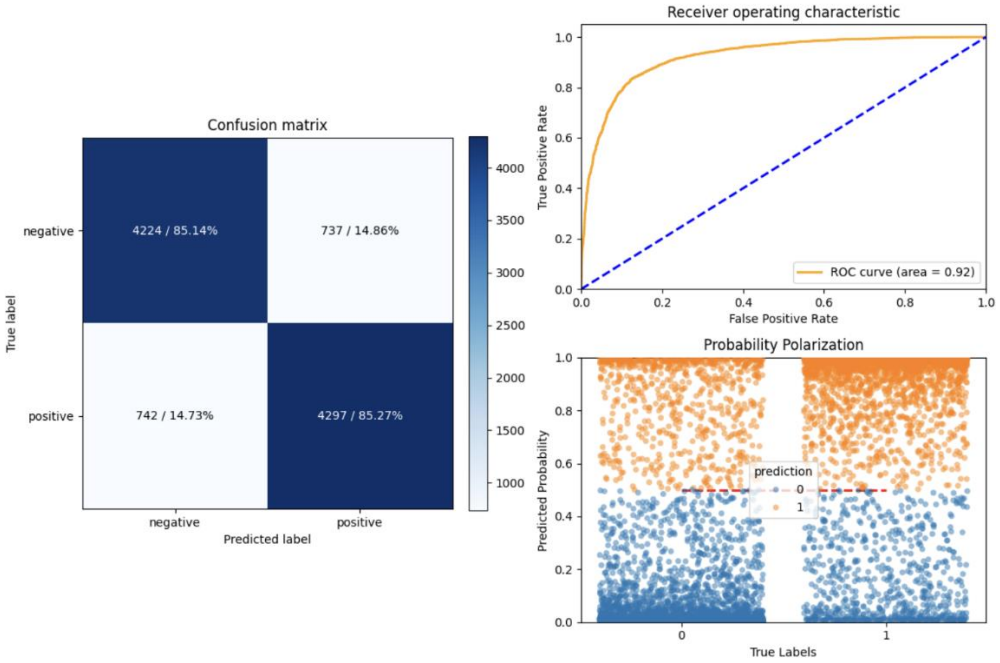
Table 21: Comparison of validation accuracy results of traditional machine learning models trained on augmented OXU dataset with combined mBart50 back translator and T5 paraphraser vs original version.

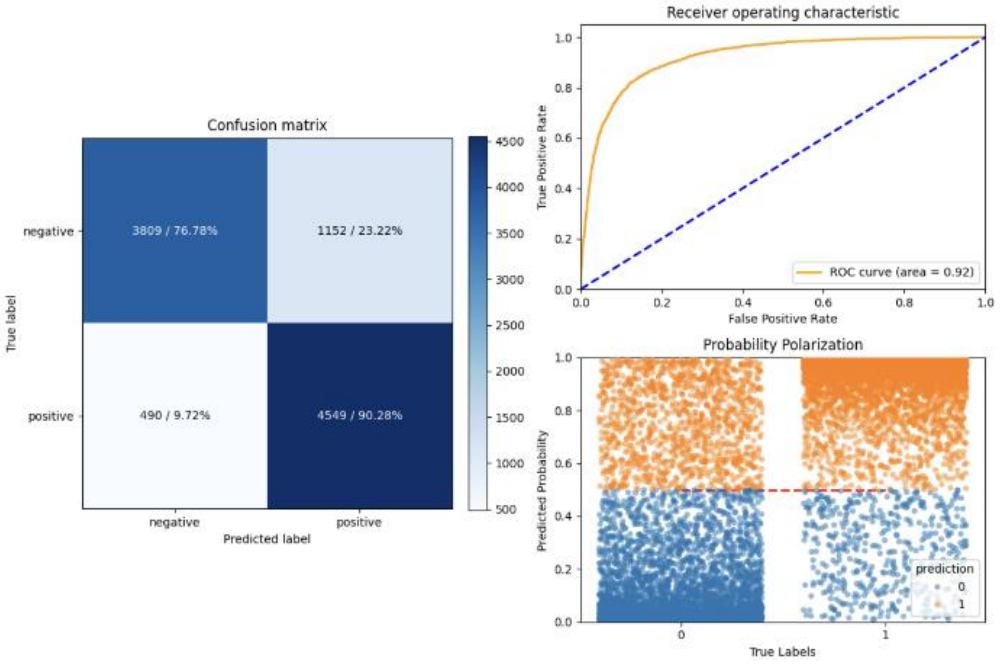
5.3 IMDB

Finally, it is time to show the results of our last dataset which is IMDB data review and relates to binary sentiment analysis subcategory. The same tools have been used to improve the accuracy of results. Since the size of IMDB dataset is not big (40 000 samples in train dataset), it was decided to take only 80% of it to test augmentation with T5 and Pegasus. However, due to model's specifications it was needed to drop some samples from original dataset in case of Pegasus, that is why after taking 80% there were 29385 samples, compared to full 32 000 in the case with T5. That is why the illustrated below results of models trained on original datasets in 2 cases are different (in Pegasus it was trained on 29385 samples, and in case of T5 there were 32000, with regards to augmented versions it is as usually twice as many as in the original versions).

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.89	0.77	0.82	0.85	0.85	0.85
1	0.8	0.9	0.85	0.85	0.85	0.85
Accuracy	0.84			0.85		
Macro average	0.84	0.84	0.83	0.85	0.85	0.85
Weighted average	0.84	0.84	0.83	0.85	0.85	0.85

Table 22: Bi-LSTM-CNN model validation accuracy result trained on augmented IMDB dataset with T5, utilizing 80% of the original dataset, vs the original dataset.





