Kazakh Large Language Model (Kaz-LLM) for Sentiment Analysis in Online Media

by

Abzal Nurgazy

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

at the

NAZARBAYEV UNIVERSITY

April 2025

Author .....
Department of Computer Science

Certified by .....
Michael Lewis
Associate Professor
Thesis Supervisor

Accepted by .....
Yelizaveta Arkhangelsky
Acting Dean, School of Engineering and Digital Sciences

Kazakh Large Language Model (Kaz-LLM) for Sentiment Analysis in Online Media

by

Abzal Nurgazy

Submitted to the Department of Computer Science
on , in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science

Abstract

The increasing volume of digital content, especially in textual form containing various multilingual and cultural features has highlighted the need in efficient analysis models. Traditional machine learning approaches such as support vector machines, simple neural networks or Bayesian models struggle to capture the complexities of languages. Recent advancements in natural language processing (NLP) and the availability of various text data have made it possible for transformer-based models to perform more in-depth analyses. However, despite these advances, LLMs still face challenges when it comes to applications of under-resourced languages, including Kazakh. This work explores the potential of integrating Retrieval-Augmented Generation (RAG) systems with standalone LLM to enhance their performance in sentiment analysis tasks. By leveraging of embedding models and fine-tuning multilingual models with specific objectives the proposed approach aims to improve the accuracy and relevance of generated final responses . Since the performance of rag systems directly depends on the characteristics of embedding models, this work will evaluate and compare metrics with baseline models. Moreover, based on the parsed data from news feeds with a wide range of different topics, a synthetic dataset will be created for both training and evaluation. Performance evaluation does include metrics such as accuracy, precision, recall and other, providing insight into the applicability of the model in real-world applications.

Thesis Supervisor: Michael Lewis
Title: Associate Professor

Contents

1 Introduction **11**

1.1 Problem Statement 11

1.2 Goals and Aims of the Current Work 16

2 Related works **19**

2.1 Sentiment Analysis for daily news 20

2.2 Role of Embedding Models in Textual Data Processing 21

2.3 Retrieval Augmented Generation 27

3 Datasets for Embedding Models **33**

3.1 Open-domain datasets 33

3.1.1 MS MARCO: A Human Generated MAchine Reading COmprehension 33

3.1.2 Natural Questions (NQ) 36

3.1.3 TriviaQA 37

3.2 Closed-domain datasets 39

3.2.1 SQuAD (Stanford Question Answering Dataset) 39

3.2.2 HotpotQA 41

3.3 Synthetically generated dataset 43

4 Methodology **47**

4.1 Foundation of proposed approach 47

4.2 News feed parsing 48

4.3 Fine-tuning of Embedding Models 52

5 Results and Discussion **57**

6 Conclusion **65**

6.0.1 Future works 67

List of Figures

1-1 Performance and cost comparison between large context LLMs and RAG-powered simple LLMs 14

3-1 Special UI created for annotation of answer passages by human editors [2] 35

3-2 Annotation example from the NQ dataset [21] 36

3-3 The majority part of annotated questions contain multiple entities [17] 38

3-4 Interface for annotators to create questions from the given passage [38] 40

3-5 Types of sample questions from the HotpotQA dataset [49] 42

3-6 Raw data from the data parsing used for the synthetic dataset creation 44

3-7 The internal structure of the custom dataset 44

4-1 Workflow of site parser 51

4-2 Fine-tuning and pre-training process of BERT model serving as a base for the most part of existing embedding models 53

5-1 Baseline *intfloat/multilingual-e5-base* model's performance on test dataset 57

5-2 Fine-tuned *intfloat/multilingual-e5-base* model's performance on test dataset 58

5-3 Fine-tuned *intfloat/multilingual-e5-large-instruct* model's performance on test dataset 58

5-4 Baseline *intfloat/multilingual-e5-large-instruct* model's performance on test dataset 59

5-5 MTEB benchmark evaluation for English language, fine-tuned *intfloat/multilingual-e5-base* 60
5-6 MTEB benchmark evaluation for Russian language, fine-tuned *intfloat/multilingual-e5-base* 61
5-7 t-SNE for English word embeddings obtained from baseline *intfloat/multilingual-e5-base* 61
5-8 t-SNE for English word embeddings obtained from finetuned *intfloat/multilingual-e5-base* 61
5-9 t-SNE for Kazakh word embeddings obtained from baseline *intfloat/multilingual-e5-base* 62
5-10 t-SNE for Kazakh word embeddings obtained from finetuned *intfloat/multilingual-e5-base* 62
5-11 MTEB benchmark evaluation for Russian language, baseline *intfloat/multilingual-e5-base* 63
5-12 MTEB benchmark evaluation for English language, baseline *intfloat/multilingual-e5-base* 64

List of Tables

4.1 The structure of data in the dataset used for Sentence Transformer training 54

Chapter 1

Introduction

1.1 Problem Statement

Early works in the field of automated human language processing started with rule-based systems capable of text parsing and generation. These systems relied on predefined sets of linguistic rules to analyze text, providing limited flexibility but serving as a basis for further improvements. The advent of machine learning has ushered in a new era of text processing methods using learnable models such as support vector machines (SVM) and naive Bayesian classifiers. However, the real breakthrough in natural language processing (NLP) came with introduction of highly complex transformer models in 2017 [45]. Transformers with their encoder-decoder architecture revolutionized NLP by introducing an attention mechanism that enables model to focus on relevant part of text regardless of the distance between words. Such property allowed to capture a long range dependencies and contextual relationships among sentence or text.

The advancement in computational power and machine learning algorithms have revolutionized Large Language models (LLMs) into a new level making it possible to process textual data at human level of comprehension. They are trained on the massive datasets so that they can understand emotional context or nuanced sentiment of a given text automatically. Traditional sentiment analysis based on simple statistical techniques struggles in understanding irony, sarcasm or ambiguous information, thus

leading to different interpretations [15]. Moreover, the digital content produced everyday is overwhelming and the manual sentiment analysis is very inefficient in terms of resources and time. The natural language processing is further complicated by the presence of multilingual materials and culturally varied content. Therefore, such operations have to be done automatically by highly capable language models.

The screening of the texts by the model compared to the human can emphasize the critical problem arising among public opinions, so that it serves as a warning system. This allows government agencies to stay informed about public reaction to newly adopted policies. However, there is still a room for advancement and especially for the LLM capable of processing materials in Kazakh language. Training of the model requires a large amount of dataset written in the Kazakh language. Since the grammatical features of the language differ from English, which is the main language for working with the model, it will be necessary to create a unique tokenizer.
The authors of the [44] noted that Kazakhstan, as one of the major countries in Central Asia, is lagging far behind global development in the field of Natural Language Processing (NLP). Such a large gap, in their opinion, is caused by the weak development of LLM and benchmarks for their assessment that take into account the cultural and linguistic characteristics of the Kazakh language. Even the texts in Kazakh are present in some multilingual benchmarks the majority part of information resources rely on English language. Accordingly, benchmarks or trained language models simply cannot fully cover the entire richness of the Kazakh language, its dialectal features, historical and cultural contexts and specific language constructions. This feature may limit the models in processing Kazakh text, which will limit their wide applicability in real-life scenarios such as sentimental analysis. The assumption is also confirmed by the author of the work [50] who talks about the small number of academic publications over the past eight years. Moreover, even those publications that have primarily focused on general applications of language models rather than the specific task of sentiment analysis. In early work, before sophisticated models capable of performing detailed analyses of text data existed, sentiment analysis researchers of [41] proposed a classification model. To create the dataset, they manually labeled more than 40,000

Kazakh and Russian news to 3 classes (positive, neutral and negative). The performance evaluation of the classifier showed 86.3% accuracy for the Russian language, while the Kazakh language with 72.8% clearly indicated a relatively small training dataset and the lack of preprocessing.

In [1], the authors aimed to explore general opinions written in social media and micrblogs with a special focus on Kazakh, Russian and English. They presented results on the application of deep recurrent neural networks and bilingual word embeddings to capture semantic similarities of words from different languages. After conducting experiments on possible language pairs (e.g. Kazakh-Russian) for semantic analysis capabilities, it was found that when using 2% of the data in Kazakh, the model achieved only 58% accuracy. Given this level of accuracy, it can be inferred that the model just guesses an answer. It is also worth noting that the share of data for the Kazakh language is 80 times less than for Russian and the presented solution in the form of recurrent neural networks is not entirely suitable for tasks requiring long memory.

More recent approaches such as [35] have leveraged the transformer-based models which have demonstrated the significant advantage before more simpler model architectures for seq2seq tasks. In particular, the study presented in [50] explored the use of various versions of multilingual Bidirectional Encoder Representations from Transformers (BERT) model for sentiment classification and highlighted the superiority in capturing contextual information. Experimental results confirmed that when classifying text into negativity and positivity (polarity) with a balanced dataset, the accuracy of multilingual BERT shows 85%. Other variants of this model showed similar results.

Large language models based on a transformer can be used for different purposes both in word generation and in word representation in a latent space. [31] by training language models of different architectures for different tasks, showed that the correct approach to training and adapting models can affect their effectiveness in different scenarios. Through experiments, the author demonstrated how architectural differences, dataset volume and quality, and preprocessing methods influence the performance of

model. Announced in 2024, KazLLM was a significant step in the development of language models adapted for the Kazakh language. Unlike other multilingual models, KazLLM was trained on a large corpus of Kazakh text, taking into account the morphological and syntactic features of the language. However, the input size called context window is limited by the model specifications, while the size of aggregated document size for the sentiment analysis may exceed that limit.

To solve this problem, it was decided to use Retrieval Augmented Generation (RAG) in the stage of data preprocessing. Large language models have achieved text understanding comparable to human but still have shortcomings in applications requiring specific knowledge in a particular domain. New information appears every day and the knowledge base is updated continuously, whereas language models were trained on static data, they do not have the built-in ability to automatically update in real time. Therefore, besides a standalone LLM capable of processing texts in three languages (eg. KazLLM), there should be efforts to integrate Retrieval-Augmented Generation (RAG) techniques to enhance its performance and reliability. The author of [12] especially noted the problem of hallucinations even in models with good performance in various benchmarks. In other words, the answer may look plausible but has no bearing on the question.

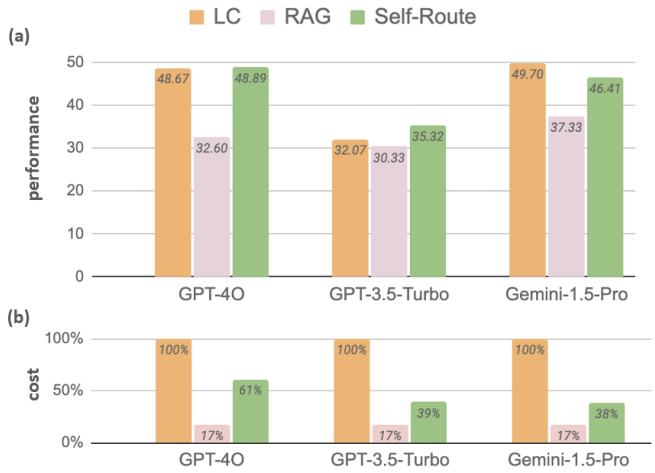

Figure 1-1: Performance and cost comparison between large context LLMs and RAG-powered simple LLMs

Moreover, the survey compiled by researchers from the work [16] emphasizes that

detection of the hallucinations in model responses complicates the integration of retrieval systems. An output response that directly influences decision-making processes may lead to the spread of false information. For example, in this proposed project, incorrect analysis of news feeds or citizens' reactions to a newly introduced law could have serious negative consequences. If the system misinterprets public sentiment, it could distort the perception of the real picture of what is happening among government agencies and analytical centers. It cannot be said that the problem with hallucinations is solved 100 percent by integrating the RAG system, since it depends on the internal features of the LLM itself. But at least the model will give answers based on external, verifiable sources, which significantly reduces the likelihood of generating completely unreliable information. In addition, if the ranking methods or selection of relevant documents do not work as intended, then the LLM will be based on irrelevant data, reducing the accuracy of the entire system. The thorough comparison between long context LLMs and RAG-powered simple LLMs [28] that the first most of time outperforms the second one in terms of general reasoning performance but at the cost of computational power. This can be seen in the Figure1-1.

Embedding models in the RAG pipelines plays a crucial role, as they serve as a foundation for document retrieval from the external sources. The success of RAG-powered systems directly depends on the quality of document representations in a large space which determines how well the model can match the embeddings constructed from an user query. Unlike the standalone LLMs heavily relying on internal memory, the RAG leverages the embedding model to retrieve contextually relevant information, improving an accuracy of the following generating stage. In the context of information retrieval, the embedding model projects user query and parsed daily news into high-dimensional vector representations. When dealing with low-resource languages (e.g Kazakh), the quality of those embeddings could be very critical. Poor representation of words or sentences lead to the retrieval of wrong documents thereby degrading the accuracy at the final part of response generation. Thus, the fine-tuning of the pre-trained multilingual embedding models on a broad dataset will help to

enhance the retrieval performance. It is important to note that the dataset on which the embedding model will be trained covers different aspects of human life, since the parsed news includes politics, economics, science and other areas.

