Information systems. Computer sciences. Issues of information security

Информационные системы. Информатика. Проблемы информационной безопасности

UDC 004.8
https://doi.org/10.32362/2500-316X-2025-13-3-21-43
EDN QKUGFZ

REVIEW ARTICLE

Knowledge injection methods in question answering

Daniil V. Radyush @

ITMO University, Saint Petersburg, 197101 Russia
@ *Corresponding author, e-mail: daniil.radyush@gmail.com*

• **Submitted:** 26.06.2024 • **Revised:** 13.02.2025 • **Accepted:** 27.03.2025

Abstract

Objectives. Despite the recent success of large language models, which are now capable of solving a wide range of tasks, a number of practical issues remain unsolved. For example, users of systems providing question answering (QA) services may experience a lack of commonsense knowledge and reasoning proficiency. The present work considers knowledge injection methods as a means of providing functional enhancements to large language models by providing necessary facts and patterns from external sources.

Methods. Knowledge injection methods leveraged in relevant QA systems are classified, analyzed, and compared. Self-supervised learning, fine-tuning, attention mechanism and interaction tokens for supporting information injection are considered along with auxiliary approaches for emphasizing the most relevant facts.

Results. The reviewed QA systems explicitly show the accuracy increase on the CommonsenseQA benchmark compared to pretrained language model baseline due to knowledge injection methods exploitation. At the same time, in general the higher results are related to knowledge injection methods based on language models and attention mechanism.

Conclusions. The presented systematic review of existing external knowledge injection methods for QA systems confirms the continuing validity of this research direction. Such methods are not only capable of increasing the accuracy of QA systems but also mitigating issues with interpretability and factual obsolescence in pretrained models. Further investigations will be carried out to improve and optimize different aspects of the current approaches and develop conceptually novel ideas.

Keywords: deep learning, nature language processing, question answering system, knowledge base, graph neural networks, knowledge injection

For citation: Radyush D.V. Knowledge injection methods in question answering. *Russian Technological Journal.* 2025;13(3):21–43. https://doi.org/10.32362/2500-316X-2025-13-3-21-43, https://www.elibrary.ru/QKUGFZ

Financial disclosure: The author has no financial or proprietary interest in any material or method mentioned.

The author declares no conflicts of interest.

ОБЗОР

Методы интеграции знаний для разработки вопросно-ответных систем

Д.В. Радюш [@]

Национальный исследовательский университет ИТМО, Санкт-Петербург, 197101 Россия
[@] *Автор для переписки, e-mail: daniil.radyush@gmail.com*

• **Поступила:** 26.06.2024 • **Доработана:** 13.02.2025 • **Принята к опубликованию:** 27.03.2025

Резюме

Цели. Несмотря на наблюдаемые в последние несколько лет успехи больших языковых моделей, которые способны решать широкий перечень задач, ряд практических проблем остается не до конца решенным. В контексте построения вопросно-ответных систем к таким проблемам можно отнести использование общих знаний и учет причинно-следственных связей. Целью статьи является рассмотрение методов интеграции знаний, которые способны усовершенствовать функционирование больших языковых моделей путем предоставления необходимых сведений и закономерностей из внешних источников.

Методы. В работе осуществляются классификация, анализ и сопоставление методов интеграции знаний, используемых в актуальных реализациях вопросно-ответных систем. В частности, рассматривается вовлечение вспомогательных сведений через самообучение, дообучение, механизм внимания и использование токенов взаимодействия, а также описываются соответствующие вспомогательные подходы для акцентирования наиболее релевантных сведений.

Результаты. Рассмотренные в обзоре вопросно-ответные системы непосредственно демонстрируют возрастание точности относительно базового решения на основе предобученной языковой модели за счет использования методов интеграции знаний на примере бенчмарка CommonsenseQA. При этом в целом более высокие результаты показывают методы интеграции знаний, основанные на использовании языковых моделей и механизма внимания.

Выводы. Представленный систематический обзор существующих методов интеграции знаний из внешних источников в работу вопросно-ответных систем фактически подтверждает эффективность и перспективность этого направления исследований. Данные методы демонстрируют не только возможность увеличить точность вопросно-ответных систем, но и в некоторой степени сгладить проблемы, связанные с интерпретируемостью результатов и устареванием знаний в предобученных моделях. Последующие изыскания способны как улучшить и оптимизировать отдельные аспекты существующих подходов, так и выработать концептуально новые.

Ключевые слова: глубокое обучение, обработка естественного языка, вопросно-ответная система, база знаний, графовые нейронные сети, интеграция знаний

Для цитирования: Радюш Д.В. Методы интеграции знаний для разработки вопросно-ответных систем. *Russian Technological Journal*. 2025;13(3):21–43. https://doi.org/10.32362/2500-316X-2025-13-3-21-43, https://www.elibrary.ru/QKUGFZ

Прозрачность финансовой деятельности: Автор не имеет финансовой заинтересованности в представленных материалах или методах.

