UDC 004.912

UA-LLM: ADVANCING CONTEXT-BASED QUESTION ANSWERING IN UKRAINIAN THROUGH LARGE LANGUAGE MODELS

Syromiatnikov M. V. – Post-graduate student of the Department of Software Engineering, Odesa Polytechnic National University, Odesa, Ukraine.

Ruvinskaya V. M. – PhD, Professor of the Department of Software Engineering, Odesa Polytechnic National University, Odesa, Ukraine.

ABSTRACT

Context. Context-based question answering, a fundamental task in natural language processing, demands a deep understanding of the language's nuances. While being a sophisticated task, it's an essential part of modern search systems, intelligent assistants, chatbots, and the whole Conversational AI field. While English, Chinese, and other widely spoken languages have gathered an extensive number of datasets, algorithms, and benchmarks, the Ukrainian language, with its rich linguistic heritage and intricate syntax, has remained among low-resource languages in the NLP community, making the Question Answering problem even harder.

Objective. The purpose of this work is to establish and benchmark a set of techniques, leveraging Large Language Models, combined in a single framework for solving the low-resource problem for Context-based question-answering task in Ukrainian.

Method. A simple yet flexible framework for leveraging Large Language Models, developed as a part of this research work, enlights two key methods proposed and evaluated in this paper for dealing with a small amount of training data for context-based question-answering tasks. The first one utilizes Zero-shot and Few-shot learning – the two major subfields of N-shot learning, where N corresponds to the number of training samples, to build a bilingual instruction-based prompt strategy for language models inferencing in an extractive manner (find an answer span in context) instead of their natural generative behavior (summarize the context according to question). The second proposed method is based on the first one, but instead of just answering the question, the language model annotates the input context through the generation of question-answer pairs for the given paragraph. This synthetic data is used for extractive model training. This paper explores both augmentation-based training, when there is some annotated data already, and completely synthetic training, when no data is available. The key benefit of these two methods is the ability to obtain comparable prediction quality even without an expensive and long-term human annotation process.

Results. Two proposed methods for solving the low-to-zero amount of training data problem for context-based question-answering tasks in Ukrainian were implemented and combined into the flexible LLM experimentation framework.

Conclusions. This research comprehensively studied OpenAI GPT-3.5, OpenAI GPT-4, Cohere Command, and Meta LLaMa-2 language understanding capabilities applied to context-based question answering in low-resource Ukrainian. The thorough evaluation of proposed methods on a diverse set of metrics proves their efficiency, unveiling the possibility of building components of search engines, chatbot applications, and standalone general-domain CBQA systems with Ukrainian language support while having almost zero annotated data. The prospect for further research is to extend the scope from the CBQA task evaluated in this paper to all major NLU tasks with the final goal of establishing a complete benchmark for LLMs' capabilities evaluation in the Ukrainian language.

KEYWORDS: large language model, question-answering, few-shot learning, generative annotation.

ABBREVIATIONS

API is an Application Programming Interface;

BERT is a Bidirectional Encoder Representations from Transformers;

BLEU is a Bilingual Evaluation Understudy;

CBQA is a context-based question answering;

CoT is a Chain-of-Thought;

DeBERTa is a Decoding-enhanced BERT with disentangled attention;

EM is an Exact Match;

FLAN is a Fine-tuned Language Net;

GPT is a Generative Pre-trained Transformer;

LLaMA is a Large Language Model Meta AI;

LLM is a Large Language Model;

LM is a Language Model;

MLM is a Masked Language Modeling;

MT is a Machine Translation;

MVP is a Minimum Viable Product;

NLP is a Natural Language Processing;

NLTK is a Natural Language Toolkit;

NLU is a Natural Language Understanding;

PaLM is a Pathways Language Model;

QA is a Question-Answering;

RLHF is a Reinforcement Learning from Human Feedback;

RoBERTa is A Robustly Optimized BERT Pretraining Approach;

ROUGE is a Recall-Oriented Understudy for Gisting Evaluation;

SQuAD is a Stanford Question Answering Dataset;

TrecQA is a Text Retrieval Conference Question Answering;

ULMFiT is a Universal Language Model Fine-tuning.

NOMENCLATURE

A is a contiguous span of words from context C that is the most acceptable answer to question Q;

A_i is a possible answer span from context C;

C is a context represented as a sequence of words;

I_{LANG} is a language used for writing the task instruction included in the prompt;

k is a number of possible answers to question Q;

L is a number of layers of the neural network;

m is a length of question Q;

N is a number of context-question-answer triplets included in the prompt;

n is a length of context C;

DOI 10.15588/1607-3274-2024-1-14

P is a number of parameters of the neural network;

Q is a question represented as a sequence of words;

q_i is an i-th word of question Q;

$S_{POSITIVES}$ is a number of matching pairs where both predicted and true answers are not empty.

T_{GEN_NEG} is a number of generated question-answer pairs with unanswerable questions used for training;

T_{GEN_POS} is a number of generated question-answer pairs with answerable questions used for model training;

TOK_{TOTAL} is a total number of tokens used for both input and generated output;

$T_{UA\text{-}SQUAD}$ is a number of training samples from the UA-SQuAD dataset used for model training;

w_i is an i-th word of context C.

INTRODUCTION

In a century defined by the persistent influx of information and the ever-expanding digital landscape, the ability to extract relevant knowledge from vast repositories of text data has become increasingly paramount. CBQA, a fundamental task in the field of natural language processing, plays a pivotal role in addressing this need by enabling machines to comprehend human language and provide precise answers to user queries within a given context. Nowadays, search engines leverage CBQA systems like Google Quick Answer to provide users with precise and contextually relevant answers to their queries. Additionally, virtual assistants like Siri and Alexa employ context-based QA to facilitate natural language interactions, allowing users to ask questions and receive informative responses.

As the demand for efficient and accurate information retrieval continues to grow, the development of robust and sophisticated context-based QA systems remains a critical pursuit. The ability to emulate human-like comprehension, going beyond mere keyword matching, has always been a tough challenge for machines. However, recent advancements in NLP have ushered in a new era of possibility. The application of the attention mechanism to encoder-decoder architecture, which unlocked deeper context understanding for tasks with long text sequences, like machine translation, can be viewed as a starting point of the "golden age of NLP". Subsequent releases of Transformer architecture and its variations like BERT, RoBERTa, and DeBERTa pushed us as close as possible to human performance on most NLU tasks, including QA [1]. Being pretrained on giant text corpora, these encoder language models require just a little fine-tuning to demonstrate reasonable performance [2]. However, this strategy doesn't work well for low-resource languages [3, 4], including Ukrainian, as they were underrepresented during the pre-training stage, and in most cases, there is just a little-to-zero amount of training samples for fine-tuning.

With a scale from millions to tens of billions of parameters, we are now at the forefront of NLP advances – generative LLMs like GPT and PaLM, which have demonstrated unparalleled capabilities in understanding, generating, and processing human language across a multitude of languages and domains [5]. The ability to follow human instructions, multitasking, and orders of magnitude larger pre-training corpora turned this class of models into a perfect candidate to deal with the low-resource NLU tasks and languages.

While the known methods for CBQA require a substantial amount of annotated data for training, which is not available for most low-resource languages, including Ukrainian, the generalization capabilities encapsulated in LLMs as a result of pre-training with billions of words give them the ability to solve the vast majority of NLU problems, including CBQA, with just a task description and few training samples (few-shot learning) or even without them (zero-shot learning) completely [6]. Moreover, the ability to follow complex instructions combined with high generalization unveils the possibility of using these generative models for data annotation.

The object of study is the process of automatic context-based question answering with the neural network for the low-resource language.

The subjects of study are zero- and few-shot context-based question answering and data annotation with generative LLMs for the low-resource Ukrainian language.

The purpose of the work is to establish and benchmark a set of techniques, leveraging large language models, combined in a single framework for solving the low-resource problem for context-based question-answering task in Ukrainian. This endeavor includes a quantitative objective of reducing the required number of training examples while minimizing any decrease in quality, thus enhancing the feasibility and accessibility of CBQA in low-resource linguistic contexts.

1 PROBLEM STATEMENT

Let C represent a context consisting of a sequence of words $C = (w_1, w_2, \ldots, w_n)$, and Q denote a question represented as a sequence of words $Q = (q_1, q_2, \ldots, q_m)$. The task of context-based question answering can be formalized as follows: find the answer A in context C such that A is the most contextually relevant and correct response to the question Q. Mathematically, we aim to find A as:

$$A = \underset{A_i, i \in [1,k]}{\arg\max}\ P(A_i \mid Q, C).$$

$P(A_i|Q,C)$ represents the probability that the span A_i is the correct answer to the question Q given the context C.

This problem involves modeling the conditional probability distribution $P(A_i|Q,C)$ using advanced language models and machine learning techniques to accurately and contextually answer a wide range of questions within the given context. The challenge lies in identifying the correct answer span that maximizes this probability, considering the nuances of natural language and context.

2 REVIEW OF THE LITERATURE

Early research in context-based QA primarily focused on rule-based approaches [7] and information retrieval techniques [8]. Systems like IBM's DeepQA, which pow-

DOI 10.15588/1607-3274-2024-1-14

ered Watson, showcased the potential of structured data and knowledge graphs for answering questions, particularly in trivia-style competitions. While these early systems achieved remarkable milestones, they were constrained by their inability to handle the breadth and depth of human language variation and context.

The advent of machine learning techniques, particularly supervised and semi-supervised approaches, marked a significant shift in the QA landscape. Researchers began to explore methods for extracting features from text, creating labeled datasets, and training models to predict correct answers to questions [9]. Notable examples include the development of the TrecQA track and the emergence of datasets like SQuAD dataset, which laid the foundation for benchmarking QA systems.

