UDC 004.021:004.75

RESEARCH OF LARGE LANGUAGE MODELS FOR TEXT GENERATION AND THEIR PRACTICAL APPLICATION

A.N. Markov
Senior Lecturer of the Department of Informatics, Deputy Head of the Center for Informatization and Innovative Developments of the Belarusian State University of Informatics and Radioelectronics
a.n.markov@bsuir.by

M.M. Zyryanova
Student of the Faculty of Computer Systems and Networks of the Belarusian State University of Informatics and Radioelectronics.
z.y.r.y.a.n.o.v.a@mail.ru

A.E. Asadchy
Student of the Faculty of Computer Systems and Networks of the Belarusian State University of Informatics and Radioelectronics.
aleh.asadchy@gmail.com

A. Markov
Graduated from Belarusian State University of Informatics and Radioelectronics. The field of scientific activity includes such areas as computing systems, cloud computing, distributed computing systems, load balancing of computer systems.

M. Zyryanova
A fourth-year student of the Belarusian State University of Informatics and Radioelectronics. The scope of scientific activity includes such areas as large language models and their additional training, as well as effective implementation.

A. Asadchy
A fourth-year student of the Belarusian State University of Informatics and Radioelectronics. The scope of scientific activity includes such areas as large language models and their additional training, as well as effective implementation.

Abstract. This research is aimed at the study of large language models designed for translation and text generation. In the course of the study, transformers of the Text2TextGeneration type were considered, their comparative analysis and testing of various situations were carried out. Large language models were also implemented into an existing software product.

Keywords: large language models, transformers, Text2TextGeneration, fine-tuning, hyperparameters, neural network, Big Data.

Introduction. The field of modern information technologies is going through a period of rapid development, which the average user was able to notice a few years ago, with the advent of widespread use of large language models. In recent years, this area has begun to develop more and more. Large language models based on deep learning and the mechanism of self-attention are able to generate texts of various lengths and contents, which can sometimes be difficult to distinguish from those written by a person. In addition, such models have already become assistants to humans, in many respects to programmers, in their work.

In the context of this research, the aim is to investigate such large language models and their potential implementation in various fields and solutions, including automatic translation, text generation and rewriting, grammar checking, etc.

Research methodologies. The key tasks are translating text from American English to British, translating from English to German, improving the quality of the text, and checking grammar and spelling. In the course of studying various platforms, it was decided to use the Huggingface platform, which provides wide access to various kinds of models and their images absolutely free of charge.

The following criteria were developed to evaluate and select large language models:

– what was taught on what it was taught;

– dataset volume;

– perplexity (how well the model predicts the text);

– relevance (relevance, i.e. the correspondence of the answer to what the person asked);

– coherence;

– overall quality;

– accuracy (accuracy of the data directly reported, i.e. the truthfulness of the data in relation to itself without taking into account the context);

– latency (delay in response from the network, usually delay to the first character);

– context;

– vulnerability of language models;

- Rapid deployment and testing of models.

We will consider and conduct a comparative analysis of large language models in the field of translation from American English to British. Two models were chosen: t5-base-us-to-uk-english and autonlp-US-to-UK-604417040.

Table 1. Comparative Analysis of Models for Translation from American English to British

Criterion	t5-base-us-to-uk-english	autonlp-US-to-UK-604417040
Dataset	249525 American and British sentences	604417040 sentences with American and British spelling
Dataset Volume	249525 offers	604417040 offers
Perplexity	5.2	4.8
Relevance	Average	High
Coherence	High	High
Overall quality	High	High
Accuracy	High	High
Latency	100-200 ms	50-100 ms
Context	Takes context into account	Takes context into account
Vulnerability of language models	Susceptible to bias, can generate incorrect information	Susceptible to bias, can generate incorrect information
Rapid deployment and testing	Fast	Fast

Both models have their advantages and disadvantages. The autonlp-US-to-UK-604417040 model has lower perplexity, higher relevance, and lower latency than t5-base-us-to-uk-english. However, /t5-base-us-to-uk-english has a higher coherence. In this case, it makes sense to have a

high consistency model for related translation of sentences rather than individual words, so in this case the t5-base-us-to-uk-english model was chosen.