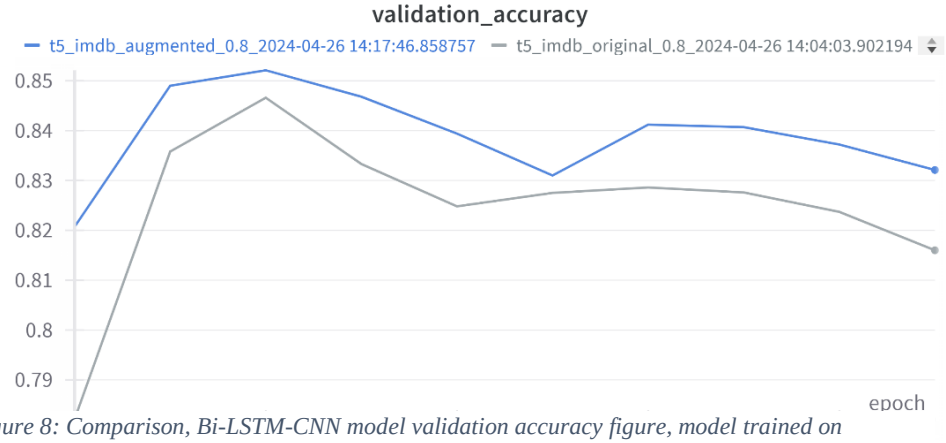
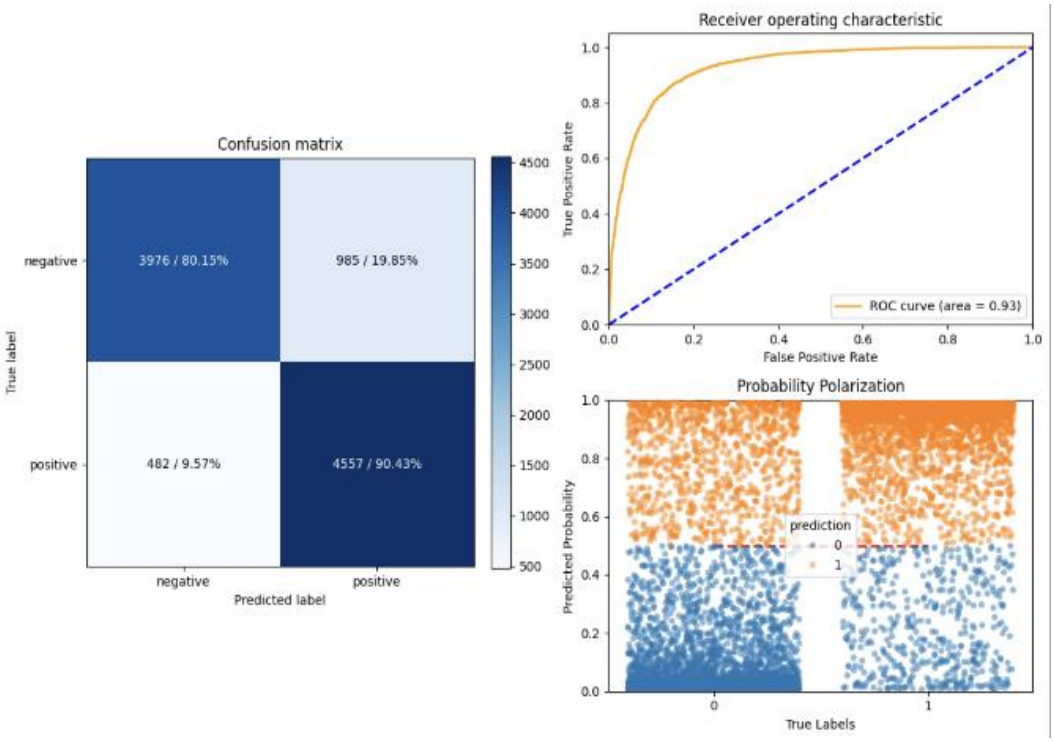


Figure 8: Comparison, Bi-LSTM-CNN model validation accuracy figure, model trained on augmented IMDB dataset with T5, utilizing 80% of the original dataset, vs the original dataset.

The table above illustrates accuracy score improvement of 1% in all accuracy metrics. Initially, 76% of true negative predictions and 90% of true positive predictions were gotten when trained on original dataset, after augmentation with T5 we observed accuracy smoothing across 2 classes and got 85% of accuracy for both of them with an accuracy rise of 1 %. Moreover, the validation accuracy comparison chart shows pure superiority of the augmented dataset.

Next, the same train dataset was augmented with Pegasus paraphraser, and again sight improvement in all accuracy scores have been observed as Table and Figures below depict, on other side augmentation with Pegasus increased the accuracy of the true negative from 79 to 80%, and the accuracy of the true positive from 83 to 90%, meanwhile the average accuracy scores again got increased to from 0.82 to 0.85. After careful analysis of both results, the superiority of Pegasus augmentation over T5 is clearly visible.

	original dataset			augmented dataset		
	Precision	Recall	F1	Precision	Recall	F1
0	0.83	0.8	0.81	0.89	0.8	0.84
1	0.81	0.84	0.82	0.82	0.9	0.86
Accuracy	0.82			0.85		
Macro average	0.82	0.82	0.82	0.86	0.85	0.85
Weighted average	0.82	0.82	0.82	0.86	0.85	0.85



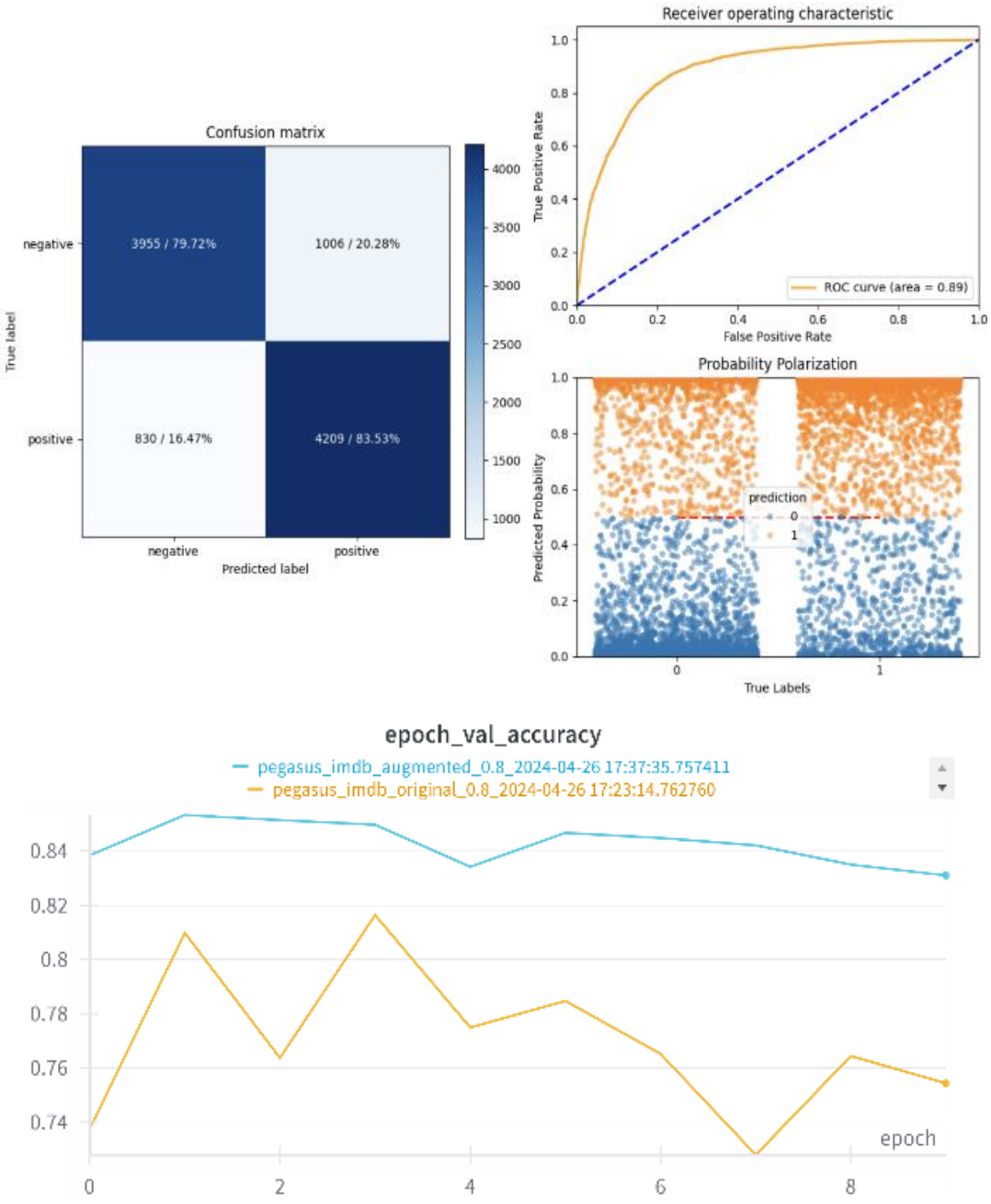


Table 23: Bi-LSTM-CNN model validation accuracy result trained on augmented IMDB dataset with Pegasus, utilizing 80% of the original dataset, vs the original dataset.

