Ministry of Science and Higher Education of the Republic of Kazakhstan

SDU University

Zhunis Karimov

Empowering Kazakh Text Generation: Developing a Meaningful Natural Language Processing Model for Kazakh Language

THESIS

Presented in Partial Fulfilment for the

Degree of Master of Technical Science in Computer Science

(degree code: 7M06102)

Department of Computer Science

Faculty of Engineering and Natural Sciences

Supervisor: **Meraryslan Meraliyev**

Kaskelen, June 2024

SDU University
Faculty of Engineering and Natural Sciences
Department of Computer Science

Dean of Faculty of Engineering and Natural Sciences

Assistant Professor, PhD Akhmedov Ramis

_____

« 04 » 06 2024

Topic of the thesis:
Empowering Kazakh Text Generation: Developing a Meaningful Natural Language Processing Model for Kazakh Language

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science", SDU University

Head of Department Zhanar Mukash

Academic Supervisor Meraryslan Meraliyev

Master student Zhunis Karimov

Kaskelen, 2024

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Zhunis Karimov

June 2024

Acknowledgements

I want to thank my supervisor professor Merarslan Meraliyev for the inspiration, trust and valuable suggestions and support which I got during the development of the master thesis and the research generally.

Next my thanks to the Google Colab for providing the best work experience as computational resources which was necessary for conducting my research and the experiments over the models covered in this research. Glad to Google Colab this research took its place as it is.

Finally, I am grateful to the dataset contributors used in this research for being generous and making it open source.

Dedication

I dedicate conducted research to the Suleyman Demirel University, especially to the Computer Science department for their trust, support, encouragement. They have been inspiring me throughout my master degree journey and their belief in my abilities was a great source of endless motivation.

Abstract

Nowadays, NLP is getting more and more popular due to the vast development of the computing units such as GPUs and CPUs. Especially, recent GPU models allow researchers to process analyse and process terabytes of text and audio data in a short period of time. The problem is that the data is not unlimited and has its own threshold value, therefore data augmentation is one of the techniques of data synthesis, the development of which can produce good effect on the Science of Kazakhstan, to be more precise on the kazakh language processing research area.

This research aimed to develop and compare state-of-the-art uni-directional and bi-directional LSTM architectures in order to determine which one is more effective in a task of generating synthetic, kazakh language based text data. Generated synthetic data can be used in a variuos NLP tasks such as training NLP models, content generation, text classification and domain adaptation. By generating synthetic data scientist can overcome data limitations, model overfitting and open new abilities in adjusting and increasing model performance.

Architectures used in this research are Uni-directional and Bi-directional LSTMs. After many training session and model parameters tuning the study shows that Bi-directional LSTM has its own pros and cons over the traditional Uni-directional LSTM. Bi-directional neural network learns well and understands the relationship between words better than Uni-directional one, but as expected the training process takes much more time as it extracts the information from the input text in both directions. The dataset used in this study consist of 10 000 unique rows of text taken from the kazakh language based Wikipedia area where each row has median as 80 word sequence. It worth to mention that in order to understand the model relationship between the words study proposes the usage of FastText pre-trained kazakh word vectors as an embedding layer for the both models. This layer turns the input words into vectors where similiar words have close vectors.

Finally, the research takes a step towards generating synthetic data with the help of which future works can avoid the problems of data limitations, on the other hand researchers can save the time by ignoring ineffective model selection problem and considering this study's models comparison.

Аннотация

В настоящее время NLP становится все более популярным благодаря широкому развитию вычислительных мощностей, как графические процессоры. В частности, последние модели графических процессоров позволяют исследователям обрабатывать, анализировать и перерабатывать терабайты текстовых и аудиоданных за короткий промежуток времени. Проблема в том, что данные не безграничны и имеют порог, поэтому увеличение объема данных является одним из методов синтеза данных, развитие которого может оказать хорошее влияние на науку Казахстана, а точнее на область исследований в области обработки казахского языка.

Это исследование было направлено на разработку и сравнение современных однонаправленных и двунаправленных архитектур LSTM, чтобы определить, какая из них более эффективна в задаче генерации синтетических текстовых данных на основе казахского языка. Сгенерированные синтетические данные могут быть использованы в различных задачах NLP, таких как обучение моделей NLP, генерация контента, классификация текста и адаптация к предметной области. Создавая синтетические данные, ученые могут преодолеть ограничение данных, переоснастить модель и открыть новые возможности для настройки и повышения производительности модели.

В этом исследовании использовались однонаправленные и двунаправленные LSTM. После многочисленных тренировок и настройки параметров модели исследование показало, что двунаправленный LSTM имеет свои плюсы и минусы по сравнению с традиционным однонаправленным LSTM. Двунаправленная нейронная сеть хорошо обучается и лучше понимает взаимосвязь между словами, чем однонаправленная, но, как и ожидалось, процесс обучения занимает гораздо больше времени, поскольку она извлекает информацию из входного текста в обоих направлениях. Набор данных, использованный в этом исследовании, состоит из 10 000 уникальных строк текста, взятых из Википедии на казахском языке, где каждая строка имеет медиану в виде последовательности из 80 слов. Стоит отметить, что для понимания взаимосвязи между словами исследование предлагает использовать предварительно подготовленные векторы казахских слов FastText. Этот слой преобразует входные слова в векторы, в которых похожие слова имеют близкие векторы.

Наконец, исследование делает шаг к получению синтетических данных, с помощью которых будущие исследователи могут избежать проблем, связанных с ограниченностью данных, сэкономить время, игнорируя проблему неэффективного выбора модели и рассматривая сравнение моделей в этом исследовании.

Аңдатпа

Қазіргі уақытта NLP технологиясы графикалық процессорлар сияқты есептеу қондырғыларының кең дамуына байланысты танымал бола бастады. Әсіресе, GPU-ның соңғы үлгілері зерттеушілерге қысқа уақыт ішінде мәтіндік және аудио деректердің терабайттарын талдауға және өңдеуге мүмкіндік береді. Мәселе мынада, қазақ тіліндегі информация шекті, сондықтан оны синтездеу арқылы көбейту әдістерінің бірі болып табылады, оның дамуы Қазақстан ғылымына жақсы әсер етуі мүмкін.

Бұл зерттеу қазақ тіліне негізделген синтетикалық мәтіндік деректерді генерациялау тапсырмасында қайсысы тиімдірек екенін анықтау үшін заманауи бір бағытты және екі бағытты LSTM архитектураларын әзірлеуге және салыстыруға бағытталған. Жасалған синтетикалық деректерді NLP модельдерін оқыту, мазмұнды генерациялау, мәтінді жіктеу және доменді бейімдеу сияқты әртүрлі NLP тапсырмаларында пайдалануға болады. Синтетикалық текстті генерациялау арқылы ғалым деректер шектеулерін еңсере алады, модельдің шамадан тыс сәйкестігін анықтай алады және модель өнімділігін реттеу мен оның пайдасын арттыруда жаңа мүмкіндіктерді аша алады.

Бұл зерттеуде қолданылатын архитектуралар бір бағытты және екі бағытты LSTM болып табылады. Көптеген жаттығулар мен модель параметрлерін реттегеннен кейін зерттеу екі бағытты LSTM дәстүрлі бір бағытты LSTM-ге қарағанда өзінің артықшылықтары мен кемшіліктері бар екенін көрсетеді. Екі бағытты нейрондық желі бір бағытты желіге қарағанда сөздер арасындағы байланысты жақсы меңгереді және жақсы түсінеді, бірақ күткендей, оқу процесі әлдеқайда ұзағырақ уақытты алады, өйткені ол екі бағытта да енгізілген мәтіннен ақпаратты алады. Осы зерттеуде пайдаланылған деректер жинағы қазақ тіліндегі Уикипедия аймағынан алынған 10 000 бірегей мәтін жолынан тұрады, мұнда әрбір жолдың медианасы 80 сөзден тұрады. Айта кету керек, сөздер арасындағы модельдік байланысты түсіну үшін зерттеу екі модель үшін де ендіру қабаты ретінде FastText алдын ала оқытылған қазақ сөз векторларын пайдалануды ұсынады.

Ақырында, зерттеу синтетикалық деректерді жасауға қадам жасайды, оның көмегімен болашақ жұмыстар деректерді шектеу мәселелерін болдырмайды, екінші жағынан, зерттеушілер модельдерді таңдаудың тиімсіз мәселесін елемеу және осы зерттеудің модельдерін салыстыруды қарастыру арқылы уақытты үнемдей алады.

Abbreviations

LSTM - Long Short-Term Memory
NLP - Natural Language Processing
GPU - Graphical Processing Unit
CPU - Central Processing Unit
TPU - Tensor Processing Units
RAM - Random Access Memory
RNN - Recurrent Neural Network
GPT - Generative Pre-trained Transformer
BLEU - BiLingual Evaluation Understudy
ROUGE - Recall-Oriented Understudy for Gisting Evaluation
METEOR - Metric for Evaluation of Translation with Explicit Ordering

Table of Contents

Chapter 1

Introduction

1.1 Background and Motivation

Recently, NLP has become one of the popular direction in deep learning area, glad to chat bots, artificial avatars and assistants in the commercial industry or text to speech tasks. NLP is a science which studies the abilities of computers to manipulate, generate and understand natural languages. The latest researches provides the machine abilities to understand and generate meaningful text data or even voice speech as an output. Based on this we can divide NLP area into two parts where natural language understanding is a branch which is responsible for semantic analysis and understanding the language whereas natural language generation is based on generating human readable, meaningful text.

Generally, there are several key points why NLP has become one of the fast pace developing area in artificial intelligence branch. One of those key points is advancements in the data processing algorithms or in another words neural architectures developed by the industry giants and the scientists. For example, the recent techniques called transformers, particularly models like BERT or GPTs have made a revolution the observed field by allowing to make more accurate and sophisticated language understanding and generation compared to the traditional state of the art techniques such as recurrent neural networks and its sub types.

From another hand, it is worth to mention the exponential in computational power units such as GPUs, CPUs or even TPUs. That advancements allow to to train large language, complex models that can process vast amounts of data quickly and efficiently. That changes in the industry lead to another point which causes the interest and scientists motivation in proceeding the studies called open source libraries or datasets.

The development of open-source NLP libraries and frameworks has increased and made an impact into field by democratizing the access to the latest technologies, allowing a wide range of developers and researchers to contribute and participate in the development of the new technologies and appearance of open source datasets.

Those open source technologies and datasets enables the growth of technology applications in real world scenarios such as chat bots, virtual assistants, translation services, sentiment analysis and etc. These application scenarios shows us the value

and practical application of NLP, driving further interest and investment in the field.

It is significant to make a contribution in the study of languages with the low level of observation which is kazakh language. Such researches make a vast contribution to the development of NLP for particular languages as they often have distinctive syntax, morphology, phonology and lexicon as kazakh language is. Syntax is a group of rules which defines the structure of the sentence in a language and structures the words in an order to form the meaning. For example, in the sentence "I go to school" we can see the structure subject-verb and object, this sentence structure is a common for many languages in the world. Morphology studies the structure of words and how they form. Typically, as a structure we propose the usage of prefixes, suffixes and etc. Phonology refers to the study of sounds and how they formed in a particular language. Finally, lexicon is a vocabulary or set of words in a language.