We will consider and conduct a comparative analysis of large language models in the field of translation from English to German. Models such as small100, wmt19-en-de and opus-mt-en-de were chosen.

Table 2. Comparative Analysis of Models for Translation from English to German

Criterion	small100	wmt19-en-de	Opus-MT-en-DE
Dataset	100 languages (small sets)	WMT'19 (English-German)	OPUS (multilingual)
Dataset Volume	1.5M offers	4.5M offers	65M offers
Perplexity	45.0	27.0	21.0
Relevance	Low	Average	High
Coherence	Average	High	Very high
Overall quality	Low	Average	High
Accuracy	Low	Average	High
Latency	Low	Average	High
Context	Limited	Average	High
Vulnerability of language models	High	Average	Low
Rapid deployment and testing	High	Average	Low

We can conclude that small100 is the fastest and simplest model, but it has low accuracy and quality. At the same time, wmt19-en-de is a model with higher quality than small100, but it is less versatile. In contrast to the above-mentioned models, opus-mt-en-de is the most accurate and high-quality model, but it is the slowest and most difficult to deploy.

Despite the low speed of the opus-mt-en-de, its precision makes it ideal for the task at hand.

We will consider and conduct a comparative analysis of large language models in the field of translation from English to German. Models such as t5_Grammar_Checker, t5-Base-Grammar-Correction, BERT-Grammar-Checker were chosen.

Table 3. Benchmarking Models for Grammar Checking

Criterion	T5_Grammar_Checker	T5-Base-Grammar-Correction	BERT Grammar Checker
What they taught	Stack Overflow, Reddit, Wikipedia	Stack Overflow, Reddit	Google Books
Dataset Volume	100M+ words	10M+ words	100B+ words
Perplexity	10.2	12.5	8.9
Relevance	High	Average	Low
Coherence	High	Average	Low
Overall Quality	High	Average	Low
Accuracy	High	Average	Low

End of table 3

Criterion	T5_Grammar_Checker	T5-Base-Grammar-Correction	BERT Grammar Checker
Latency	200-300 ms	100-200 ms	50-100 ms
Context	Takes context into account	Takes context into account	Doesn't take context into account
Vulnerability of language models	Vulnerable to bias, noise, attacks	Vulnerable to bias, noise, attacks	Vulnerable to bias, noise, attacks
Rapid deployment and testing	Average	High	High

T5_Grammar_Checker is the best model out of the three. It has high perplexity, relevance, coherence, overall quality, accuracy and latency. T5-Base-Grammar-Correction is a faster model, but it is less accurate and relevant. BERT-Grammar-Checker is the fastest model, but it is the least accurate, relevant, and consistent. The chatgpt_paraphraser_on_T5_base model was chosen as a model for improving the quality of the text due to the good result during the prompting testing.

The latter was the selection of a model to improve the text. There was a slightly different approach to the choice, since we evaluated the transformed text not as a set of some parameters, but as its content and semantic value. After making comparisons on certain prompts, it was decided to choose the first one from the two models humarin/chatgpt_paraphraser_on_T5_base and ramsrigouthamg/t5_sentence_paraphraser due to the better output text.

Architecture and training of large language models. One of the important parts of the development of large language models was the invention of new neural network architectures. In particular, a significant breakthrough in this area was the mechanism of transformers, which uses self-attention.

The transformers model has a significant difference from the models that preceded it: it efficiently processes text without using either recurrent or convolutional layers, which in turn allows for parallel learning and directly increases the speed of learning.

The architecture of the transformer can be represented in the form of the following layers:

– Embeddings Layers. Converts each word in the input sequence into a vector representation. Word vectors and positions can be combined to convey information about the sequence of words and their relative positions.

– Multi-Head Self-Attention Layers. These layers draw the model's attention to different parts of the input sequence. And the Multi-Head mechanism allows the model to look at the input sequence from different perspectives, so the model better understands the context.

– Feed-Forward NN Nonlinear activation functions for processing data from the self-attention layer.