One significant disadvantage of existing machine learning approaches was their isolated training process requiring a lot of training samples for reasonable performance [10] and generalization ability. The second rise of language models, like ULMFiT, empowered by deep neural networks, popularized domain adaptation and transfer learning – techniques aimed at pre-training of model on a giant amount of textual data to build a foundation for task-specific training with a higher generalization and less annotated examples required for the latter [11].

The subsequent introduction of encoding Transformer-based LMs, such as BERT and its descendants, RoBERTa and DeBERTa, catalyzed a quantum leap in most NLU tasks. These models, pre-trained on vast text corpora, demonstrated the ability to understand human language at an unprecedented scale. With the release of decoding or generative Transformer-based LMs like GPT, the question-answering was divided into two parts:

– extractive QA, where the encoding model tries to answer the question by predicting the most relevant span in the context;

– generative QA, where the model generates an answer to the given question; the context here is not mandatory since large generative models may retain the knowledge from the training stage.

Evaluations demonstrate that while extractive encoders like RoBERTa and DeBERTa perform better for rare terms or domains, generative decoders (GPT) and encoder-decoders (T5) are beneficial for long contexts [12]. For extractive models, the drawback is the required context, and for generative models, it's their hallucination – a tendency to output text sequence that may be structured correctly but wrong from the factuality side [13, 14].

While the fine-tuning stage with a reasonable amount of training data was initially required for both extractive and generative models, scaling from millions to billions and from billions to tens or hundreds of billions of parameters helps to mitigate this problem for the latter. Recent LLMs like PaLM 2 and LLaMA 2 outperform fine-tuned extractive QA models in a single-shot manner on the TriviaQA and BoolQ benchmarks [15, 16].

However, the bigger the language model is, the easier it is for her to hallucinate – generate incorrect facts and harmful information, or not follow the instructions given due to overfitting and a massive amount of controversial data in the training corpora. There are two main strategies to increase the model controllability: prompt engineering and instruction-based fine-tuning.

Prompt engineering generalizes a set of techniques that help to increase language model steerability without adjusting its weights, just through manipulating input (prompt) structure. Widespread techniques are instruction prompting and chain-of-thought. Instruction prompting augments the input sequence, usually a zero-shot or few-shot [17], with natural language instructions, so for example, instead of feeding just a context and a question for the CBQA task, one will also add "Extract the answer on the question using the context below" to the input. Chain-of-thought prompting is a more descriptive way to insert instructions into the input: instead of writing an abstract instruction, it uses one or multiple examples (one- or few-shot learning) to demonstrate how to derive a solution with a sequence of steps (thoughts). While complex tasks like math problems significantly benefit from CoT [18], it brings little to no gain for standard NLU tasks.

In contrast to prompt engineering, instruction-based fine-tuning connects natural language instructions with a diverse set of tasks to increase steerability by adjusting language model weights. FLAN LLM demonstrated that instruction tuning substantially improves zero-shot performance on unseen tasks [19]. Also, a more sophisticated strategy for instruction-based fine-tuning called Reinforcement Learning from Human Feedback, where human evaluations of language model predictions are used for its tuning to minimize bias and harmful generations, adapt it for the chat environment and zero-shot prompting [20]. The list of RLHF-powered models includes Gopher, InstructGPT, and well-known ChatGPT.

The key drawbacks of instruction-based fine-tuning compared to prompt engineering are a greater risk of overfitting, as the model becomes more tailored to the specific instructions or examples used during fine-tuning, potentially limiting its generalizability to a broader range of tasks, a more extensive and labor-intensive data collection process, and the higher computational power required for tuning model with billions of parameters. At the same time, prompt engineering, a more streamlined and efficient approach is mostly limited to high-resource languages as recent research on the multilingual capabilities of LLMs has indicated a significant decrease in quality when applied to low-resource languages [21].

3 MATERIALS AND METHODS

We propose two methods, extending the instruction prompting, to solve the data problem for the CBQA task in the low-resource Ukrainian language: N-shot bilingual instruction prompting and generative data annotation for extractive model training, both using LLMs.

In the proposed N-shot bilingual instruction prompting (Fig. 1), the English language is used for writing natural language instruction for the CBQA task and keywords, specifying the start of context, question, and answer written in Ukrainian. The resulting prompt consists of instruc-

DOI 10.15588/1607-3274-2024-1-14

tion, N annotated examples, where N>=0, and context with the question to be answered by the model. The motivation for using English as the primary language for instructions is based on the fact, that despite pre-training and fine-tuning corpora for LLMs include millions of texts, in most cases, they are dominated by English, and low-resource languages, including Ukrainian, are still underrepresented. For benchmarking purposes, we also evaluated monolingual instruction prompting with instructions and keywords written in Ukrainian. For N-shot learning, we explored both zero-shot and few-shot cases.

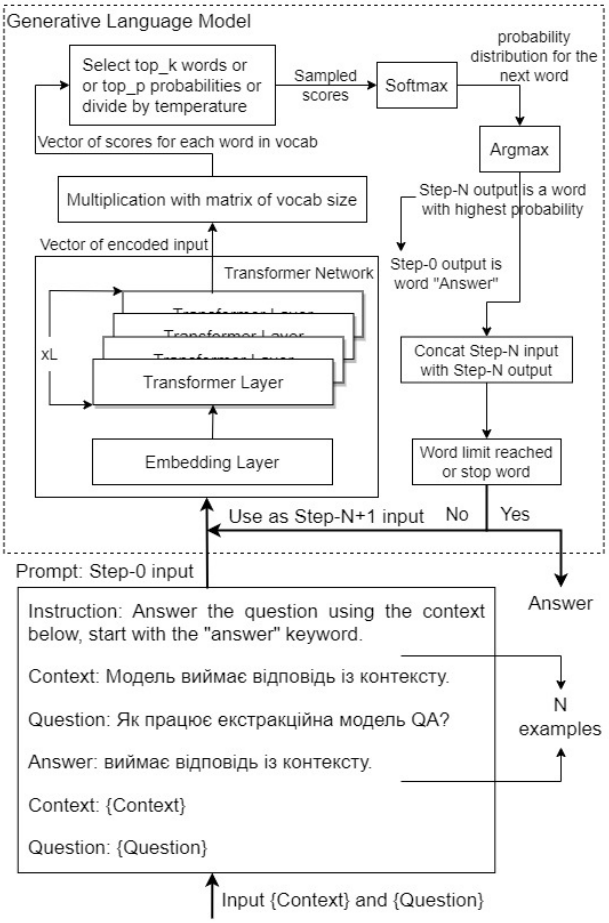

Figure 1 – Visualization of N-shot bilingual instruction prompting method

For simplicity, in Figure 1 above, a “word” term was used to demonstrate a language model’s decoding stage. However, most LMs work with tokens instead, where the token may be a complete word or just a part of it, depending on the word frequency in the training corpora. The word-to-token ratio depends on the language and the tokenization algorithm, but in general, a text with 750 words in English will be encoded into 1000 tokens, while the same number of words in Ukrainian and other low-resource languages may result in 1500–3000 tokens.

The question-answering with a generative language model, demonstrated in Figure 1, follows the standard autoregressive generation process iteratively predicting the next word in a sequence by leveraging the context of previous words. On each prediction step the model utilizes its knowledge acquired during training to estimate the probability distribution over the vocabulary by crafting the context representation of the input sequence and multiplying it with the weight matrix of vocab size. An important sampling step with top-K words, top-P probabilities, or softmax temperature strategies allows to parametrize and control creativity and diversity during the next word selection.

The second proposed method – a generative data annotation for extractive model training (Fig. 2), is based on the first one but introduces three key changes:

1. Gather texts from neutral domains like news portals or encyclopedias.

2. Instead of just answering the question with the given context, the LLM’s task is to annotate these gathered contexts, i.e. generate question-answer pair for each, where the answer span must be entirely presented in the context.

3. Depending on the available amount of data, use these generated annotations from the previous step either as the complete training set or to augment the existing dataset for the extractive QA model training.

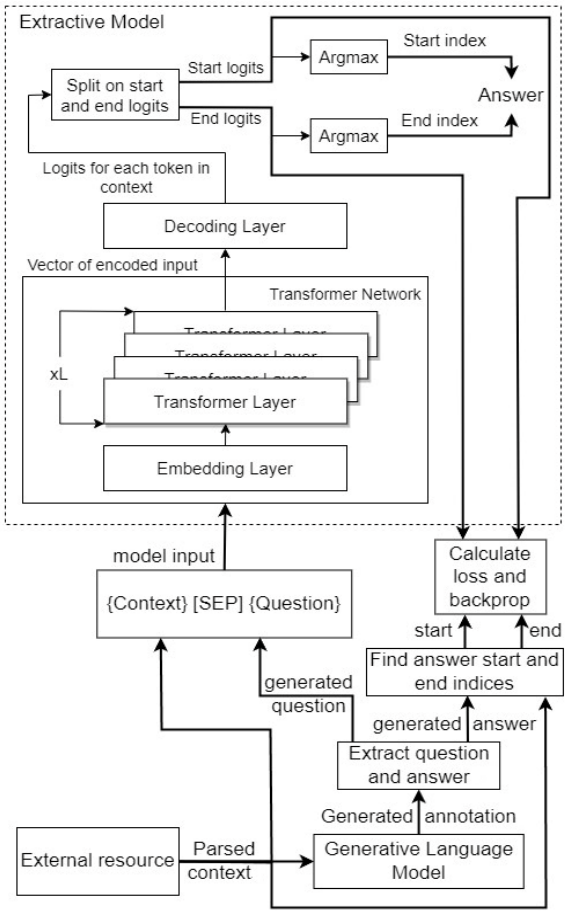

Figure 2 – Visualization of a generative data annotation method for extractive model training

DOI 10.15588/1607-3274-2024-1-14

In contrast to generative LLMs, where this can be achieved simply by adjusting the prompt, to adapt the encoding LM for any downstream task like context-based question answering, it is required to place a specific decoding layer on top of pre-trained LM for projecting encoding sequence context into desired dimension. In Figure 2, one can see the decoding layer placed after the stacked encoding transformer layers, which is just a linear projection. The dot product of sequence context with this linear layer results in two logits for each input token: start and end score. The higher the score is, the more confident the model is that this token is the start or the end of the answer. Therefore, the argmax operation applied to lists of start and end positions logits returns boundaries for the predicted answer. This approach for solving CBQA tasks with encoding Transformer LMs was first introduced with the BERT model release and later became the de facto standard for solving any machine reading comprehension task with transformers.