Kazakh language is a rich in every section of language formation mentioned above. It is one of the richest languages in the world as the lexicon or in another words vocabulary of the kazakh language contains approximately 2.5 million words. Language is characterized by the addition of suffixes and endings as most of the world languages, but the uniqueness is that prefixes are not used as they performed by suffixes and post positions.

Those distinctive features make kazakh language difficult to learn, understand and analyse, as shown by the number of studies conducted over this language in the computer science area. That is the main point of providing the research in this direction.

1.2 Objectives

The purpose of the study is to contribute to the development of the science of Kazakhstan in the terms of analysing and generating the meaningful kazakh language based text data. Research expects achieving this goal by comparing the performance of the constructed two types of LSTM [1] models which are Uni-directional and Bi-directional LSTMs. The training process took it place on the Google Colab Pro [2] platform using TPU as a processing unit with the 334.6GB RAM memory with the same values for such parameters like learning rate, epoch number, batch size, sequence length and etc.

Models trained over data consisting of 10 000 unique text rows, where each row at average contains 80 words sequence, totally passing the sequence consisting of 800 000 word to the models. The data before being fed into models has passed the preprocessing and cleaning steps. In general, at the time research being conducted there is a lack of digital kazakh language based text data and the number of researches which provides text generation methods are at a low level, which in turn is the motivation for the conducted research work.

Despite the fact that currently state-of-the-art GPT [3] architecture outperforms LSTM and RNN architectures, the study objective is to see the results and provide the performance of the LSTMs in the task of kazakh text generation. Primary reason of choosing the LSTM architecture is that the most of the research

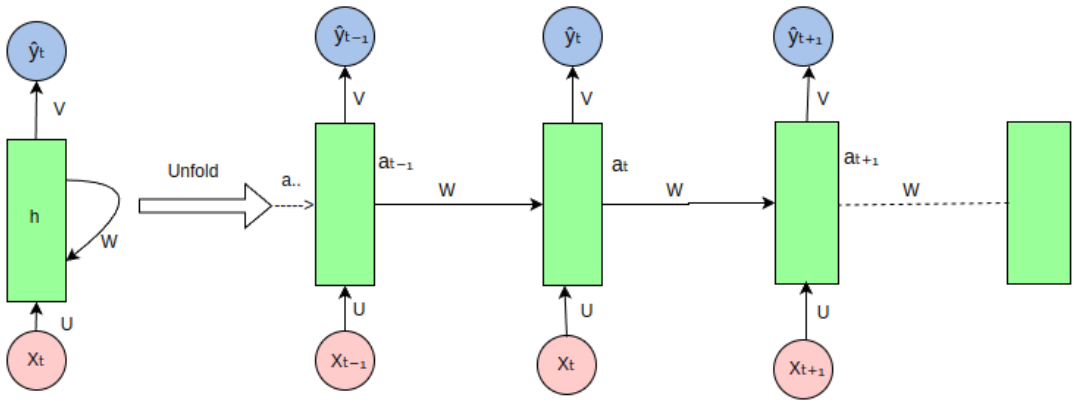

Figure 1.1 - Illustration of RNN architecture

Source: https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef

is in this direction of text generation task are using GPT and RNN architectures ignoring the LSTMs.

Finally, by comparing the output in terms of generated kazakh language based text of Uni-directional and Bi-directional LSTMs [4], study tries to identify the output difference and quality of both text generation models. From the other hand research demonstrates for kazakh language text generation task which architecture, model parameters, preprocessing techniques and machine metrics consider to use for the future works.

1.3 Problem definition

Without a doubt NLP has become one of the fast-pace developing area in the field of artificial intelligence and achieved great results in text generation, speech to text, text to speech, voice synthesis areas, but the fact is that mostly the progress is seen only for widely spoken and used languages such as english or russian, leaving less-resourced languages the shadows of progress such as kazakh language. There are several reason why is this happening. Primary reason is an access to the large database of digital data in terms of text, audio and etc. for those widely spoken languages. The plenty of data lead to the progress and quantity of the researches connected with the NLP field. In the case of less-resourced languages there is a lack of digital data, as a result there is a low level of researches, lack of language corpus and absence of pre trained models for processing and analysing the language data.

In our case, those results has the opposite effect on developing kazakh language based technologies such as chat bots, virtual assistants and more applications related to the kazakh spoken society. Which leads to the gap in the development of technology and the degradation of society. This research attempts to fill the progress gap and stimulate growth of the studies in kazakh language processing area. By developing text generation models for kazakh language, the study contribute to the developing of technologies for less-resourced language.

Such kind of studies make a vast contribution to the development of low re-

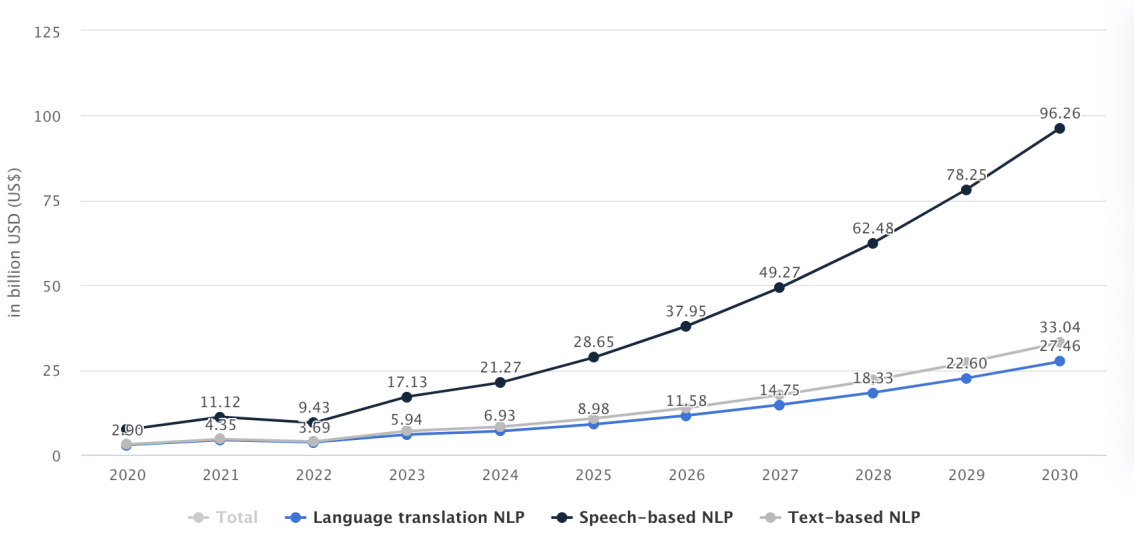

Figure 1.2 - The market size of the Natural Language Processing

Source:
https://www.statista.com/outlook/tmo/artificial-intelligence/natural-language-processing/worldwide#market-size

sourced languages as kazakh is in the field of natural language processing by producing adapted language models and developing the datasets for future works where an important factor regarding the datasets is the quality and the structure of the collected data.

1.4 Research questions

This section supposed to demonstrate the specific research questions addressed to kazakh text generation using uni-directional and bi-directional LSTM architectures. The research questions below reveal the purpose and the meaning of the study:

1. What is the difference in terms of performance and the quality of the generated text measured by the human evaluation between unidirectional LSTM and bidirectional LSTM models?
2. How does the size of training dataset affect the quality of the generated text for both models?
3. What are the limitations of uni-directional and bi-directional LSTM architectures for kazakh language based text generation?
4. How does FastText as an embedding layer and model parameters affect the performance of LSTM models?
5. Do the observed architecture shows better results in terms of text generation compared to the state of the art GPTs for kazakh language?

The research questions provided above can be considered as a roadmap for our research of generating meaningful kazakh text using LSTM architectures. By answering each questions we will dive deepear and find the pros and cons, potential

improvements of the uni-directional and bi-directional models.

Throughout this process, research aims to make a contribution in the developing the language model, sharing the experience and providing an inspiration for future works related to the kazakh language based text generation techniques.

Chapter 2

Literature Review

2.1 Overview

This section provides description of text generation foundations, analysis of the previous works and finally proposes a consensus by providing the context of the study, showing the gaps for further researches and informing the progress of the current work.

The first steps toward to mimic human intelligence using machine were held in 1950 by Alan Turing using the tool called Turing Test [5]. This test involves evaluator person who interacts both with another human and machine using text based communication. The aim was to convince the evaluator that he was communicating with another person, although in fact it was machine, if so the test is passed.

Talking about the term AI it was first introduced in Dartmouth Conference which took its place in 1956. The community in a head of John McCarthy, Marvin Minsky, Allen Newell, and Herbert Simon theoretically proposed and reasoned a machine intelligence which can mimic the human intelligence.

Years later, in 1970s and 1980s there have appeared expert systems. That is the programs that mimic decision making ability by using set of rules and knowledge base. They produce the decision depending on the input without performing some mathematical operations.

The term neural network appeared in the 1980s - 1990s where the researchers Geoffrey Hinton, Yann LeCun and Yoshua Bengio provided a significant contribution to this field by developing the backpropagation algorithm and introducing improved neural network architectures.

The start of deep learning era took its place in late 2000s due to the progress in computing units, data availability and size. Years later leading to the deep learning resurgence in 2010.

Text generation as a subset of the NLP area is fast developing research branch. It takes the start from the early researches in the computational linguistics which were focused on the developing algorithms for text generation based on pre-defined grammatical rules and linguistic patterns. Those algorithms are the basis for the developments which took their place later in the field of statistical modeling, machine and deep learning methods for text generation task.

Statistical modeling is an text generation approach which uses probabilistic models for estimation of the specific word sequence appearance in a give context. It uses the methods such as n-grams, hidden Markov models and probabilistic context-free grammars.

Machine learning approach considers using supervised and unsupervised learning algorithms. Where supervised learning is based on the training the model over the labeled text data in order to learn patterns, but unsupervised learning supposed to find hidden relationship among the text data.

Deep learning based approach proposes the neural networks usage with the famous architectures like RNN, LSTM, GPT which have made a drastical changes in the context of text generation by demonstrating impressive results in the form of human-like text by learning hierarchical representations of language and context.