– Normalization and Residual Connection Layers. They are used to improve the stability of learning and prevent gradient attenuation [1].

The architecture of the transformer is shown in Figure 1.

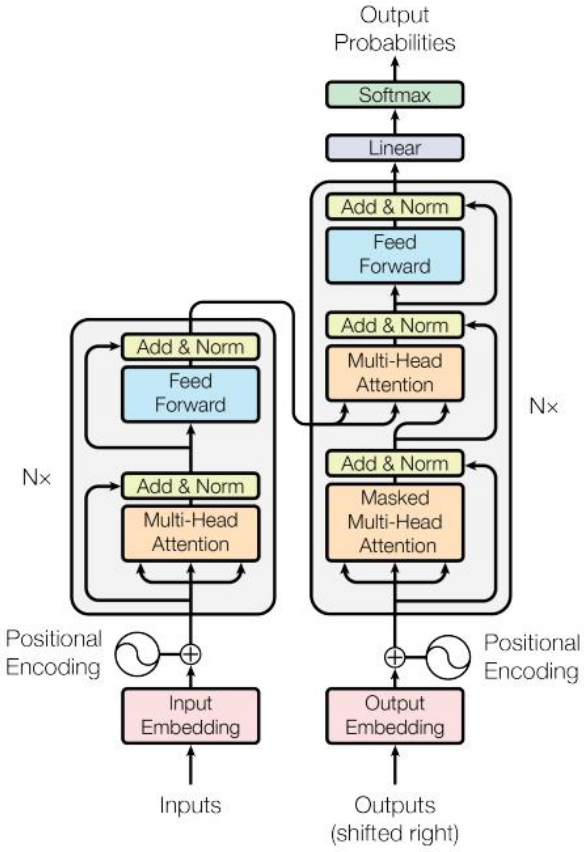

Figure 1. Transformer Architecture

The principle of operation is as follows: the encoder receives words as input, converts them into embeddings by running each word through layers of fully-connected layers, some through shortcut connections, and most importantly through Multi-head attention, i.e. through several attention mechanisms in parallel. Thanks to this mechanism, the model pays attention to the positions of words in a sentence if they are important. Next comes the decoder: it runs one word at a time, by receiving the last word as input and giving the next one. Here again, the self-attention mechanism is used to work with words and their positions, but here we use words that have already been encoding. At the end, there is a regular softmax, which already gives out the probabilities of words. During decoding, each vector interacts with all the others. During decoding, each next word interacts with the previous ones and with the encoder vectors [2].

The main advantage of such models is the ability to refer to any word, regardless of the length of the context, as well as the possibility of parallel learning.

Speaking about the learning process, there are several stages:

– Pre-training. Here, the model is trained on a large amount of text data without a specific target. The model simply learns general knowledge about the language and its patterns.

– Fine-tuning. Once a model has been trained on general data, it's time to train it for a specific task where it will be used, such as translation or text generation.

– Hyperparameter tuning is also an important step in training the model.

Use of large language models. Due to their efficiency and ability to work with large long, large language models are applicable in different fields.

One of the most popular areas is text generation itself. This feature is often used by copywriters, as it is well suited for generating articles, reviews, news, or even literary works. This function is highly dependent on hyperparameters, and if you set them up correctly, you can

actually get almost literary works. This feature can also be used to generate content for marketing purposes, advertising slogans, product descriptions, and more.

Automatic translation. Models of this type are successfully used in systems of this type. They are capable of generating high-quality translations between different languages without additional customization in a specific language pair. The best example would be YandexTranslate, DeepL, which openly states the use of transformers architecture within its models.

Summarization of text. Thanks to the model's ability to capture context even in large amounts of text, llms can be used to automatically summarize long texts, and they can also generate concise and informative summaries of articles of various genres, and in general, almost any document.

One of the most popular applications of such models is dialogue systems and chatbots. Due to the model's ability to generate natural and understandable answers to user questions, such systems can be used to communicate with users in their native language. They can be embedded in a huge number of websites, applications, and services. The most popular models at the moment are ChatGPT, Bard. Every day, a huge number of people use it for a variety of purposes: from writing scientific articles and using it for all sorts of household trifles to helping IT specialists in their work.