1.2 Goals and Aims of the Current Work

Processing the huge amount of news and content created on social networks requires both human resources and time. With the current computational assistance of supercomputers, it has become possible to widely use resource-intensive language models to analyze large corpora of data. However, even language models by themselves do not possess the knowledge gained after training and requires integration of systems providing new information flows. Therefore, this proposed work intends to test how well the language model along with the RAG system is able to produce accurate and relevant answers. The extracted data and metadata from the daily news plays role of external knowledge repository. All these data will be saved as text format in the SQL database and its vector form in the vector database (e.g Pincecone, FAISS, Qdrant). Thus, the language model will have knowledge not only learned during training on various topics, but also relevant information obtained from daily updated news sources.

As was said earlier, the effectiveness of the RAG system directly depends on how well the embedding model represents words or a sentence. The selection of an appropriate embedding model that captures the semantic meaning of words across different language in the common space is very important for ensuring high quality of representations. It is important to consider that text processing involves three languages, and in particular the Kazakh language, which has the least developed support in modern language models. This creates additional challenges in processing, tokenization and vector representation of texts, requiring adaptation of models and the use of specialized data corpora. In order to solve this problem, it was decided to fine-tune models in the public domain and which support multilingualism. The best examples can be models like *E5-base*, *E5-large-instruct* and *gte-multilingual-base*. To evaluate

their retrieval performance we aim to calculate the precision, recall, accuracy and any other relevant metrics.

Chapter 2

Related works

The LLM-powered applications are limited by the knowledge gained during training on a huge corpus of data while having no idea about the information released after training. The model is capable of providing information about the history of Roman Empire but struggle to provide details about current weather. To solve this problem, the Retrieval-augmented generation (RAG) systems began to be widely used, complementing the wide functionality of language models. To enrich the responses of language models RAG retrieves relevant information from external sources or knowledge databases. This could be APIs, SQL databases, or even personal data to generate more context-aware responses. The workflow of RAG consists of two main parts: the retrieval step for fetching relevant documents from the database and the generation step for generating accurate responses. Since this project is doing text processing on data extracted from daily news, it needs to be stored somewhere efficiently. This is handled by the vector database by projecting textual data into latent space in the form of representation vectors. Words having similar semantic meanings are placed close to each other while words with different meanings are placed farther apart. So the representation power of the embedding model responsible for the conversion of texts to numerical form implicitly affects how accurate the response of LLM is. It should be noted that depending on the type of embedding model the output vector can represent the word inside the sentence or even the sentence itself. Moreover, the model architecture with high complexity potentially could lead to latency during

the inference time. Recently, much work has been done to construct a highly effective RAG pipeline for retrieval tasks. This chapter discusses the proposed methods and what are the shortcomings. The major focus is given to the sentiment analysis methods, embedding models and different modifications of RAG method.

2.1 Sentiment Analysis for daily news

The author of [52] in his paper compared different small specialized models fine-tuned on domain-specific data with pre-trained LLMs on general data. Additionally, there are three main applications of sentiment analysis, namely Unified Aspect-Based Sentiment Analysis (UABSA), Aspect Sentiment Triplet Extraction (ASTE), and Aspect Sentiment Quadruple Prediction (ASQP) [52]. The empirical experiments conducted for these tasks revealed that LLMs are still outperformed by smaller fine-tuned language models when the micro F1 score is used as the main metric. While the large models show good performance in simple sentiment analysis tasks, particularly classification, they fall short in the extraction of fine-grained sentiments. It should be noted that LLMs are initially trained on general data for general purposes and not specifically for sentiment analysis. Both large and small models inherit biases related to different prompts, making direct comparisons challenging. Another problem raised by [52] is the absence of a common benchmark. Many experiments in the field of language models do not provide comprehensive assessments or evaluate behavior in other applications, as researchers often construct models for a single task and use selectively chosen datasets. Another work describing the common methods used for the analysis of sentiment on social media classify them into 3 categories: lexicon-based, machine learning (ML) and hybrid methods [8]. The lexicon-based method does not require training data and instead relies on predefined set of words (lexicon) to classify the sentiments. However it is important to understand that even if specific words have polarity scores, they can also be used differently depending on the context. Sentences that lack such vocabulary can also express emotions or opinions. Machine learning (ML) methods, on the other hand, require training on annotated data, which allows

models to find patterns in the dataset and categorize text according to its emotional tone.

Building on the findings of [52] regarding the extraction of fine-grained human sentiments through LLMs, [43] suggests a new framework where the language model acts as both a discriminator and a generator. In this framework, one model (the generator) analyzes the provided text for sentiments, while the other model (the discriminator) evaluates the credibility of the output. Such an iterative, negotiation-based method addresses the problem raised in in the previous works, namely the limitations of a single round of reasoning in conventional LLM sentiment analysis. This traditional approach may struggle to understand ambiguous sentences, especially those with ironic or sarcastic characteristics. The proposed framework of iterative LLM models enhances accuracy by alternating their roles, ensuring the model reasons from different perspectives [27]. Experiments conducted by the researchers included ablation studies to determine how the removal of one component affects overall performance. These studies demonstrated that multi-LLM frameworks are highly sensitive to such adjustments, making them more effective than conventional LLMs in sentiment analysis tasks.

2.2 Role of Embedding Models in Textual Data Processing

With the emergence of NLP, researchers started using N-gram statistical language modeling for the prediction of word sequences and understanding patterns in the text [19]. Such simple models heavily rely on a large corpus of pre-processed data to estimate the various combinations of words, often facing problems with the capture of long-range dependencies among words. These shortcomings have prompted the creation of more sophisticated models for studying semantic relationships between words. Authors of work [32] introduced a new technique for the construction of vec-

tor spaces, which significantly improved the representation of words in a vocabulary. Previously proposed representation methods of texts have not been trained on huge amounts of data consisting of hundreds of millions of words. It became possible to represent words on several levels of similarity, not only on the basis of their proximity in the text. This approach allowed the model to take into account both syntactic and semantic relationships between words. It was found that simple algebraic operations (e.g shifting or addition) performed on the vector representation of words result in the new vector encoding the semantic attributes. The classical example "vec(king)-vec(queen) = vec(man) - vec(woman)" specifies the relationships like gender, royalty, or any other social attributes.

A two-layer neural network underlying the Word2Vec architecture was used to study the representation of texts in numerical form. For better control during inference, the hyperparameter N determines how many previous words before the current word should be used to predict the next words. It should be noted that the model inputs only numerical values and the words initially have their own one-hot encoded vector. The hidden layer following after projection layer makes up the majority of parameters. During the inference of the second part of the network, the parameters between the hidden and output layers will be removed, and the projection layer weight matrix will be used as a lookup table for the input words. However, the model accuracy depends on the context window size defined by hyperparameter N. In terms of long-range dependencies and computational cost, the proposed method is still not efficient. Therefore, the authors of the paper [33] suggested to use Recurrent Neural Network (RNN) to save all history information in one hidden/latent vector which is then used for the next word prediction. Only the weights of the recurrent network and weight matrix connecting hidden and output layers are involved.

The aforementioned methods like Word2Vec or its modified version with RNN rely on the distance between the word vectors to capture the semantic relationships. So the distance or angle in case of cosine similarity is used as the primary evaluation metric to assess the intrinsic quality of the set of vector representations. Instead of

solely focusing on the distance metrics the idea proposed by [36] takes into account the directional differences of two vectors. In simple terms, linear transformations have to be performed in order to move from one vector representation to another. Similar to the example "vec(king)-vec(queen) = vec(man) - vec(woman)" subtraction of the "queen" vector from the "king" one will have the same direction as the operation done for the words man and women. Additionally, the GloVe method in [36] based on Latent Semantic Analysis efficiently leverages the global statistics of words from the entire dataset and not only the local context as in the case of Word2Vec. Probability computations make up the majority part of the computational works because there is no neural network involved with many parameters. All collected statistical information is given as a matrix showing the occurrence frequency of specific words in the context of another. Given these tabular data, the proposed algorithm to find the interconnection between two words i and j address to the third vector k. The following steps include the calculation of the conditional probabilities of the vector k occurrence in the context of two other vectors. The process of learning vector representations exploits the fact that the conditional probability itself does not contribute enough, and their ratios are used instead. In other words, the ratio value emphasizes how strongly the probe vector k is associated with the vector i compared to the vector j. Representation power is restricted not only to the semantic connection between words but also to the relative distance. Despite that, the method is purely statistical, and in the case of long-term dependencies, it does not consider other contextual words for the generation of word embeddings.

With the advent of transformers [45] and the attention mechanism [53], there has been a significant breakthrough in natural language processing, which has led to the development of new embedding models. The attention heads are able to effectively capture dependencies between words using a larger contextual window. Unlike Word2Vec or GloVe algorithms with limited contextual words for the embedding generation the attention matrix projects the relationship of one specific word with others with the context window [45]. The simple transformer has encoder and decoder com-

ponents, both of which utilize attention heads to process the token or word sequences. Each token in the input sequence before the encoder will be mapped to a high dimensional vector and that projection unit is also learnable during training. To add positional information of words within the sequence the encoder performs integration of positional encodings to the resultant vector from the previous step. Once the input sequence is converted to numerical representation and positional encodings are added the encoder passes the pre-processed vector through the layers of multiple attention heads and dense layers to capture the structure and content of the input.

While the decoder goes through the same processes with slightly different neural network layers. [7] introduced the stacked version of the encoders capable of understanding the input sequence and its internal structure in a higher level of abstraction so that even humans can not interpret it properly. Each encoder layer refines the vector representation generated from the previous layer, progressively learning interconnections between the tokens. The transformer-based embedding model's advantage as opposed to the simple algorithms based on the statistical data is in their contextualized embeddings. The static nature of the simple representation learning algorithms cannot adapt effectively to the dynamic nature of the inputs. The context and surrounding of a specific word could change its meaning entirely. Moreover, a pre-trained BERT [7], with the addition of several layers of a fully connected network, can be further fine-tuned to perform other downstream tasks. This universality is based on a deep understanding of the semantic connections of words at a more abstract level so its projection with hidden layers just aligns for specific requirements. In other words, it is mapping of rich vector representation into some desired outputs.

Having good representation power the DistilBERT model discussed in [42] retains its original version's performance while being optimized for inference time and computational power. This makes it suitable for real-time applications and running on mobile devices. Many researchers and private companies increased prediction accuracy or any other metrics by increasing the model's complexity. The growing memory and computational requirements make it impossible to adopt the LLM-based solutions for the less powerful hardware. Thus, the idea of knowledge distillation given in [42]

compresses the experience of heavyweight model (teacher) to train the lighter model (student) in order to reproduce its capabilities. Ordinary BERT [7] acting as a teacher directs the training of DistilBERT as a student while training on the same huge data corpus. To make the training process even easier, it was decided to remove the Next Sentence Prediction (NSP) training objective and only use Masked Language Modelling (MLM). The changes also affected the architecture of the model itself, which reduced the number of parameters from 110 to 66 million, which is almost 40 percent. The above improvements allowed not only to reduce the time spent on the training process but also to optimize the inference time up to 60% maintaining the same level of accuracy. According to the evaluations of [42], the DistilBERT retains 95% of original BERT's performance with fewer number of parameters.

[29] as a variation of BERT extended it to handle multiple languages in one model and trained on Wikipedia corpora of 104 languages. Unlike monolingual BERT mainly trained for the English language, it uses the shared embedding space. This means that semantically close words, despite the difference in spelling, will be located close to each other. Such functionality allows the mBERT [29] to be used for cross-lingual transfer learning, where knowledge learned in one language can be applied to another. Moreover, the mBERT was not trained on language-specific settings or supervision and it treats all languages equally. The model can accept input in one language generalizing the underlying conceptual meaning to other languages. It is useful for tasks such as question answering and text classification. Although it is not explicitly trained for translation, mBERT embeddings can be further fine-tuned for translation tasks just by aligning word representations in shared space.

[6] addresses key challenges in multilingual representation learning, especially for low-resource languages by using various datasets and optimized training methods. Authors introduced the new term called "curse of multilinguality" after what the addition of more languages will not increase the cross-lingual performance for low-resource languages. However, this phenomenon can be alleviated by simpling increasing the model complexity or size. Compared to mBERT their proposed XLM-RoBERTa

(XLM-R) model shows 23% better accuracy results on cross-lingual classification tasks and a 5% increase in average accuracy on Cross-lingual Natural Language Inference (XNLI) from the previous state-of-the-art model. Such a significant performance increase became possible thanks to pre-training on a massive 2.5 terabyte text corpus called CommonCrawl. This is one of the largest and most diverse multilingual datasets used for training. It should be noted that the volume of documents in the Kazakh language is 6.4 GB while the overwhelming majority of texts in English with 300.8 GB. As with previous models, XLM-R adopts the BERT-based architecture, the stack of transformer encoder layers created for bidirectional contextual learning. The Masked Language Modeling (MLM) served as a primary objective for training, where random tokens in a sequence are masked and those tokens have to be predicted by the model. The training also relies on a shared vocabulary which allows the model to handle several languages efficiently. This vocabulary contains more than 250,000 tokens encompassing words, tokens, and different characters across all supported languages. It ensures that even low-resource languages are represented appropriately, although bias can exist toward the high-resource languages.