The direct usage of LLM for predicting answers proposed in the first method may be beneficial in the short-term due to implementation and serving simplicity: the ability to use one of the existing providers and almost zero requirement for annotated data. However, the inability to provide a completely deterministic behavior for LLM's inference due to hardware limitations and their hallucination may be inappropriate for some CBQA use cases, for which the answer's span precision is crucial. The proposed method of generative data annotation for extractive model training helps to avoid the issues described above by applying LLM not for the inference but for the extractive model's training stage. With this data level knowledge distillation, the extractive model (student) learns to mimic only the correct behavior of LLM (teacher) while being completely deterministic and cheaper due to orders of magnitude smaller size.

The following generative LLMs were used for the evaluation of the proposed methods: OpenAI GPT-3.5 Turbo, OpenAI GPT-4 and GPT-4 Turbo, Cohere Command, and Meta LLAMA 2 Chat. We followed two essential rules for the language model selection process:

– language support: the model has to support both English and Ukrainian languages;

– publicly available, the LLM has to be available for everyone either in the form of API or an open-sourced checkpoint to make results reproduction possible.

Let's briefly review each of the selected generative language models:

1. GPT-3.5 Turbo, also known as ChatGPT – language model from OpenAI, based on GPT-3 and optimized for conversations and instructions with RLHF fine-tuning while retaining most of the knowledge seen during pre-training. The model hyperparameters were not disclosed, but the original GPT-3 consists of 96 transformer decoder layers with a hidden size of 12288, totaling 175 billion parameters [22]. GPT-3.5 Turbo is available through OpenAI API in two variations: with a maximum input of 4098 and 16386 tokens.

2. GPT-4 – the latest generation LLM from OpenAI. GPT-4 outperforms GPT-3.5, demonstrating state-of-the-art results on NLU tasks and professional and academic exams [23], as well as achieving human-level performance on many of them [5]. The architecture details of GPT-4 were not disclosed, but according to rumors, the LLM leverages the Mixture-of-Experts technique with 8 GPT-3 expert models, resulting in approximately 1.7 trillion parameters. Standard GPT-4 is accessible through OpenAI API with a maximum input length of 8192 and 32768 tokens, while GPT-4 Turbo – the latest model released by OpenAI with improved instruction following, supports context of up to 128000 tokens. In this research, both standard and turbo versions were evaluated.

3. Command – a generative model built by Cohere and optimized for instruction-like prompts. Cohere provides access through API for the two versions of this model: command with 52.4 billion parameters and command-light with only 6.1 billion parameters. In this paper, the standard version is evaluated as it demonstrated reasonable quality on QA tests of the HELM benchmark [24].

4. LLaMA 2 – is a second generation of the open-source LLM family by Meta. Three different base models with 7, 13, and 70 billion parameters were pre-trained on 2 trillion tokens with a context size of 4096 [16]. Each of these models also has a fine-tuned version for chat (LLaMA Chat) and code (Code LLaMA). A version with 13 billion parameters fine-tuned for chat on over 1 million human annotations was selected for this research as it's an optimal trade-off between model size and performance.

In addition to generative LLMs, the following encoding LMs were used to build extractive models as a part of the generative data annotation for extractive model training and to establish a baseline:

1. XLM-RoBERTa – multilingual version of RoBERTa model by Meta pretrained on 2.5TB of text containing 100 languages in a self-supervised fashion with MLM objective [4]. More precisely, this language model based on encoding Transformer architecture was trained to predict masked words in the input sequence leveraging bidirectional context. For this research, the smaller version of XLM-RoBERTa with only Ukrainian and English tokens kept in the embedding layer was used.

2. DeBERTa – a family of Transformer-based encoding models by Microsoft demonstrating state-of-the-art performance and surpassing human performance on the SuperGLUE benchmark [25]. The usage of disentangled attention and enhanced mask decoder allows DeBERTa to outperform BERT and RoBERTa on most NLU tasks, including CBQA [25]. The third version of multilingual DeBERTa improved by ELECTRA-Style pre-training with Gradient Disentangled Embedding Sharing [26] was selected for this research.

For prediction evaluation, the following string-based metrics were selected:

1. SQuAD F1 – the harmonic mean of precision and recall adapted for SQuAD, a machine reading comprehension benchmark. The precision of the question-answering models is determined by the ratio of the number of cor-

DOI 10.15588/1607-3274-2024-1-14

rectly predicted words of the answer to the total predicted number of words of the answer and recall – the ratio of correctly predicted words of the answer to the total number of words in the true answer [27].

2. SQuAD Exact match – another metric used in the SQuAD benchmark along with F1. EM measures the exact match between strings on character level: EM = 1 for the exact match of the predicted answer span with the ground truth, and 0 in all other cases. For negative annotations with an empty string as a ground truth, any predicted text will result in EM = 0, even if it is a single character.

3. Any match F1 – similar to classification or detection F1 measure. This metric was used to measure LLM's hallucination and steerability: Any is true positive if there is a ground truth answer and the model predicted any span from the context, or if there is an empty true answer and the prediction is empty as well.

3. Partial match F1 – a stricter subset of Any metric, for which a true positive case happens only when there is an index-based overlap between the predicted answer span and ground truth.

4. BLEU – a metric initially implemented for evaluating MT systems by measuring the n-gram similarity between the translated text and high-quality reference translation [28]. While having an obvious drawback when applied for MT evaluation due to comparing tokens and not meaning, BLEU fits well for CBQA, allowing to evaluate LLM's predictions with augmentations or hallucinations.

5. ROUGE – a metric inspired by BLEU and used to score summarization algorithms. While both BLEU and ROUGE use n-grams to calculate similarity, the former is precision-oriented as it measures the number of n-grams from prediction appearing in the reference text, and the latter is recall-oriented instead, measuring how many n-grams from reference are presented in the prediction [29].

In order to simplify the LLM's inferencing and evaluation process for CBQA and make it flexible, the following experimentation framework named "UA-LLM" was designed and implemented (Fig. 3). The framework consists of 5 modules written in Python chained with OmegaConf-based config module by Hydra configuration framework.

Hydra's key features are the ability to dynamically create a hierarchical configuration by composition and override it through config files and the command line [30], as well as automatic recursive instantiation of Python objects during the runtime based on the config provided.

The entry point defined in main, accessible from the command line, expects a path to the OmegaConf file in YAML format with task config. The task module contains abstract classes with logic for prediction, evaluation, and annotation, along with their implementation for CBQA. Each task requires data reader and writer objects from the Data module and the model object from the LLM module. For each iteration of its run, the task gets the portion of data, combines it with the preconfigured prompt, and then feeds the obtained input to get the generated prediction. Depending on the task, the reader is used to load input data or predictions from an arbitrary source, and the writer is used to output predictions, annotations, or evaluation results. CSV tabular format reader and writer were implemented for CBQA experimentation. The data module also supports streaming for reading and writing to deal with large files.

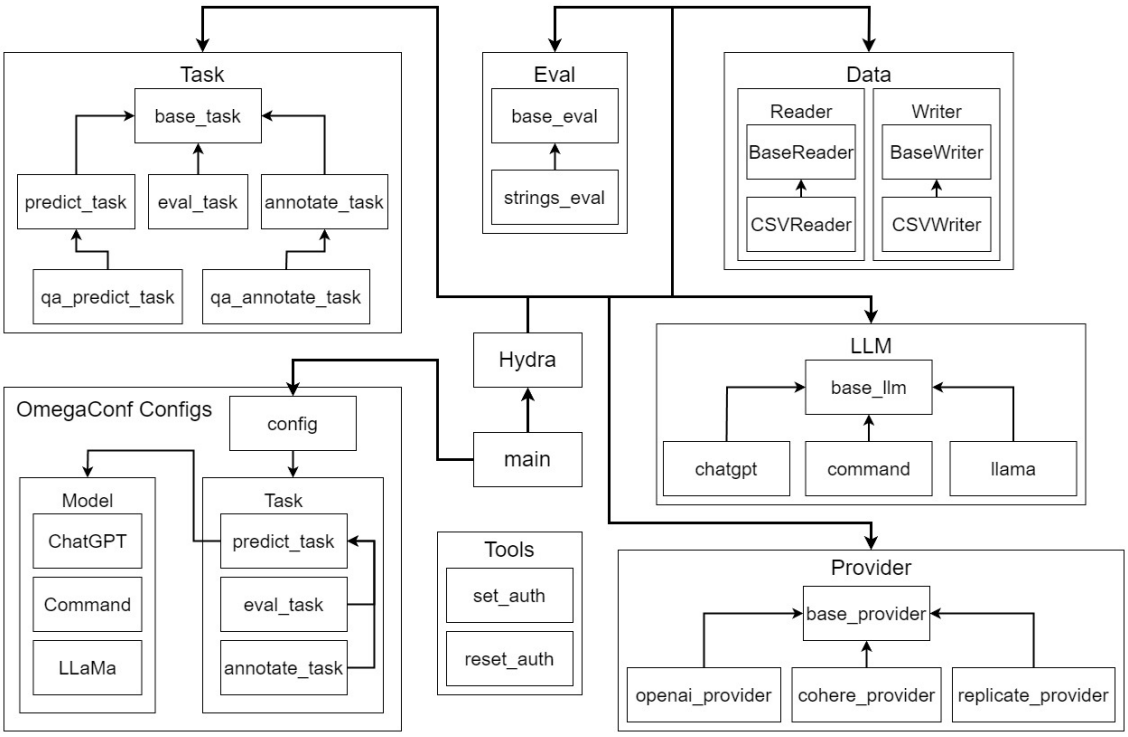

Figure 3 – Architecture diagram of the experimentation framework "UA-LLM"

DOI 10.15588/1607-3274-2024-1-14

Each model from the LLM module is just a simple wrapper for the provider object, which performs all low-level operations like establishing a connection to the model server, making requests, processing errors, and collecting session statistics. The framework contains providers for interaction with models accessible through API from OpenAI, Cohere, and Replicate. The eval module is a place for metrics grouped by evaluation level that could be used for prediction scoring. While strings_eval implements all metrics for strings comparison listed in the section above, one can select a subset with only desired metrics included by adjusting the argument in the task config.