Thus, large language models are not just a theoretically promising direction of development, but literally practically successful. At the moment, almost every large company wants to have its own model with completely different areas of application, although usually for internal needs, but this shows that this type of model is successful and, moreover, necessary [3].

Implementation of large language models. A topic of direct interest to us was the introduction of large language solutions into ready-made projects in order to increase their efficiency and convenience. Such a project was a CV generation service, which was supplemented by another service containing Transformers.

The main advantage of the project is the use of ready-made transformers models trained on big data and already familiar with the language with which they will work. Using already trained models allows you to save a lot of money and man-hours and focus your efforts exclusively on fine-tuning the model, tuning hyperparameters, as well as role-based prompting (the technique of making the right query on the model), which will lead to a significant increase in performance.

After comparing the models, a service was created that had several endpoints and performed the following functions:

– Paraphrasing with enhancement of written text – beautify.

– Translation from English to German.

– Translation from American English to British in one-sentence and multi-sentence mode.

– Checking the grammar of the written text.

Hyperparameters and their configuration. One of the most important things when working with large language models is hyperparameters, which can be adjusted to improve the performance of the pipeline. The most common hyperparameters are the following:

– input_ids:

– Description: A tensor of input token identifiers representing the input text to be generated.

– Type: torch.Tensor.

– Example: input_ids.

– temperature:

– Description: A temperature parameter to control the degree of randomness in the lasing. A high temperature (greater than 1.0) makes the lasing more random, while a low temperature (less than 1.0) makes it more deterministic.

– Type: float.

– Example: temperature=0.7.

– repetition_penalty:

– Description: Penalty for repeating tokens. A value greater than 1.0 increases the likelihood of rejecting repeated phrases.

– Type: float.

– Example: repetition_penalty=1.2.

– num_return_sequences:

– Description: The number of alternate generation sequences that will be returned.

– Type: int.

– Example: num_return_sequences=3.

– no_repeat_ngram_size:

– Description: The maximum n-gram size to be prevented during generation (no repetitions).

– Type: int.

– Example: no_repeat_ngram_size=2.

– num_beams:

– Description: The number of beams to generate. A higher number of beams can improve the quality of generation, but slow down the process.

– Type: int.

– Example: num_beams=5.

– num_beam_groups:

– Description: Number of beam groups. Ray groups help control the diversity of lasing within each group.

– Type: int.

– Example: num_beam_groups=3.

– max_length:

– Description: The maximum length of the generated sequence (including input tokens). Limits the length of the generation.

– Type: int.

– Example: max_length=100.

– diversity_penalty:

– Description: A parameter that affects the degree of diversity of the sequences generated. A value greater than 1.0 increases variety.

– Type: float.

– Example: diversity_penalty=1.5.

Changing each of these parameters can affect the text generation process. Changing the temperature towards higher values makes the generation more random and creative, but can also result in less crisp and less cohesive text. Increasing repetition_penalty can help reduce repetitions in generation. Increasing num_return_sequences gives more alternative generation options. Increasing num_beams and num_beam_groups improves generation quality and variety, but can also slow down the process. Increasing the max_length limits the length of the generation. Changing diversity_penalty affects the degree of diversity of the alternate sequences generated.

Evaluate the performance and quality of models. Let's take a look at how models work on test input data. Figure 2 illustrates how the error correction model works.

Input	Output
See you s00n	See you soon!
I is a trainee at iTechArt Group right now.	I am a trainee at iTechArt Group right now.
I have ability of writing, creating content, to build communication.	I have the ability to write, create content, and build communication.

Figure 2. How the Error Correction Model Works

Figure 3 shows how the American English to British translation model works.

Input	Output
The intermission between graduation and finding the first job was 3 months.	The interval between graduation and finding the first job was 3 months.
I studied math at school.	I studied maths at school.
I have gotten serious about my future career.	I have got serious about my future career.