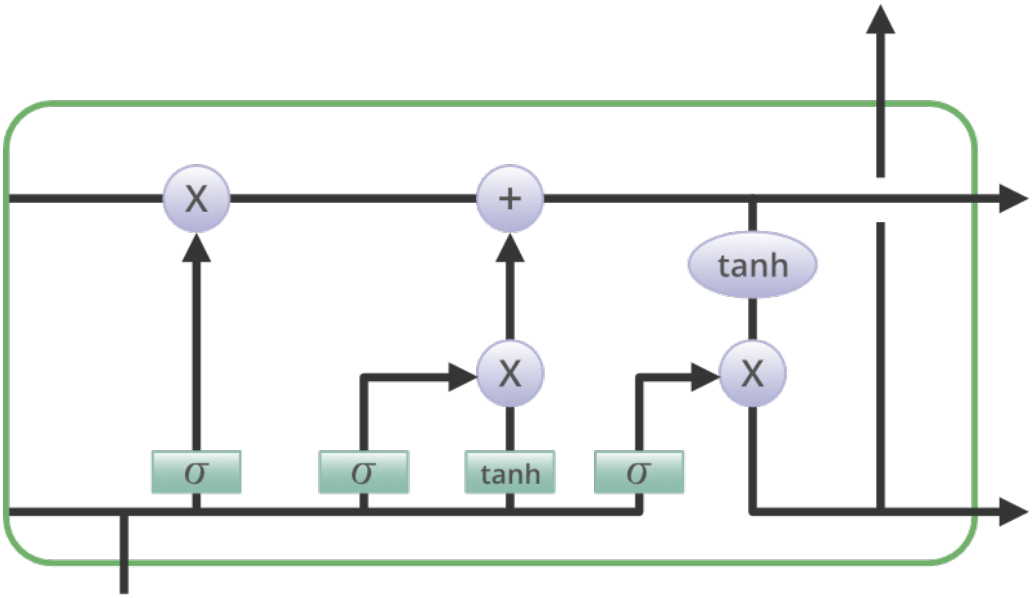

Figure 2.1 - Illustration of Uni-directional LSTM architecture

Source:
https://www.geeksforgeeks.org/understanding-of-lstm-networks/

In this research we propose the usage of uni-directional and bi-directional LSTM architecture which were used for both of the models accordingly. The LSTM architecture visually demonstrated in the figure 2.2. This model is a type of RNN architecture which was designed to overcome the problem of vanishing gradients which usually happens on a long sequence data. In the case of text generation vanishing can lead to very small gradient value, therefore it will update weights inefficient causing slow training process. LSTM solve this problem by initializing special state cells where each of them contains several components. The components are:

1. Cell State: in the figure from left to right top straigth input is Cell state. It represents the entire cell memory and can be dynamicly updated during the training process by removing or adding information depending on the current state and input data.

2. Hidden State: in the figure from left to right bottom to top input is a hidden state. It represent the cells output and passes the information to the next time step. This state is computed based on the input, the previous hidden state and the current state.
3. Gates: in the figure from left to right they can be observed as a sigmoid activation function in 3 places. First one is forget gate, second one is input gate and the last one is output gate. Those 3 gates decides whether to forget information, which information to accept and return as an output.

Generally LSTM is capable of selectively updating and spreading the information over the time, capturing long-range dependencies. As LSTM has those abilities it makes sense to use it for a wide range of tasks involving kazakh text generation task.

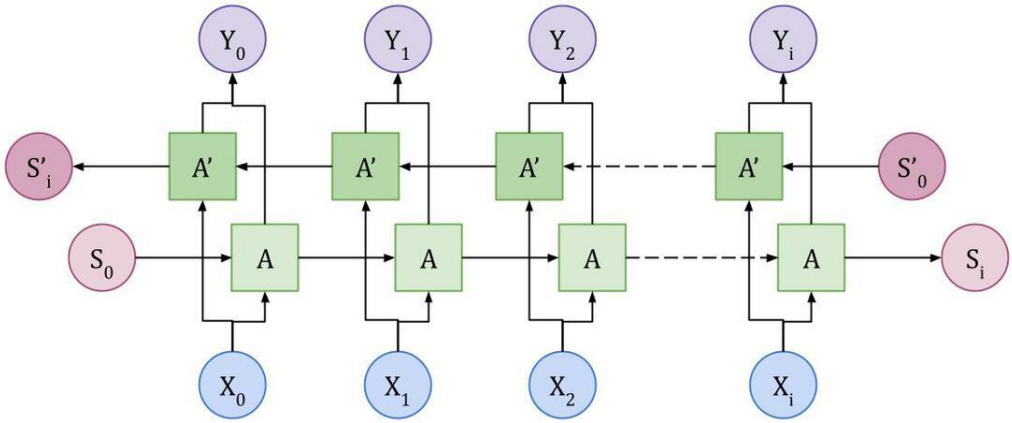

Figure 2.2 - Illustration of Bi-directional LSTM architecture

Source: https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/

The key difference between uni-directional and bi-directional LSTM is that last one captures the information from sequential data in both directions while uni-directional captures information only from past to future. Also the bi-directional LSTM uses two LSTM layers where first one is used to process the information for forward direction and the second one is used to process the information backward. Usually, bi-directional LSTM is used in a tasks where the contexts from the both past and future are important, this allows them to capture more contextual information rather than uni-directional ones.

By diving into those foundational concepts, this section provides comprehensive understanding of the basic principles which are lying in the basis of this research. In the next sections we review the previous studies related to the research topic, their methodologies and finally reaching the consensus of this chapter.

2.2 Previous work

Previous researches in the domain of kazakh language processing have used various methods and techniques starting from creating a model for text to speech tasks, sentiment analysis, text generation and others. But we will review only ones

related to our research, particularly text generation models. There are we review recent key findings and methodologies from previous works.

The study [6] titled "Generative Pre-Trained Transformer for Kazakh Text Generation Tasks" proposes an objective of evaluation text generation models adapted for kazakh language. Author uses a transformer based neural network for the research.

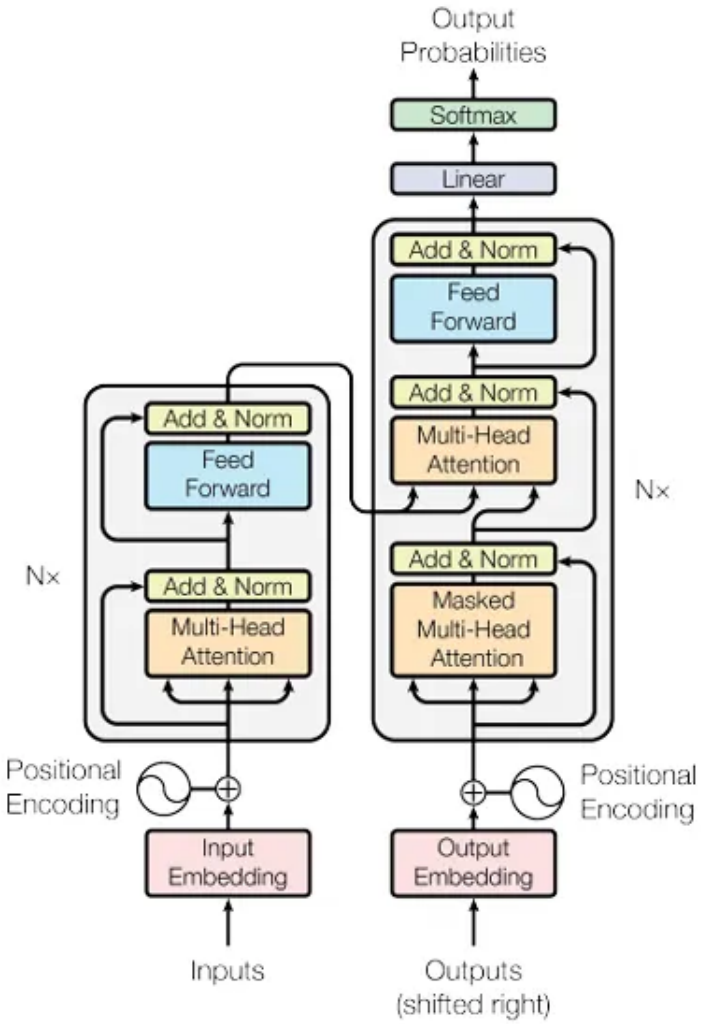

Figure 2.3 - Illustration of transformer architecture

Source: https://medium.com/@tsaiabhi.cool/explaining-gpt-3-architecture-and-working-d0219c79202c

Their approach is to train large language model for the kazakh language using substantial text corpus. After the training process author fine-tuned the model using specialized question answering dataset. At the end proposed pre-trained large language model goes under the evaluation step where the process achieved with the help of conversation response generation task.

Findings of this research shows that it is difficult to achieve good results in text

generation task for languages with low resource due to the lask of data in order to properly train despite the fact that study used the state-of-the-art solution. Even so fine-tuned large language model shows promising results. More precisly, the BLEU [7] score shows result as a 8.5 percent which means that generated text some how is similiar to the referenced one.

The two datasets were used in this research. The first one is a large dataset taken from the different domains in order to train the model. And the second one is collected for the question-answering task which were used for the evaluation purposes.

The second research [2] called "The Task of Generating Text Based on a Semantic Approach for a Low-Resource Kazakh Language" proposes a method for text generation task where they choose semantic approach, using machine learning techniques.

The methodology consists of semantic analysis over the input text including word, stop word and symbol counts. After the TF-IDF [8] metric is used to identify the semantically important parts of the given text. Finally the input text which was annotated based on the provided semantic analyse and passed forward to the model in order to generate text data.

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{2.2.1}$$

In the equation above t is the word being evaluated, d is the document containing the word t, D is a collection of documents. TF(t,d) is a frequence of word t in document d. IDF(t, D) is a frequence of word t among D collection of documents.

TF-IDF statistical metric used to measure the significance of a word in a document in compare of a collection of documents. This metric consists of two parts. TF which stands for term frequency and inverse document frequency IDF.

TF metric measures how often the word appears in a document. Depending on the frequency of the work it shows it importance in document context. IDF has the similiar measure except that it shows the frequence of a word among collection of documents. The least frequency word appear among documents, the high level of importance it has.

In this research, TF-IDF metric is used to identify semantically important words in the kazakh language training dataset.

As the for model the state-of-the-art GPT-3 [9] architecture was chosen for text generation task. Finally the study shows that using semantic analysis gives significant opportunities in the kazakh text generation task.

Talking about the GPT-3 architecture, it was developed by OpenAI, and as it name stays the model is based on transformer architecture. This model was trained over the vast amount of the data and demonstrated remarkable success in terms of generating human like text data.

In the context of the research the model is used to generate kazakh language text.

2.3 Consensus

Provided both researches makes a huge contribution to the development of the text generation task for the kazakh language. The only thing that connects this studies is that they use transformer architectures for their research, but the applications are different in term of the adaptation of the pre-trained large language models and how do they use the attention mechanism in order to get the high performance results and meaningful generated text.

Findings

The paper "Generative Pre-Trained Transformer for Kazakh Text Generation Tasks" proposed a the usage of large transformer-based language model which was trained over kazakh language dataset. Finally was fine-tuned using question answering dataset. The study showed promising result with the BLEU score achieving an 8.5 percent, meaning that generated and referenced data has some similiarities.

The paper "The Task of Generating Text Based on a Semantic Approach for a Low-Resource Kazakh Language" this research address the problem of kazakh text generation using semantic analysis approach. Where before feeding the training data into the transformer model, they provided semantic analysis with the annotation over the training data.

Common

Both studies addressed the problem of kazakh language based text generation, highlighting the importance of creating the kazakh text generation models. Also they share the same concept of using transformer based architectures. While the first paper was aimed to use pre-trained large language model and tune it according to the kazakh language, the second one proposes the usage of semantic analyses by extracting the necessary information and annotating the training data.

Conclusion

In conclusion, observed researches makes a valuable contribution into text generation branch for the kazakh language. Those studies show the effectiveness of different methods and algorithms in text generation task. As mentioned above, one emphasizes the usage of pre-trained large transformer-based model, while last one shows the significance of semantic analysis of training data in order to improve the quality of the generated data.