Автор заявляет об отсутствии конфликта интересов.

INTRODUCTION

The development of question answering (QA) systems in recent years has been significantly influenced by the emergence and subsequent improvement of pre-trained language models [1]. The effectiveness of such models is based on the processing of large text corpora containing heterogeneous information, which makes it possible to capture both certain linguistic regularities and specific facts in the model weights [2]. Nevertheless, due to the peculiarities of natural languages, a significant amount of relevant information about the surrounding world may not always presented in the text in an explicit form, making it difficult to identify such information by language models at the training stage. In the first place, such information concerns various kinds of social interactions, including their psychological aspects, but also basic physical principles, which are learned by human beings at an early age. Examples of the latter include understanding the need to take care when crossing the road or cool down excessively hot food before eating it.

In order to compensate for this disadvantage, different kinds of knowledge sources can be used, in which such data will be recorded in an unambiguous form. For example, Cyc[1], which set out to collect general ideas about the world around us, can be considered as one of the first examples of a combined ontology and knowledge base. The main idea consists in the information record in the form of logical rules, which corresponded to the main direction of development of applications in the field of artificial intelligence of that time. A number of similar sources have been developed to date, albeit having somewhat different approaches to knowledge description, such as ATOMIC [3] and ConceptNet[2].

From a formal point of view, several arguments in favor of using external knowledge sources in the development of QA systems can be distinguished. Here, the primary motivation is directly to obtain more accurate and satisfying results for the user. By providing additional context to the query, a model can be expected to be able to answer a number of questions for which the internal representations of pretrained language models may not be sufficient. On the one hand, such questions include those where some causal relations are omitted, while, on the other, there is uncertainty in terms of identifying the semantics of some words due to their polysemy and insufficient context. This is largely determined by the limitations identified in the analysis of the application of transformers, which tend to rely only on the superficial and statistically most likely meanings of individual words [4], while for logical inference they rely heavily on heuristics learned from the training sample [5].

Even language models with a large number of weights, which demonstrate high results on many benchmarks, can not only make mistakes, but sometimes produce answers that have no relation to reality, which are popularly known as hallucinations [6]. In this regard, there is even a separate research area dedicated to methods of extracting query-relevant information and including it into the input data to improve the quality of answers [7] and using retrieval-augmented generation to reduce the number of hallucinations [8].

External knowledge sources can be used to reduce the requirements for the necessary computational resources to utilize pretrained language models. In particular, the purposeful involvement of additional information may enable the use of models having fewer weights while maintaining system accuracy at a comparable level [9]. This approach can be used to simplify QA system exploitation, as well as offsetting the cost of extracting and processing auxiliary data.

No less important is the use of knowledge bases from the position of explainable artificial intelligence (XAI). Due to the structured nature of knowledge bases, information extracted from them can provide a sequence of logical reasoning to the user to serve as a justification of the system's result. This property can be extremely important from the point of view of practical application since it is often the lack of interpretability of the results obtained with the help of neural networks that restricts their application in areas where there is a high risk of error and corresponding potential damage to society. In general, in order to effectively evaluate the adequacy of a system, it is desirable to have the most complete understanding of its operation.

Finally, the problem of updating the facts captured in the weights of language models is significant. Training of such models is typically performed on specific data sets and takes quite a long period of time. At the same time, a huge number of events occur every day, which leads to the change of a part of knowledge and the appearance of new facts. One way to solve this problem is to extract such information from external databases.

In this regard, it is quite relevant to consider how to use auxiliary general information to solve specific problems, such as the development of QA systems. In particular, the successful operation of the system requires that the information obtained is sufficient but not redundant, as otherwise it may impede its functioning and degrade the results. Also of equal importance is the way in which the additional knowledge is processed, as this will largely determine the effect of its use in the system. Thus, since the procedure of knowledge injection can be influenced by a substantial number of aspects, this paper presents an attempt to systematically analyze and compare existing

[1] Cycorp. https://cyc.com. Accessed December 01, 2024.

[2] ConceptNet. An open, multilingual knowledge graph. https://conceptnet.io. Accessed December 01, 2024.

approaches in order to draw a complete picture of the relevant ideas.

KNOWLEDGE BASES

In general, several directions can be distinguished in the field of QA system development depending on the methods used to provide additional data. For example, open domain QA implies the absence of a specialized knowledge base and is focused on the use of information from general-purpose sources. For this purpose, Wikipedia[3] is typically used and thus, due to its considerable volume and structural heterogeneity of content, the focus shifts significantly towards methods of searching for relevant information.