Figure 3. How the Translation Model Works

Below are the options for the text rewriting model. According to the design, the model should give us three different versions of the rewritten text per sentence. In particular, the model should work well regardless of the length of the input text. This is illustrated in Figures 4 and 5.

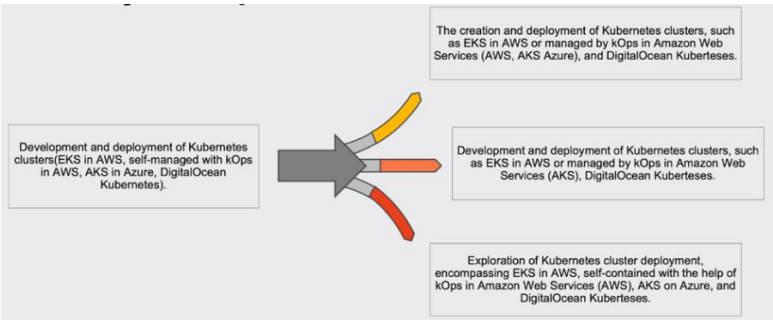

Figure 4. How the text rewriting model works

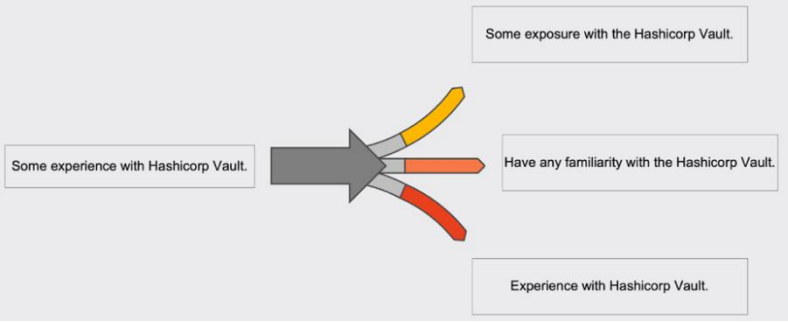

Figure 5. The Single Sentence Text Rewriting Model Works

Disadvantages. However, like everything based on neural networks, the current one has drawbacks. Because it is impossible to absolutely completely learn a model and make it flawless. In our specific examples, the easiest way to see this is in the text rewrite model: since all text-2-text models work on the principle of building text based on maximum probability, there are often mishaps (Fig. 6).

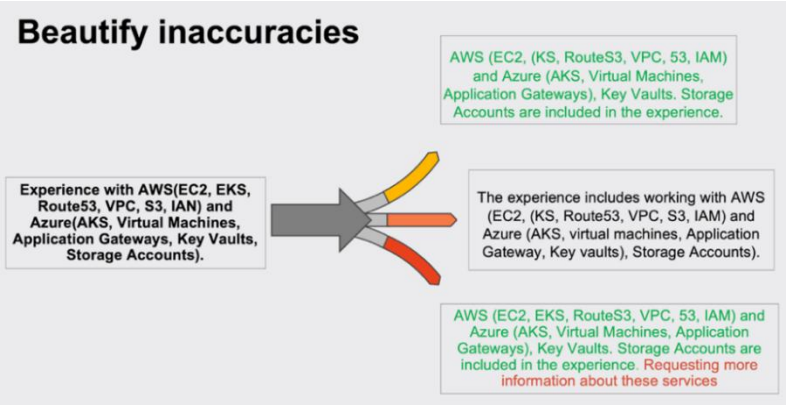

Figure 6. Example of possible errors

The same applies to other models, as, for example, when translating into another language, the names of programming languages can be translated, like python, although in our case this is incorrect. All such cases need to be corrected, which can be done by retraining the model on specific data, for example, by taking the model of a German translator who learned the general base of the language, and then learning it in words from the everyday life of an IT specialist. So we used transfer training.

Transfer learning is a machine learning technique that allows you to reuse a model trained on one task and fine-tune it for a related but different task.

There are different approaches to transfer learning, but one of them, finetuning, is particularly widespread. In this approach, the team takes a pre-trained model and removes or retrains the last layers of that model to focus on a new, similar task.

The advantages of such a model are the use of an existing knowledge base, increased model performance, and low cost.