Recent advancements in pre-trained models raised natural language processing to a new level of contextual learning. Among these models, BERT introduced bidirectional contextual learning setting new benchmarks for the NLP tasks. However, subsequent studies have shown that it is possible to optimize the original BERT in terms of pre-training strategy. RoBERTa, proposed by [30], introduces modification upon the conventional BERT by removing the Next Sentence Prediction (NSP) objective, replacing the static masking pattern with a dynamic one, and optimizing the training stage. During pre-training, the first objective is the Masked Language Model (MLM). 15% percent of input sentences will be selected for possible replacement with mask tokens and only 80% of those tokens inside that list will be further replaced by special tokens. Authors of the paper [30] pay attention to static masking, where the model masks the same word in a given sequence across all epochs. Once the word is masked it remains the same whenever we use that instance. Instead of static masking, they adopted the dynamic version, meaning for the same sentence at each iteration differ-

ent masking token will be chosen. The model does not memorize the fixed positions of the masked token but instead learns to predict when it changes its mask position. According to the results of the F1 score given in [30], the dynamic masking does not add any significant contribution to overall performance, and the metric value increases by 0.4%. Additionally, [7] observed that the removal of NSP loss as a training objective will result in performance degradation, especially for benchmarks namely Multi-Genre Natural Language Inference (MNLI), Question Natural Language Inference (QNLI) and Stanford Question Answering Dataset (SQuAD). Some researchers [22, 18] in e eir works questioned the need of NSP objective. In order to test this hypothesis, several benchmark tests were conducted. It reveals that the NLP loss removal slightly improves downstream tasks or matches the results given in [7]. Another proposal is the usage of large mini-batches (2K/8K) instead of small 256 batch size. This idea is supported by observations from a scientific article [51] which demonstrates the efficiency of large batch sizes in terms of faster convergence and stable training without compromising the model performance. For example, the 256 batch size gives a 3.99 perplexity score compared to 3.68 with the 2K batch. In simple terms, the perplexity represents the uncertainty level of a model and it has to choose one word among 3.99 words.

2.3 Retrieval Augmented Generation

As mentioned in the beginning of the chapter standalone LLM is limited in functionality and knowledge when the input request requires the knowledge to appear after the model training. To address this gap in LLM-based applications RAG systems have been developed, combining the capabilities of LLMs in text processing with external sources of knowledge. The RAG integrates a retrieval component that fetches relevant documents to the query, from APIs, vector databases, or search engines.. The most conventional RAG uses this retrieved information to augment the language model's output with up-to-date and contextually appropriate answers. For real-time applications, the time allotted for query processing throughout the whole RAG pipeline is

very critical. Furthermore, the final output of the pipeline has to be not only fast enough during inference time but also reliable, accurate, and contextually relevant to the requested query. Achieving this trade-off between processing time and response accuracy requires optimization of the retrieval and generation components of the RAG pipeline.

As described in [23], LLM parameters store obtained knowledge after the training and its knowledge manipulation capabilities are limited. The question of how exactly to update the weights so that the model takes into account new knowledge about the world when making a decision remains open. [23] evaluated parametric memory represented as knowledge stored in the pre-trained neural network's weights and the non-parametric memory supported by neural knowledge retrieval. Combined memory units create an end-to-end pipeline with retrieval and generation parts. This is a so-called hybrid memory for knowledge processing. The pipeline contains a pre-trained retriever (non-parametric memory) along with a pre-trained sequence-to-sequence (seq2seq) transformer model (parametric memory) to generate responses conditioned on the retrieved information. Since the system serves as a basis for the following advanced modifications the pipeline components are pre-trained and pre-loaded to access the fresh knowledge without any training. The retrieval part is initialized with weights of Dense Passage Retrieval (DPR) which provides some kind of retrieval supervision from datasets such as Natural Questions and TriviaQA [20]. Moreover, unlike the DPR QA systems relying on additional re-ranking techniques after the retrieval of relevant documents, RAG [23] bypasses it. Such an approach not only saves computational power but also decreases the inference time.

RAG-empowered LLM pipelines shows remarkable results in generating contextually accurate responses but it still can have hallucinatory answers. Most of RAG implementations relies on the initial query for the text retrieval, overlooking the ambiguity without any clarification. Thus, authors of [3] suggested to use so-called Refine Query for Retrieval Augmented Generation (RQ-RAG) method in order to expand the functionality of the model with query decomposition and refinement components. Their experimental results shows that the integration of the refinement

method to 7B Llama2 model leads to almost 2% improvement over the previous state-of-the-art (SOTA) model in question answerting datasets. The retrieval part of the conventional RAG described in [23] have been modified to increase the clarity of the input query. This is done through iterative refinement processes like query decomposition, rewriting and context-aware refinements. For example, the query decomposition breaks down the complex sequence into the simpler sub-queiries so the model can construct comprehensive response. [3] created special dataset reflecting the behavior of the refinement process across the pipeline during inference time. It is not simple dataset like question-answer pairs but the answer refined throughout the multiple iterations of modifications. To guide the model in dataset generation, [3] uses the pre-defined token "SPECIAL" explicitly telling what kind of refinement operation to perform. Iteratively improving query results will lead to better document retrieval and ultimately more accurate inferences. As a result, evaluation of the RQ-RAG outperformed foundation models for both non-retrieval and retrieval configurations. For the latter retrieval setting the proposed model surpassed 7B Llama2 by 33.5% revealing the implicit problems that could be faced by LLMs. One of the indicators of the model's advantage compared to baseline models is evaluation on the multi-hop question-answering (QA) benchmarks. To respond the user query appropriately the model combines information from the multiple sources of evidence rather than directly finding the answer from the single source of information.

Similar work to [3], [26] addresses the problem of noise or ambiguity in the input query. It introduces multiple query-rewriting techniques acting on different levels of information. In addition, the authors of the paper further optimized the rewriting process by integrating adaptive strategy selection, which minimizes the number of refinement operations. There are variety ways of query rewriting approaches and the most of them generate single rewritten user input. The retriever part fetches redundant or similar documents, reducing the possibility of covering all relevant information. Some important knowledge will not be available to the model, which results in low recall values. People usually have difficulty mentally expressing their question,

which ultimately leads to the extraction of non-relevant information. In addition, [26] mentions the criterion for rewriting user input, namely the uniqueness of each rewritten query. Each unique query should not repeat the content of other generated queries in order to increase the diversity. To match search engine preferences, they created a Keyword Rewriting (KWR) method that retains the same amount of information. The main idea is the extraction of query keywords (nouns and subjects). By doing so, this method allows to avoid unnecessary loads on the search engine and at the same time find the necessary documents. Another type of rewriting involves reducing information to remove unnecessary details. [26] defined this approach as Core Content Extraction (CCE). If a RAG pipeline contains reranking or generating components it creates a significant load on them. Besides this there is also information expansion strategy incorporating prior knowledge to extract more diverse set of documents. Depending on the task at hand, the above rewriting strategies can be combined together to create a set of dynamic rewriting methods. When new methods appear, it is possible to simply include them in the set, which makes Diverse Multi-Query Rewriting (DMQR-RAG) framework universal and flexible. Empirical experiments done with Ambiguous Natural Questions (AmbigNQ) dataset show 68.57% of F1 score for the DMQR-RAG while RQ-RAG [3] scored 63.96% on the same metric. However, evaluations using the Multi-Hop Question Answering (HotpotQA) dataset, designed to test RAG's performance on tasks that require multiple steps to find an answer, show a slightly different picture. RQ-RAG scored the highest value 54.11% F1 surpassing DMQR-RAG by 0.12%. The benchmark AmbigNQ specifically created to test how the model handles ambiguity in question-answering tasks serves as one of best performance indicators. RAG-Fusion [37] demonstrates state-of-the-art results on that benchmark with 68.59% F1 score.

The idea of user query reformulation is further studied by [37], where the author studies methods to enhance retrieval quality by refining user queries before document retrieval. It uses reciprocal score as a fusion mechanism to combine multiple results from the reformulated queries, ensuring the retrieval of the most relevant documents. Conventional RAG pipelines integrated in the virtual assistants ranks retrieved doc-

uments by their relevance to the query. The distance between embedded vectors defines the relevance. [13] found that revaluation mechanisms play a very important role, on which the accuracy and completeness of the answer depends. The retrieval component of RAG described in [37] remain the same and provides pre-defined number of documents. But before sending queries and corresponding documents to a large language model to generate the final answer to the user input, it goes through a reranking stage. The reciprocal rank fusion (RRF) assign some score and reorder the relevant information according to those scores. One of the challenges that might be faced in RAG-Fusion is the relatively high response time compared to simple RAG pipeline. Although it is difficult to generalize accurate running time the average end-to-end pipeline evaluation shows that RAG-Fusion lagged behind the RAG by 1.77 times [37].

Chapter 3

Datasets for Embedding Models

When training embedding models various datasets can be used to learn word, sentence and even the whole document and then represent/convert them into numerical format. The open source datasets proposed in the paper consider representations at different levels of abstraction: sentence and document levels. Given the focus on embedding news articles, this work will prioritize datasets that are specifically designed for this purpose. Good examples are MS MARCO, Natural Questions, Stanford Question Answering datasets. The chapter ends with a discussion of why such high-quality datasets cannot be used for embedding news articles, and the need to create synthetic datasets instead.

3.1 Open-domain datasets

3.1.1 MS MARCO: A Human Generated MAchine Reading COmprehension

Open-domain dataset [2] contains a collection of question-answer (QA) pairs that are not restricted to a specific topic or domain. Topics can vary widely, from historical texts to social media blogs. They are designed to evaluate and train models that can answer user questions without being tied to any predefined topic. One of the main topics in the Artificial Intelligence (AI) field is to construct intelligent agents

capable of reading and understanding textual data in a way similar to humans. It is called machine reading comprehension (MRC) in NLP. The importance of this topic is growing every day as more and more people move from traditional browser-based information search to voice interfaces that support natural language dialogue through questions and answers. Such a system has to be able to extract and process unstructured data from the user input. Therefore, a large dataset is required to train neural networks with a large number of parameters that produce high accuracy.
Many of the general datasets available for MRC and QA are created synthetically rather than with human annotations. Moreover, synthetic datasets have a common property, namely, questions created by crowd workers from various types of documents and paragraphs. In order to eliminate this property, questions in MS MARCO correspond to real online queries in the Bing search engine, which comes closer to the natural diversity of the information sought. As practice shows, most texts may have grammatical errors, abbreviations, and transcription errors when it comes to speech interfaces. Huge sources of information such as Wikipedia, on the contrary, consist of high-quality texts. Authors of [2] highlight the key limitation of MRC tasks - the majority of models are trained to extract useful information from one document or text span while real-world scenarios require models to aggregate and reason across multiple textual data. In simple terms, answers to the question could be spread in different parts of different documents, and the model has to be capable of aggregating those pieces to output the comprehensive response.

The data generation process begins with the extraction of over 1 million Bing user questions with unique answers from the system's history. All fetched questions are then manually annotated with well-written responses. During the generation stage authors continuously observed the resultant answers to ensure high quality, accurate, and complete outputs. Some of the question samples are ambiguous or the formulation may not be clear due to non-standard structure of human-generated queries. For instance, the sentence "will I qualify for osap if i'm new in Canada" represent of the human behavior when a person is searching for some information and is difficult to interpret even for a human [2].

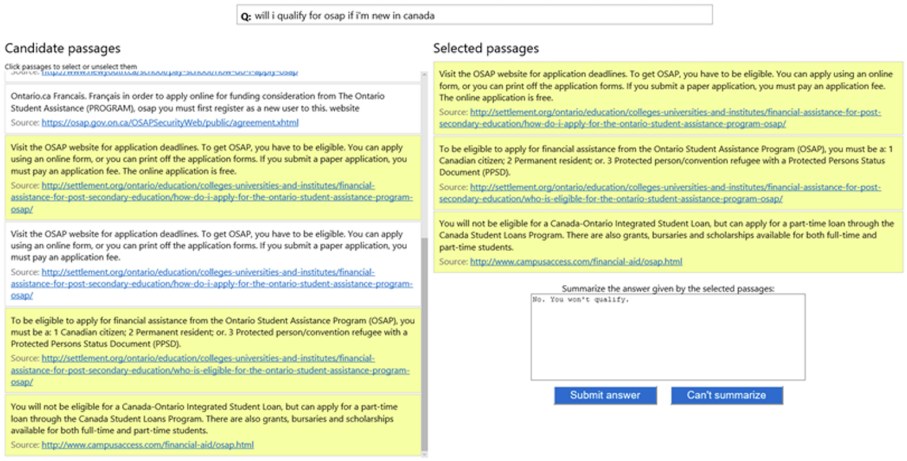

Figure 3-1: Special UI created for annotation of answer passages by human editors [2]

Thus, this dataset can serve as a good benchmark to test the performance of MRC models. The distribution analysis of the top-5 questions in the dataset shows that "What" type queries are the most frequent accounting for almost 35% while the second place is given to "How" type with 16.8%. This distribution shows that many people in search engines are looking for a more detailed explanation of various phenomena, processes and concepts.