When using more narrowly focused queries to search for answers to more specialized questions, closed domain QA approaches include those complicated by the need to perform logical inference and take specific information into account. In this regard, specialized bases of structured knowledge, for example, knowledge graphs, can act as external knowledge sources that simplifies to a certain extent information retrieval, as well as implementation of logical inference and auxiliary operations. A relevant example is the knowledge graph DBpedia[4].

In the context of developing approaches to knowledge injection in the field of QA systems, structured knowledge bases are of primary interest. To some extent, this can be justified by the current state of affairs in this area. In particular, the emergence and subsequent development of pretrained language models has significantly reduced the need for contextual requirements to be provided to the query. As a result, some models after fine-tuning are now able to show results comparable to those of humans. Consequently, the main interest shifts towards analyzing the cases in which humans outperform existing QA systems. As a rule, such queries are those requiring out-of-context general ideas about the structure of the surrounding world, as well as analyzing cause-and-effect relations between individual facts.

In such circumstances, structured common knowledge bases can be particularly useful. In the first place, they directly provide the system with missing facts, which can be extracted taking into account existing relationships among themselves and together with other related information. Moreover, the structured nature of the knowledge greatly simplifies its machine processing and hence its use in practice. Thus, it seems possible to solve, to some extent, simultaneously the problems related to the class of queries that can be considered challenging for existing QA systems.

While Wikipedia can still be of use as an external source of additional data, due to the lack of systematization and large redundancy of information, the Wikidata[5] structured knowledge base which is based on data from Wikipedia has become of increasing relevance. The Wikidata graph consists of more than 100 mln entries describing elements of human knowledge in some way. Since each element of the graph corresponds to a certain set of properties that characterize it and establish its relationships with other elements, the content of Wikidata can be represented as for other knowledge graphs as a set of so-called subject–predicate–object triplets for which the object is a set of specific property values or a reference to another entity.

The ConceptNet knowledge base is also frequently used as a knowledge source in the context of building QA systems. This knowledge base, in addition to unique general information, partially includes information from other relatively frequently used sources such as the previously mentioned Cyc and DBpedia. Within ConceptNet, words and phrases are grouped based on several dozen relations. Comparable to the Wikidata resource discussed earlier, ConceptNet contains over 30 mln entries, although one must keep in mind that a significant portion of this value is due to the presence of effectively duplicate entries due to the existence of counterparts in another language, single-rooted words, and symmetric relationships. In addition, a slightly greater emphasis in ConceptNet is placed on linguistic properties, for example by capturing synonyms, antonyms and etymologically related words for a word. Finally, a feature of ConceptNet is the existence of weights for each relationship between elements, which heuristically reflects the degree of probability or importance of a given relationship.

Among the relatively recent general knowledge bases, ATOMIC, which contains more than 1 mln elements, can also be emphasized. The peculiarity of ATOMIC is the reflection of information in the form of abstract events and their results, which can be used emphasize complex cause-and-effect relations existing in the surrounding world. In particular, for example, based on some event in ATOMIC, it is possible to identify any of its consequences, as well as the intention, desire, or other characteristic of one of the participants, which can provide models with potentially missing knowledge about social interactions.

Table 1 presents examples of information that can be extracted from the knowledge bases discussed above. In general, they are somewhat similar, except for the more specific purpose of the ATOMIC database.

[3] https://www.wikipedia.org/. Accessed December 01, 2024.

[4] The DBpedia Knowledge Base. https://www.dbpedia.org. Accessed December 01, 2024.

[5] Wikidata. The free knowledge base. https://www.wikidata.org. Accessed December 01, 2024.

Table 1. Examples of information extracted from the knowledge bases

Knowledge base	Data example
DBpedia	DBpedia subject SemanticWeb
Wikidata	Wikidata uses semantic technology
ConceptNet	ConceptNet motivated by goal let computers understand what people already know
ATOMIC	Person X pays Person Y a compliment. Person X wanted to be nice

KNOWLEDGE INJECTION METHODS

A classification of knowledge injection methods based on the analysis of current research on the topic is presented in Fig. 1. Accordingly, the main ideas of knowledge injection methods can be considered in the context of developing QA systems with the corresponding examples and taking into account the peculiarities of specific selected classes of methods. The main place in this classification is occupied by the division of knowledge injection methods according to the use of knowledge bases, which is understood as processing the information directly when inferencing answers to queries and thus excludes cases of involving knowledge bases in the process of pretraining models. In turn, both language and graph models can be used to extract features from knowledge base data.