Together that studies prove that text generation is not only the use of a certain template, architecture or dataset. But provides different approaches of achieving text generation task for kazak language.

Chapter 3

Methodology

3.1 Dataset

The initial dataset [10] used in this research contains 300 000 unique rows of text where each row has an average length of sequence equal to 88 words. The dataset was formed by scrapping the kazakh language based Wikipedia dump. It represent the raw data containing symbols, special characters and numbers. For the research reason 10 000 random text rows were chosen as a subset and preprocessed using techniques such as lowercasing, deleting special characters, newlines, symbols and numbers. Finally each row was split into the separate sequence.

3.2 Architecture

The study proposes the usage of uni-directional and bi-directional LSTM architectures in order to compare and determine which one outperforms another in a term of text generation task.

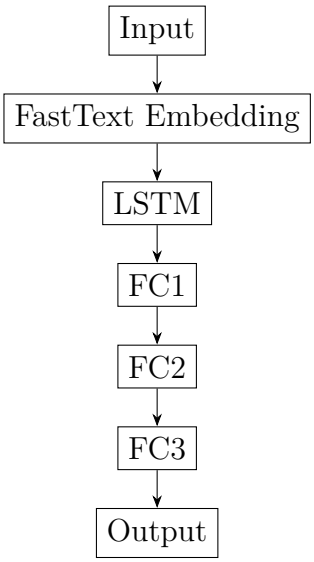

Figure 3.1 - General architecture for LSTM models used in this study.

The main distinguishing feature from other LSTMs is that study uses the FastText [11] pre-trained kazakh vord vectors as an embedding layer. There are several reasons for that usage:

1. Semantic relationship: using FastText pre-trained kazakh word vectors allow to the model to capture the semantic relationship between the words and improving the generated text quality.
2. Generalization: as we use pre-trained FastText word embeddings it means that the model have seen a lot of data which leads to better generalization for unseen data.
3. Size reduction: FastText pre-trained word vectors have low dimensionality rather than random initialized embedings or one-hot encodings. This makes computation more efficient and reduces the risk of overfitting considering that the study already use a small part of the initial dataset.
4. Vocabulary handling: FastText pre-trained kazakh word vectors can handle words that are out of vocabulary by breaking down them into character n-grams.
5. Transfer learning: With the help of FastText pre-trained kazakh word vectors our models at initialization step do not start from scratch, but have already some knowledge about the observed kazakh language.

FastText is a library for efficient learning and understanding the word representations in textual data and text classification. This library was presented by the Facebook's AI research team. It was presented in order to solve the limitation problem of traditional embedding methods such as Word2Vec [12] by capturing the subword information and providing the embeddings for the words which are out of vocabulary. In the 6.5 the process of embedding formation is shown over the text "I have expertise in python".

The first step is a text preprocessing, where the text is tokenized into individual words. Then each word is break down into character n-grams. For example, "python", "pyth", "hon" and so on.

In the second step, FastText applies skip-gram model to learn those word representations. Trying to predict context words bases on a target word. In our case the context words can be "I" or "expertise".

Third step considers using of hierarchical softmax for word embeddings computation. The words are organized into binary tree where each node represents a word in the vocabulary. This technique has overcome traditional softmax as during the training process model traverses the tree which makes the computation of probability distribution over the words more efficient.

Worth to mention that ability to generalize well on the words out of vocabulary is because of FastText captures subword information as described in the first step. With the help of this subword FastText understands the similiarity of the words. As an example, in our case "python" and "expertise" can have common subwords like "py", "th" and etc.

In the forth step, finally after training process each word in the vocabulary represented as a dense vectors or in another words embeddings in a high-dimensional space. Those final embeddings contain the semantic and morphological information of the words in a given context.

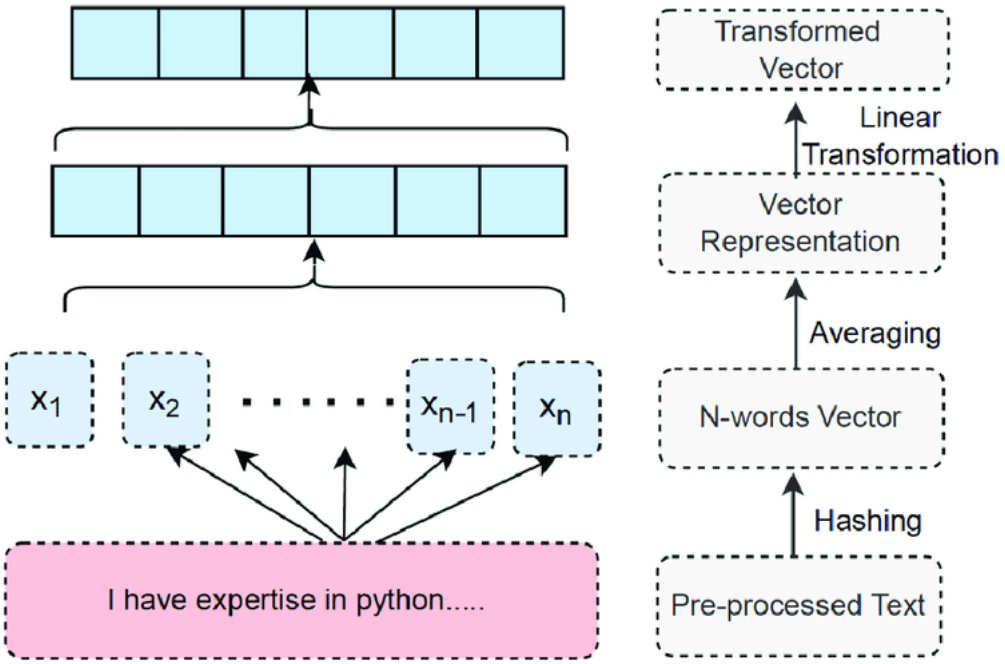

Figure 3.2 - Illustration of FastText embeddings

Source: https://www.researchgate.net/figure/Architecture-of-fastText-embedding_fig1_361845657

Finally, the usage of FastText pre-trained word vectors ad an embedding layer is a good experience for text generation task. The reasons mentioned above in total, provide a good effect on the performance of the model and the quality of the generated text.

3.3 Research setup

Research was conducted on the web platform called Google Colab using Python 3 programming language and Pytorch framework. After many training processes the research process, especially the development of the models and their training were provided and moved to the Google Colab Pro platform where as a processing unit study proposed the TPU with the RAM size of 334.6GB. For the models creation we used PyTorch framework in a pair with downloaded and initialized pre-trained FastText kazakh word vectors. For the storage purposes research considered using of Google Drive Pro in order to store the dataset and saved models after their training process.

Chapter 4

Text Generation Models

4.1 Unidirectional LSTM

Study proposed the usage and comparison of the traditional uni-directional LSTM model with significant changes in the model architecture. The primary reason of choosing the LSTM architecture is that ability to overcome the problem of vanishing gradients. Especially for long sequence data as a text.

Vanishing gradients fenomenon usually happens during the training process of deep neural networks, especially when training the recurrent neural networks. Technically this problem is caused when gradients of the loss function becomes extremely small which leads to the slow training process or even the stop.

The architecture proposed by the study has several changes and optimization techniques in order to properly train the uni-directional LSTM model.

Model architecture

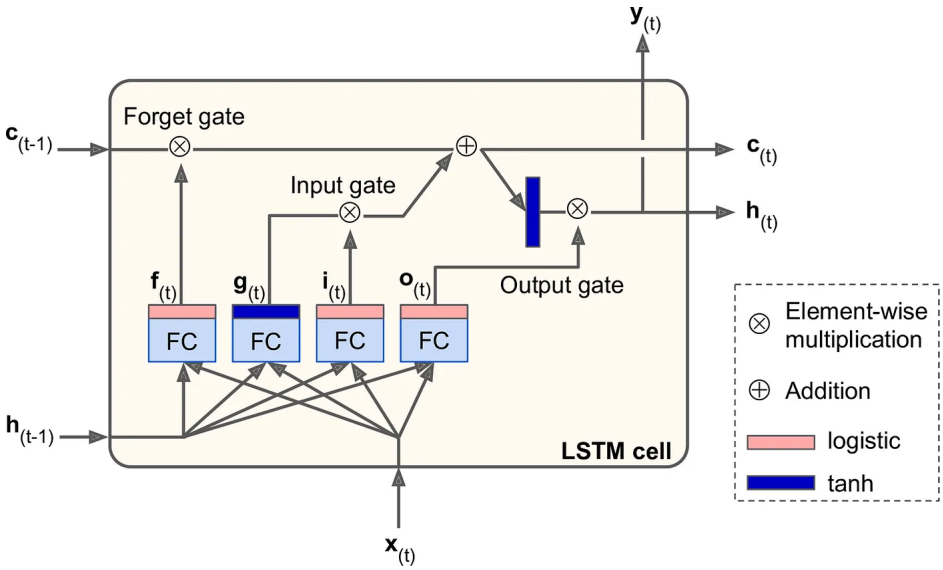

Figure 4.1 - Illustration of LSTM architecture

Source: https://www.oreilly.com/library/view/neural-networks-and/9781492037354/ch04.html

The uni-directional LSTM model proposed by the study consists of single LSTM layer and stacked 3 fully connected layer. As mentioned before, the model uses FastText pre-trained kazak word vectors as its embedding layer in order to represent the words. The LSTM layer captures the information and processes the input in one direction, from past to future manner. Finally the LSTM output is passed to the stacked fully connected layers in order to produce the output predictions.

Distinctive features of our model are stacked fully connected layers at the end and the usage of FastText pre-trained kazakh word vectors as embedding layer. There are definitions of their usage:

1. 3 stacked fully connected layers are used in order to increase the depth of the proposed model, as those deep layers results better performance compared to shallow networks with less layers. Another reason is that additional layers provides more parameters to learn leading to better performance in terms of generating the output.
2. The use of pre-trained FastText word embeddings allows the model to start already with some information and semantic and morphological information between the kazakh words, allowing it to deal with the words that are out of vocabulary. Generally it is a good practise to use it in order to generate quality text, especially for language like kazakh.

Training process

During the training process model is trained using forward and backward sequences to learn input data representations in both directions. The training process repeats until reachs the number of declared epochs. Where at each epoch the model proceeds entire dataset using the batches where the batch size is adjustable parameter.

The parameters of the model are updated using the Adam optimizer based on the gradients computed in the process of backpropagation.

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log(\hat{y}_{ij}) \tag{4.1.1}$$

The equation above represent CrossEntropyLoss function where $\mathbf{y}$ is the ground truth and $\hat{\mathbf{y}}$ is the predicted probability distribution, N is the number of samples in a batch, and C is the number of classes in our case size of the vocabulary.

As a measurement for difference between predicted and actual output the model uses CrossEntropyLoss function. The reason why study uses this function for loss calculations is that in text generation task each word in a vocabulary represents a class, and our aim is to predict the probability distribution over those words for the next word.

In order to prevent gradient accumulation the model uses detach method after each iteration over hidden and cell state.