Following this, the closed-ended questions "Yes-No" make up 7.46%, emphasizing demand for binary or decision-based questions. The rest of questions "Who", "Where", "Which" collectively contribute to the smaller yet noticeable portion of the dataset. On average each question has 10 chunks of documents that contain answers to that question. This entire extraction process is accompanied by additional editing by people and if there is no answer in any passage, it is marked as zero. It is important to mention that a passage extracted from any document includes metadata such as URL, document body, and title. During the extraction of documents from the Bing index (vector store), about 300,000 could not be extracted because the document was simply deleted or the content was changed. All questions were then labeled by a machine learning classifier into 5 categories: NUMERIC, ENTITY, LOCATION, PERSON, and DESCRIPTION [2].

3.1.2 Natural Questions (NQ)

[21] presents a large-scale question-answer dataset derived from real user queries of the Google search system. The annotator is given one question taken from anonymous sources asked to Google during a search and the top 5 pages from Wikipedia as an answer to the question. The subject matter of the CA pairs can differ dramatically from each other, making it a good benchmark for testing a trained model on natural language understanding (NLU) tasks. The authors of the article note the problem of data collection, namely the sources of the questions asked and the annotation of the answers to them. The annotation process described in [21] starts with the annotation of (the Wikipedia page, question) pair followed by the generation of the (long answer, short answer) pair. An example of annotation is given in Figure 3-2.

Example 1
Question: what color was john wilkes booth's hair
Wikipedia Page: John_Wilkes_Booth
Long answer: Some critics called Booth "the handsomest man in America" and a "natural genius", and noted his having an "astonishing memory"; others were mixed in their estimation of his acting. He stood 5 feet 8 inches (1.73 m) tall, had jet-black hair , and was lean and athletic. Noted Civil War reporter George Alfred Townsend described him as a "muscular, perfect man" with "curling hair, like a Corinthian capital".
Short answer: jet-black

Figure 3-2: Annotation example from the NQ dataset [21]

When the answer does not present or information is spread across multiple passages in Wikipedia page then the annotator labels the long answer (l) as NULL. This decision directly affects the short answer due to hierarchical dependency between them. If there exists the long answer then the short answer is derived from the long. In case of absence of the long answer no short answer is assigned. The annotation process is performed through a dedicated interface by 50 human annotators, taking an average of 80 seconds for each question. After careful annotation, the number of samples for each stage of development is as follows.

- 307,373 training samples - exposes model with wide variety of question types and answer formats

- 7842 development samples - used for fine-tuning and optimization purposes
- 7830 testing samples - hold-out part for evaluation of the trained models on unseen data (generalization)

The input data to a model is the training question with a corresponding Wikipedia page. In parallel to the input data, targets are also supplied in the form of a long paragraph containing the answer or a short one if there is a long one. Such kind of task mimics the real-world QA chatbot systems where the model must read, process and respond to the provided question. Moreover, the model must not only produce correct answers but also determine the presence of an answer. This approach help to prevent the generation of irrelevant responses and mitigating problems connected with false information (hallucination).

Human evaluation of the quality of the dataset annotations showed that for long questions the F1 metric was 87%, while for short questions the value dropped to 76%. These values show that even if the annotation is done well by people, the complexity of the task of extracting answers from context remains high, especially for short ones. In addition to the aforementioned metrics, the proposed NQ dataset's development and testing samples have 5-way annotations. So, every sample in that set have 5 annotations from different human editors. This multi-annotator approach provides with various interpretations and valid answers resulting in more comprehensive evaluation framework.

3.1.3 TriviaQA

The TriviaQA is another large-scale question-answering (QA) dataset designed to evaluate the machine learning comprehension models for complex tasks [17]. Those tasks could be difficult for a number of reasons. The first of these is that questions can be compositional and complex in their semantic structure. The answer to the question will not be obvious, but will require the model to combine facts from different parts of the documents. It happens that the model must be able to extract relevant pieces of information scattered in different time chronologies. Existing QA

dataset including previously mentioned MS MARCO [2] dataset is well suited for testing models in different conditions but still have some limitations. For example, the WikiQA [48] dataset is not large enough compared to MS MARCO or even have multiple evidence sources for one question.

The word “evidence” in context of the question-answering simply implies the supporting text passage or the piece of a document that contain needed information to answer the question. Overall, the number of question-answer-evidence triplet in the TriviaQA make up about 650,000 samples. They are obtained from a combination of 95,000 trivia question-answer pairs and referencing an average of six supporting documents. The dataset structure allow models to better generalize unlike QA dataset like SQuAD (Stanford Question Answering Dataset) that consist of questions having direct answer in one passage. In other words, this benchmark tests the model to see how effectively it copes with retrieval and synthesizing information from relevant documents.

Property	Example annotation	Statistics
Avg. entities / question	Which politician won the **Nobel Peace Prize** in 2009?	1.77 per question
Fine grained answer type	What **fragrant essential oil** is obtained from Damask Rose?	73.5% of questions
Coarse grained answer type	**Who** won the Nobel Peace Prize in 2009?	15.5% of questions
Time frame	What was photographed for the first time in **October 1959**	34% of questions
Comparisons	What is the appropriate name of the **largest** type of frog?	9% of questions

Figure 3-3: The majority part of annotated questions contain multiple entities [17]

As a source of questions, 14 quis league platforms were selected, while questions with less than four tokens were filtered out. To ensure that the model can find an answer from the corpus of documents, the researchers of [17] extracted evidence from 2 main sources: web-based api with the top retrieved 50 URLs and Wikipedia-based sources. The final collection of evidence sources includes news articles, social media blogs and many other document types. According to statistics given in [17] the average length of a question is 14 words while the average document length is equal to 2895 words. Further analysis of question properties shows that 73.5% of them contain special descriptive phrases (Which, Who, What). In the example "What chemical element is represented by the symbol "Au"?" the answer can be extracted directly without any complex linguistic processing. These kinds of questions fall into

the fine-grained category. Getting an answer to the question "Who painted the Mona Lisa?" will require more reasoning capabilities from the model. The share of such questions is about 15.5%.

3.2 Closed-domain datasets

3.2.1 SQuAD (Stanford Question Answering Dataset)

In addition to the open data sets discussed earlier, there are also closed-domain ones. They represent a specialized corpus of texts in which a sample of question-answer pairs is limited to a specific subject area. Unlike open-domain datasets TriviaQA [17] or MS MARCO [11] which cover general knowledge, the closed ones require from the model the expert level knowledge about some area. Researchers of [38] noted that most of the datasets for Reading Comprehension (RC) tasks have 2 disadvantages. The first is that high-quality data, whether it is an image or text, is too small in quantity for complex models [40]. On the other hand, those that have an advantage in the number of samples are half synthetic and do not have much in common with real RC questions [15]. Thus, [38] suggests high quality and a large amount of textual data called SQuAD (Stanford Question Answering Dataset) with publicly available pre-trained models.

The process of creating a dataset begins with annotations, which are given a segment of some text or document from Wikipedia, and each question is compiled based on them. That is, for each manually composed question the target will be a segment of text containing the answer. The total number of question-answer pairs is about 107,000 extracted from more than 500 Wikipedia articles. Compared to the previously labeled MCTest dataset [40], it is more than 100 times larger in size. One of the features of the proposed dataset is that it poses different challenges to the trained models. Models with previous datasets were offered to choose an answer from several options, but now SQuAD removes this option, complicating the work of neural networks. Instead, the model is required to extract the answer directly from the doc-

ument by selecting the correct piece of text from among several possible candidates. Despite being extractive, the dataset includes a variety of questions that can be descriptive (what?), temporal (when?), or even causal (why?). This way, samples are more closely related to real-world use cases. Another innovation was the division of edited questions by difficulty using dependency tree analysis [38]. The entire analysis is automated through a special interface within which a logistic regression model is implemented. Syntactic complexity, structure, and variation in answer types are taken into account to determine the difficulty of a question. The "When was the Eiffel Tower built?" question has a direct answer without any complicated reasoning while "Why was the Eiffel Tower built?" needs an understanding of sentence structure and relationships between words.

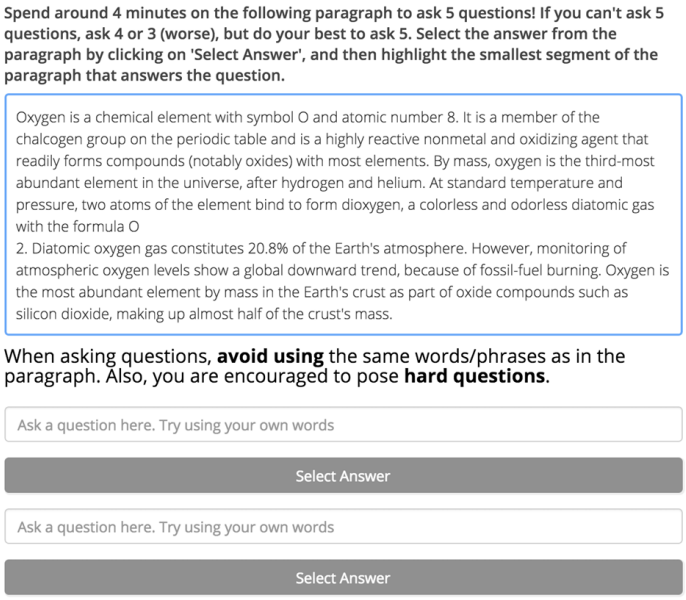

Figure 3-4: Interface for annotators to create questions from the given passage [38]

Evaluations for some types of models representing different approaches shows significant variations in their performance. A sliding window baseline using keyword matching to find relevant text spans shows the worst result with 20% F1 score. In contrast, a logistic regression model incorporating more lexical and syntactical information achieves 51%. Further development of the application of neural networks in text data processing has significantly surpassed previous methods by almost 20% [46].

Despite these improvements, the upper bound is still held by annotators with an F1 score of 86.8%, highlighting the gap between machine learning models and human capability in comprehension.

3.2.2 HotpotQA

QA benchmarks provide an objective evaluation of the model's ability to analyze, understand, and reason over a long sequence of texts. However, most are related to so-called single-hop reasoning, where the answer can be extracted from one passage. They don't perform a sufficient stress test on the system's ability to handle questions requiring multi-hop reasoning. In other words, the benchmark tests how well a model can synthesize information from different sources, thereby simulating human critical thinking capabilities. The previous SQuAD dataset [38] questions are created to be processed given one paragraph, and the majority of them can be answered just by mapping a question to a sentence. Moreover, the existing multi-hop benchmarks like QAngaroo [47] rely heavily on knowledge bases (Wikipedia, Freebase) having a fixed structure. Since the bases store knowledge in a fixed way, the possible resulting questions are predictable. Compared to human-generated ones, there are no such large variations in wording, and as a result, the model will have a poor understanding of natural language in real-world scenarios. Questions based on factual knowledge will outweigh questions that require deep reasoning, contextual understanding, or inference beyond simple fact retrieval. Third, the distant supervision approach provided by these datasets shows the model its correct answer but not the reasoning step that led to it.

As a result, the trained model will guess the reasoning path, often relying on statistical patterns instead of logical inference. To solve the aforementioned problems, the paper [49] suggests a new large-scale dataset, HotpotQA. The dataset has the following core properties: it is not limited to some rigid knowledge base, provides explainable reasoning steps, and contains questions requiring multi-hop reasoning. Scientific articles from Wikipedia served as a source of information. The crowd workers were then provided with several pieces of supporting context and asked to construct

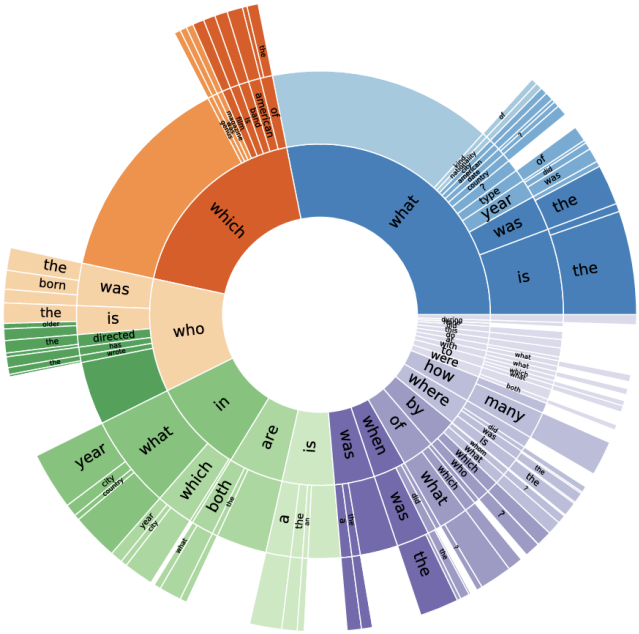

Figure 3-5: Types of sample questions from the HotpotQA dataset [49]

several questions based on them. The idea is that the answer will not be in a specific document but will require the model to collect pieces of the answer from different parts and connect them in the right way. In addition to the answer, supporting context or a document is also attached. According to Figure 3-5, the dataset includes a variety of question types centered around locations, descriptions, dates, and entities.