As for two other datasets augmentation did not show any significant improvement in the case of traditional machine learning models.

model_name	data type	accuracy	f1_score	precision	recall
LogisticRegression	augmented	0.849	0.848	0.849	0.848
LogisticRegression	original	0.847	0.847	0.848	0.847
RandomForestClassifier	augmented	0.838	0.838	0.838	0.838
MultinomialNB	augmented	0.836	0.836	0.836	0.836
MultinomialNB	original	0.834	0.834	0.835	0.835
SVC	original	0.832	0.832	0.832	0.832
RandomForestClassifier	original	0.830	0.829	0.830	0.829
SVC	augmented	0.821	0.821	0.821	0.821

Table 24: Comparison of validation accuracy results of traditional machine learning models trained on augmented IMDB dataset with Pegasus.

5.4 Intrinsic evaluation

While the results described above are extrinsic, the intrinsic evaluation also has been done. For that Rouge and Bleu scores as well as Bert embedding cosine similarity score have been utilized. The table below showcases the results:

Model	Data	Bleu score	Rouge score	Bert embedding cos similarity
T5	AG News	0.21	0.49	0.94
Pegasus	AG News	0.43	0.65	0.92
T5	IMDB	0.17	0.42	0.94
Pegasus	IMDB	0.30	0.55	0.88
T5	Oxu	0.10	0.22	0.82
Pegasus	Oxu	0.09	0.15	0.70
mBart	Oxu	0.15	0.34	0.84

Figure 9: Intrinsic evaluation.

From the results above it is crystally clear that Peagasus transformer works better for shorter sequences since its value for Ag News which has shorter sampler are much higher in Bert similarity score, while for datasets with longer samples, IMDB and Oxu, it showed lower scores. While T5 generally has higher values, in English datasets in extrinsic evaluation scores Pegasus showed better results. The reason standing behind it is the fact that Bleu and Rouge scores showcase how close the paraphrased samples to their original counterparts, Bert embedding similarity scores illustrate the contextual similarity, and high cosine similarity and low Rouge and Bleu scores of Pegasus mean that it altered the original samples and but saved their semantic details and contextual meaning.

In the case with Azerbaijani dataset, Oxu, we see that its Bert embedding, contextual score is low, which means that in paraphrasing process it changed its contextual meaning a lot, which corrupted the dataset and led to degradation of extrinsic evluation scores – accuracies, F1 scores, Recalls and Precisions. Meanwhile, T5 and mBart could preserve high Bert embedding scores while altering the original samples enough, that is why those methods showcased better accuracy scores in extrinsic evaluation.

6 Summary and Future work

Augmentation type	Dataset	Model	Improvement
Pegasus	IMDB	NN	3%
T5	IMDB	NN	1%
mBart50	IMDB	NN	0%
Pegasus	AG news	NN	13%
T5	AG news	NN	4%
mBart50	AG news	NN	0%
Pegasus	OXU news	NN	-3%
T5	OXU news	NN	8%
mBart50	OXU news	NN	7%
T5 + mBart50	OXU news	NN	13%

Table 25: Experiment results

The Table above summarizes the results of all experiments. It is crystally clear that for binary classification datasets augmentation was not as productive as for dataset containing multiple classes. Hence, experiments with AG news dataset showed 13% and 4% of average rise in different accuracy metrices, although back translation with mBart did not showcase any elevation in accuracy scores. The main reason behind ineffectiveness of the backtranslation is the fact that mBart is not the best translator for English language, and the main reason for choosing it was its translation quality for Azerbaijani language, which has been confirmed with the recorded rise of 7% in accuracy after augmentation of Azerbaijani multiclass dataset, OXU, with backtranslation. While Pegasus decreased the performance of the model, T5 model could increase the accuracy by 0.08 points. Finally, after combining 2 main augmentation techniques even higher accuracy has been reached (13%). While those results obtained for neural network models are impressive, traditional machine learning models did not showcase any improvement.

In terms of future work there are 2 possible ways. The first one is relatively easy and does not require any financial or computational support. The main idea is testing those augmentation ways on attention-based models like Bert Classifier. Moreover, generating models as well as their respective datasets can be also tested. One of such models could be summarizing dataset, samples of which can be divided into sentences and passed to paraphrasers. After which those generative models can be tested to see whether the summarization capabilities of the models are improved.

The second way is difficult and requires financial and computation support since its main idea is developing our own generative paraphraser for Azerbaijani language. The main issue in this project will be shortage of dataset which can be solved by utilizing large language models and paraphrasing tools. Here we have 2 options: use tools like ChatGpt to generate the models from scratch or use same LLMs or translating tools to translate already available dataset with paraphrased sample pairs from Microsoft company. Both of those options require financial support to pay for API credits to utilize those tools. After acquiring the dataset, we can train generative models like Pegasus and T5 from scratch. This way we will ensure that our augmentation model will paraphrase and generate high quality text in Azerbaijani language without need to utilize translation tools.

7 References

1. A. L. Maas and R. E., "Learning Word Vectors for Sentiment Analysis," 2011.
2. A. Y. Atabay Ziyaden, "Text data augmentation and pre-trained Language Model for enhancing text classification of low-resource languages," 2023.
3. X. S. Ernie Chang, "Neural Data-to-Text Generation with LM-based Text Augmentation," n.d.
4. A. P. Gaurav Kolhatkar, "Team Converge at ProbSum 2023: Abstractive Text Summarization of Patient Progress Notes," 2023.
5. E. C. Hugo Queiroz Abonizio, "Toward Text Data Augmentation for Sentiment Analysis," 2022.
6. Y. Z. Jingqing Zhang, "PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization," 2020.
7. K. T. Jugal Kalita, "Abstractive Text Summarization Using the BRIO Training Paradigm," 2023.
8. Y. F. Potsawee Manakul, "CUED at ProbSum 2023: Hierarchical Ensemble of Summarization Models," 2023.
9. F. Retkowski, "The Current State of Summarization," 2023.
10. D. P. Snajder, "Data Augmentation for Neural NLP," n.d.
11. J. Z. Xiang Zhang, "Character-level Convolutional Networks for Text Classification," n.d.
12. K. Zhou and F. Long, "Sentiment Analysis of Text Based on CNN and Bi-directional LSTM Model," 2015.
13. M. Lewis et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," 2020.
14 Z. Jiang et al., "Reinforced Dynamic Reasoning for Conversational Question Answering," 2021.
15. H. Wang et al., "Structured Neural Summarization," 2019.
16. A. See et al., "Get To The Point: Summarization with Pointer-Generator Networks," 2017.
17. C. Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," 2020.
18. J. Devlin et al., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," 2019.
19. Y. Liu et al., "RoBERTa: A Robustly Optimized BERT Pretraining Approach," 2019.
20. N. F. Rajani et al., "Explain Yourself! Leveraging Language models for Commonsense Reasoning," 2019.
21. A. Radford et al., "Improving Language Understanding by Generative Pre-training," 2018.
22. I. Sutskever et al., "Sequence to Sequence Learning with Neural Networks," 2014.
23. D. Bahdanau et al., "Neural Machine Translation by Jointly Learning to Align and Translate," 2015.
24. K. Cho et al., "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," 2014.
25. L. Dong et al., "Unified Language Model Pre-training for Natural Language Understanding and Generation," 2019.
26. M. Zaheer et al., "Big Bird: Transformers for Longer Sequences," 2020.
27. P. J. Liu et al., "Generating Wikipedia by Summarizing Long Sequences," 2018.
28. O. Vinyals et al., "Grammar as a Foreign Language," 2015.