Pros and cons

The good side of usign the uni-directional LSTM is that they are computationally efficient and simple to implement what makes them suitable to the use in many text generation and sequence tasks.

The bad side is that they can not handle the information in the both direction and the second problem is that there is no ability to capture long-term context in

a sequence text.

4.2 Bi-directional LSTM

The second model used in this research is a bi-directional LSTM model. This architectures solves the problem of uni-directional LSTM in a way that it can process the input data in the both forward and backward directions. With the ability of capturing the information from the past and the future it can producre rich semantic and syntactic features.

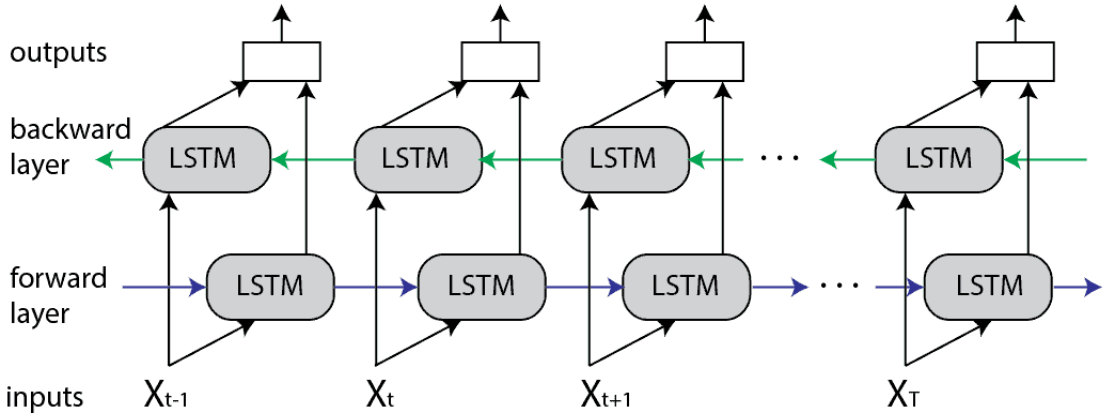

Figure 4.2 - Illustration of BiLSTM architecture

Source: https://dagshub.com/blog/rnn-lstm-bidirectional-lstm/

Model architecture

The bi-directional LSTM consists of two LSTM layer where the first one processes the input sequence in forward directions and the second one in backward direction.

Similiar to the uni-directional LSTM, this model also uses FastText pre-trained kazakh word vectors as an embedding layer for word representations.

Finally, the stacked fully connected layers added to the model in order to process the bi-directional features and produce the output.

Training process

Training process repeats the same logic as in uni-directional LSTM except the moment that bi-directional version learns bidirectional representations of the input data using both forward and backward directions.

Pros and cons

The advantage of using this architecture is that it can handle semantic information for the long sequence data. But the bad side is performance in terms of computing units. This architecture needs powerful units and large datasets in order to properly process the training compared to uni-directional ones.

4.3 Model comparison

In previous sections the study described the both architectures and training processes for uni and bi directional LSTMs. In this section we finally summarize this comparison below:

1. The uni-directional LSTM captures the information over the sequence in one direction from past to future while bi-directional LSTM handles this information in both directions and in longer sequences which outperforms the uni-directional LSTM by generating higher quality text. But uni-directional LSTM can be better at the task where information from the past is necessary in order to predict the future tokens.
2. While the bi-directional LSTM shows the best performance it can cost a lot of computational resources and data compared to the uni-directional one.

Concluding the comparison, the choice between uni-directional LSTM and bi-directional LSTM lies under the options of the performance and resource requirements.

Chapter 5

Evaluation and Metrics

5.1 Evaluation techniques

Qualitative assessment

This method involves experts based evaluation over the defined criteria such as fluency, coherence, relevance and readability. The judges compare the generated text and can leave a feedback or rating over the sample or mark specific parts of the text. Finally among the generated data experts can choose the best one to make an overall evaluation. There are some examples of this technique:

1. Content analysis: the experts analyse the content generated by the language model for some criteria like relevance, accuracy and correctness. They compare the generated output with the referenced text in order to evaluate its accuracy.
2. Visual inspection: the judges inspect the visually the generated text and search for correctness in such aspects like font size, symbols, spacing and etc. In order to evaluate the models output for aesthetic appeal and readability.
3. Comparative analysis: in this case, the experts compare generated outputs of different language models in order to determine their strengths, weaknesses and output quality.

Quantitative evaluation

This approach proposes the usage of automated tools in order to evaluate the generated text. Those tools provide the numerical scores such as similarity between generated and referenced text, their diversity or semantic coherence. Example of this tools can be seen from the researches in a previous work from the chapter 2 where they used metric called BLEU score in order to evaluate the quality for generated text.

Human studies

This metric assumes that generated text can be evaluated without experts of the field but with the help of end-user. Usually they can be asked to perform work or give a feedback about the quality, readability and relevance of the generated text. Human studies can produce a valuable feedback on model performance and the quality of the generated text. There are several examples of this metric:

1. Survey analysis: researchers design a survey where the participants will be asked to read and evaluate the generated text using some criteria such as

grammatics or overall text quality using some predefined rating system.
2. Focus group: researchers provides groups consisting of small amount of persons where the aim is to collect group's opinion, suggestions from the different perspectives regarding the generated data.
3. Interview analysis: in this methods the researchers provides one to one interviews with participants in order to collect the opinion, experiences and perceptions after exploring the generated data.

5.2 Performance metrics

BLEU Score

BLEU stands for Bilingual Evaluation Understudy. This metric measures the n-gram overlap between the referenced and generated text by calculating the precision for n-gram size and combines them using geometric mean. The range of this score lies between 0 and 1 where high score means high similiarity to the referenced text.

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{5.2.1}$$

ROUGE Score

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. This score evaluates the overlapping between the generated and referenced text. The score range vary between 0 and 1 where value closer to 1 means high level of similiarity to the referenced text.

$$\text{ROUGE} = \frac{\text{number of overlapping words}}{\text{total words in reference}} \tag{5.2.2}$$

F1-Score

This metric calculates the harmonic mean of precision and recall. Usually this metric is used in text classification tasks, but can also be adapted to the use in text generation problems.

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{5.2.3}$$

METEOR Score

METEOR stands for Metric for Evaluation of Translation with Explicit Ordering. This metric evaluates the quality of machine translation by measuring the accordance of generated and referenced text.

$$\text{METEOR} = \frac{P \times R}{\alpha \times P + (1 - \alpha) \times R} \tag{5.2.4}$$

Overall it is worth noting that most of the metrics used in text generation model evaluation tries to calculate the difference between generated and referenced text proposed by the model.

5.3 Perplexity-Based Evaluation

In the language processing area the method called perplexity is significant metric which is used in various language models including those for text generation task. This metric is a measure of language model uncertainty when predicting the next word in an sequance. The low value of metric means that the model is accurate enough to generate the quality text.

$$\text{PPL}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1, w_2, ..., w_{i-1})}} \tag{5.3.1}$$

where the N is the number of words in a test set and $P(w_i|w_1, w_2, ..., w_{i-1})$ is conditional probability of word w_i given set of previous words $w_1, w_2, ..., w_{i-1}$.

Summing up, there are various metrics and techniques to evaluate large language models starting from human evaluation and ending with automated tools we covered above. Evaluation of the models is a significant part of the model development as it shows the model performance in terms of how they good at generating the text toward to referenced ones and its quality.

Chapter 6

Results

6.1 Metrics

The training process for the both LSTM models was provided several times, where each time there was a changing in such hyperparameters like learning rate, batch size, epoch number, optimizer algorithms, dropout probability rate and manipulation with data.

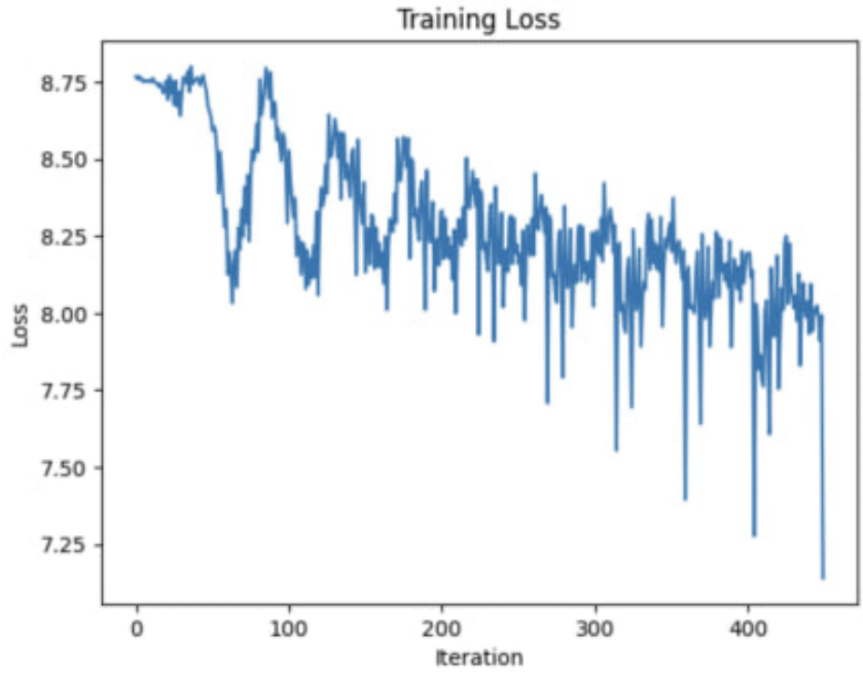

Figure 6.1 - Illustration of simple LSTM model training epoch - loss graph

The first try was not good as expected as we can see in the figure above, the training loss was jumping to high whenever new epoch starts. Also the architecture of the model was not original as described in this study. For the first try it was the same LSTM model but without FastText pre-trained kazak words embedding layer and optimizer algorithm. The following parameters were set:

1. Learning rate - 0,00001.

2. Batch size - 32
3. Epoch number - 500
4. Sequence length - 8

The reason why loss is jumping too high at the beginning of the epoch is addressed to the small batch size and sequence lenght parameters. They cause the fluctuations in the each epoch loss.