To see how well the reimplemented baseline model of [5] performs on the multi-hop QA dataset. In cross-validation testing, the model could answer 60% of questions with a high level of confidence. Later, these correctly answered samples, numbering just over 56,000, were classified as medium training data. It helps to refine the model on moderately complex reasoning tasks. Other correctly answered questions with simple models having a small number of trainable parameters are marked as train easy. After classifying parts of light and medium complexity, the remaining samples for which the model is unable to give a convincing answer become a train-hard class. The most complex training data forces the model to think more broadly and find logical connections between disparate facts. As a result, the number of questions with difficult samples is 15661, while the medium level is approximately 56800.

3.3 Synthetically generated dataset

The above-mentioned datasets have a common limitation, namely the lack or small amount of data in the Kazakh language for training the embedding model. This constraint poses a significant challenge since deep learning methods heavily relies on large-scale and diverse dataset to capture language representations. Even they are widely used for training and evaluation purposes on tasks such as classification, machine reading comprehension, question answering, or retrieval, the English-language content still outnumbers all others combined. After fine tuning, it is necessary to somehow check the performance of the embedding model for processing Kazakh text, respectively, a test dataset or an open source benchmark that does not exist will be required. So, the best choice would be the creation of own synthetic dataset, split for training and evaluation. A synthetic dataset based on high-quality text, edited by editors and verified by human editors, can be a very valuable tool for training models, especially in the context of the Kazakh language. The raw data is parsed data from news feeds in three languages: English, Kazakh, and Russian. The total number of news items for the period September 11 to December 12 (recent 3 months) was 37,255, including all metadata. The metadata includes the following fields:

- Object ID - the unique identifier automatically generated inside the MongoDB database
- Article URL - link to the source from which the information was removed
- Article Title - a brief description of a daily news
- Article Body - the main part of the news with all textual and visual descriptions
- Scrapped and publication date - date of parsing and day of publication of the article

To address this challenge, synthetic datasets have emerged as a powerful solution according to the specific task. Since pre-trained embedding models already have extensive knowledge of natural language processing, specific tasks will require additional

training (fine-tuning) on a specific dataset. For example, the XLM-R [6] model is pre-trained on a large corpus of textual data where the share of the Kazakh language is only 6.4 GB while the largest share is made up of texts in English with a volume of 300 GB. As a rule, the volume of data for fine-tuning relative to the dataset for pre-training is several times smaller and only 2-5 epochs will be required.

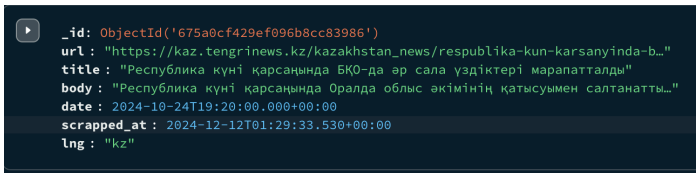

Figure 3-6: Raw data from the data parsing used for the synthetic dataset creation

It is important to mention that not all pre-trained embedding models support the Kazakh language, even if they are multilingual models. This problem arises due to the lack of qualitative training data during the pre-training phase and, as a result, the poor representation of words in latent space. The final dataset on which the embedding model will be trained has the following characteristics:

- 83998 question-answer pairs used for fine-tuning of selected models
- 3000 pairs are reserved for evaluation purposes after the training

Figure 3-7: The internal structure of the custom dataset

The internal structure of synthetic dataset is organized into 3 parts: queries, corpus, and relevant documents. An example part of the dataset is shown in Figure 3-7. All information about the texts is encoded with unique identifiers, and in the database, it will point to a specific piece of information. All these manipulations

with texts are done to effectively save data and also return it when necessary. This can be seen in Figure 3-7, where all questions and answers have their unique ID. The "queries" key in the dictionary represents a mapping between ID and its corresponding textual description with a question mark. Next comes the data "corpus," also with its own ID numbers and text information received from the body of scrapped daily news containing an answer to the questions. The last part of the dataset, "relevant docs" represents the crucial component that establishes a connection between the "corpus" and "queries". To be more precise, it functions as the mapping from the question ID (found in the queries section) to the document ID (corpus section), indicating the source of information from which it was generated. To compile a synthetic dataset, the ChatGPT API was used, which made it possible to significantly simplify and speed up the process of generating text data. The data generation itself took about 40 hours, with 43,000 requests to the API service.

Chapter 4

Methodology

4.1 Foundation of proposed approach

Given that the research aims to process a large corpus of textual data for sentiment analysis and extraction of important insights from daily news. The classical machine learning (ML) methods described in [25, 24, 9] have a limited contextual understanding of a large volume of text and problem with ambiguous words. Moreover, these models result in shallow word representations that fail to capture the deeper semantic relationships between phrases or concepts. As the size and complexity of the data grows, ML algorithms such as Support Vector Machines (SVM) and Naive Bayes will require additional manual feature extraction, making text analysis even more time-consuming. In contrast, RAG-based LLM uses deep learning methods to discover long-term dependencies between sentences, contextual meaning, and semantic relationships in vector space in ways that classical machine learning algorithms cannot. It is also important to understand that the final goal of the project is integration with systems that require real-time text processing with contexts from external knowledge sources (e.g Vector stores, SQL databases, APIs, tools).

4.2 News feed parsing

This project apply a comprehensive methodology to analyze daily news on government websites, with a focus on extracting relevant information for further analysis. The data collection process involves the usage of web scraping tools like BeatifulSoup or Scrapy capable of efficiently extracting large volumes of textual data. These tools are designed to handle dynamic web content including the date of publication, the author of an article, and other useful metadata from the structured HTML script. To ensure high diversity in the data set and comprehensive coverage of aspects of life in a given country the special configuration file is written to target reputable websites in three languages. Additionally, the scrapping process can be easily incorporated with an automated scheduling mechanism to facilitate daily news updates for the Retrieval Augmented Generation (RAG) pipeline. This collected data can serve a dual purpose within the proposed methodology. The first application is the integration of an SQL database of parsed data with the pipeline, which retrieves relevant articles from the database according to user queries. It is important to note that the received data is stored both in text format in the SQL database and also in compressed vector form. In this context, the scrapped daily news is indexed into the vector stores to enable faster search capabilities for real-time applications. The second application involves using this data to create synthetic dataset for training and evaluating embedding models. All vector encoding and decoding operations in the system must be performed using the same embedding model. If this rule is not followed, the pipeline will simply not process requests and will display an error.

In the original version of the news parser, the code was hardcoded, that is, for each site, its own unique function was created. This is done to match the structure of the site and a function written for one site will not work for another. Therefore, when adding new sites, it is necessary to add a new function on top of the existing ones. The hard-coded approach, while initially showing a working scheme of parsing for specific websites presents significant drawbacks in terms of scalability and maintainability. As the number of government websites or social media channels increases,

this will eventually lead to code redundancy and an increased likelihood of errors when managing multiple functions for parsing. Moreover, any structural change in the website's HTML document requires manual updates of the function code which reduces the system's adaptability to a dynamic environment.

To address this challenge, the daily news parser has been redesigned to be more flexible and modular. Instead of writing separate functions for each website, the newly refactored code does the same work but with common scrapping, link collector, and metadata extractor functions. These functions apply to the website-specific parameters written in JSON config files, describing the structure of a website. This approach allows the main parser to remain unchanged while enabling seamless integration of new websites by simply adding a definition to the configuration file. The abstraction of website-specific details into the configuration file makes further development easier and there will be no need to change the main structure of the parser all the time. In addition, in case of unexpected changes from the website, the registration and error handling mechanisms skip the current site, thereby preventing the breakdown of the entire system. That is, the parser will simply continue its function while notifying about the error. It is still worth noting that when writing the structure of a site for a parser, it must follow the logic of the functions created for all sites. Otherwise, the inconsistency in the configuration file may lead to incorrect data extraction, requiring manual debugging.

Most of the publications on news sites are given through pagination to display hundreds of pages in an efficient and compact way. This method breaks down a large set of news into smaller pieces rather than dumping all information at once. In addition to traditional pagination, there is infinite scrolling, which is widely used on news sites, social networks, and blogs. This type of content loading automatically loads new articles when the user scrolls down the page, instead of clicking on the "Next page" or "Load more" button. Parsing this kind of scrolling causes difficulties because the site's HTML document changes dynamically. When collecting data, automatically loading a web page complicates the process and does not allow for proper

tracking of changes. Compared to more static pagination, infinite scrolling requires more advanced techniques of data extraction. With infinite scrolling, the developer will need to intercept Ajax requests that return new content or simulate user interaction through an automatic tool such as Playwright or Selenium.

As was said earlier, the structure of sites can differ dramatically and it is necessary to manually enter the path to the metadata in the configuration file. The news database are created under the assumption that date of publication of news is normalized and brought to standard format. In the process of standardization, the languages in which they are written, various variants of the ending of the names of months and the structure of the text storing information about the date are taken into account. All this transformation allows to avoid unnecessary errors during sorting and other types of operations in the database. It is possible to write a parser that creates a function for each site separately, which was basically used in earlier versions of the parser. Instead, it was decided to use a *dataparser* library capable of converting a text date into a standard digital form. For example, the sentence containing publication date "By Staff Report in Tourism on 22 February 2025" is automatically converted to "2025-02-22T00:00:00.000+00:00" with information accurate to the hour (if it presents). Moreover, another useful feature of the given library is the support of almost any date format, whether it is an absolute date (e.g "15 October 2024") or a relative date (e.g "tomorrow, two weeks ago").

Similar to the most of programming language parsers, the data parsing stage consist of two parts: lexer and syntactic analyzer. The lexer or lexical analyzer as the early stage of the parser converts a stream of raw characters into the sequence of tokens. These tokens represent the smallest possible element bearing some meaningful information such as identifiers, symbols, literals and so on. The next following stage after the lexer is the syntactic analyzer. It takes the generated by lexer sequence of tokens and analyzes them according to pre-defined rules/grammar of some language. This is done by construction of abstract syntax tree (AST) which makes the hierarchical structure of the source code for subsequent downstream stages. Compared to the usual brute force parsing of a site, which checks each line for a given pattern, the tree

construction method allows to find the necessary piece of information much faster and more efficiently. Essentially, the proposed parser creates an XML tree from an HTML script to find specific page elements that store the necessary metadata. All of those collected metadata coming with news main body will be then stored in SQL database for dataset generation or for retrieval purposes in RAG pipeline.

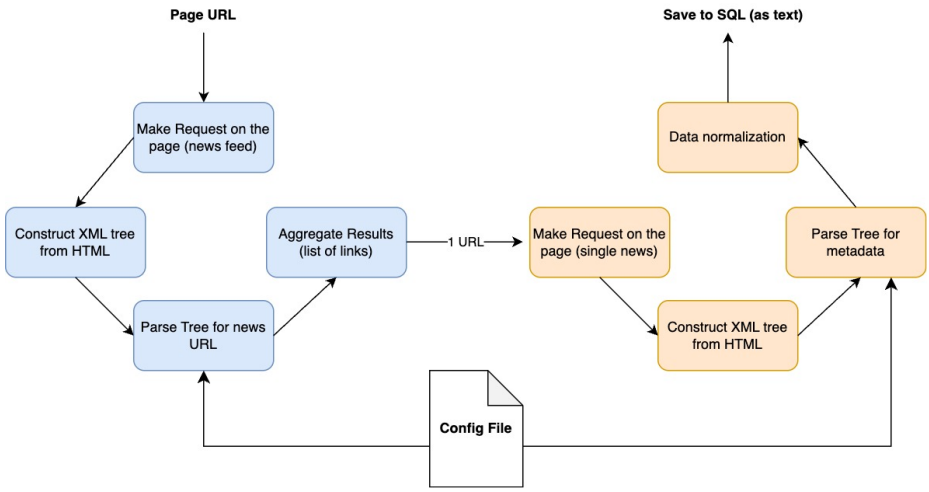

Figure 4-1: Workflow of site parser

As shown in the figure, the parser workflow first receives a URL of a web page containing links to each news item. To efficiently and quickly scan a page for the required elements, the next step is to build an XML tree from the HTML document. This approach allows us to build a more convenient data structure, which greatly simplifies the application of XPath and CSS selectors for faster extraction of news's main body along with its metadata. This element is the key to extract links to articles and is unique for each site. Having access to all the necessary news pages, all of this is then collected into one list for the subsequent stage of data extraction for a specific piece of news. Next, the resulted list is passed through the information extraction pipeline, where each individual web page is processed in more comprehensive way. The same method used for the news feed can be applied when constructing a tree, enabling efficient identification of key elements of a page, such as the article title, publication date, author, and body text. It is worth noting that the parsing of the required element in both cases occurs using a configuration file showing which traversal path

needs to be taken to reach this element. At the final point, extra characters, special symbols are removed from the parsed texts, and in particular, the publication dates are converted to a standard format. All collected news and metadata will be saved to the database.