Though the implemented framework for LLM experimentations primarily focuses on CBQA, it's easy to adapt it for any other NLU task since its decomposed architecture with most independent components provides high flexibility and generalization.

4 EXPERIMENTS

The localized version of the SQuAD 2.0 dataset was selected for methods evaluation. Stanford Question Answering Dataset (SQuAD) is a CBQA dataset consisting of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable [31]. The second version combines 100,000 questions from version 1.1 with more than 50,000 unanswerable questions written by annotators as well. However, work on the Ukrainian version of this dataset is still in progress, and so UA-SQuAD contains only 13,859 question-answer pairs, among which 2,927 are unanswerable questions [32].

This dataset with almost 14 thousand examples was grouped by 2620 unique contexts, and then these groups were randomly split into train/validation/tests sets with 80%/10%/10% proportion resulting in 11,173 pairs for train and 1,343 – validation and test.

For the N-shot bilingual instruction prompting method experiments, 0-shot, 2-shot, and 4-shot cases were selected. The annotated examples for few-shot prompting were randomly sampled from the train set. Only the temperature hyperparameter of LLM's generation process was tuned, and the set of parameters for each model was determined based on the manual evaluation of 100 predictions with a focus on steerability maximization and hallucination minimization. Evaluation with a temperature equal to 0 was prioritized for each model as it provides almost complete deterministic behavior. In addition to evaluations of bilingual instructions, monolingual instructions were evaluated to establish a baseline.

The complete set of successful experiments for N-shot bilingual instruction prompting method evaluation is listed in Table 1. The pricing columns correspond to the model request price for text processing and generation.

Table 1 – Parameters for N-shot bilingual instruction prompting method evaluation
* – OpenAI has not disclosed hyperparameters for GPT-3.5-turbo, GPT-4, and GPT-4-turbo models

Id	Model	Architecture			Prompt			Pricing, $ per 1000 tokens	
		L	*P*, bn	Context	*N*	Instruction language	Temperature	Input tokens	Output tokens
1	LLaMA-2-Chat	40	13	4096	0	UA	0.1	0.003	0.003
2	LLaMA-2-Chat	40	13	4096	0	UA	0.5	0.003	0.003
3	LLaMA-2-Chat	40	13	4096	0	UA	0.9	0.003	0.003
4	LLaMA-2-Chat	40	13	4096	0	EN	0.1	0.003	0.003
5	LLaMA-2-Chat	40	13	4096	0	EN	0.5	0.003	0.003
6	LLaMA-2-Chat	40	13	4096	0	EN	0.9	0.003	0.003
7	Command	N/A	52.4	4096	0	UA	0.0	0.015	0.02
8	Command	N/A	52.4	4096	0	EN	0.0	0.015	0.02
9	Command	N/A	52.4	4096	2	UA	0.0	0.015	0.02
10	Command	N/A	52.4	4096	2	EN	0.0	0.015	0.02
11	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	UA	0.0	0.0015	0.002
12	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	UA	0.5	0.0015	0.002
13	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	UA	0.9	0.0015	0.002
14	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	UA	0.0	0.0015	0.002
15	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	UA	0.5	0.0015	0.002
16	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	UA	0.9	0.0015	0.002
17	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	4	UA	0.0	0.0015	0.002
18	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	4	UA	0.5	0.0015	0.002
19	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	EN	0.0	0.0015	0.002
20	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	EN	0.5	0.0015	0.002
21	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	0	EN	0.9	0.0015	0.002
22	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	EN	0.0	0.0015	0.002
23	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	EN	0.5	0.0015	0.002
24	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	2	EN	0.9	0.0015	0.002
25	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	4	EN	0.0	0.0015	0.002
26	GPT-3.5-turbo-06.13	up to 96*	up to 175*	4096	4	EN	0.5	0.0015	0.002
27	GPT-4	N/A	1700*	8192	0	UA	0.0	0.03	0.06
28	GPT-4	N/A	1700*	8192	0	EN	0.0	0.03	0.06
29	GPT-4-turbo-preview	N/A	N/A	128000	0	UA	0.0	0.01	0.03
30	GPT-4-turbo-preview	N/A	N/A	128000	0	EN	0.0	0.01	0.03

DOI 10.15588/1607-3274-2024-1-14

Few-shot experiments with LLAMA-2-Chat were not included in the final table since with $N>0$ the model stopped to follow instructions and hallucinated in all test generations, making answer extraction impossible. In addition, limited experiments were conducted with Command, GPT-4, and GPT-4 Turbo models due to usage limits and significant inference pricing.

English and Ukrainian instructions used in both methods were written with a focus on coherence and clarity and were not optimized for any observed LLM. For few-shot instructions, examples were sampled randomly from the training set. All LLM prompts were fed as a single message, without splitting them into system prompt and user prompt, since it did not affect prediction quality for the GPT and Command models and demonstrated higher steerability for the LLaMA model. Prompts used for both methods are listed on the paper's page[1] in the GitHub repository.

The best model and set of parameters in terms of quality and pricing determined from evaluations of the N-shot bilingual instruction prompting method in the results section was later selected as an annotator for generative data annotation for the extractive model training method.

External contexts for the data annotation process were crawled from the TSN news portal. Since news texts usually consist of multiple paragraphs or media objects, collected texts were preprocessed to remove unknown symbols, common phrases like "Read more" or "Similar articles", multiple spaces, and join paragraphs in case there are multiple of them. Some of these texts were then filtered out due to being shorter than 120 or longer than 350 tokens, as the maximum sequence length for extractive models was established at 384 tokens, for which a bit more than 30 tokens were reserved for questions. Finally, 2000 contexts for annotation were randomly sampled from the resulting amount.

The prompt from the best experiment was adapted for the data annotation task: instead of answer extraction, the request became to generate five question-answer pairs for the given context, where four pairs are positive – questions are answerable, and the last one, negative, should be unanswerable. Given the 2000 contexts for annotation, this should result in 10,000 question-answer pairs ideally, however, the unavoidable hallucinations like word order change in answer span or plurality and inflection modifications make the annotation postprocessing pipeline filter some pairs.

Training configs for extractive models used to establish a baseline and those used for generative data annotation for the extractive model training method evaluation are demonstrated in Table 2. For this method, both training only on generated annotations and augmentation-based training approaches with adding generated positive question-answer pairs were evaluated. All extractive models were trained on a single GPU for three epochs with a batch size of 12, a learning rate of 3e-05, a max sequence length of 384, a dropout rate of 0.1, and an Adam optimizer.

Table 2 – Parameters for extractive models training

Id	Model	P, mln	$T_{\text{UA-SQUAD}}$	$T_{\text{GEN\_POS}}$	$T_{\text{GEN\_NEG}}$
31	XLM-RoBERTa	110	11173	0	0
32	DeBERTa	278	11173	0	0
33	XLM-RoBERTa	110	0	6156	1724
34	DeBERTa	278	0	6156	1724
35	XLM-RoBERTa	110	11173	6156	0
36	DeBERTa	278	11173	6156	0

For augmentation-based experiments, where both $T_{\text{UA-SQUAD}}$ and $T_{\text{GEN\_POS}}$ were used for the extractive model training, the final train set was created by concatenation of these two sets with subsequent random shuffling. All experiments, including those with $T_{\text{UA-SQUAD}}$ equal to 0, used the validation set of UA-SQUAD dataset. The intermediate evaluation step aimed at filtering unpromising parameter combinations [33] was based on N-shot bilingual instruction prompting and resulted in only generated positives being used for augmentation.

5 RESULTS

Along with quality, the pricing aspect of LLM utilization plays an important role in this research as it defines whether proposed methods for solving the low-resource problem of language could be reproduced and adopted. Table 3 demonstrates the costs of the N-shot bilingual instruction prompting method for each unique set of parameters. In the table, $\text{TOK}_{\text{TOTAL}}$ and price columns were calculated as the sum of values for each of all 1343 examples from the test set. The detailed stats for the temperature parameter weren't included in the table since it affects only the length of the generated answer, so its influence on the total price is close to zero compared to the instruction language parameter I_{LANG} and the number of training examples N added to the prompt.

Table 3 – Influence of N-shot bilingual instruction prompting method parameters on inference pricing

Model	N	I_{LANG}	$\text{TOK}_{\text{TOTAL}}$	Price, $
LLaMA-2-Chat	0	UA	706272	2.12
LLaMA-2-Chat	0	EN	524563	1.58
Command	0	UA	1227015	18.58
Command	0	EN	794381	12.02
Command	2	UA	2447285	36.77
Command	2	EN	2025534	30.46
GPT-3.5-turbo-06.13	0	UA	968301	1.47
GPT-3.5-turbo-06.13	0	EN	676416	1.03
GPT-3.5-turbo-06.13	2	UA	1976186	2.97
GPT-3.5-turbo-06.13	2	EN	1686699	2.54
GPT-3.5-turbo-06.13	4	UA	3663482	5.51
GPT-3.5-turbo-06.13	4	EN	3372538	5.07
GPT-4	0	UA	961822	29.42
GPT-4	0	EN	679322	21.01
GPT-4-turbo-preview	0	UA	968681	10.01
GPT-4-turbo-preview	0	EN	684890	7.19

Results of the quality evaluation for the N-shot bilingual instruction prompting method are demonstrated in Table 4.