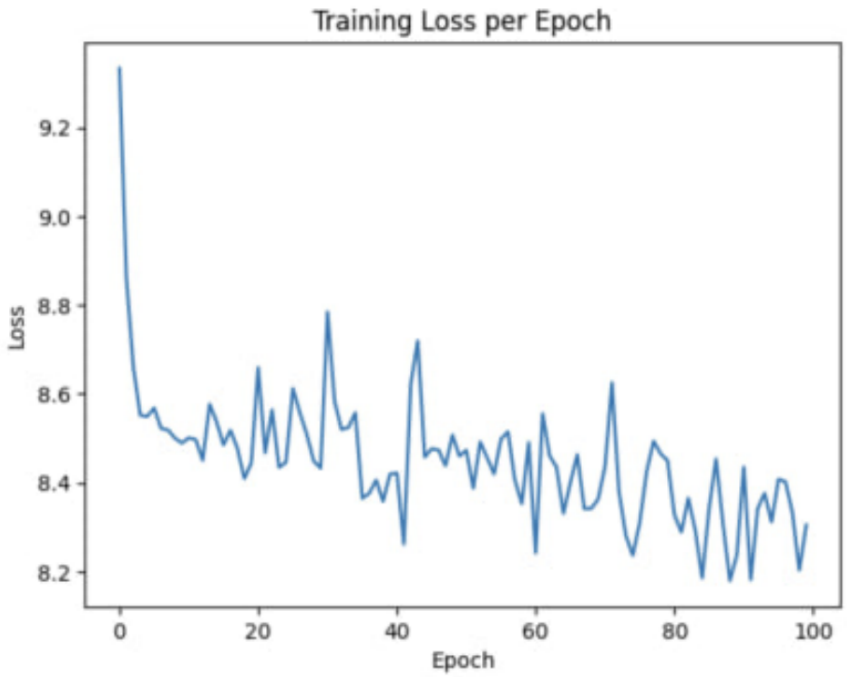

Figure 6.2 - Illustration of simple LSTM model training epoch - loss graph

After increasing the batch size and the sequence length with learning rate we got a bit smoother graph, but as we see the loss is still jumps and causes the fluctuations at the each epoch loss. The applied parameters are written below:

1. Learning rate - 0,0001.
2. Batch size - 64
3. Epoch number - 100
4. Sequence length - 16

After tuning the hyperparameters of the models, study proposes the usage of optimizer algorithms and increasing the model parameters. Result below shows the loss - epoch graph of the second try with the following hyperparameters set:

1. Learning rate - 0,0001.
2. Batch size - 128
3. Dropout - 0.1
4. Adam optimizer algorithm
5. Epoch number - 10
6. Sequence length - 24

Now the the loss in both models seems to be smoothly decreasing and this is the metric which tell us that models are started properly learning. To make the training process better the models were tuned and again trained in order to achieve

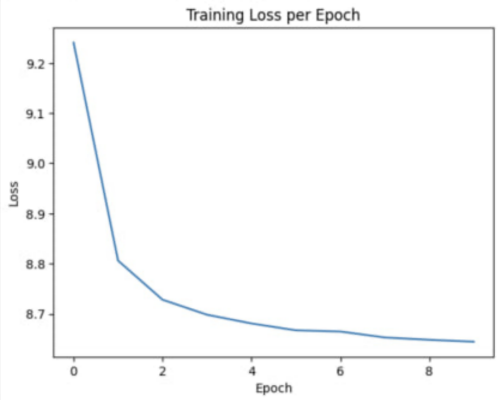

Figure 6.3 - Illustration of training Loss - Epoch graph for unidirectional LSTM

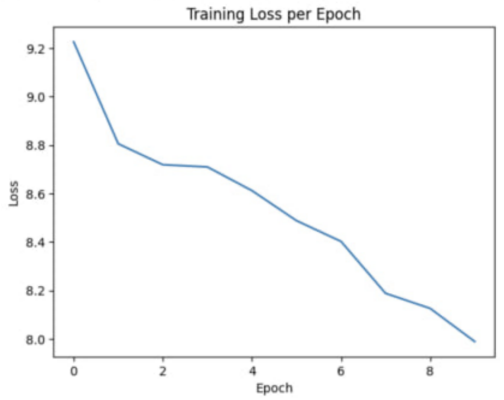

Figure 6.4 - Illustration of training Loss - Epoch graph for bi-directional LSTM

better loss - epoch results.

Finally to achieve better results, the models was tuned and trained again with the following models parameters:

1. Learning rate - 0,0001.
2. Batch size - 256
3. Dropout - 0.5
4. Adam optimizer algorithm
5. Epoch number - 100
6. Sequence length - 18

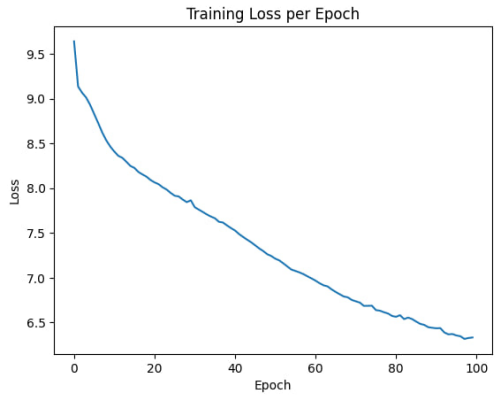

Figure 6.5 - Illustration of training Loss - Epoch graph for unidirectional LSTM

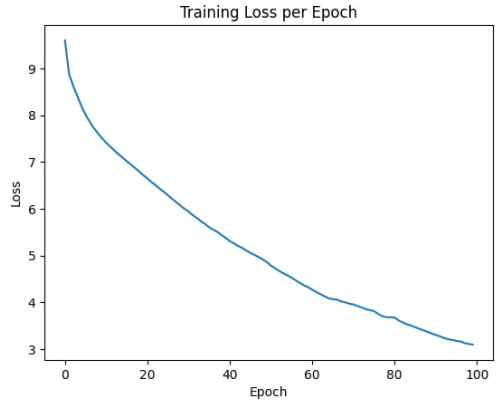

Figure 6.6 - Illustration of training Loss - Epoch graph for bi-directional LSTM

It worth to mention that in both graphs of the LSTM models the loss is still decreasing meaning that we can repeat the training process with the increased epoch number. Unfortunately in this study because of the computing unit limitations we could not provide the training process with a large number of epochs in order to see how the models will behave in such situation.

After this decision, the study decided to provide another try and train the

bi-directional LSTM with increased model parameters before reaching the computational limit of the machine. The parameters are shown below:

1. Learning rate - 0,0001.
2. Batch size - 256
3. Dropout - 0.5
4. Adam optimizer algorithm
5. Epoch number - 5, 50 and 100
6. Sequence length - 80

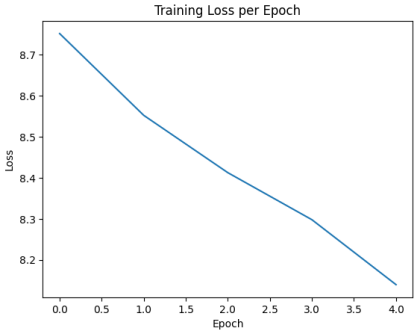

Figure 6.7 - Epoch 5 and sequence length 80

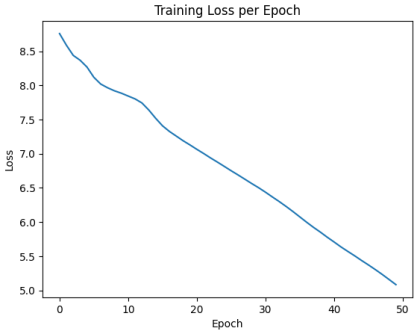

Figure 6.8 - Epoch 50 and sequence length 80

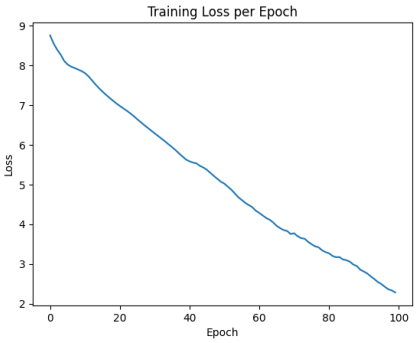

Figure 6.9 - Epoch 100 and sequence length 80

Finally, through several training processes with different model parameters but the same for both architectures we see that with the same dataset and development environment, the bi-directional LSTM outperforms its uni-directional version in the task of text generation. At least we can state that bi-directional model shows better results in the training process compared to the uni-directional LSTM. This experiment proves that for future work when there is a choice between uni-directional and bi-directional the better one is the bi-directional LSTM model for text generation task especially for low-resourced kazakh language.

Similarly, the study proposes the comparison of the training and test loss compared to the epoch of the both models in one graph as shown in the figure 6.10:

From the results of the figure 6.10 it is supposed to be the over fitting problem due to test loss value increase compared to the epoch number, but it is worth to mention that the amount of the data passed to the test process is relatively small

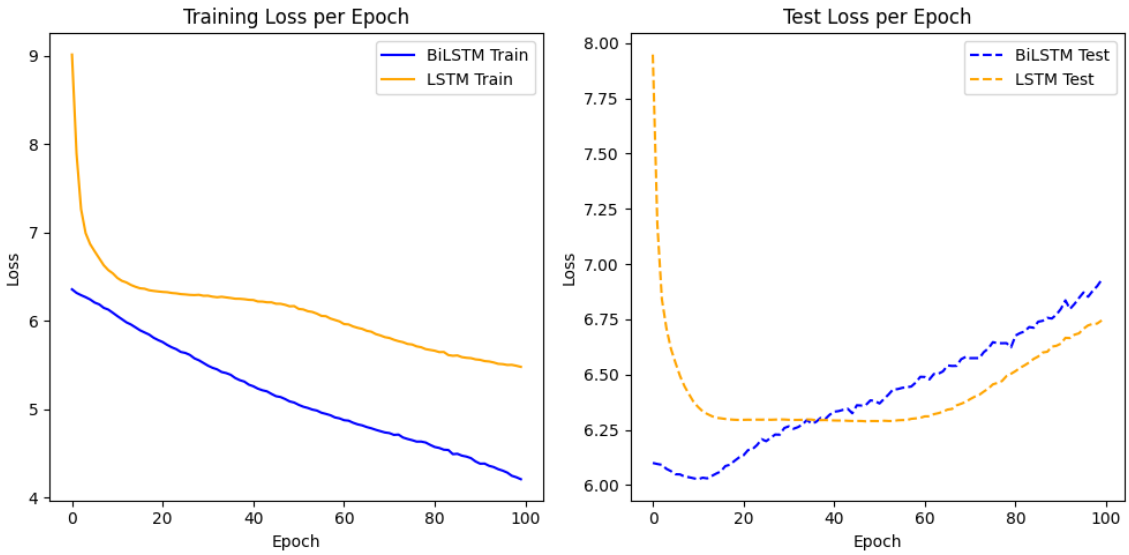

Figure 6.10 - Illustration of uni and bi-directional LSTMs epoch - loss graph

due to the size of the original dataset used in this study. Again it all comes down to the lack of performance units we faced during the training process in a Colab Pro platform. In order to solve this problem the study requires higher computing units to train the models properly over the full dataset and increase models capacity.

Concluding, in this section the study showed model training and evaluation results by comparing both models performance and according loss - epoch graph. The process of developing the models was difficult from the point of view computing units. Even purchasing the paid subscription for Google Colab Pro could not untie our hands to provide more experiments with increased dataset and larger model parameters.

6.2 Analysis

The objective of this section is to analyse produced text from the both trained models and compare them in a context of visual, semantic perception. The algorithm of the proposed model as follows:

1. Define sequence lenght
2. Define start word
3. Run the generate process

In the figures below the research shows generated outputs of the uni-directional and bi-directional models. The starting word defined by the user is aligned with a red box, whereas the words semantically close and similiar are marked with yellow box. As a result, models generate the kazakh text based on the start word, sequence length and the data it observed during the training process. In order to achieve better results the passing sequence length should be the same as was during the training process. If we pass the value bigger, then after reaching the trained sequence length the model will produce duplicate words.