29. T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing," 2018.
30. E. M. Bender et al., "Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data," 2020.
31. S. Merity et al., "Pointer Sentinel Mixture Models," 2016.
32. G. Lample and A. Conneau, "Cross-lingual Language Model Pretraining," 2019.
33. Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2019). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization Andrew L. Maas, R. E. (2011). Learning Word Vectors for Sentiment Analysis.
Atabay Ziyaden, A. Y. (2023). Text data augmentation and pre-trained Language Model for enhancing text classification of low-resource languages.
Ernie Chang, X. S. (n.d.). Neural Data-to-Text Generation with LM-based Text Augmentation.
Gaurav Kolhatkar, A. P. (2023). Team Converge at ProbSum 2023: Abstractive Text Summarization of Patient Progress Notes.
Hugo Queiroz Abonizio, E. C. (2022). Toward Text Data Augmentation for Sentiment Analysis.
Jingqing Zhang, Y. Z. (2020). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization.
Jugal Kalita, K. T. (2023). Abstractive Text Summarization Using the BRIO Training Paradigm.
Potsawee Manakul, Y. F. (2023). CUED at ProbSum 2023: Hierarchical Ensemble of Summarization Models.
Retkowski, F. (2023). The Current State of Summarization.
Snajder, D. P. (n.d.). Data Augmentation for Neural NLP.
Xiang Zhang, J. Z. (n.d.). Character-level Convolutional Networks for Text Classification.
Zhou, K., & Long, F. (2015). Sentiment Analysis of Text Based on CNN and Bi-directional LSTM Model.

8 Appendix

Appendix A: Figures

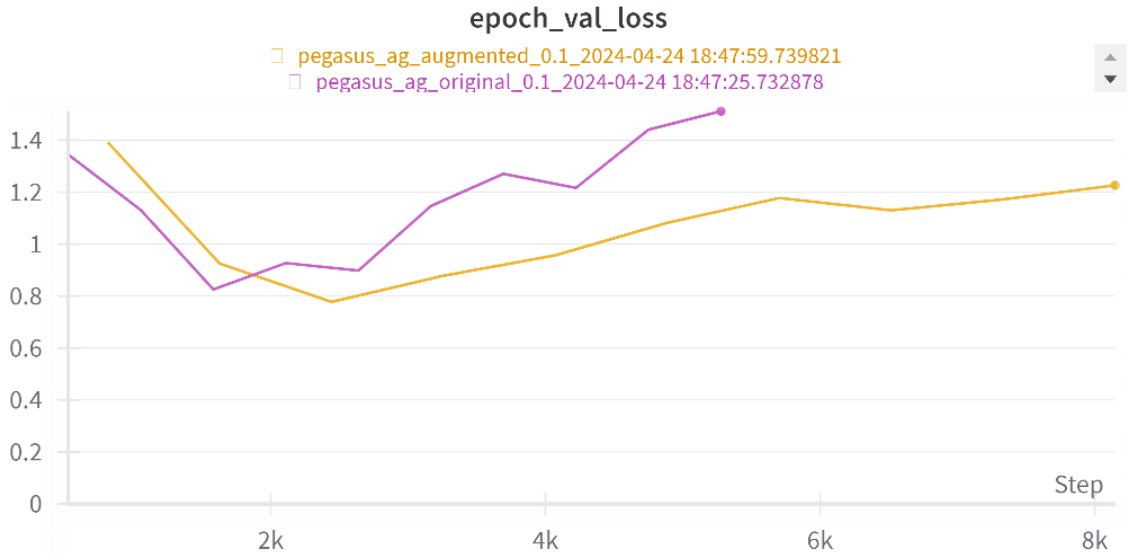


Figure 10: Validation Loss Comparison. The graph showcases the results of an LSTM model trained with 10% of Ag news, augmented with Pegasus versus the original version.

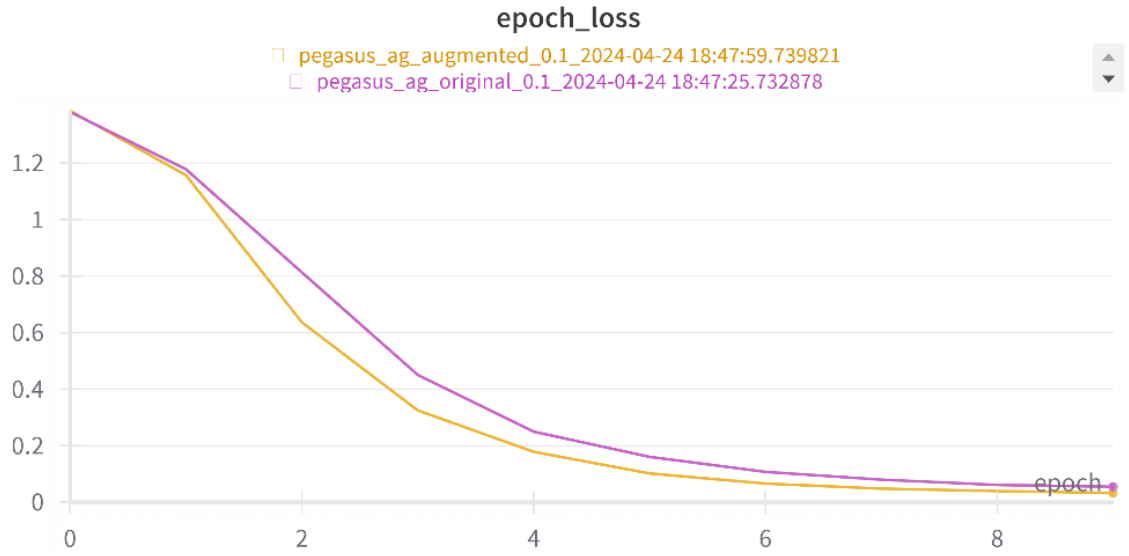


Figure 11: Training Loss Comparison. The graph showcases the results of an LSTM model trained with 10% of Ag news, augmented with Pegasus versus the original version.

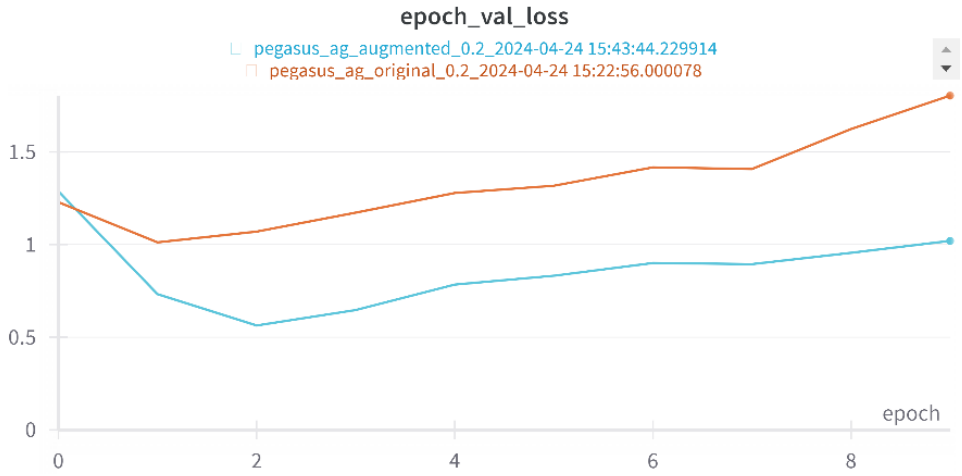


Figure 12: Validation Loss Comparison. The graph showcases the results of an LSTM model trained with 20% of Ag news, augmented with Pegasus versus the original version.