1 - https://github.com/NLPForUA/UA-LLM/blob/main/paper/README.md

DOI 10.15588/1607-3274-2024-1-14

Table 4 – Results of N-shot bilingual instruction prompting method evaluation, numbers in bold represent the highest metric score per model

Id	Model	SQuAD Metrics					N-gram Metrics			Index-based Metrics	
		EM	F1	Pos EM	Pos F1	Neg F1	BLEU-1	ROUGE-1	$S_{POSITIVES}$	Partial F1	Any F1
1	LLaMA-2-Chat	13.70	25.89	16.70	31.72	0.79	0.24	0.28	1089	0.28	0.30
2	LLaMA-2-Chat	13.48	25.95	16.33	31.69	**1.19**	0.25	0.29	1089	0.27	0.30
3	LLaMA-2-Chat	11.39	22.47	13.94	27.59	0.40	0.21	0.25	1089	0.24	0.26
4	LLaMA-2-Chat	**24.57**	**41.05**	**30.09**	**50.40**	0.79	**0.41**	**0.46**	1090	**0.48**	**0.52**
5	LLaMA-2-Chat	23.98	40.03	29.36	49.14	0.79	0.40	0.45	1090	0.47	0.51
6	LLaMA-2-Chat	23.38	38.35	28.53	46.98	**1.19**	0.39	0.43	1090	0.45	0.49
7	Command	26.43	40.13	**31.47**	**48.34**	4.74	0.42	0.48	1048	0.60	0.73
8	Command	27.48	39.09	29.63	43.94	18.18	0.45	0.51	910	0.58	0.74
9	Command	29.56	40.99	26.70	40.78	**41.90**	0.42	0.49	879	0.56	0.68
10	Command	**32.02**	**44.76**	**31.47**	47.17	34.39	**0.46**	**0.54**	924	**0.62**	**0.76**
11	GPT-3.5-turbo-06.13	36.04	54.34	34.40	56.96	43.08	0.50	0.57	1035	0.59	0.63
12	GPT-3.5-turbo-06.13	35.96	54.43	34.86	57.61	40.71	0.51	0.58	1038	0.59	0.63
13	GPT-3.5-turbo-06.13	32.91	51.87	31.19	54.55	40.32	0.47	0.54	1040	0.55	0.60
14	GPT-3.5-turbo-06.13	45.27	61.15	44.31	63.88	49.41	0.58	0.63	1011	0.65	0.69
15	GPT-3.5-turbo-06.13	43.34	59.67	42.57	62.69	46.64	0.57	0.63	1009	0.64	0.68
16	GPT-3.5-turbo-06.13	42.37	58.90	41.01	61.38	48.22	0.55	0.60	1010	0.63	0.66
17	GPT-3.5-turbo-06.13	44.30	60.78	43.12	63.42	49.41	0.57	0.62	1013	0.63	0.67
18	GPT-3.5-turbo-06.13	43.26	59.33	42.48	62.28	46.64	0.57	0.62	1005	0.63	0.66
19	GPT-3.5-turbo-06.13	40.95	57.85	36.79	57.61	**58.89**	0.55	0.61	979	0.66	0.71
20	GPT-3.5-turbo-06.13	40.58	57.96	36.88	58.30	56.52	0.55	0.62	981	0.65	0.70
21	GPT-3.5-turbo-06.13	38.42	56.30	34.77	56.80	54.15	0.54	0.60	974	0.62	0.66
22	GPT-3.5-turbo-06.13	46.01	62.85	**47.52**	**68.26**	39.53	**0.61**	0.66	1054	**0.72**	**0.76**
23	GPT-3.5-turbo-06.13	45.72	62.30	47.61	68.05	37.55	**0.61**	0.66	1050	**0.72**	**0.76**
24	GPT-3.5-turbo-06.13	43.48	60.62	44.77	65.88	37.94	0.58	0.64	1050	0.69	0.73
25	GPT-3.5-turbo-06.13	**46.24**	**63.36**	45.69	66.79	48.62	**0.61**	**0.67**	1027	0.71	0.75
26	GPT-3.5-turbo-06.13	**46.24**	62.81	46.06	66.47	47.04	0.60	0.65	1035	0.70	0.75
27	GPT-4	55.47	72.76	**52.57**	**73.88**	67.98	0.68	0.74	1061	0.87	0.89
28	GPT-4	52.94	71.35	49.45	72.13	67.98	0.66	0.72	1059	0.87	0.89
29	GPT-4-turbo-preview	**58.08**	**74.64**	51.47	71.88	86.56	**0.71**	**0.77**	1003	**0.88**	**0.90**
30	GPT-4-turbo-preview	55.92	72.73	48.44	69.15	**88.14**	0.70	**0.77**	960	0.87	**0.90**

The id column from the results table above corresponds to the same column from Table 1 with experiment parameters. Pos and Neg in the SQuAD metrics section are abbreviations for positive or answerable questions and negative or unanswerable questions. BLEU-1 and ROUGE-1 measure unigram similarity between the prediction and true answer using the NLTK's tokenizer to split sequences on words during the preprocessing. These N-gram metrics only support cases when both prediction and true answer are non-empty strings. The total number of these positive pairs for each evaluation is demonstrated in $S_{POSITIVES}$ column. All evaluations were performed on the test set with 262 unique contexts and 5.12 questions on average per context, reaching 1343 question-answer pairs in total, 1090 of which are positive and 253 – negative.

One of the best sets of parameters in terms of demonstrated quality and prediction price – 2-shot bilingual prompting for GPT-3.5 Turbo model with instruction in English and temperature parameter equal to zero (Id 22 in Tables 1 and 4) was chosen for generative data annotation. The generative annotation process for 2000 external contexts resulted in 9972 generated question-answer pairs, 2092 of which were filtered out due to the answer span not being presented in the context. Among the 7880 valid pairs, 6156 were positive and 1724 – negative. The total price for the annotation with 2.8 million tokens used for input and 526 thousand output tokens is $5.79.

The complete results of the evaluation of baseline extractive models trained on the UA-SQuAD dataset, as well as those trained entirely on or augmented by generated annotations, are demonstrated in Table 5.

Table 5 – Results of extractive models evaluation, numbers in bold represent the highest metric score per model

Id	Model	SQuAD Metrics					N-gram Metrics			Index-based Metrics	
		EM	*F1*	Pos EM	Pos F1	Neg F1	BLEU-1	ROUGE-1	$S_{POSITIVES}$	Partial F1	Any F1
31	XLM-RoBERTa	53.83	66.56	52.39	68.06	**60.08**	0.69	0.75	976	0.83	0.89
32	DeBERTa	59.05	**72.91**	57.06	74.15	**67.59**	**0.73**	**0.79**	1017	**0.88**	0.92
33	XLM-RoBERTa	35.82	48.67	36.97	52.81	30.83	0.56	0.62	899	0.73	0.80
34	DeBERTa	39.61	53.33	41.10	58.00	33.20	0.63	0.69	889	0.76	0.81
35	XLM-RoBERTa	**54.58**	**67.30**	**53.39**	**69.06**	59.68	**0.70**	**0.76**	983	**0.84**	**0.90**
36	DeBERTa	**60.31**	72.80	**58.81**	**74.19**	66.80	**0.73**	0.78	1023	0.87	**0.93**

DOI 10.15588/1607-3274-2024-1-14

The id column from the extractive models' evaluation table above corresponds to the column with the same name from Table 2 with parameters for extractive models training. In addition, for extractive models evaluation, the same test set and metrics as for generative language models (Table 4) were used, so the results in these two tables are comparable.

6 DISCUSSION

In this section, we present a detailed discussion of the results obtained from our experimental evaluations of proposed methods for context-based question answering: N-shot bilingual instruction prompting and generative data annotation for extractive model training. Our study aimed to investigate the efficacy of these novel approaches in enhancing the performance of low-resource QA where only little to no training data is available.

Our results demonstrate that N-shot bilingual instruction prompting can indeed be an effective strategy for improving the LLM's capabilities for solving CBQA tasks in low-resource languages, particularly in Ukrainian. For all language models, except GPT-4 and GPT-4 Turbo, zero-shot bilingual prompts with instruction written in high-resource English demonstrate a significant increase for all evaluated metrics compared to the same parameters set with instruction in Ukrainian.

The biggest gain from applying a bilingual prompt was achieved for the smallest generative LM evaluated – LLaMA-2-Chat with only 13 billion parameters: a comparison of evaluations from Table 4 with id 1 (monolingual) and 4 (bilingual) demonstrate a 10.87 and 15.16 points increase of SQuAD's EM and F1 respectively for the latter, as well as up to 1.7 times higher scores for BLEU, ROUGE, Partial and Any F1. Such a vast difference could be explained by the relatively small model size and low Ukrainian language presence of 0.07% in the training corpora since the absolute gain decreases for the larger models, like Command and GPT-3.5 Turbo, with orders of magnitude more parameters. Cohere Command with zero-shot instruction prompting demonstrates similar performance for monolingual and bilingual cases, with the former being better for answerable questions and the latter – for unanswerable. At the same time, the same setup for GPT-3.5 Turbo (id 19 in Table 4) allows to achieve SQuAD EM of 40.58 or 13.6% increase, SQuAD F1 of 57.85 or 6.5% increase, as well as up to 10% growth for n-gram metrics BLEU and ROUGE and up to 12% – for index-based Partial F1 and Any F1. The best zero-shot CBQA performance across all evaluated LLMs was demonstrated by OpenAI GPT-4 Turbo with monolingual prompting (id 29): 2 points higher scores than the ones by the proposed bilingual approach (id 30) on average for SQuAD metrics. The reason behind this is that GPT-4 Turbo is a significantly bigger LLM and better optimized for instructions and multilingualism compared to GPT-3.5 Turbo [34].