4.3 Fine-tuning of Embedding Models

Before fine-tuning the embedding model, it is essential to clarify the dataset, model architecture, and loss function to ensure effective training. Depending on its architecture, training dataset, and target domain, the model can be used for many different types of tasks and objectives. For example, the QA datasets mentioned in the section "Dataset" focused on extractive question answering (e.g., SQuAD) or generative question answering (e.g., MS MARCO, Natural Questions), where the model learns to either extract an answer from a given passage or generate a response based on retrieved information. This project, similar to these datasets, considered the generation of a synthetic dataset that included samples in three languages, not just English, as in most.

The overwhelming part of embedding models' backbone constitutes the Bidirectional Encoder Representations from Transformers (BERT), with various modifications, new training objectives, and optimizations to improve efficiency, generalization, and performance in different tasks. According to [34] Massive Text Embedding Benchmark (MTEB) benchmark, the multilingual-e5 models are ranked at the top of the list across 9 task types (e.g summarization, clustering, reranking, retrieval). It is worth mentioning the reason why the BERT model serves as the base model for all models used in embedding. Unlike the earlier static word representations obtained from the Word2Vec [32] or GloVe [36], BERT provides the deep contextualized representation for words or even sentences. It fully attends to all tokens bidirectionally, thus capturing long-range dependencies among the elements.

The foundation BERT model is initially pre-trained on the Masked Language Modeling (MLM) and the Next Sentence Prediction (NSP) objectives to obtain general

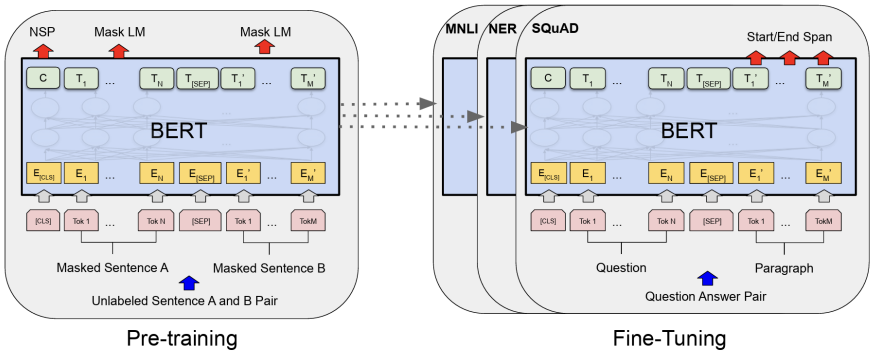

Figure 4-2: Fine-tuning and pre-training process of BERT model serving as a base for the most part of existing embedding models

knowledge in various domains. After large training on general data, the downstream tasks like question-answers (QA) will require further training (fine-tuning) . Since the parsed data contains high-quality text verified by the editors of the news sites on extensive topics, it was decided to use it as a source for creating a synthetic dataset. The figure 4-2 shows overall schema of the fine-tuning procedure. In general practice, in addition to the stack of encoders, the transformer adds layers of neural networks so that they produce answers that correspond to the task. The structure of the synthetic dataset consists of questions with their unique ID, a part of the text containing the answer to this question (also with an ID) and mapping between two ID numbers. More detailed explanation can be found in the section 3.3.
It is important to understand that language models are limited by context windows and therefore before generating a dataset, it will be necessary to divide the text passages into smaller chunks. This ensures the coherence and fluency in the chunked text, meaning that sentences should be arbitrarily cut off somewhere in the middle of story. At worst, such partitioning technique will lead to lost and fragmented information.

More recent embedding models project the text input into a fixed latent space, where semantically similar texts are matched to closer points [39]. To come up with static embedding space, the average pooling layer is connected to the output of main model (e.g BERT). The dimension of the transformed dense vector can vary from 128

Structure of data	**Dataset Examples**	**Loss Functions/Objectives**
Sentence pair and its similarity score	SNLI, MNLI, STS-B	SoftMax Loss, CosineSimilarity Loss, Constrastive Loss
Triplet format (anchor, positive and negative samples)	DPR, MS MARCO	Triplet Loss
Query-Document	BEIR, NQ	Multiple Negative Ranking Loss (MNRL)

Table 4.1: The structure of data in the dataset used for Sentence Transformer training

to 8000, depending on the complexity of the problem. The larger the vector size, the greater the memory footprint and computational cost for storage, retrieval, and processing. Therefore, the trade-off between the representative power of the implementation model and the retrieval process of the RAG system requires careful adjustment. The document size is not too big so we decided to set 1024 as a boundary and check the functionality of models with sizes not exceeding this limit. For the smallest model *intfloat/multilingual-e5-small* this is 384 while the *intfloat/multilingual-e5-large* or *intfloat/multilingual-e5-large-instruct* have the highest possible dimension (1024).

The Multiple Negative Ranking Loss (MNRL) is defined as:

$$Loss = \sum_{i=1}^{P} \sum_{j=1}^{N} \max(0, f(q, p_i) - f(q, n_j) + margin)$$

where:

- P – Number of positive samples.
- N – Number of negative samples.
- q – Query or input sample.
- p_i – The i^{th} positive sample associated with the query.
- n_j – The j^{th} negative sample, which is considered incorrect for the query.

- f – A similarity function that measures the relationship between the query and a given sample.

- $margin$ – A hyperparameter that defines the required separation between positive and negative samples.

To train the given models on the QA task, the Multiple Negative Ranking Loss (MNRL) is used to further optimize the model by ensuring that positive samples are ranked higher than the negative ones in a batch. [14]. Other loss functions and their corresponding application are given in Table 4.1. All of them have a common goal to bring words with similar semantic meanings as close as possible and to push away vectors of words with different meanings. This approach resemble with the idea implemented in the Simple Framework for Contrastive Learning of Visual Representations (SimCLR) paper [4], where the goal is to make closer word representations sharing the similar semantics.

The NT-Xent Loss used in SimCLR is given by the following equation [4]:

$$L(i,j) = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

- z_i and z_j are the vector representations of the anchor (e.g user query) i and the positive sample j, respectively.

- $\text{sim}(z_i, z_j)$ is the similarity function (usually cosine similarity):

$$\text{sim}(z_i, z_j) = \frac{z_i \cdot z_j}{\|z_i\| \|z_j\|}$$

- τ is the temperature hyperparameter.

- $1_{[k \neq i]}$ is the indicator function that takes the value 1 if $k \neq i$, and 0 otherwise. The anchor samples are excluded from calculation

- $2N$ is the total number of samples in the batch.

The embedding model includes multiple negative samples, which provide a richer context by improving the model's ability to distinguish between relevant and irrelevant samples. Moreover, negative samples prevents model from overfitting to some specific positive point. It has some degree of regularization effect during the training stage.

Chapter 5

Results and Discussion

This chapter presents the key findings of the study, which analyzes the performance of embedding models in the sentiment analysis of daily news. The results are structured around the main evaluation metrics such as MRR, accuracy, precision, recall, etc. We analyze the impact of model fine-tuning on a synthetically generated QA dataset and then compare with the same pre-trained baseline models. Baseline models trained on more common objectives such as classification and semantic similarity may differ from instruct models in their ability to generalize across diverse tasks, adapt to specific prompts, and capture contextual nuances. This distinction can lead to variance in performance.

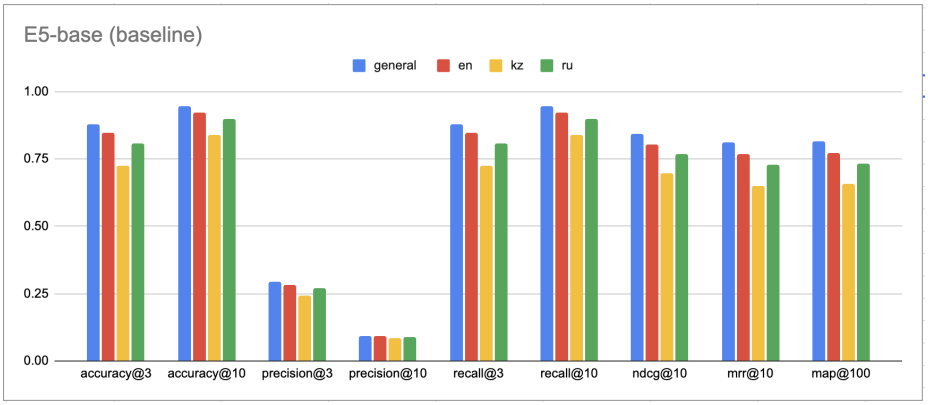

Figure 5-1: Baseline *intfloat/multilingual-e5-base* model's performance on test dataset

As shown in Figure 5-1, the results demonstrate evaluation metrics for the *multilingual-e5-base* baseline model on four test datasets. The test datasets are the same format

as the training datasets, but the model did not see them during training in order to study how well the model copes with generalizations.

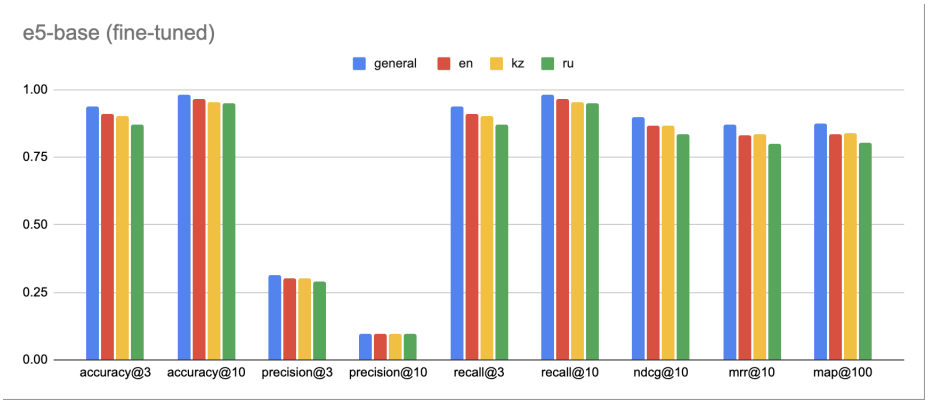

Figure 5-2: Fine-tuned *intfloat/multilingual-e5-base* model's performance on test dataset

On other hand, Figure 5-2 shows results of fine-tuned *multilingual-e5-base* and it outperforms the baseline version across all key metrics. It demonstrates noticeable improvements in accuracy, recall, precision and ranking quality. Accuracy@1 metric incresed from from 73.04% to 79.86% (+6.82%) indicating correctly identified top-ranked prediction, meanwhile Recall@10 jumped from 94.6% to 97.97% (+3.37%). It should be noted that recall measures the ratio between the number of retrieved relevant items and total number of relevant items.

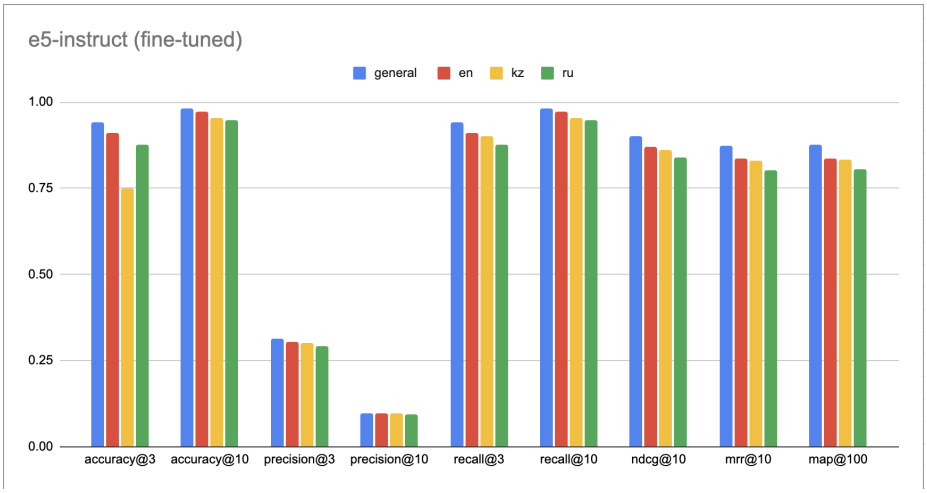

Figure 5-3: Fine-tuned *intfloat/multilingual-e5-large-instruct* model's performance on test dataset

Other metrics such as NDCG@10 and MRR@10 responsible for ranking the quality and relevance of retrieved documents also reveal an increase of 6-7%. Upon closer examination of the graph, the fine-tuning of the model on the dataset where the texts in the Kazakh language are more prevalent, then almost all metrics have improved by 5-7% on average. As if the indicators for the Kazakh language were almost equal to those of other languages. The same trend is observed for other languages, but with an increase of 2-3%, which is less than that of the Kazakh language.