Performance. Speaking of performance, it's worth noting that we've achieved good performance of the pre-trained models by breaking down the task into smaller ones, namely, in the places of our application, where possible, we send single applications to the service with models, for example, a certain record of the skill of the developer who then passes this model, which allowed us to minimize the response time of the model. Figure 7 shows the average result of our model tests.

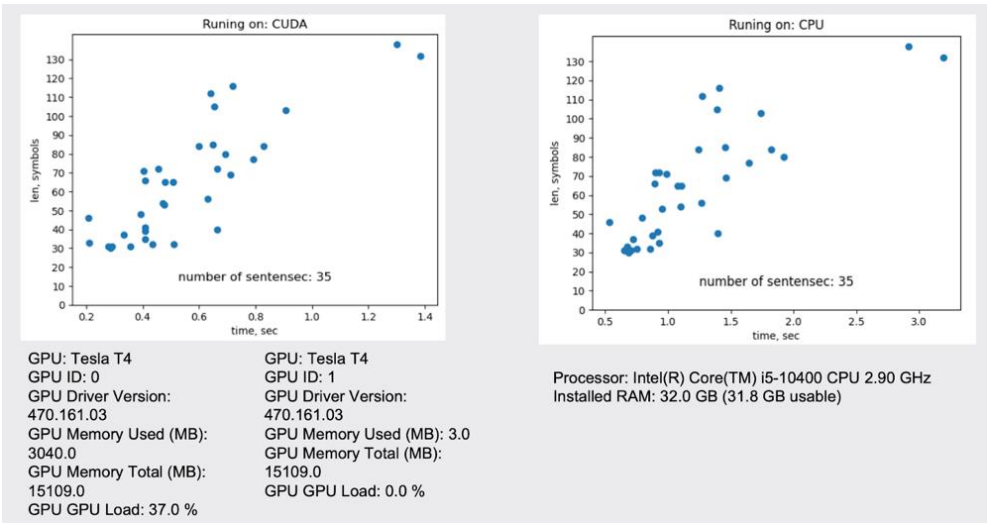

Figure 7. Model Performance

Conclusion. Thus, after adding a service using language models to the existing application, the efficiency of CV writing by the company's employees has increased, the time that an employee spends on a CV has decreased, and most importantly, the quality of the CV has increased and the number of grammatical errors has decreased, and the CV can now be translated into another language at the click of one button, which eliminates the need to sit and translate it by hand.

Reference list

[1] S. Samarasinghe. Neural Networks for Applied Sciences and Engineering, 2006.
[2] M.Nielsen. Neural Networks and Deep Learning, 2019.
[3] C.Olah. Deep Learning, NLP, and Representations, 2014.

author's contribution

Alexey Markov – the head of the study, responsible for setting the tasks and goals of the scientific work.

Maria Zyryanova – study of large language models, implementation and improvement of models responsible for translation from British English into American and grammar checking.

Aleh Asadchy – study of large language models, implementation and improvement of models responsible for translation from English into German and rewriting.

ИССЛЕДОВАНИЕ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ ДЛЯ ГЕНЕРАЦИИ ТЕКСТОВ И ИХ ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ

А.Н. Марков
Старший преподаватель кафедры информатики, заместитель начальника Центра информатизации и инновационных разработок Белорусского государственного университета информатики и радиоэлектроники.

М.М. Зырянова
Студентка факультета компьютерных систем и сетей Белорусского государственного университета информатики и радиоэлектроники.

О.Э. Осадчий
Студент факультета компьютерных систем и сетей Белорусского государственного университета информатики и радиоэлектроники.

Аннотация. Данное исследование направлено на изучение больших языковых моделей, предназначенных для перевода и генерации текста. В ходе исследования были рассмотрены трансформаторы типа *Text2TextGeneration,* проведен их сравнительный анализ и тестирование различных ситуаций. Большие языковые модели также были внедрены в существующий программный продукт.

Ключевые слова: большие языковые модели, трансформеры, *Text2TextGeneration*, *fine-tuning*, гиперпараметры, нейронные сети, *Big Data*.