One noteworthy observation from our experiments is the influence of the size of the N-shot prompt on model performance. We found that as the number of example question-answer pairs in the second language increased, so did the model's ability to handle questions in that language. However, the rate of improvement tended to diminish beyond a certain point, suggesting that a modest-sized N-shot prompt can yield substantial benefits without overwhelming the model. From Table 4 with the evaluation results, it can be seen that N-shot bilingual prompting demonstrates the best scores for the Command model (id 10) with N=2, while the highest score for GPT-3.5 Turbo is the one with N=4. In addition, it was observed that few-shot monolingual prompting with instruction in Ukrainian also helps to increase the quality of predictions: two-shot prompt (id 14) improved SQuAD EM from 36.04 to 45.27 (25.6% gain) and F1 from 54.34 to 61.15 points (12.5% gain), BLEU and ROUGE were also increased by 16% and 10.5% correspondingly.

The best GPT-3.5 Turbo scores achieved with bilingual prompt and N=4 (id 24) are 46.24 and 62.81 for SQuAD EM and F1, with a relative increase of 28.3% and 16.6% over the monolingual zero-shot baseline (id 11). While the 4-shot bilingual instruction prompting demonstrates the highest scores among all GPT-3.5 Turbo evaluations, the difference for all metrics is no more than 1% between N=2 (id 22) and N=4, illustrating our observation around the quality improvement decay with an increase of N beyond the certain point.

Comparing the results of LLMs evaluation with extractive models (Table 5), one can see that both monolingual and bilingual instruction prompts allow GPT-4 and GPT-4 Turbo to outperform the baseline XLM-RoBERTa model (id 31) with more than 11 thousand annotated examples used for training in contrast to 0 samples used for the GPT. Moreover, GPT-4 Turbo with zero-shot prompting surpasses baseline DeBERTa's (id 32) SQuAD F1 score (74.64 vs 72.91) and strongly outperforms its Negative F1 (86.56 vs 67.59). While the GPT-4 and GPT-4 Turbo are prohibitively expensive, an order of magnitudes cheaper model, the GPT-3.5 Turbo with 4-shot bilingual instruction prompting allows to achieve 95.2% and 86.9% of baseline XLM-RoBERTa and DeBERTa models performance according to SQuAD F1.

The results of the evaluation of generative data annotation for extractive model training presented in Table 5 demonstrate that with just 2000 general-domain texts and $5.79, it is possible to obtain annotations sufficient to train XLM-RoBERTa or DeBERTa, achieving 73.1% (ids 33–34) of SQuAD F1 score of the same model trained with significantly more expensive human-annotated data (ids 31–32). Also, usage of the generative annotation for training data augmentation increased XLM-RoBERTa's scores (id 35) by 1 point for all metrics except for Negative F1, compared to the baseline result (id 31), while for DeBERTa (id 36), only 1–2 points gain for SQuAD EM, Positive F1, and Any F1 was observed.

The advantage of the N-shot bilingual instruction prompting method proposed in this paper for solving CBQA tasks in low-resource language is the possibility of achieving up to 95% of the extractive model's quality trained on tens of thousands of annotated samples with

DOI 10.15588/1607-3274-2024-1-14

just 2 to 4 examples, thus solving the problem of close-to-zero amount of annotated data. The drawback of this method, however, is its high price: serving LLM with tens or hundreds of billions of parameters is enormously more expensive than inferencing an extractive model with only one or two hundred million parameters. Additionally, most LLMs available through API are billed per input and output text lengths.

The advantage of the generative data annotation for the extractive model training method proposed in this research is that it helps to achieve up to 73.1% of the prediction quality of the extractive model trained with significantly more expensive human annotations while using only cheap generative annotations. In addition, combining generative and human annotations for extractive model training slightly increases its overall performance. However, the key benefit of this method is that it takes the best of two worlds: distilling LLM's knowledge into the extractive model through training on generative annotations makes it possible to achieve MVP solutions with reasonable quality relatively fast while saving on the inference pricing due to expensive LLM being used only for data annotation. At the same time, the key disadvantage of this method is the lack of ability to evaluate the correctness of annotations generated by LLM without additional spending on human annotators.

CONCLUSIONS

The scientific novelty of the obtained results is that the N-shot bilingual instruction prompting method for solving CBQA tasks in low-resource Ukrainian language with a low-to-zero amount of training data is first proposed. This method, compared to existing, utilizes large language models in a zero- or few-shot manner with instruction written in high-resource English to increase the model steerability and prediction quality, taking advantage of the fact that most large language models during the training stage have seen more text in English than in all other languages combined.

Based on the first method, a method of generative data annotation for the extractive model training was also proposed as a solution for low-resource CBQA. In contrast to existing solutions with only question or answer generation, this method leverages LLM as a data annotator to generate complete question-answer pairs for the given contexts, allowing the extractive model to be trained without annotated data.

The application of the N-shot bilingual instruction prompting method with the GPT-3.5 Turbo model and four examples (N=4) achieved a 28.3% (46.24 vs 36.04) and 16.6% (62.81 vs 54.34) relative increase for SQuAD EM and F1 metrics, a 22% (0.61 vs 0.50) and 17.5% (0.67 vs 0.57) relative gain for unigram BLEU and ROUGE, as well as 20.3% (0.71 vs 0.59) and 19% (0.75 vs 0.63) improvement for Partial and Any F1 compared to baseline zero-shot (N=0) monolingual method. Also, this method solves the key problem of CBQA tasks in low-resource languages, the lack of annotated training data, retaining 95.2% and 86.9% of baseline XLM-RoBERTa and DeBERTa models prediction quality according to SQuAD F1 metric, while using 2793.3 times less annotated examples (4 instead of 11173).

Training extractive models on generated annotations achieved 73.1% of the practical ceiling of these models trained on human annotations, spending only $5.79 on the labeling process. Moreover, combining these generative annotations with the existing training set increased XLM-RoBERTa's results by 1 point for SQuAD, N-gram, and index-based metrics.

Lastly, to the best of our knowledge, this paper provides the first comprehensive study and evaluation of publicly available large language models' capabilities in Ukrainian context-based question answering.

The practical significance is that the open-source experimentation framework implementing the two methods proposed in this paper for solving context-based question-answering tasks with large language models is developed. The proposed methods formed the basis of the developed program system for the fully automated development of task-agnostic language models.

In addition, the conducted experiments and their evaluation results demonstrate the possibility of applying these methods for building components of search engines and intelligent chatbot applications, as well as standalone general-domain CBQA systems with Ukrainian language support and a low-to-zero amount of annotated data for training.

The limitations of this research should be duly acknowledged. First, the study primarily focuses on a single context-based question-answering dataset and one specific low-resource language, Ukrainian. While the findings provide valuable insights, they may not be universally applicable to other low-resource languages or fully encompass the diverse range of real-world contexts, questions, and answers.

Second, the evaluation metrics used to assess the effectiveness of N-shot bilingual instruction prompting and generative data annotation for the extractive model training methods may not capture all aspects of performance, and further research into more nuanced evaluation methodologies is warranted.

The prospect for further research is to broaden the investigation scope beyond the context-based question-answering task, which has been the primary focus of this research. The ultimate objective is to create a comprehensive benchmark that thoroughly assesses the capabilities of large language models within the context of the Ukrainian natural language understanding and generation, providing a holistic perspective on their performance and adaptability.

ACKNOWLEDGEMENTS

We would like to extend our heartfelt gratitude to the OpenAI team for providing us with early access to the GPT family of models. We also appreciate the contributions of Meta AI for making the LLaMA model family openly accessible.