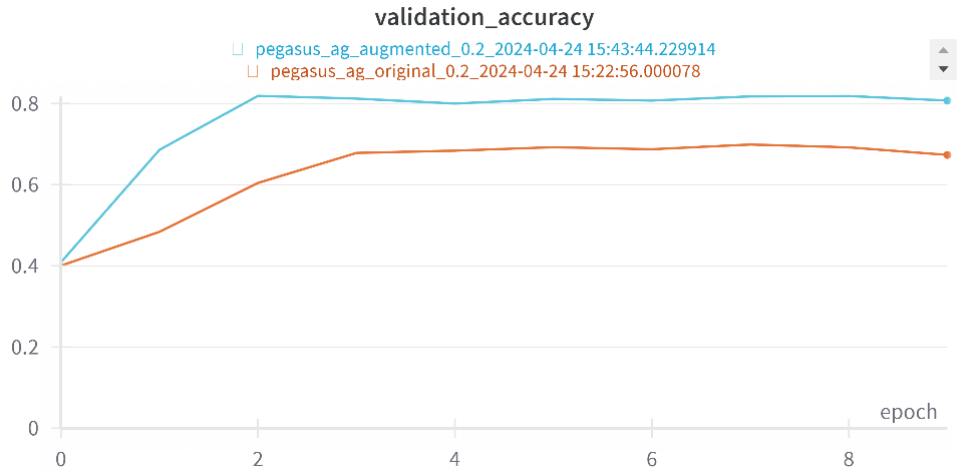


Figure 13: Validation Accuracy Comparison. The graph showcases the results of an LSTM model trained with 20% of Ag news, augmented with Pegasus versus the original version.

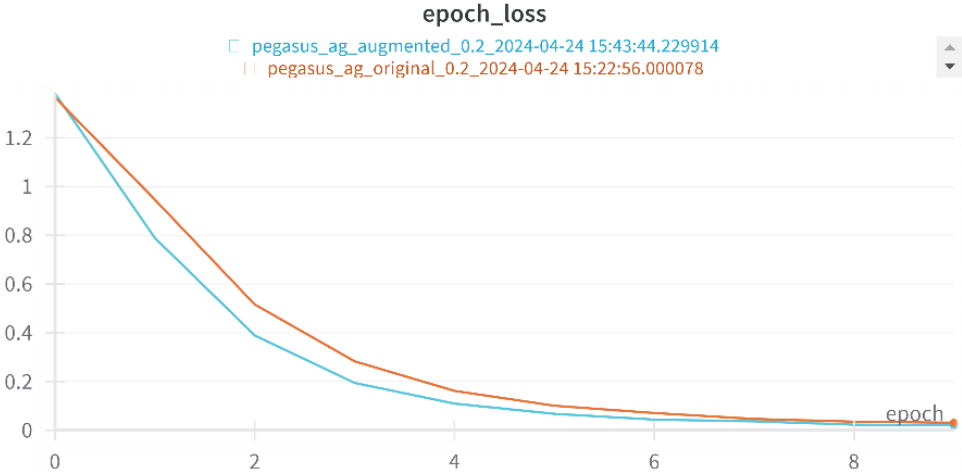


Figure 14: Training Loss Comparison. The graph showcases the results of an LSTM model trained with 20% of Ag news, augmented with Pegasus versus the original version.

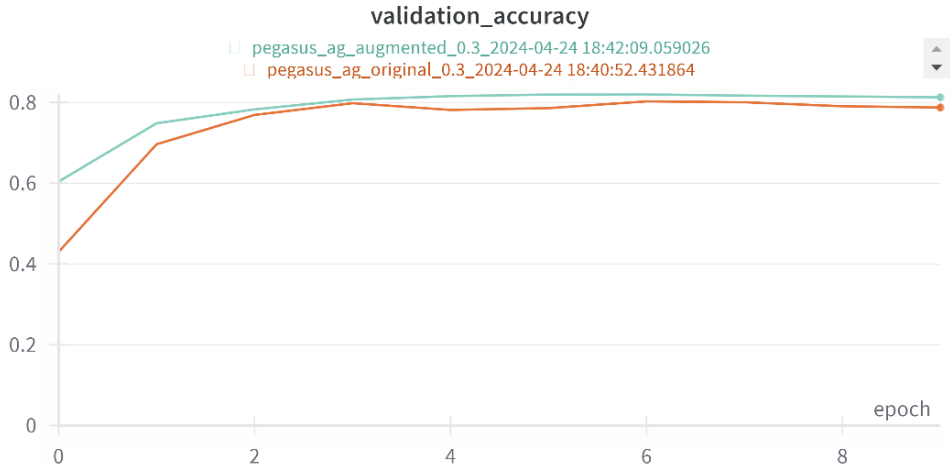


Figure 15: Validation Accuracy Comparison. The graph showcases the results of an LSTM model trained with 20% of Ag news, augmented with Pegasus versus the original version.

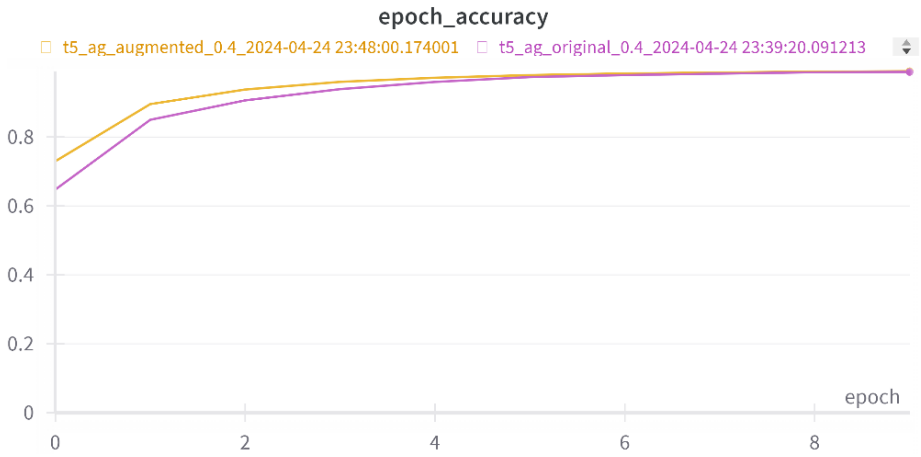


Figure 16: Training Accuracy Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 40% of Ag news, augmented with T5 versus the original version.