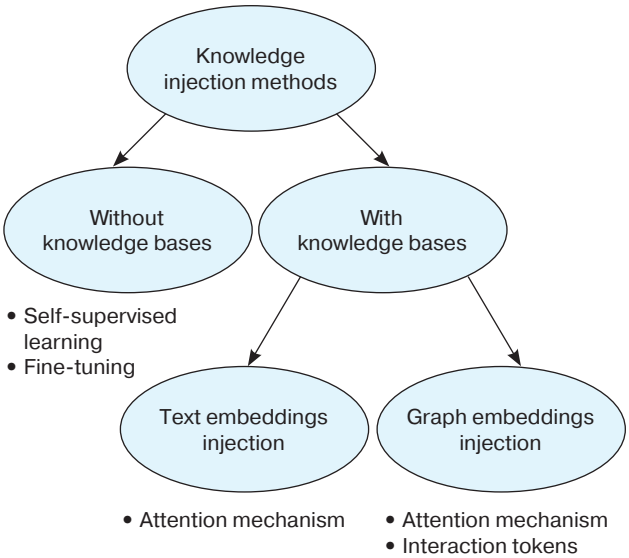

Fig. 1. Classification of knowledge injection methods

METHODOLOGICAL BASIS

Despite the differences in the approaches used for knowledge injection, it is possible to identify a common methodological basis in the QA systems discussed below. In particular, this applies both to the problem formulation itself and the supporting methods used.

The use of additional context in the system creates a certain specificity in terms of operation. In this regard, such auxiliary stages as retrieval of relevant data to the query also begin to acquire significant importance. In general, this stage implies determination of some number of entities n: $(q_1, \ldots, q_n)$ in the received query. For this purpose, classical methods from the field of natural language processing, such as lemmatization and part-of-speech tagging, continue to be mostly used in practice. The subsequent part of the process may vary depending on the specific task.

Many works on knowledge injection in QA systems assume that a question has answer choices. Accordingly, the goal of the system is to estimate the probability of each answer and select the most probable one. This allows us to significantly simplify and unify the construction and evaluation of systems. Therefore, in such cases, we will assume that the identified n entities correspond to m similarly extracted entities from the answer choices: $(a_1, \ldots, a_m)$.

The next step involves some knowledge base, which can be formalized as $G = (V, E)$, where V is the set of entities in the knowledge base, and $E \subseteq V \times R \times V$ is the set of triplets of the knowledge base of the entity–relation–entity kind. In practice, the established form of representation of such knowledge bases is a graph. Based on this, it is possible to construct a set of paths between entities defined in the context of the used knowledge base of the following form:

$$p = ((q_i, r_l, v_l), (v_l, r_{l+1}, v_{l+1}), \ldots, (v_{k-1}, r_k, a_j)),$$

where $i \in (1, \ldots, n)$, $j \in (1, \ldots, m)$; k is the path length in the graph; $l \in (1, \ldots, k)$; q_i is the ith entity from the query; a_j is the jth entity from the answer; v_l and r_l are the lth entity and relation in the graph, respectively.

Subsequently, a knowledge base subgraph or set of paths is used as additional context to determine the most likely answer.

One of the main problems in this setting is to determine the most relevant information in relation to the query. A possible tool for solving this problem is the Attention mechanism [10], which allows us to calculate the so-called Attention Weights quantitatively evaluating the degree of importance of this or that information from the context. Formally, the attention mechanism is defined by the expression:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\times\mathbf{K}^{\mathrm{T}}}{\sqrt{d_{\text{model}}}}\right)\cdot\mathbf{V} = \\ = \textbf{Attention weights}\cdot\mathbf{V}, \quad (1)$$

where $\mathbf{Q}$ = Query = $\mathbf{X}\times\mathbf{W}_{\text{q}}$; $\mathbf{K}$ = Key = $\mathbf{X}\times\mathbf{W}_{\text{k}}$; $\mathbf{V}$ = Value = $\mathbf{X}\times\mathbf{W}_{\text{v}}$; $\mathbf{X}\in\mathbb{R}^{N\times d_{\text{model}}}$ is the embedding[6] of the input data; $\mathbf{W}_{\text{q}}\in\mathbb{R}^{d_{\text{model}}\times d_k}$, $\mathbf{W}_{\text{k}}\in\mathbb{R}^{d_{\text{model}}\times d_k}$, $\mathbf{W}_{\text{v}}\in\mathbb{R}^{d_{\text{model}}\times d_v}$, N is the number of vectors in the input data; d_{model}, d_k, d_v are the dimensions of the embedding in the model and matrices $\mathbf{K}$ and $\mathbf{V}$, and $\text{softmax}(X_i) = \dfrac{\exp(X_i)}{\sum_{j=1}^{N}\exp(X_j)}$.

Thus, the attention weights, when multiplied by the embedding of the context, can adjust the influence of its individual elements on the result. In practice, it is common to use several groups of different matrices (so-called *heads*) to take into account different aspects of the data within the mechanism; the ensuing result of their application is concatenated and projected into the desired dimension using one more matrix, which is called Multi-Head Attention:

$$\text{Multi-Head Attention} = \\ = \text{Concatenation}(\mathit{Attention}_1, ..., \mathit{Attention}_z)\times W_{\text{o}}, \quad (2)$$

where $\mathit{Attention}_i$ is the ith result of the Attention block, $W_{\text{o}}\in\mathbb{R}^{zd_v\times d_{\text{model}}}$, while z is the number of the attention heads.