The uni-directional model generated the output which is demonstrated in the

figure 6.11. As we see in the context of the generated text there are only one work semantically close to the start word. This is due to the fact that the uni-directional LSTM are poor for capturing the information for a long sequences leading to the loss of meaning in generated text. From the other hand, probably we could improve its performance by adding more training data and reducing the sequence length parameter.

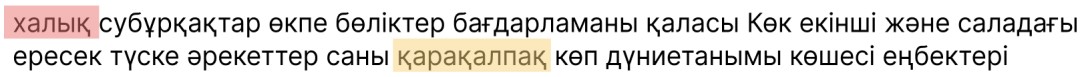

Figure 6.11 - Illustration of the output provided by uni-directional LSTM

The bi-directional version shows better results in a way that there are more semantically related words to the start word. The reason is that this model captures the information in both direction as for forward as for backward process.

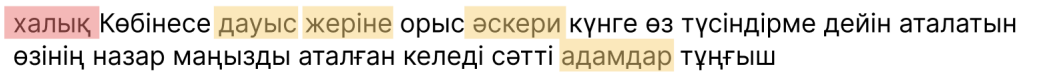

Figure 6.12 - Illustration of the output provided by bi-directional LSTM

Based on the output provided from the both models we can state that the bi-directional model outperforms the uni-directional model in terms of the number semantically related to each other words. There are some aspects of bi-directional architecture which study proves:

Text quality

The advantage of the bi-directional LSTM is that they handle the context information in both directions which leads to better context understanding and capture the dependencies in the text. That the reason why quality of the generated text is higher than uni-directional one in terms of the number of co-related words in the same context which depends on the start input word.

Complexity

Generally, bi-directional LSTM is more complicated compared to the uni-directional architecture because of the need to analyse the data in both directions. This complexity leads the need of huge computing units and time making this architecture less scalable especially in the cases when dealing with large datasets or deploying the model in the environment with limited resources.

Requirements

The bi-directional LSTM require state saving for both activations and gradients during the forward and backward process. This leads to the usage of high processing units compared to the uni-directional LSTM. Such requirements can cause the problems during the training process, especially when working with vast datasets or processing long sequences of text.

Finally, the study proposes the evaluation metrics for both models in the face of the perplexity based evaluation as shown in the figure **??**.

From the results of the perplexity score it is obvious that the models are not fitted well and are uncertain about the next word in the predicting sequence. This

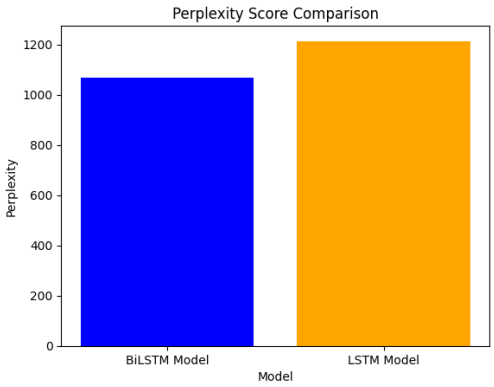

Figure 6.13 - Illustration of training Loss - Epoch graph for uni-directional LSTM

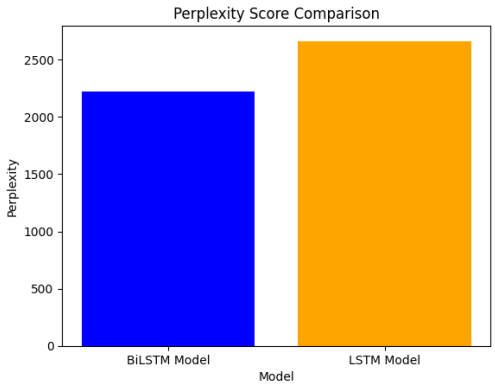

Figure 6.14 - Illustration of training Loss - Epoch graph for bi-directional LSTM

problem is caused due to the low probability score to predict the right word in a sequence which results as a high value of the perplexity score. From the other hand, if the models will show the high probability for predicting the right words in a sequence this will lead to low value of the perplexity score meaning that the models fits well.

The point to mention in the comparison of the both models perplexity score is that in the figure 6.13 we have initial validation dataset over which the perplexity score was gained. In the figure 6.14 we increased the size of the dataset by 50% and as we see the difference of the BiLSTM and UniLSTM models perplexity score increases also, which means that in any case the bi-directional one outperforms the uni-directional one in a Kazakh language based text generation task.

Thus, the choice which model architecture to use lying under the choice of performance, text quality and scalability decisions. Even in provided research to achieve some quality results we used the maximum processing units in a Google Colab Pro platform to train and generate some meaningful text data. For future works, the decision of the model choice should consider the application requirements and observe an environment for computational resources. Due to the fact that training the large language model consumes a lot of computational power and time.

Chapter 7

Discussion

7.1 Challenges

Data availability

During this research one of the faced challenges was quality data availability in order to train the models for text generation task. Despite the fact that the study tried to collect the data we faced the problems like difference of lingustic styles among the resources across different domains and data sparsity. Finally we decided to use the dataset consisting of 300 000 thousand unique rows of text with different number of word sequences. All that data was taken from the kazakh language based Wikipedia dump. The data was raw and needed preprocessing steps such as lowercasing, removing empty lines, symbols and special characters with numbers.

Model complexity

Another significant problem happened because of the LSTM model complexity. The difficult part of this architecture is that the study needs a lot of time and computational resources in order to train and tune the models hyper parameters in order to achieve some good results.

Computing resources

The primary challenge was connected with computing resources, especially for the tasks of data preprocessing and model training. In an average each training session on 100 epochs using 10 000 rows of data took from 9 to 14 hours. And every time after tuning the model there was a need to repeat that training process to see whether the changes make an effect on model performance or not.

The study was using pytorch framework in order to provide the computing process using GPU or TPU units, but still the training process was reaching the maximum machine resources available on Google Colab Pro platform. To conduct the study, we decided to reduce the original dataset size in order to finish the training process of the observed models.

7.2 Opportunities

The problem of kazak language based text generation can be solved using state-of-the-art GPT architecture. Some of the recent works in text generation area

especially for kazakh language use this technique in order to generate kazakh text. The most common approach is to set as a base pre-trained large GPT language model and fine tune it using kazakh text dataset. There are general steps:

1. Choose the appropriate GPT model. Usually large models shows the best performance but requires more computational power.
2. Data preparation step involves preprocessing techniques such as lowercasing, removing symbols and special characters, tokenization.
3. Fine tuning is an optional step, but for kazak text generation is required. The process updates model parameters based on provided dataset to improve the model performance on text generation task.
4. The last step is text generation. This process is achieved by passing some prompt or sentence to the model and letting it generate the rest of the text.

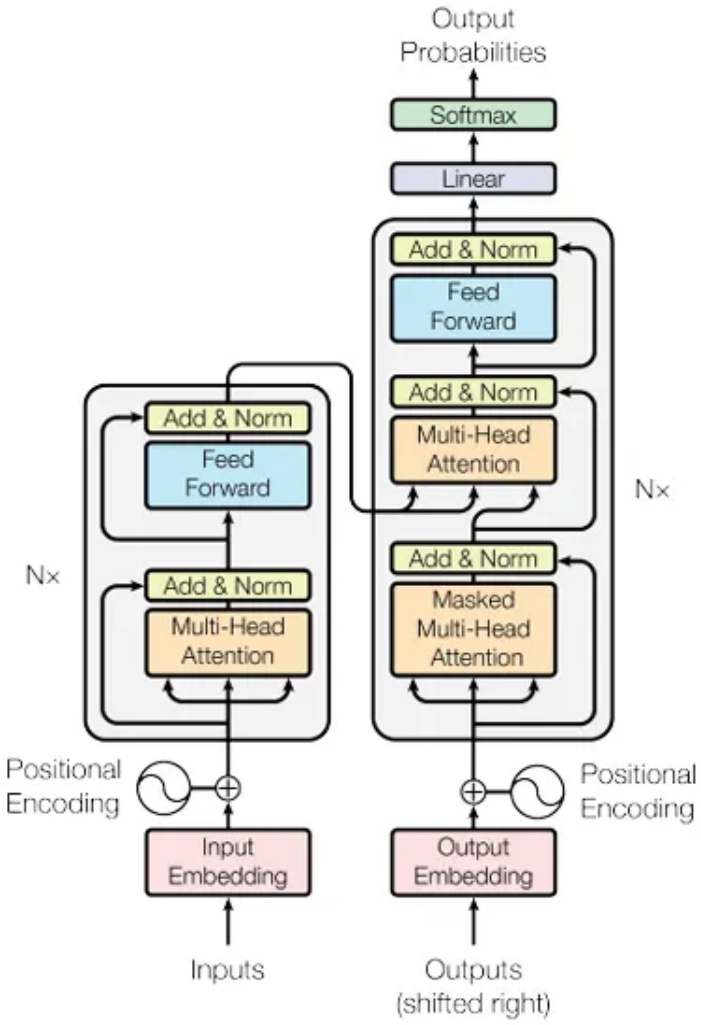

Figure 7.1 - Illustration of GPT architecture

Generally, the GPTs are highly efficient in NLP tasks and there are several reasons, making them so :

Architecture GPT models are built using transformer architecture which allows them to capture long sequence dependens in the text effectively. Unique part in this architecture is that they use self attention mechanism allowing the model to understand context and semantics through out the input text.

Pre-training Those models usually are pre-trained on a vast amount of the data allowing it to understand the language, its linguistic patterns and styles. This advantage allows to GPTs perform well over wide amount of NLP tasks without being fine tuned.

Adaptability GPT models are highly adaptive in a way that they can produce different text on various languages, genres and styles. Also they can be tuned on domain specific data which makes this models a valuable and usedful tool for the researchers in NLP area.

7.3 Implications

This study contributes to the development of NLP for the low resourced kazakh language by developing and comparing uni-directional and bi-directional LSTMs in order to determine which model outperforms another in kazakh language based text generation task. With the help of such researches the future studies can save their time in architecture selection or data augmentation processes.

Additionally, this study not only makes contribution to the field of natural language processing for the kazakh language but also makes valuable effect into the specific challenges associated with the processing and generating text in a low-resource languages as kazakh language is. On the base of the study where we are evaluating and comparing the performance of uni-directional and bi-directional LSTM models, this study highlights the strengths and limitations of each approach in the context of kazakh text generation, especially we can see the difference how the models fit in small dataset.

Future researchers can leverage these findings to make further decisions about model selection and optimization strategies, potentially saving the time and increasing the efficiency of their work. Also, the study proves that state of the art models are good in order to use pre-trained large language model as a base to construct hybrid models as we covered in the related works section.

From the other hand, the findings proposed by this study can affect on the creation of more comprehensive and representative datasets for kazakh language processing rather than using the current ones. As the vast part of the available data needs preprocessing and sorting for the corresponding domain.

The analysis provided in thos study by comparing of uni-directional and bi-directional LSTMs can serve as a benchmark for the later, following studies, by providing a reference point for improvements and innovations in terms of architecture development or the dataset usage. This work enables future researchers to build on established knowledge, potentially incorporating more advanced techniques such as attention mechanisms, transformer architectures, or integrating large language models to enhance the model performance.