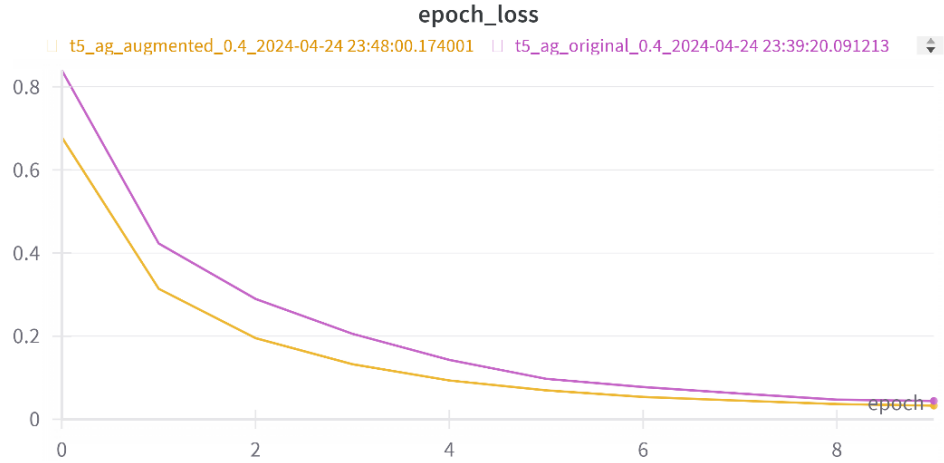


Figure 17: Training Loss Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 40% of Ag news, augmented with T5 versus the original version.

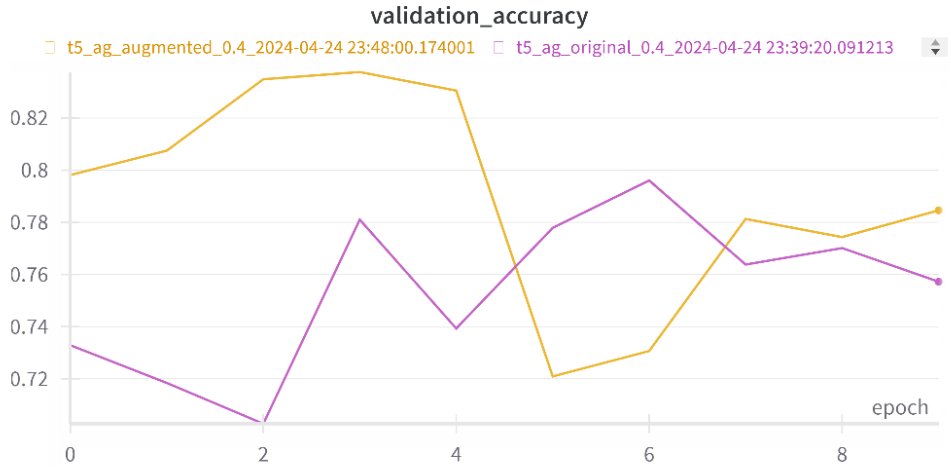


Figure 18: Validation Accuracy Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 40% of Ag news, augmented with T5 versus the original version.

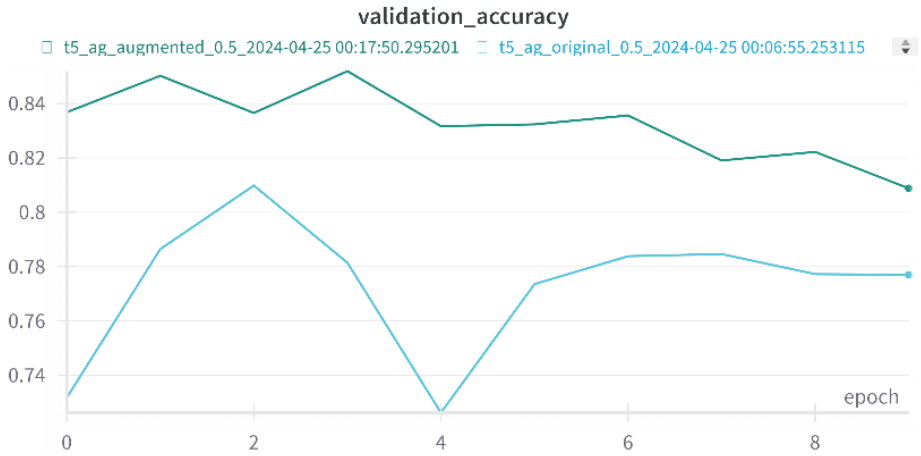


Figure 19: Validation Accuracy Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 50% of Ag news, augmented with T5 versus the original version.

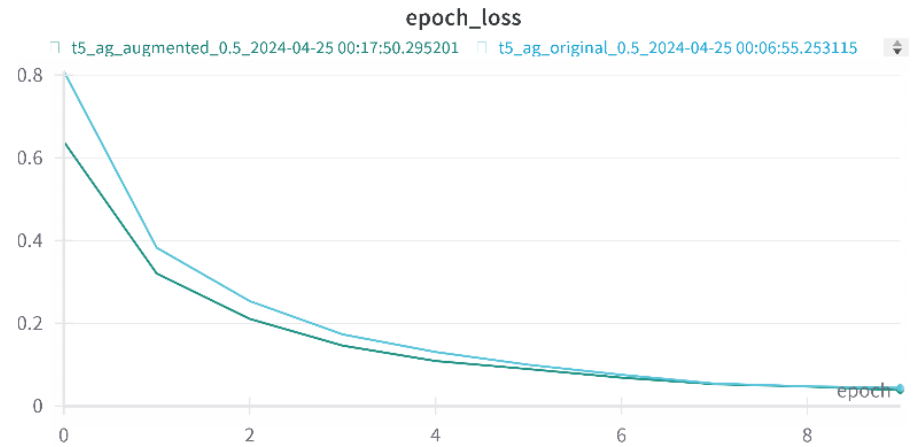


Figure 20: Training Loss Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 50% of Ag news, augmented with T5 versus the original version.

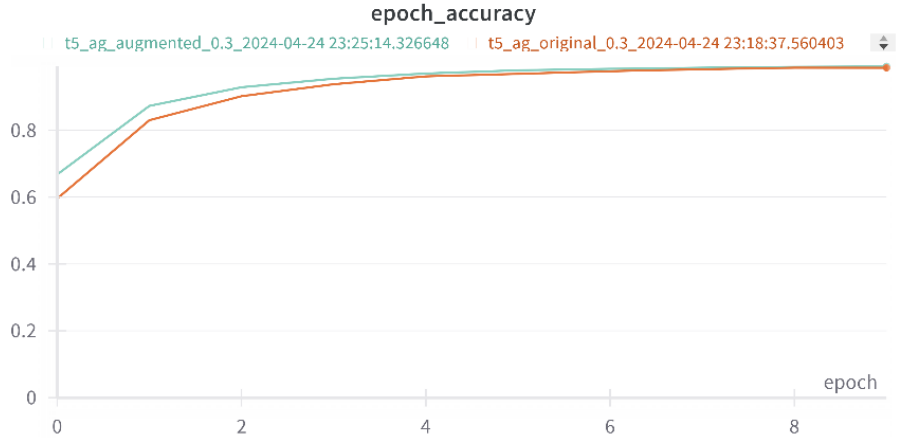


Figure 21: Training Accuracy Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 30% of Ag news, augmented with T5 versus the original version.

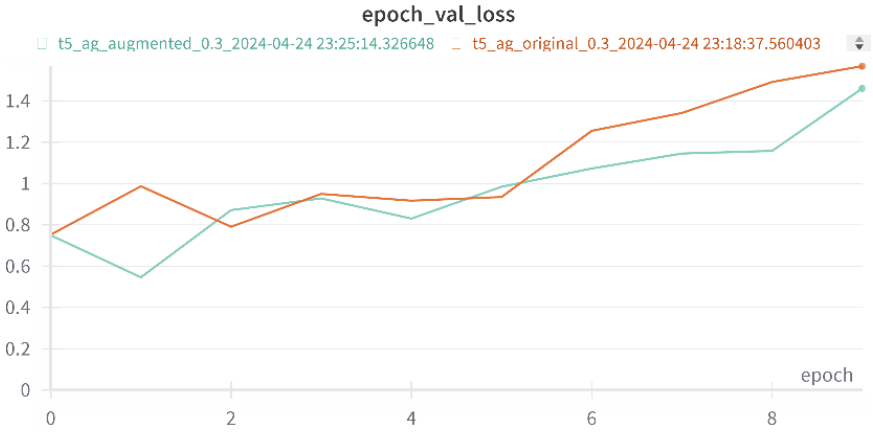


Figure 22: Validation loss Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 30% of Ag news, augmented with T5 versus the original version.