DOI 10.15588/1607-3274-2024-1-14

REFERENCES

1. Rajpurkar P. The Stanford Question Answering Leaderboard [Electronic resource]. Access mode: https://rajpurkar.github.io/SQuAD-explorer/
2. Devlin J., Chang M., Lee K. et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, *17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, 2–7 June 2019, proceedings. Stroudsburg, Association for Computational Linguistics, 2019, Vol. 1, pp. 4171–4186. DOI: 10.18653/V1/N19-1423
3. Conneau A., Lample G. Cross-lingual language model pretraining, *33rd International Conference on Neural Information Processing Systems*, Vancouver, 8–14 December 2019, proceedings. New York, Curran Associates Inc., 2019, pp. 7059–7069.
4. Conneau A., Khandelwal K., Goyal N. et al. Unsupervised Cross-lingual Representation Learning at Scale, *58th Annual Meeting of the Association for Computational Linguistics*, Online, 5–10 July 2020, proceedings. Stroudsburg, Association for Computational Linguistics, 2020, pp. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747
5. Bubeck S., Chandrasekaran V., Eldan R. et al. Sparks of Artificial General Intelligence: Early experiments with GPT-4. ArXiv preprint, 2023, Vol. 2303.12712.
6. Kojima T., Gu S., Reid M. et al. Large language models are zero-shot reasoners, *36th Annual Conference on Neural Information Processing Systems*, New Orleans, November 28 – December 9, 2022, proceedings. New York, Curran Associates Inc., 2022, Vol. 35, pp. 22199–22213.
7. Riloff E., Thelen M. A rule-based question answering system for reading comprehension tests, *2000 ANLP/NAACL Workshop on Reading comprehension tests as evaluation for computer-based language understanding systems*, Seattle, 4 May 2000, proceedings. Stroudsburg, Association for Computational Linguistics, 2000, Vol. 6, pp. 13–19. DOI: 10.3115/1117595.1117598
8. Radev D., Fan W., Qi H. et al. Probabilistic question answering on the web, *11th international conference on World Wide Web*, Honolulu, 7-11 May 2002, proceedings. New York, Association for Computing Machinery, 2022, pp. 408–419. DOI: 10.1145/511446.511500
9. Radev D., Prager J., Samn V. Ranking suspected answers to natural language questions using predictive annotation, *6th conference on Applied natural language processing*, Seattle, 29 April 2000, proceedings. Stroudsburg, Association for Computational Linguistics, 2000, pp. 150–157. DOI: 10.3115/974147.974168
10. Ruder S., Peters M., Swayamdipta S. et al. Transfer Learning in Natural Language Processing, *17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, 2–7 June 2019, proceedings. Stroudsburg, Association for Computational Linguistics, 2019, Tutorial Abstracts, pp. 15–18. DOI: 10.18653/v1/N19-5004
11. Howard J., Ruder S. Universal Language Model Fine-tuning for Text Classification, *56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, 15–20 July 2018, proceedings. Stroudsburg, Association for Computational Linguistics, 2018, Vol. 1, pp. 328–339. DOI: 10.18653/v1/P18-1031
12. Luo M., Hashimoto K., Yavuz S. et al. Choose Your QA Model Wisely: A Systematic Study of Generative and Extractive Readers for Question Answering, *Decoupling Logic from Knowledge: 1st Workshop on Semiparametric Methods in NLP*, Dublin, 27 May 2022, proceedings. Stroudsburg, Association for Computational Linguistics, 2022, pp. 7–22. DOI: 10.18653/v1/2022.spanlp-1.2
13. Guerreiro N., Voita E., Martins A. Looking for a Needle in a Haystack: A Comprehensive Study of Hallucinations in Neural Machine Translation, *17th Conference of the European Chapter of the Association for Computational Linguistics*, Dubrovnik, 2–6 May 2023, proceedings. Stroudsburg, Association for Computational Linguistics, 2023, pp. 1059–1075. DOI: 10.18653/v1/2023.eacl-main.75
14. Zheng S., Huang J., Chang K. C. Why Does ChatGPT Fall Short in Providing Truthful Answers? ArXiv preprint, 2023, Vol. 2304.10513.
15. Anil R., Dai A. M., Firat O. et al. PaLM 2 Technical Report. ArXiv preprint, 2023, Vol. 2305.10403.
16. Touvron H., Martin L., Stone K. et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. ArXiv preprint, 2023, Vol. 2307.09288.
17. Ye S., Hwang H., Yang S. et al. In-Context Instruction Learning. ArXiv preprint, 2023, Vol. 2302.14691.
18. Wei J., Wang X., Schuurmans D. et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models, *36th Annual Conference on Neural Information Processing Systems*, New Orleans, November 28 – December 9, 2022, proceedings. New York, Curran Associates Inc., 2022, Vol. 35, pp. 24824–24837.
19. Wei J., Bosma J., Zhao M. et al. Finetuned Language Models Are Zero-Shot Learners, *International Conference on Learning Representations*, Online, 25–29 April 2022, proceedings. ArXiv, 2022, Vol. 2109.01652.
20. Ruis L., Khan A., Biderman S. et al. Large language models are not zero-shot communicators, *37th Annual Conference on Neural Information Processing Systems*, New Orleans, 10–16 December 2023, proceedings. San Diego, NeurIPS, 2023.
21. Bang Y., Cahyawijaya S., Lee N. et al. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. ArXiv preprint, 2023, Vol. 2302.04023.
22. Brown T., Mann B., Ryder N. et al. Language models are few-shot learners, *34th Annual Conference on Neural Information Processing Systems*, Vancouver, 6–12 December, 2020, proceedings. New York, Curran Associates Inc., 2020, Vol. 33, pp. 1877–1901.
23. OpenAI. GPT-4 Technical Report. ArXiv preprint, 2023, Vol. 2303.08774.
24. Liang P., Bommasani R., Lee T. et al. Holistic Evaluation of Language Models, *Transactions on Machine Learning Research*, 2023. ArXiv, Vol. 2211.09110.
25. He P., Liu X., Gao J. et al. DeBERTa: Decoding-enhanced BERT with Disentangled Attention, *International Conference on Learning Representations*, Online, 3–7 May, 2021, proceedings. ArXiv, 2021, Vol. 2006.03654.
26. He P., Gao J., Chen W. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing, *International Conference on Learning Representations*, Online, 3–7 May, 2023, proceedings. ArXiv, 2023, Vol. 2111.09543.
27. Rajpurkar P., Zhang J., Lopyrev K. et al. SQuAD: 100,000+ Questions for Machine Comprehension of Text, *2016 Conference on Empirical Methods in Natural Language Processing*, Austin, 1–5 November 2016, proceedings. Stroudsburg, Association for Computational Linguistics, 2016, pp. 2383–2392. DOI: 10.18653/v1/D16-1264
28. Papineni K., Roukos S., Ward T. et al. Bleu: a Method for Automatic Evaluation of Machine Translation, *40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, 6–12 July 2002, proceedings. Stroudsburg, Association for Computational Linguistics, 2002, pp. 311–318. DOI: 10.3115/1073083.1073135
29. Lin C. ROUGE: A Package for Automatic Evaluation of Summaries, *Text Summarization Branches Out: ACL-04 Workshop*, Barcelona, 25–26 July 2004, proceedings. Stroudsburg, Association for Computational Linguistics, 2004, pp. 74–81.

DOI 10.15588/1607-3274-2024-1-14

30. Yadan O. Hydra – A framework for elegantly configuring complex applications [Electronic resource]. Access mode: https://github.com/facebookresearch/hydra.
31. Rajpurkar P., Jia R., Liang P. Know What You Don't Know: Unanswerable Questions for SQuAD, *56th Annual Meeting of the Association for Computational Linguistics*, Melbourne, 15–20 July 2018, proceedings. Stroudsburg, Association for Computational Linguistics, 2018, Vol. 2, pp. 784–789. DOI: 10.18653/v1/P18-2124
32. Ivanyuk-Skulskiy B., Zaliznyi A., Reshetar O. et al. ua_datasets: a collection of Ukrainian language datasets [Electronic resource]. Access mode: https://github.com/fido-ai/ua-datasets.
33. Krisilov V., Komleva N. Analysis and evaluation of competence of information sources in problems of intellectual data processing, *Problemele energeticii regionale*, 2019, Vol. 40, Issue 1, pp. 91–104. DOI: 10.5281/zenodo.3239185.
34. Ahuja K., Diddee H., Hada R. et al. MEGA: Multilingual Evaluation of Generative AI. ArXiv preprint, 2023, Vol. 2303.12528.

Received 18.12.2023.
Accepted 29.01.2024.

УДК 004.912

UA-LLM: ПОКРАЩЕННЯ ВІДПОВІДІ НА ЗАПИТАННЯ ЗА КОНТЕКСТОМ УКРАЇНСЬКОЮ МОВОЮ З ВИКОРИСТАННЯМ ВЕЛИКИХ МОВНИХ МОДЕЛЕЙ

Сиром'ятніков М. В. – аспірант кафедри Інженерії програмного забезпечення Національного університету «Одеська Політехніка», Одеса, Україна.
Рувінська В. М. – канд. техн. наук, професор кафедри Інженерії програмного забезпечення Національного університету «Одеська Політехніка», Одеса, Україна.

АНОТАЦІЯ

Актуальність. Відповідь на запитання за контекстом, фундаментальне завдання обробки природної мови, вимагає глибокого розуміння мови. Будучи складною задачею, вона є невід'ємною частиною сучасних пошукових систем, інтелектуальних помічників, чат-ботів і всієї сфери розмовного штучного інтелекту. У той час як англійська, китайська та інші широко поширені мови налічують велику кількість наборів даних, алгоритмів і тестів, українська – з її багатою лінгвістичною спадщиною та складним синтаксисом залишається серед малоресурсних мов, що ще більше ускладнює задачу відповіді на запитання за контекстом.

Мета роботи. Мета роботи полягає у розробці та оцінюванні методів на базі великих мовних моделей, об'єднаних у фреймворк для вирішення проблеми низькоресурсності задачі відповіді на запитання за контекстом в українській мові.

Метод. Простий, але гнучкий фреймворк для використання великих мовних моделей, розроблений в рамках цієї дослідницької роботи, висвітлює два ключові методи для вирішення проблеми даних у задачі відповіді на запитання за контекстом, запропоновані та оцінені в цій статті. Перший метод використовує Zero-shot і Few-shot learning – дві основні гілки N-shot learning, де N відповідає кількості тренувальних прикладів, для побудови двомовної стратегії підказок на основі інструкцій для роботи з мовними моделями у екстрактивний спосіб (пошук сегменту відповіді у контексті) замість їхньої природної генеративної поведінки (генерація відповіді на основі контексту). Другий запропонований метод базується на першому, але замість простої відповіді на запитання мовна модель розмічає вхідний контекст шляхом генерації пар запитання-відповідь. Отримані синтетичні дані використовуються для тренування екстрактивної моделі. У цій статті розглядається як навчання на основі аугментації даних, коли вже є деякі розмічені дані, так і повністю синтетичне навчання, коли дані відсутні. Ключовою перевагою запропонованих методів є можливість отримати якість передбачень на рівні натренованих екстрактивних моделей навіть без дорогого та довготривалого процесу розмітки даних людьми.

Результати. Два запропонованих методи для розв'язання проблеми недостатньої кількості тренувальних даних у задачі відповіді на запитання за контекстом для української мови було реалізовано та об'єднано в гнучкий фреймворк для роботи з великими мовними моделями .