The attention mechanism, which plays a major role in many deep learning models, is widely used in developing approaches for knowledge injection. A feedforward neural network is also often used in conjunction with multi-head attention, which together form the main part of the model called a transformer. A multilayer perceptron is a specific implementation of feedforward neural network; from a practical point of view, the role of this component of the transformer is considered in the context of storing and retrieving patterns learned in the process of the training.

[6] Embedding means a vector representation.

METHODS OF KNOWLEDGE INJECTION WITHOUT KNOWLEDGE BASES

The involvement of auxiliary knowledge does not necessarily imply the use of specific knowledge bases. For example, similar examples with an indication of the correct answer can be used as information to help obtain a more accurate answer to a query. For example, [11] and [12] demonstrate the positive effect of adding queries from the training sample to the input data based on the similarity of their embeddings to the embedding of the original query.

Another type of approach is based on the idea of directly accessing the information learned in the process of the model pretraining; retrieved depending on the query, it is used as additional input data. For this purpose, [13] proposes to ask the pretrained model clarifying questions using templates, and use the answers as useful context. A similar approach in [14] also involves the exploitation of auxiliary data generated for a query in the QA system. Specially trained for knowledge generation models as described in [15] generate structured information in the format of knowledge base paths. Thus, the approaches discussed above are based on the idea of providing additional information as input, for which pretrained and other auxiliary models can be used, while fine-tuning of the language models may not be required.

Quantitatively, the most extensive group of approaches comes from the concept of pretraining models. Many experimental results support the idea that models with a large number of weights, trained on as much diverse information as possible, are able to show better results when they are subsequently adapted to specific conditions [16]. This concept relies heavily on the self-supervised learning methodology, which enables the extraction of representations from text corpora without the need of labels. To accomplish this, special tasks are developed according to the model training requirements. In particular, two such tasks were used in the development of the bidirectional encoder representations from transformers (BERT) language model [1]. The first task is the prediction of words in a sentence masked by a special token. For this purpose, some tokens from a sentence are selected with a probability of 15%, then 80% of these tokens are masked, 10% are replaced by a random token, while the remaining 10% are left unchanged. Cross-entropy can be used as a measure of error for the model's masked token prediction:

$$\text{Cross-entropy} = -\frac{1}{N}\sum_{i=1}^{N}\mathbf{y}_i\times\log(\bar{\mathbf{y}}_i), \quad (3)$$

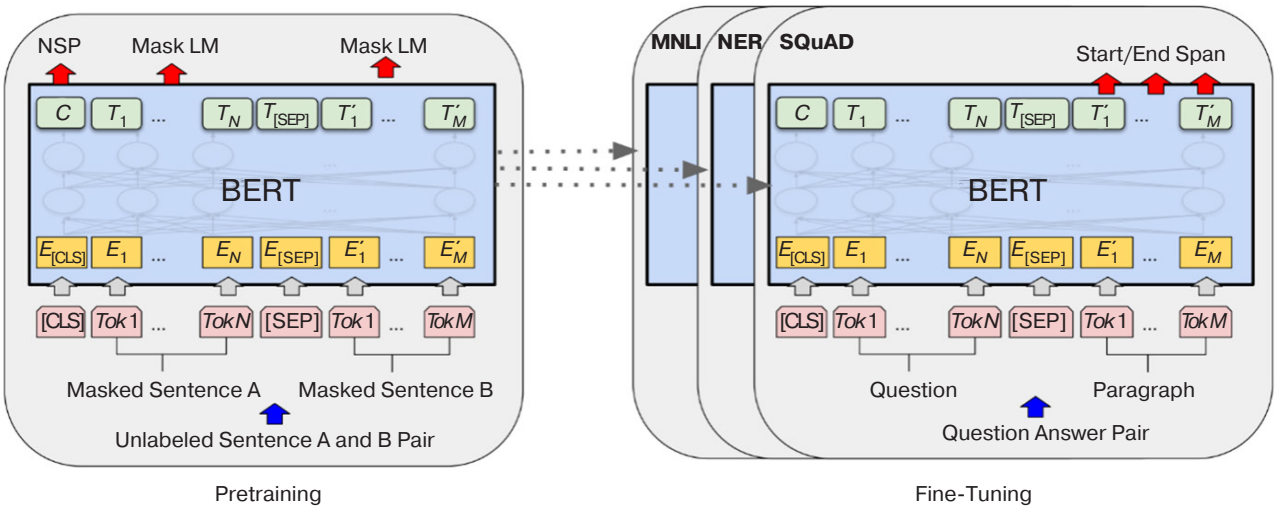

Fig. 2. BERT model training scheme [1]

where N is the total number of examples; $\mathbf{y}_i$ is one-hot-vector[7] encoding the correct answer for the ith example; $\bar{\mathbf{y}}_i$ is the model prediction vector for the ith example denoting the probability of matching each possible answer choice within the problem.