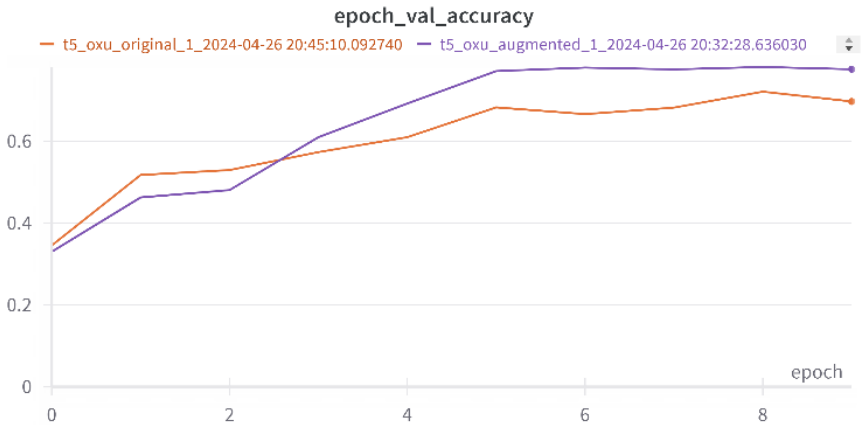


Figure 23: Validation accuracy Comparison. The graph showcases the results of a Bi-LSTM-CNN model trained with 100% of OXU, augmented with T5 versus the original Azerbaijani version.

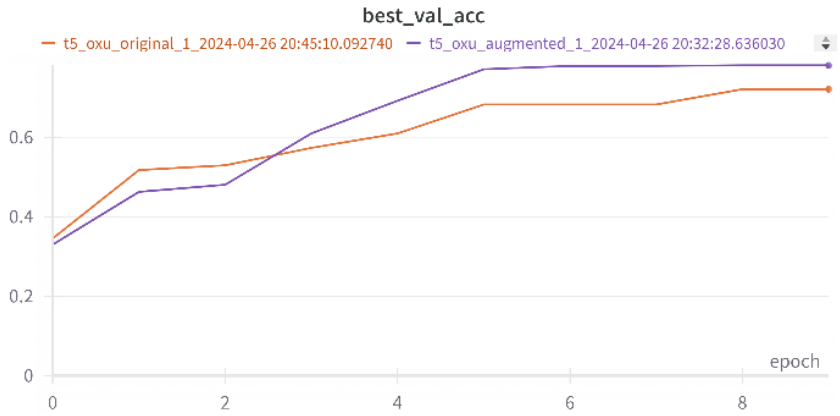


Figure 24: Best Validation accuracy Comparison. The graph showcases the results of an LSTM model trained with 100% of OXU, augmented with T5 versus the original Azerbaijani datasets.