Висновки. Дана робота демонструє результати всеосяжного дослідження рівня розуміння мови моделями OpenAI GPT-3.5, OpenAI GPT-4, Cohere Command і Meta LLaMa-2 на прикладі вирішення задачі відповіді на запитання за контекстом для низькоресурсної української мови. Ретельна оцінка запропонованих методів за різноманітним набором показників доводить їх ефективність, розкриваючи можливість побудови компонентів пошукових систем, інтелектуальних чат-ботів та автономних систем відповіді на запитання з підтримкою української мови та близькою до нуля кількістю розмічених тренувальних даних. Перспектива подальших досліджень полягає у розширенні сфери застосування від завдання відповіді на запитання за контекстом, розглянутого у цій статті, до усіх основних задач розуміння природної мови з кінцевою метою встановлення повного тесту для оцінювання можливостей великих мовних моделей в українській мові.

КЛЮЧОВІ СЛОВА: велика мовна модель, відповідь на запитання, few-shot learning, генеративна розмітка даних.

ЛІТЕРАТУРА

1. Rajpurkar P. The Stanford Question Answering Leaderboard [Electronic resource] / P. Rajpurkar. – Access mode: https://rajpurkar.github.io/SQuAD-explorer/.
2. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding / [J. Devlin, M. Chang, K. Lee et al.] // Human Language Technologies : 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics, Minneapolis, 2–7 June 2019 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2019. – Vol. 1. – P. 4171–4186. DOI: 10.18653/V1/N19-1423
3. Conneau A. Cross-lingual language model pretraining / A. Conneau, G. Lample // Neural Information Processing Systems : 33rd International Conference, Vancouver, 8–14 December 2019 : proceedings. – New York : Curran Associates Inc., 2019. – P. 7059–7069.
4. Unsupervised Cross-lingual Representation Learning at Scale / [A. Conneau, K. Khandelwal, N. Goyal et al.] // 58th Annual Meeting of the Association for Computational Linguistics, Online, 5–10 July 2020 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2020. – P. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747
5. Sparks of Artificial General Intelligence: Early experiments with GPT-4 / [S. Bubeck, V. Chandrasekaran, R. Eldan et al.] // ArXiv preprint. – 2023. – Vol. 2303.12712.
6. Large language models are zero-shot reasoners / [T. Kojima, S. Gu, M. Reid et al.] // Advances in Neural Information Processing Systems : 36th Annual Conference on Neural Informa-

DOI 10.15588/1607-3274-2024-1-14

tion Processing Systems, New Orleans, November 28 – December 9, 2022 : proceedings. – New York : Curran Associates Inc., 2022. – Vol. 35. – P. 22199–22213.
7. Riloff E. A rule-based question answering system for reading comprehension tests / E. Riloff, M. Thelen // Reading comprehension tests as evaluation for computer-based language understanding systems : 2000 ANLP/NAACL Workshop, Seattle, 4 May 2000 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2000. – Vol. 6. – P. 13–19. DOI: 10.3115/1117595.1117598
8. Probabilistic question answering on the web / [D. Radev, W. Fan, H. Qi et al.] // 11th international conference on World Wide Web, Honolulu, 7–11 May 2002 : proceedings. – New York : Association for Computing Machinery, 2022. – P. 408–419. DOI: 10.1145/511446.511500
9. Radev D. Ranking suspected answers to natural language questions using predictive annotation / D. Radev, J. Prager, V. Samn // 6th conference on Applied natural language processing, Seattle, 29 April 2000 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2000. – P. 150–157. DOI: 10.3115/974147.974168
10. Transfer Learning in Natural Language Processing / [S. Ruder, M. Peters, S. Swayamdipta et al.] // Human Language Technologies : 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics, Minneapolis, 2–7 June 2019 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2019. – Tutorial Abstracts. – P. 15–18. DOI: 10.18653/v1/N19-5004
11. Howard J. Universal Language Model Fine-tuning for Text Classification / J. Howard, S. Ruder // 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, 15–20 July 2018 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2018. – Vol. 1. – P. 328–339. DOI: 10.18653/v1/P18-1031
12. Choose Your QA Model Wisely: A Systematic Study of Generative and Extractive Readers for Question Answering / [M. Luo, K. Hashimoto, S. Yavuz et al.] // Decoupling Logic from Knowledge : 1st Workshop on Semiparametric Methods in NLP, Dublin, 27 May 2022 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2022. – P. 7–22. DOI: 10.18653/v1/2022.spanlp-1.2
13. Guerreiro N. Looking for a Needle in a Haystack: A Comprehensive Study of Hallucinations in Neural Machine Translation / N. Guerreiro, E. Voita, A. Martins // 17th Conference of the European Chapter of the Association for Computational Linguistics, Dubrovnik, 2–6 May 2023 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2023. – P. 1059–1075. DOI: 10.18653/v1/2023.eacl-main.75
14. Zheng S. Why Does ChatGPT Fall Short in Providing Truthful Answers? / S. Zheng, J. Huang, K. C. Chang // ArXiv preprint. – 2023. – Vol. 2304.10513.
15. PaLM 2 Technical Report / [R. Anil, A. M. Dai, O. Firat et al.] // ArXiv preprint. – 2023. – Vol. 2305.10403.
16. Llama 2: Open Foundation and Fine-Tuned Chat Models / [H. Touvron, L. Martin, K. Stone et al.] // ArXiv preprint. – 2023. – Vol. 2307.09288.
17. In-Context Instruction Learning / [S. Ye, H. Hwang, S. Yang et al.] // ArXiv preprint. – 2023. – Vol. 2302.14691.
18. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models / [J. Wei, X. Wang, D. Schuurmans et al.] // Advances in Neural Information Processing Systems : 36th Annual Conference on Neural Information Processing Systems, New Orleans, November 28 – December 9, 2022 : proceedings. – New York : Curran Associates Inc., 2022. – Vol. 35. – P. 24824–24837.
19. Finetuned Language Models Are Zero-Shot Learners / [J. Wei, J. Bosma, M. Zhao et al.] // International Conference on Learning Representations, Online, 25–29 April 2022 : proceedings. – ArXiv, 2022. – Vol. 2109.01652.
20. Large language models are not zero-shot communicators / [L. Ruis, A. Khan, S. Biderman et al.] // Neural Information Processing Systems : 37th Annual Conference, New Orleans, 10–16 December 2023 : proceedings. – San Diego: NeurIPS, 2023.
21. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity / [Y. Bang, S. Cahyawijaya, N. Lee et al.] // ArXiv preprint. – 2023. – Vol. 2302.04023.
22. Language models are few-shot learners / [T. Brown, B. Mann, N. Ryder et al.] // Neural Information Processing Systems : 34th International Conference, Vancouver, 6–12 December, 2020 : proceedings. – New York : Curran Associates Inc., 2020. – Vol. 33. – P. 1877–1901.
23. OpenAI. GPT-4 Technical Report / OpenAI // ArXiv preprint. – 2023. – Vol. 2303.08774.
24. Holistic Evaluation of Language Models / [P. Liang, R. Bommasani, Lee. T et al.] // Transactions on Machine Learning Research. – 2023. ArXiv. – Vol. 2211.09110.
25. DeBERTa: Decoding-enhanced BERT with Disentangled Attention / [P. He, X. Liu, J. Gao et al.] // International Conference on Learning Representations, Online, 3–7 May, 2021 : proceedings. – ArXiv, 2021. – Vol. 2006.03654.
26. He P. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing / P. He, J. Gao, W. Chen // International Conference on Learning Representations, Online, 3–7 May, 2023 : proceedings. – ArXiv, 2023. – Vol. 2111.09543.
27. SQuAD: 100,000+ Questions for Machine Comprehension of Text / [P. Rajpurkar, J. Zhang, K. Lopyrev et al.] // 2016 Conference on Empirical Methods in Natural Language Processing, Austin, 1–5 November 2016 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2016. – P. 2383–2392. DOI: 10.18653/v1/D16-1264
28. Bleu: a Method for Automatic Evaluation of Machine Translation / [K. Papineni, S. Roukos, T. Ward et al.] // 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, 6–12 July 2002 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2002. – P. 311–318. DOI: 10.3115/1073083.1073135
29. Lin C. ROUGE: A Package for Automatic Evaluation of Summaries / C. Lin // Text Summarization Branches Out: ACL-04 Workshop, Barcelona, 25–26 July 2004 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2004. – P. 74–81.
30. Yadan O. Hydra – A framework for elegantly configuring complex applications [Electronic resource] / O. Yadan. – Access mode: https://github.com/facebookresearch/hydra.
31. Rajpurkar P. Know What You Don't Know: Unanswerable Questions for SQuAD / P. Rajpurkar, R. Jia, P. Liang // 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, 15–20 July 2018 : proceedings. – Stroudsburg : Association for Computational Linguistics, 2018. – Vol. 2. – P. 784–789. DOI: 10.18653/v1/P18-2124
32. Ivanyuk-Skulskiy B. ua_datasets: a collection of Ukrainian language datasets [Electronic resource] / [B. Ivanyuk-Skulskiy. A. Zaliznyi, O. Reshetar et al.]. – Access mode: https://github.com/fido-ai/ua-datasets.
33. Krisilov V. Analysis and evaluation of competence of information sources in problems of intellectual data processing / V. Krisilov, N. Komleva // Problemele energeticii regionale. – 2019. Vol. 40, Issue 1. – P. 91–104. DOI: 10.5281/zenodo.3239185.
34. MEGA: Multilingual Evaluation of Generative AI / [K. Ahuja, H. Diddee, R. Hada et al.] // ArXiv preprint. – 2023. – Vol. 2303.12528.

DOI 10.15588/1607-3274-2024-1-14