In addition to technical contribution, this study underscores the importance of cultural and linguistic preservation through technology. By focusing on the low

resourced kazakh language, it not only makes a contribution to the development of practical applications such as automated translation, text summarization, and sentiment analysis for kazakh language speakers but also helps in maintaining the linguistic heritage by making the language more adaptable to the modern technologies and time.

In conclusion, this study makes valuable contribution as technical insights into NLP for the kazakh language and also sets a starting point for the future researches.

Chapter 8

Future Directions

8.1 Applications

Proposed models can be used in different domains starting from content creation, machine translation and ending with language education. From research point of view the provided models are suit for using in data augmentation process. In the reality, kazakh language based content is only a small part of the accessible data on the internet, as a result in order to fill this gap text generation models can be used to generate kazakh synthetic data. As an example, generated text can be used for text to speech applications.

From the other hand, future studies can use this models as a base and produce even modernized version which outperforms current results using the metrcis and suggestions for the future works.

The general applications of the text generation models can be seen in a wide variety of domains, leveraging their ability to generate coherent and contextually relevant text.

Content creation Text generation models can be applied in automated generation of news articles, blog posts and other written content.

Conversational agents Such models can also be used in online assistants and chat bots by generating responses in real-time conversations.

Translation and localization Translating text from one language to another while maintaining context and fluency. Also can be used for adapting the content to suit different cultural and regional preferences.

These applications showcase the importance and potential of text generation models to enhance efficiency, creativity, and personalization across various industries and the use cases.

8.2 Limitations

During the research we faced many limitations in terms of computing power and good quality data availability. It is worth to mention that for future studies it is better to use powerful processing units that outperforms Google Colab Pros units. Also for future studies, if the work planning to build the large language model based on the Google Colab platform, then it is suggested to use the Pro plan in

a pair with TPU processing unit with Tenserflow framework. As this framework allows to open and use the whole power and performance of the TPU processing unit.

The study proposes during the training process the hyper parameter called sequence length, which determines the length of a sequence to learn from the input rows of the dataset. The average sequence length of the each row in a dataset was 90 words, but due to the lack of computing power, the study proposes the 20 to 30 sequence length for training process. The usage of high computing units means increased dataset and sequence length parameter, which increases the model capacity and its performance in predicting the words in a sequence.

8.3 Importance

Since the Kazakh language is a low resourced language there are lack of studies in the kazakh language NLP area. This study makes another one contribution in the development of science area especially for kazakh language. From another hand, nowadays there is a digital century where the survival of the language depends on its usability and application in digital spaces.

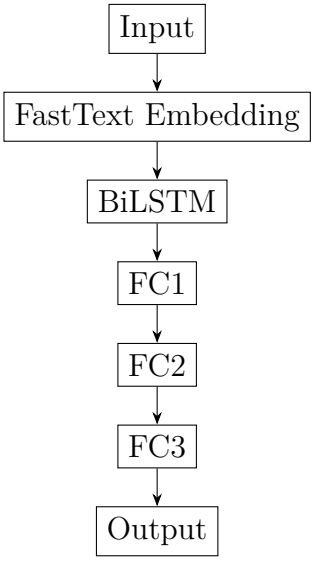

Figure 8.1 - Final architecture for BiLSTM model used in this study.

It is worth to mention that during the study we noticed that LSTM fits better on a small dataset compared to the BiLSTM architecture, this conclusion can be proved by showing the figure 8.2. Where the train loss decreases in a both architectures, but when the turn comes to the test loss the BiLSTM loss is increasing drastically compared to the LSTM architecture.

In academics, such study can support and help the researchers to generate kazakh synthetic text in order to process and find its application in their studies. The thing to notice is that this study can be used further as a data augmentation technique for application which needs kazakh language based synthetic data. For example, text to speech works, where the researchers can not only collect the data

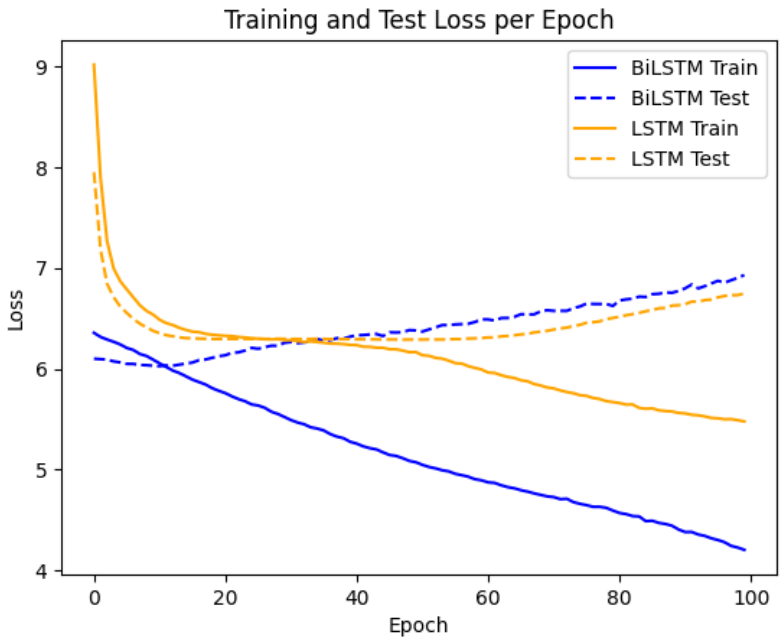

Figure 8.2 - Illustration of simple LSTM model training epoch - loss graph

or use the official corpus, but also generate the text by feeding those data into proposed models.

It is important from the architecture point of view, to tune the models by adding the self attention mechanism as used in GPT architectures in order to increase the models performance as mentioned in the observed works in a chapter 2. This option allows the models to capture long-range dependencies and understanding of the context.

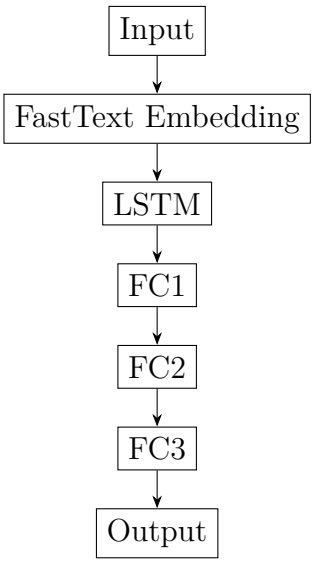

Figure 8.3 - Final architecture for Uni-LSTM model used in this study.

Talking about the attention mechanism it can be used as different method and techniques such as adding the embedding layer to the model architecture or

using the state of the art algorithms such as TF-IDF and etc. The usage of such mechanisms allows to increase the model performance by providing the information how the appereance of some word depends on the contextual information.

Attention mechanisms revolutionized the field of natural language processing by allowing models to focus on relevant parts of the input sequence and understand the context, thereby improving the handling of long-range dependencies and understand the relationships within the data. The attention mechanism in this study is used within a neural network model as a FastText pre-trained kazakh word vectors, highlighting its role in enhancing performance by dynamically weighting the importance of different words in a sentence. WIth the help of this layer the model can handle even the cases when the original words are not appearing in the original context.

Chapter 9

Conclusion

9.1 Summary

In this study we provided comprehensive investigation in text generation models for kazakh text generation, the research focuses on developing and comparing uni-directional and bi-directional LSTM architectures. Our research is to evalute those models in generating kazakh coherent and contextually relevant synthetic data.

First we proposed the usage of kazakh Wikipedia dump consisting of 300 000 unique rows where each row has different lenght of words sequence. After we introduced FastText pre-trained kazakh word vectors as an embedding layer for our LSTM models in order to capture semantic information from the input text and increase the performance of observed models.

The experiment involved training multiple times the proposed LSTM models with fine tuning the hyperparameters such as learning rate, droupout probability rate, batch size, epoch number, optimizer algorithm to increase the performance.

Research results showed several key findings regarding the performance and the quality of the generated text of both models. We found out that bi-directional LSTM proposes more semantically similiar words in a result sequence compared to the uni-directional one. However, it is worth to mention that even bi-directional version shows great performance at the same time this architecture limiting as in a term of computational units and scalability.

In conclusion, the research provides valuable insight into the question of effectiveness between the unidirectional and bidirectional LSTM architectures for kazakh language based text generation task. Our study proves that the usage of state-of-the-art language models can have a positive effect on the result of the observed area. Generally, the development of large kazakh language model is a great achievement and opportunity to make a vast contribution to the development of kazakh language based applications and studies in order to increase the number of works in domain of low-resourced kazakh language area. Such works can find their application in various scenarios starting from text synthesis, text to speech application and ending with image description or even generating the personal artificial assistant in a kazakh language domain.

9.2 Recommendations

Looking back to the provided research there are several suggestions and recommendations to the future studies which are planning to use uni-directional or bi-directional architectures in order to generate kazakh synthetic text data.

First, the study suggest careful choice of the model depending on the problem it is planning to solve. Using LSTM architecture requires high computational resources in a pair with time to train the model.

Second suggestion, it is better to use attention mechanism in a pair with FastText embedding layer in order to understand the context and key points in the input text.

Third recommendation, it is better to use grouped by the domains dataset rather than one big mixed domains dataset and provide a deep analysis of the kazakh languages semantics, structure and language vocabulary as since while kazakh language is a low-resourced at the same time it is very rich language.

The last suggestion for future works, If you are faced with the task of rapid development of language model for text generation task, it is better to use pre-trained large GPT language model. By tuning this GPT with the provided domains dataset you will achieve the results in much faster and efficient way in the time point of view. But, GPT models requires the high computational resources in order to be well trained and fine tuned using specific domains dataset.

Finally, the study shows that in order to properly use the Google Colab's TPU units, it is better to develop the models using the framework Tenserflow [13], as it allows to use the tensors of the TPU unit to maximaze the performance of the proposed processing unit for training processes.

Bibliography

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 1997.

[2] Google. Colab notebook [computer software], 2024.

[3] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training, 2018.

[4] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks : the official journal of the International Neural Network Society*, 18:602–10, 2005.

[5] Cristiano Castelfranchi. Alan turing's "computing machinery and intelligence". *Topoi*, 32, 2013.

[6] D. Rakhimova, S. Abilay, and A. Kuralay. The task of generating text based on a semantic approach for a low-resource kazakh language. In *Recent Challenges in Intelligent Information and Database Systems. ACIIDS 2023. Communications in Computer and Information Science*, volume 1863. Springer, Cham, 2023.

[7] Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. Bleu: a method for automatic evaluation of machine translation. 2002.

[8] K. Jones. A statistical interpretation of term specificity in retrieval. *Journal of Documentation*, 60:493–502, 2004.

[9] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, and Dario Amodei. Language models are few-shot learners. 2020.

[10] D. Goldhahn, T. Eckart, and U. Quasthoff. Building large monolingual dictionaries at the leipzig corpora collection: From 100 to 200 languages. In *Proceedings of the 8th International Language Resources and Evaluation (LREC'12)*, 2012.

[11] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. En-

riching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 2016.

[12] Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at ICLR*, 2013.

[13] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL Workshop: Text Summarization Braches Out 2004*, 2004.