Appendix B: Code

```
#_________________mBart Paraphraser_____________________________

mbart = MBartForConditionalGeneration.from_pretrained("facebook/mbart-large-50-many-to-many-mmt")
tokenizer_az = MBart50TokenizerFast.from_pretrained("facebook/mbart-large-50-many-to-many-mmt", src_lang = "az_AZ")
tokenizer_en = MBart50TokenizerFast.from_pretrained("facebook/mbart-large-50-many-to-many-mmt", src_lang = "en_XX")
mbart.to_bettertransformer()
mbart.to('cuda')
mbart = DataParallel(mbart)
mbart = mbart.module
mbart.eval()


def az_en(input_seq):
    model_inputs = tokenizer_az(input_seq, max_length = 2048, padding=True, truncation=True, return_tensors="pt").to('cuda')



    generated_tokens = mbart.generate(
        **model_inputs,
        forced_bos_token_id=tokenizer_az.lang_code_to_id["en_XX"]
    )

    return tokenizer_az.batch_decode(generated_tokens.to('cuda'), skip_special_tokens=True)


def en_az(input_seq):
    model_inputs = tokenizer_en(input_seq, max_length = 2048, padding=True, truncation=True, return_tensors="pt").to('cuda')



    generated_tokens = mbart.generate(
        **model_inputs,
        forced_bos_token_id=tokenizer_en.lang_code_to_id["az_AZ"]
    )

    return tokenizer_en.batch_decode(generated_tokens.to('cuda'), skip_special_tokens=True)

#_____________________Pegasus_____________________________

model_name = 'tuner007/pegasus_paraphrase'
torch_device = 'cuda' if torch.cuda.is_available() else 'cpu'
tokenizer_pegasus = PegasusTokenizer.from_pretrained(model_name)
model_pegasus = PegasusForConditionalGeneration.from_pretrained(model_name).to(torch_device)

model_pegasus.eval()
model_pegasus.to_bettertransformer()

model_pegasus = model_pegasus.cuda()
model_pegasus = DataParallel(model_pegasus)
model_pegasus = model_pegasus.module



def get_response_pegasus(input_text: str, num_return_sequences = 1, num_beams = 10) -> List[str]:
    batch = tokenizer_pegasus(input_text, max_length=60, truncation = True, padding = 'longest', return_tensors="pt").to(torch_device)
    print(batch['input_ids'].shape)
    with torch.no_grad():
        translated = model_pegasus.generate(**batch, max_length=60, num_beams=num_beams, num_return_sequences=num_return_sequences, temperature=1.5)
    tgt_text = tokenizer_pegasus.batch_decode(translated, skip_special_tokens = True)
    return tgt_text

splitter = SentenceSplitter(language='en')

def count_words(sentence):
    return len(sentence.split())

def paraphrase_text_by_sentence(input_text: str, num_return_sequences: int, num_beams: int) -> str:
    sentences = splitter.split(input_text)
    paraphrased_sentences = get_response_pegasus(sentences, num_return_sequences, num_beams)
    paraphrased_text = ' '.join(paraphrased_sentences)
    return paraphrased_text
```

```
#____________________T5_____________________________

tokenizer_t5 = AutoTokenizer.from_pretrained("humarin/chatgpt_paraphraser_on_T5_base")
model_t5 = AutoModelForSeq2SeqLM.from_pretrained("humarin/chatgpt_paraphraser_on_T5_base")
model_t5.eval()
model_t5.to_bettertransformer()

model_t5 = model_t5.cuda()
model_t5 = DataParallel(model_t5)
model_t5 = model_t5.module

def divide_into_pairs(text):
    sentences = splitter.split(text)
    pairs = []


    for i in range(0, len(sentences), 2):
        if i + 1 < len(sentences):
            pair = sentences[i] + " " + sentences[i + 1]
        else:
            pair = sentences[i]
        pairs.append(pair)

    return pairs

def paraphrase(
    input_texts,
    num_beams=5,
    num_beam_groups=5,
    num_return_sequences=1,
    repetition_penalty=10.0,
    diversity_penalty=3.0,
    no_repeat_ngram_size=2,
    temperature=0.7,
    max_length=512
):

    input_texts = ['paraphrase: ' + input_text for input_text in input_texts]
    input_ids = tokenizer_t5(
        input_texts,
        return_tensors="pt", padding="longest",
        max_length=max_length,
        truncation=True,
    ).input_ids.to(device)

    outputs = model_t5.generate(
        input_ids, temperature=temperature, repetition_penalty=repetition_penalty,
        num_return_sequences=num_return_sequences, no_repeat_ngram_size=no_repeat_ngram_size,
        num_beams=num_beams, num_beam_groups=num_beam_groups,
        max_length=max_length, diversity_penalty=diversity_penalty
    )

    res = tokenizer_t5.batch_decode(outputs, skip_special_tokens=True)


    res = " ".join(res)

    return res
```

```
# INTRINSIC EVALUATION

model = BertModel.from_pretrained('bert-base-uncased')
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

model_az = BertModel.from_pretrained('bert-base-multilingual-cased')
tokenizer_az = BertTokenizer.from_pretrained('bert-base-multilingual-cased')

def get_embedding_az(text):
    inputs = tokenizer_az(text, return_tensors="pt", padding=True, truncation=True)
    with torch.no_grad():
        outputs = model_az(**inputs)
    return outputs.last_hidden_state.mean(dim=1).detach().numpy()

def get_embedding(text):
    inputs = tokenizer(text, return_tensors="pt", padding=True, truncation=True)
    with torch.no_grad():
        outputs = model(**inputs)
    return outputs.last_hidden_state.mean(dim=1).detach().numpy()


def get_bleu_score(original, paraphrase):
    reference = original.split()
    candidate = paraphrase.split()
    return sentence_bleu([reference], candidate)

def get_rouge_score(original, paraphrase):
    rouge = Rouge()
    scores = rouge.get_scores(paraphrase, original)
    return scores[0]['rouge-l']['f']
```

```
#LSTM CLASSIFIER

class LSTMClassifier(nn.Module):
    def __init__(self, vocab_size, embedding_dim, hidden_dim, output_dim):
        super(LSTMClassifier, self).__init__()
        self.embedding = nn.Embedding(vocab_size, embedding_dim)
        self.lstm = nn.LSTM(embedding_dim, hidden_dim, dropout = 0.1, batch_first=True)
        self.fc = nn.Linear(hidden_dim, output_dim)

    def forward(self, x):
        # Embedding layer
        x = self.embedding(x)
        # LSTM layer
        x, _ = self.lstm(x)
        # Select the last hidden state
        x = x[:, -1, :]
        # Fully connected layer
        x = self.fc(x)
        return x
```

```python
# SIMPLE MODEL WITH LINEAR EMBEDDING


class CNN_BiLSTM(nn.Module):

    def __init__(self, vocab_size):
        super(CNN_BiLSTM, self).__init__()
        self.hidden_dim = 256
        self.num_layers = 4
        V = vocab_size
        D = 128
        C = 1
        self.C = C
        Ci = 1
        Co = 256
        Ks = [3, 4]
        self.embed = nn.Embedding(V, D, padding_idx=0)
        # CNN
        self.convs1 = nn.ModuleList([nn.Conv2d(Ci, Co, (K, D), padding=(K//2, 0), stride=1) for K in Ks])
        # BiLSTM
        self.bilstm = nn.LSTM(D, self.hidden_dim, num_layers=self.num_layers, dropout=0.1, bidirectional=True, bias=True)
        # linear
        L = len(Ks) * Co + self.hidden_dim * 2
        self.hidden2label1 = nn.Linear(L, L // 2)
        self.hidden2label2 = nn.Linear(L // 2, C)
        # dropout
        self.dropout = nn.Dropout(0.1)

        self.sg = nn.Sigmoid()
    def forward(self, x):
        embed = self.embed(x)
        # CNN
        cnn_x = embed
        # cnn_x = torch.transpose(cnn_x, 1, 2)
        cnn_x = cnn_x.unsqueeze(1)
        cnn_x = [conv(cnn_x).squeeze(3) for conv in self.convs1] # [(N,Co,W), ...]*len(Ks)
        cnn_x = [F.tanh(F.max_pool1d(i, i.size(2)).squeeze(2)) for i in cnn_x] # [(N,Co), ...]*len(Ks)

        cnn_x = torch.cat(cnn_x, 1)
        cnn_x = self.dropout(cnn_x)
        # BiLSTM
        bilstm_x = embed.view(len(x), embed.size(1), -1)
        bilstm_out, _ = self.bilstm(bilstm_x)
        #bilstm_out = torch.transpose(bilstm_out, 0, 1)

        bilstm_out = torch.transpose(bilstm_out, 1, 2)
        bilstm_out = F.max_pool1d(bilstm_out, bilstm_out.size(2)).squeeze(2)
        bilstm_out = F.tanh(bilstm_out)
        # CNN and BiLSTM CAT
        cnn_x = torch.transpose(cnn_x, 0, 1)
        bilstm_out = torch.transpose(bilstm_out, 0, 1)
        cnn_bilstm_out = torch.cat((cnn_x, bilstm_out), 0)
        cnn_bilstm_out = torch.transpose(cnn_bilstm_out, 0, 1)
        # linear
        cnn_bilstm_out = self.hidden2label1(F.tanh(cnn_bilstm_out))
        output = self.hidden2label2(F.tanh(cnn_bilstm_out))
        return self.sg(output)
```

```python
# DATA PROCESSING AND LOADING
def sample(df, n):
    sampled_df = df.sample(frac=n, random_state=42).reset_index(drop = True)
    original_df = sampled_df[['original', 'sentiment']]
    original_df = original_df.rename(columns={'original': 'review'})
    augmented_samples = sampled_df[['review', 'sentiment']]
    augmented_df = pd.concat([original_df, augmented_samples], ignore_index=True).sample(frac=1, random_state=42).reset_index(drop = True)
    return original_df, augmented_df

def into_loader_vocab(df_true, text):
    df = df_true.copy()
    df['tokenized'] = df[text].apply(lambda x: x.split())
    all_words = [word for tokens in df['tokenized'] for word in tokens]
    word_counts = Counter(all_words)
    sorted_vocab = sorted(word_counts, key=word_counts.get, reverse=True)
    vocab = {word: i + 2 for i, word in enumerate(sorted_vocab)}
    vocab['<PAD>'] = 0
    vocab['<UNK>'] = 1
    return vocab

def into_loader(df_true, text, label, vocab):
    df = df_true.copy()
    df['tokenized'] = df[text].apply(lambda x: x.split())
    df['indexed'] = df['tokenized'].apply(lambda x: [vocab.get(word, vocab['<UNK>']) for word in x])
    max_length = max(len(x) for x in df['indexed'])
    df['padded'] = list(pad_sequences(df['indexed'], maxlen=max_length, padding='post', value=vocab['<PAD>']))
    class TextDataset(Dataset):
        def __init__(self, text_sequences, labels):
            self.text_sequences = text_sequences
            self.labels = labels
        def __len__(self):
            return len(self.text_sequences)
        def __getitem__(self, idx):
            return torch.tensor(self.text_sequences[idx], dtype=torch.long), torch.tensor(self.labels[idx], dtype=torch.long)
    X = df.loc[:, 'padded'].tolist()
    y = df.loc[:, label].tolist()
    dataset = TextDataset(X, y)
    dataloader = DataLoader(dataset, batch_size=32)
    return dataloader
```