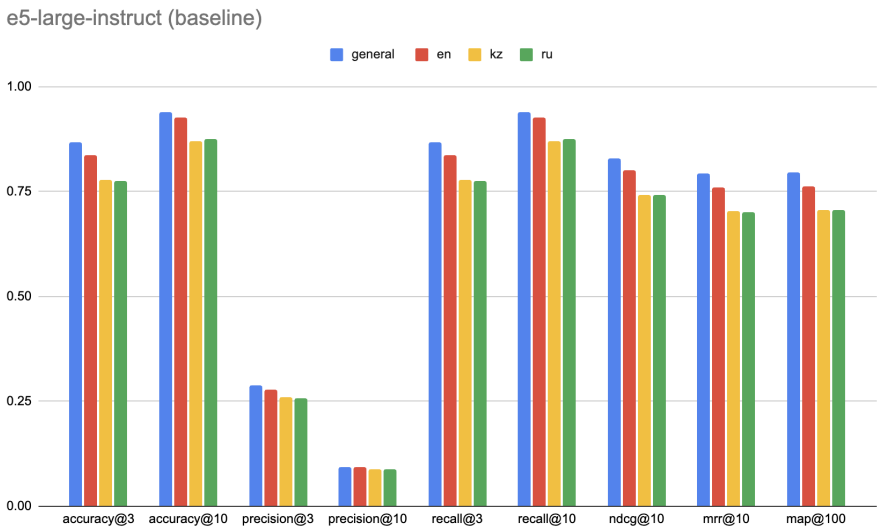

Figure 5-4: Baseline *intfloat/multilingual-e5-large-instruct* model's performance on test dataset

The results given in Figure 5-2 and Figure 5-3 indicate that after fine-tuning, there is no significant difference between the large and base models in terms of retrieval performance. The improvement in metrics remain in the 0.5-1.5% margin. This could imply that the base model, once fine-tuned, captures most of necessary task-specific information, making additional number of parameters less impactful. It's easy to see how much Precision lags behind other metrics. Regardless of model complexity and pre-training objective, the precision value for the top 3 (Precision@3) and 10 (Precision@10) continue to decrease. As the retrieval list of documents gets larger, the RAG system with embedding model includes less relevant items. The same trend is observed even in the baseline models without any further training. We have tested

on three languages along with mixed dataset and English as the main language show poor results. This indicates that the root cause of this problem does not stem from the quality of the data set or the imbalance within it. One of the factors that can lead to such a result is the number of test samples, the number of which for each language is about 3000. While in large benchmarks like MTEB only for retrieval task there are 15 datasets. In this study we have created our own small benchmark because there are no available benchmarks for evaluating fine-tuned embedding models in Kazakh. Meanwhile, benchmarks are available for English and Russian texts, we decided to develop own test dataset for the mentioned languages to ensure consistency with the Kazakh dataset.

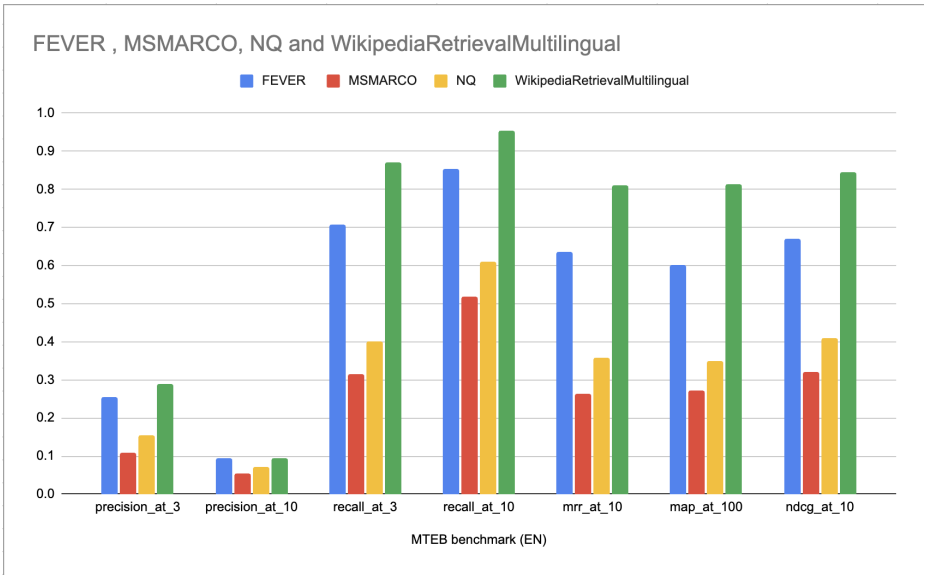

Figure 5-5: MTEB benchmark evaluation for English language, fine-tuned *intfloat/multilingual-e5-base*

As shown in Figure 5-5 reveals performance of four benchmark: Fact Extraction and Verification (FEVER), Microsoft Machine Reading Comprehension (MSMARCO), Natural Questions (NQ) and WikipediaRetrievalMultinlingual. The problem of low precision remains relevant. There is certainly a noticeable difference between Figures 5-5 and Figure 5-3 in terms of their performance across the evaluation metrics. This is especially true for benchmarks MSMARCO and NQ. If such figures are understandable for a NQ benchmark that is more focused on question answering

tasks, the case with MSMARCO is bit different. The latter benchmark evaluates how well an embedding model retrieve and rank relevant documents from external data source. Most likely this may happen because MSMARCO contains diverse set of real-world queries and involve highly ambiguous user questions.

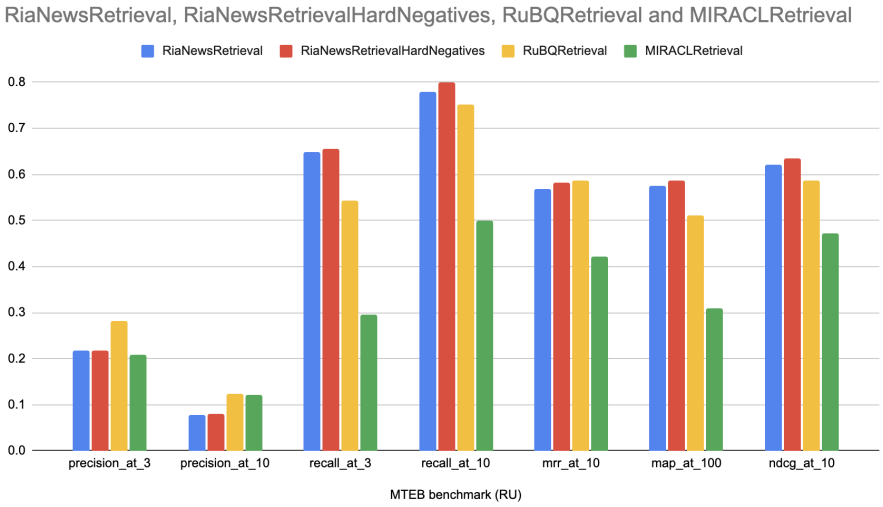

Figure 5-6: MTEB benchmark evaluation for Russian language, fine-tuned *intfloat/multilingual-e5-base*

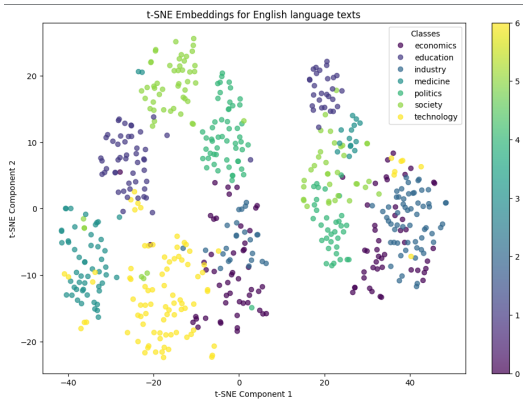

Figure 5-7: t-SNE for English word embeddings obtained from baseline *intfloat/multilingual-e5-base*

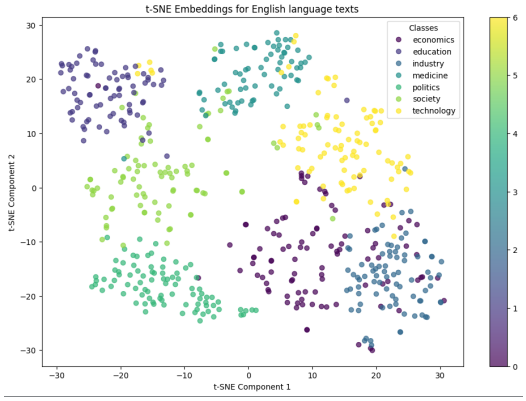

Figure 5-8: t-SNE for English word embeddings obtained from finetuned *intfloat/multilingual-e5-base*

The plots given in Figures 5-7 and 5-8 how different classes (economics, society, technology and etc) are distributed in a latent space. To test how well the embedding model embeds sentence vectors, a static method for projections, such as t-SNE

(t-Distributed Stochastic Neighbor Embedding) was used. This method is very effective in compression of high-dimensional vectors into 2 or 3 dimensional space while preserving the greatest variations and structure between data points. In a good embedding space, the sentences of the same category should form a distinct cluster and placed close to each other. For English words, clustering with the baseline model works well because it is clear how they are separated even if there are small overlaps. After fine tuning, it is clear that these overlaps have become a little smaller. This is most likely due to the fact that the pre-trained model was already trained on huge amounts of English data, which allowed it to achieve good results. The same trend can be seen for Kazakh sentences shown in Figures 5-10 and 5-9. When compared with English word clusters, the overlap in Kazakh language clusters is noticeably greater.

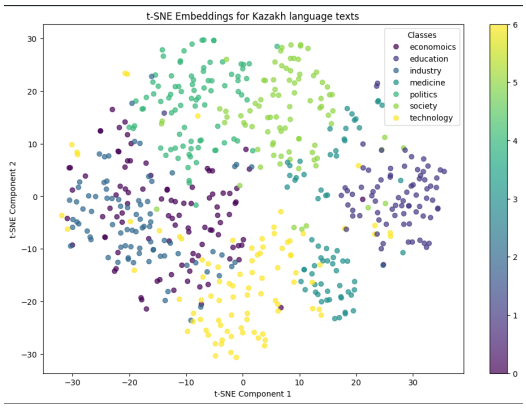

Figure 5-9: t-SNE for Kazakh word embeddings obtained from baseline *intfloat/multilingual-e5-base*

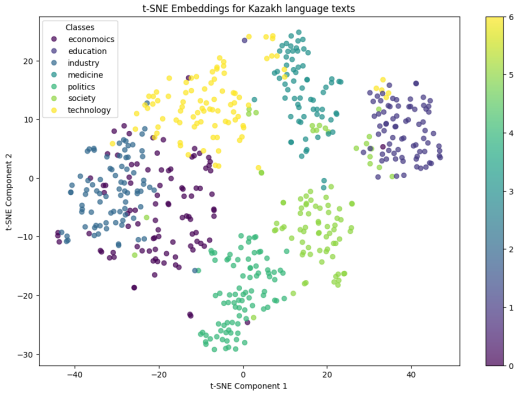

Figure 5-10: t-SNE for Kazakh word embeddings obtained from finetuned *intfloat/multilingual-e5-base*

Despite this, fine tuning made the variance and overlaps among vectors smaller, which was shown by the improvement in metrics mentioned earlier. The data points for industries and economics are located close to each other indicating similarities in context or relationships between these areas. This may also tell that many texts fall into these classes have common topics or use similar terminology.

In all the graphs one common trend can be seen, namely low indicators for precision. The *precision at* k represents evaluated in the MTEB benchmark refers to how

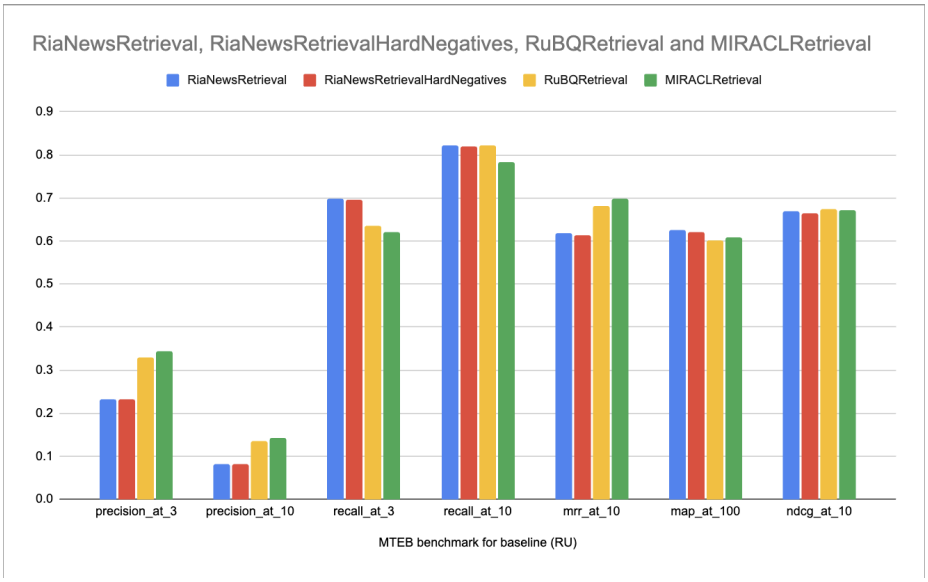

Figure 5-11: MTEB benchmark evaluation for Russian language, baseline *intfloat/multilingual-e5-base*

many of top k retrieved documents are actually relevant to the requested query. For example, *Precision@5* with 60% shows that 3 out of 5 are relevant. Those numbers are then averaged over all queries in the dataset. It should be that the top-ranked documents are more relevant and placed higher. That is why value for precision at 3 is noticeably higher than at 10. As more lower ranking items (eg. 4-10) are included, irrelevant results begin to appear.