The second task concerns determining the correct order of two sentences in a text. This is realized through adding a special token [CLS] to the input data during training, representing information from the entire sentence, and is reduced to a binary classification task. The goal of this classification is to determine whether some sentence B is a continuation for the sentence A based on their resulting embeddings in the output of the model for the [CLS] tokens. In this learning framework, in 50% of cases, B is a random sentence, while in the other 50% of cases, B is a correct continuation. Cross-entropy can also function as the measure of loss for the task. The total model loss during training is considered as the sum of losses for each task. The overall training scheme of the BERT model is shown in Fig. 2:

In the pretraining phase, some tokens of unlabeled sentence A and B pair are masked, after which the embeddings ($E_{[CLS]}, E_1, \ldots, E_N, E_{[SEP]}$, and $E_1', \ldots, E_M'$) of the tockens (*Tok* 1, …, *Tok N* and *Tok* 1, …, *Tok M*) of the masked sentence A and masked sentence B with addition of generalization and separation tokens ([CLS] and [SEP]) are fed to the input of the BERT transformer. The resulting final embeddings ($C, T_1, \ldots, T_N, T_{SEP}, T_1', \ldots, T_M'$) are used to predict masked tokens and sentence order. In the Fine-Tuning phase, the format of the input data and the predicted data changes depending on the task (MNLI[8], NER[9], SQuAD[10]). In the case of QA SQuAD dataset, the input is a Question and the corresponding Paragraph, and the output is a predicted position in the context of the correct answer (Start/End Span).

Subsequently, this methodology has been modified and adapted in the context of pretraining of other language models. In the context of QA systems, many approaches have been developed based on modifying and extending BERT self-supervised learning tasks or replacing them with others. In general, when building these kinds of models, it is most common to modify the masking procedure by imposing constraints on what should be masked in a sentence and changing the masking parameters during training.

One of the first and most significant developments in this direction was the Enhanced Language Representation with Informative Entities (ERNIE) model [17], the scheme of which is shown in Fig. 3. Its main idea is that if one also pretrains to predict the masked named entities identified in the text based on the knowledge base as an additional task for self-supervised learning, it can improve the model's language understanding as well as contextualize its certain knowledge about the world. Specifically, for this purpose, for text tokens, the corresponding named entity is replaced by a random entity in 5% of the cases, and in 15% of the cases, the entity is masked and should be predicted from the text tokens. In addition, the paper introduces an interaction

[7] One-hot vector is a binary vector in which only one element has the value 1, and remaining elements are equal to 0.
[8] Multi-Genre Natural Language Inference—dataset for the Natural Language Inference task—establishes a logical relationship between the text fragments.
[9] Named Entity Recognition is the task of recognizing named entities in the text.
[10] Stanford Question Answering Dataset is a QA dataset, which implies automatic answers to questions in natural language.

mechanism between the embeddings of entities and the corresponding text tokens, which can bring additional information to both types of embeddings, thereby increasing the accuracy of predicting the correct tokens. To this end, an intermediate embedding is introduced that combines information at the level of tokens and named entities, due to which the initial embeddings of tokens and entities are subsequently updated, which is defined by the expressions:

$$\begin{aligned} \mathbf{h}_j &= \sigma(\widetilde{\mathbf{W}}_{\mathrm{t}}\widetilde{\mathbf{w}}_j + \widetilde{\mathbf{W}}_{\mathrm{e}}\tilde{\mathbf{e}}_k + \tilde{\mathbf{b}}), \\ \mathbf{w}_j &= \sigma(\mathbf{W}_{\mathrm{t}}\mathbf{h}_j + \mathbf{b}_{\mathrm{t}}), \\ \mathbf{e}_k &= \sigma(\mathbf{W}_{\mathrm{e}}\mathbf{h}_j + \mathbf{b}_{\mathrm{e}}), \end{aligned} \tag{4}$$

where $\mathbf{h}_j$ is the aggregate embedding of the token number j, σ is a given nonlinear activation function, $\widetilde{\mathbf{w}}_j$ and $\mathbf{w}_j$ are embeddings of the token j before and after knowledge injection, $\tilde{\mathbf{e}}_k$ and $\mathbf{e}_k$ are the embeddings of the named entity k matching the token j before and after knowledge injection, $\widetilde{\mathbf{W}}$, $\mathbf{W}$, $\tilde{\mathbf{b}}$, and $\mathbf{b}$ are model parameters.