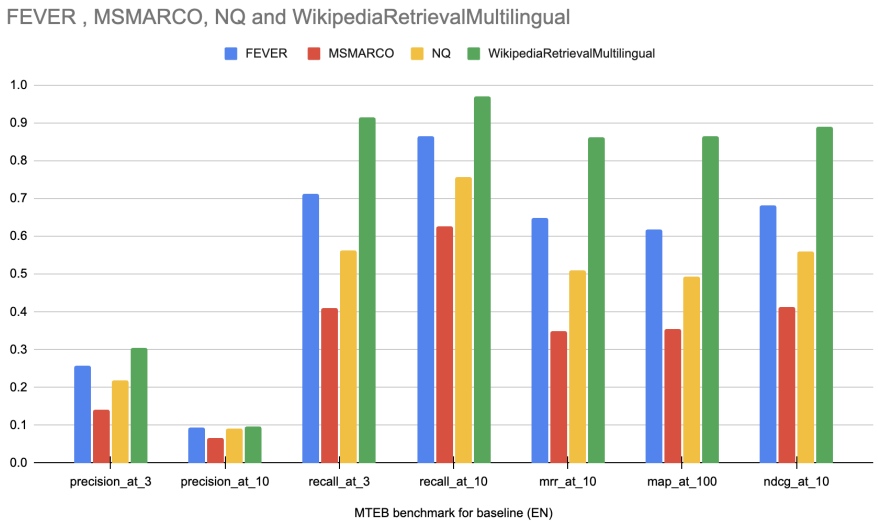

Figure 5-12: MTEB benchmark evaluation for English language, baseline *intfloat/multilingual-e5-base*

Chapter 6

Conclusion

The objective of this work is to explore the effectiveness of standalone language models with integrated Retrieval Augmented Generation (RAG) in providing accurate and relevant responses. Such approach enables language models to access not only pre-existing knowledge obtained during the pre-training stage but also real-time information from the external sources like SQL databases or vector stores. The performance of the system directly depends on the choice of a suitable embedding model that can accurately represent sentence and even the whole document. In particular, it is related to Kazakh language having limited support in multilingual models. To address this challenge, the open-source models like *E5-base*, *E5-large-instruct*, and *gte-multilingual-base* were fine-tuned and evaluated using precision, accuracy and retrieval task-related metrics.

Large Language Models (LLM) now is able to process textual data at human level of comprehension and reasoning so it can easily catch the emotional tone and sentiment of a given context. These models surpassed traditional machine learning (ML) models for sentiment analysis, which struggled with irony and sarcastic expressions. Since social media content in Kazakhstan are mainly written in native language there is still problem with cultural and linguistic nuances. This can result in models failing to capture the true meaning behind the social media posts.

This work was fragmented into six parts. The Introduction part provides the foundational background for the field of sentiment analysis and Natural Language Processing

(NLP), emphasizing the importance of these algorithms in understanding text's emotional tone. In addition, the objectives and goals were outlined along with the detailed definition of the thesis structure. The following Literature review part discusses previously completed scientific work around sentimental analysis with a particular focus on the embedding model and its integration in RAG systems. It also identifies gaps in the research area, including how current models perform on Kazakh texts and the shortcoming of traditional algorithms.

Chapter three dedicated for datasets introduces open and closed domain corpus of data for training/evaluation purposes. The large-scale datasets comprises a diverse range of topics drawn from different social media resources, enabling the model to generalize across varying linguistic context. In addition, this chapter presents data collection methods and preprocessing to improve the performance of embedding models. The chapter concludes with a discussion of the limitations of such high-quality datasets for news article embedding, highlighting the lack of representations of Kazakh-language texts.

The methodology part was focused on the architecture of the embedding model, the training procedure, and how daily news parsing was implemented to provide the system with continuously updated textual data. In other words, the collected data were used as additional context for augmenting responses in RAG systems as well as for generating a synthetic dataset for training. Since the project involves embedding at the paragraph and text level, a more suitable loss function Multiple Negative Ranking Loss (MNRL) was selected. The model learns to correctly determine which part of the document the query refers to. As the results in the Results and Discussions section show, the process of fine tuning a model with a synthetic dataset improves performance by an average of 3-7 percent for all languages, including Kazakh. This can be seen by comparing the two Figures 5-1 and 5-2. All metrics showing improvement in model performance were obtained using the test dataset for all three languages. It is worth mentioning that there are no benchmarks for testing embedding models for the ability to work with texts in the Kazakh language. Figure 5-6 representing fine-tuned models for the evaluation of the Russian language has bet-

ter results and resembles Figure 5-2. English language benchmarks are massive and consist of complex sentences, allowing AI models to effectively test their ability to cope with a variety of grammatical structures, lexical features, and cultural context. The quality and volume of the collected data corpus is also important here. Based on the procedure of collecting and processing datasets defined in Natural Questions (NQ), MS MARCO or Stanford Question Answering Dataset (SQuAD), the manual annotation is at high level. This indicates that manual annotation will be required in the future to improve the quality of the dataset with a focus on the Kazakh language. Moreover, the improvements can be made by expanding the context rich samples from social media sources and not only daily news.

6.0.1 Future works

Recent research papers in the area of representation learning is shifting toward language agnostic representation which aim to embed the texts regardless of its origin and language into one space. In other words, we abstract from language and the model begins to think at the level of concepts. This new paradigm enables the development of models that can reason and analyze about text's meaning across several languages without the need for direct translation into another language. Unlike some embedding models focusing on token-level, the Sentence-Level Multimodal and Language-Agnostic Representations (SONAR) [10] emphasizes sentence and it is good candidate for summarization, classification and semantic search tasks. Since multilingual embedding models were initially focused on processing English-language sources, the analysis of Kazakh texts required additional training on a high-quality dataset. In other words, information or knowledge obtained in English-language datasets can be used to process Kazakh texts since there is no language barrier and stored in a common embedding space. Another promising direction could be the application of contrastive learning to text embeddings. Similar to MNRL loss function there exist Triplet loss mentioned in Table 4.1 which offers a structured way to optimize sentence embeddings. The Triplet loss operates by minimizing the distance between anchor and positive samples (semantically similar texts), while pushing away the negative sam-

ples from the anchor having dissimilar context. To make a triplet dataset for training a model with such a loss function, you will need to collect three columns (anchor, negative,, and positive sentences) based on the parsed data. Each parsed sentence goes through the embedding process of the model (teacher) having the highest scores and metrics in the retrieval task. The embedding model from OpenAI (text-embedding-3-large) with a dimensionality of 1536 is well suited for this purpose. The search stage of hard negative and positive samples for some anchor sentence embedding will take too much time and computational power. To address the computational overhead involved in identifying hard negatives and positives for each anchor, the Facebook AI Similarity Search (FAISS) is used to provide efficient similarity search in high-dimensional space. After creating the dataset, all that remains is to train and test the finite embedding model.

Bibliography

[1] Y. B. Abdullin and V. V. Ivanov. Deep learning model for bilingual sentiment classification of short texts. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 17(1), January–February 2017.

[2] P. Bajaj et al. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[3] Chi-Ming Chan, Chen Xu, Ruihong Yuan, Han Luo, Wei Xue, Yuhang Guo, and Jie Fu. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*, 2024.

[4] Xinlei Chen, Simon Kornblith, Mohammad Noroozi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1597–1607. PMLR, 2020.

[5] C. Clark and M. Gardner. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia, 2018.

[6] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishal Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *Facebook AI*, 2020.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[8] Z. Drus and H. Khalid. Sentiment analysis in social media and its application: Systematic literature review. *Procedia Computer Science*, 161:707–714, 2019.

[9] Z. Drus and H. Khalid. Sentiment analysis in social media and its application: Systematic literature review. *Procedia Computer Science*, 161:707–714, 2019.

[10] Paul-Ambroise Duquenne, Holger Schwenk, and Benoît Sagot. Sonar: Sentence-level multimodal and language-agnostic representations. *arXiv preprint arXiv:2308.11466*, 2023.

[11] J. Gao, M. Galley, and L. Li. Neural approaches to conversational ai. *arXiv preprint arXiv:1809.08267*, 2018.

[12] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv*, 2312.10997, 2023.

[13] Michael Glass, Gaetano Rossiello, Md Faisal Mahamud Chowdhury, Atharva Ramesh Naik, Pengshan Cai, and Alfio Gliozzo. Re2g: Retrieve, rerank, generate. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2701–2715, Seattle, WA, USA, 2022.

[14] Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. Efficient natural language response suggestion for smart reply. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4697–4710. Association for Computational Linguistics, 2017.

[15] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28, pages 1693–1701, 2015.

[16] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv*, 2311.05232, 2023. Accepted by ACM Transactions on Information Systems (TOIS).

[17] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017.

[18] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*, 2019.

[19] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Pearson, Upper Saddle River, NJ, USA, 3rd edition, 2023.

[20] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020.

[21] T. Kwiatkowski et al. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[22] Guillaume Lample and Alexis Conneau. Crosslingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.

[23] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[24] S. Lewis, D. Shuster, M. P. Roland, D. B. Joshua, and R. Ming-Wei. Retrieval-augmented generation for knowledge-intensive nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*, pages 945–959. Association for Computational Linguistics, 2020.

[25] Z. Li, Y. Zou, C. Zhang, Q. Zhang, and Z. Wei. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL 2021)*, Online, 2021. Association for Computational Linguistics.

[26] Zhen Li, Jian Wang, Ziyu Jiang, Hongyu Mao, Zhenyu Chen, Jing Du, Yifan Zhang, Fan Zhang, Dongdong Zhang, and Yang Liu. Dmqr-rag: Diverse multi-query rewriting for retrieval-augmented generation. *arXiv preprint arXiv:2411.13154*, 2024.

[27] Zhen Li, Yuliang Zou, Chao Zhang, Qi Zhang, and Zhongyu Wei. Learning implicit sentiment in aspect-based sentiment analysis with supervised contrastive pre-training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*, 2021.

[28] Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 881–893, Miami, Florida, US, 2024. Association for Computational Linguistics.

[29] Chi-Liang Liu, Tzu-Yu Hsu, Yung-Sung Chuang, and Hung-yi Lee. What makes multilingual bert multilingual? *arXiv preprint arXiv:2010.10938*, 2020.

[30] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[31] Akylbek Maxutov, Ayan Myrzakhmet, and Pavel Braslavski. Do llms speak kazakh? a pilot evaluation of seven models. 2024.

[32] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[33] Tomas Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In *11th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1045–1048, Makuhari, Japan, 2010.

[34] Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*, 2022.

[35] Junichiro Niimi. Dynamic sentiment analysis with local large language models using majority voting: A study on factors affecting restaurant evaluation. 2024.

[36] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[37] Zachary Rackauckas. Rag-fusion: A new take on retrieval-augmented generation. *International Journal on Natural Language Computing*, 13(1):37–47, February 2024.

[38] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, USA, 2016.

[39] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.

[40] M. Richardson, C. J. Burges, and E. Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 193–203, Seattle, WA, USA, 2013.

[41] Narynov Sergazy Sakenovich and Arman Serikuly Zharmagambetov. On one approach of solving sentiment analysis task for kazakh and russian languages using deep learning. In *International Conference on Computational Collective Intelligence*, volume 9876 of *Lecture Notes in Computer Science*, pages 537–545, 2016.

[42] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[43] Xiaofei Sun, Xuan Li, Sheng Zhang, Shuo Wang, Fei Wu, Jian Li, Tao Zhang, and Guoren Wang. Sentiment analysis through llm negotiations. *arXiv preprint arXiv:2311.01876*, 2023.

[44] Mukhammed Togmanov, Nurdaulet Mukhituly, Diana Turmakhan, Jonibek Mansurov, Maiya Goloburda, Akhmed Sakip, Zhuohan Xie, Yuxia Wang, Bekassyl Syzdykov, Alham Fikri Aji, Ekaterina Kochmar, Preslav Nakov, and Fajri Koto. Kazmmlu: Evaluating language models on kazakh, russian, and regional knowledge of kazakhstan. *Department of Natural Language Processing, MBZUAI*, 2024.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[46] S. Wang and J. Jiang. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*, 2016.

[47] J. Welbl, P. Stenetorp, and S. Riedel. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018.

[48] Z. Yang et al. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, San Diego, CA, USA, 2016.

[49] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium, 2018. Association for Computational Linguistics.

[50] Rustem Yeshpanov and Huseyin Atakan Varol. Kazsandra: Kazakh sentiment analysis dataset of reviews and attitudes. *Institute of Smart Systems and Artificial Intelligence, Nazarbayev University*, 2024.

[51] Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. Reducing bert pre-training time from 3 days to 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[52] Wenxuan Zhang, Yang Deng, Bing Liu, Sinno Pan, and Lidong Bing. Sentiment analysis in the era of large language models: A reality check. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024.

[53] Zifan Zheng, Yezhaohui Wang, Yuxin Huang, Shichao Song, Mingchuan Yang, Bo Tang, Feiyu Xiong, and Zhiyu Li. Attention heads of large language models: A survey. *arXiv preprint arXiv:2409.03752*, 2024.