In the process of training the ERNIE model, the input text token embeddings (Token Input) pass through N layers of the transformer (T-Encoder), after which, together with the named entity embeddings (Entity Input) they are processed by M layers of the aggregator (K-Encoder). At each layer i of the aggregator, the entity (e_1 and e_2) and text (w_1, …, w_n) embeddings pass through corresponding или related Multi-Head Attention block, and the corresponding updated entity ($\tilde{e}_1$ and $\tilde{e}_2$) and text ($\tilde{w}_1$, …, $\tilde{w}_n$) embeddings are fed into the knowledge injection block (Information Fusion), the output of which, according to the formulas (4), produces the embeddings of entities (Entity Output) and text (Token Output) taking into account the knowledge injection.

A similar method is at the heart of the KnowBert model [18], but the injection of external information occurs at the level of embeddings of entities, which are updated through the attention mechanism and by adding pretraining entity embeddings from the knowledge base, which subsequently also affects the embeddings of all tokens through the attention mechanism, according to the formula:

$$\mathbf{H}'_i = \mathrm{MLP}(\mathrm{MHA}(\mathbf{H}_i, \mathbf{S}'^e, \mathbf{S}'^e), \tag{5}$$

where $\mathbf{H}'_i$ is the embedding of the token i after knowledge injection, MLP is the multilayer perceptron, MHA is Multi-Head Attention, $\mathbf{H}_i$ is the embedding of the token i before knowledge injection, $\mathbf{S}'^e$ are updated embeddings of the identified named entities.

A conceptually similar architecture to ERNIE is proposed in [19], the main difference being the use of information about relations between entities, the prediction of which is represented by a separate task for pretraining. In the Weakly Supervised Knowledge-Pretrained Language Model [20], instead of masking entities in the pretraining phase, the model is trained to predict whether entities in the input data have been replaced by others of the same type within the Wikidata knowledge base. The architecture of the model in this case is consistent with BERT, but token masking is only performed at 5% rather than 15% to avoid masking too large a fragment of context, as entities can consist of multiple words. In [21], word combination masking is applied in addition to masking words and named entities, which improves the model's understanding of word combinability, and the injection is done in stages: at each stage, a BERT-like model is trained on only one type of masking. The use of multiple training modes, in which the model switches from word prediction to phrase prediction depending on which mode between the last two consecutive iterations had the largest reduction in model loss relative to the total reduction in loss over all iterations, is a major innovation [22].

The authors of the study [23] pretrain a BERT-based model, aiming to learn masked entity prediction from their descriptions, as well as to converge embeddings of synonymous entity descriptions and distance antonymous ones, for which a special loss function is used:

$$L = -\sum \log \frac{f(\mathbf{h}_{\mathrm{ori}}, \mathbf{h}_{\mathrm{syn}})}{f(\mathbf{h}_{\mathrm{ori}}, \mathbf{h}_{\mathrm{syn}}) + f(\mathbf{h}_{\mathrm{ori}}, \mathbf{h}_{\mathrm{ant}})}, \tag{6}$$

where $f(\mathbf{h}_i, \mathbf{h}_j) = \exp(\mathbf{h}_i \mathbf{h}_j)$, $\mathbf{h}_{\mathrm{ori}}$ is the embedding of the masked entity description, $\mathbf{h}_{\mathrm{syn}}$ is the embedding of the synonymous entity description, $\mathbf{h}_{\mathrm{ant}}$ is the embedding of the antonymous entity description.

In order to deal with specific tasks this model is used in pair with BERT as an additional source of knowledge in the form of embeddings of identified entities, and their injection is performed through concatenation of model outputs with BERT outputs with optional application of attention mechanism to take into account the importance of data on specific entities. The use of the attention mechanism is considered for both output embeddings from the last model layers and output embeddings across model layers with averaging applied, and the result of the attention mechanism is concatenated with the output of the BERT model instead of the output of the auxiliary model. Linguistic features also play an important role in [24], in which an additional task for self-supervised learning is to determine the semantic similarity of a pair of words, while the model is trained by alternating between the BERT self-supervised learning task and the auxiliary task. A similar idea is presented in [25], where the model is trained to classify words into groups with similar meaning based on WordNet[11] data.

[11] A lexical database of the English language developed at Princeton University. https://wordnet.princeton.edu/. Accessed December 01, 2024.

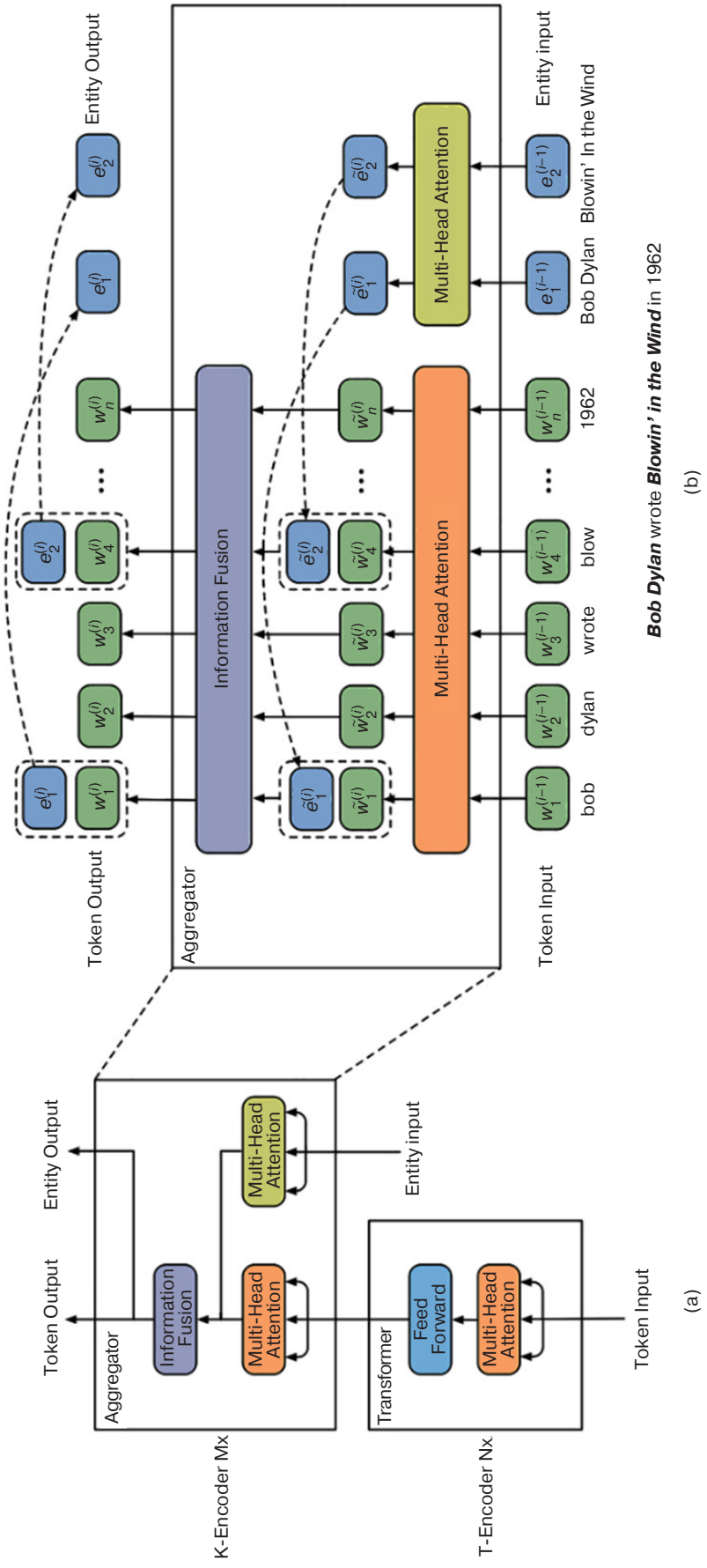

Fig. 3. ERNIE model training scheme [17]:
(a) model architecture; (b) aggregator architecture

A logical development of entity and relation masking approaches is the use of predicting structured knowledge units in the form of triplets as the pretraining stage task, which may enable learning more general principles and relationships similar to those contained in knowledge bases. Thus, in Knowledge Embedding and Pretrained Language Representation (KEPLER) [26], embeddings of triplet elements are treated as embeddings of their descriptions from a knowledge base, obtained using the same model that is used to generate embeddings of text tokens in the masked token prediction task. In this case, the scoring function of TrancE knowledge graph embedding model is applied to compute the loss in the triplet prediction task [27]:

$$d(\mathbf{h}, \mathbf{t}) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|, \quad (7)$$

where **h** is the embedding of the subject in the triplet, **r** is the embedding of the relation in the triplet, **t** is the embedding of the object in the triplet.

In [28], a set of triplets from a single subgraph is given as input to the model, and therefore the attention mechanism additionally utilizes the adjacency matrix to account for existing relationships, and training is performed in a triplet reconstruction format, which involves composing triplets from updated vertex embeddings and using the scoring function (7). The study [29] contains the idea of pretraining three functions based on an encoder model to predict each triplet element from the other two, which should facilitate the learning of possible combinations. In this setting, the answer score is the product of the similarity scores of the values of the three pretrained functions and the corresponding real triplet elements, where the subject represents the context of the question, the relation is the question itself, and the object is the specific answer choice. A pretraining function that will participate in finding answers to queries by identifying the most likely relationships with auxiliary data from the knowledge base is proposed in [30]. With the help of this function, the extracted auxiliary facts for each answer choice are compared by the degree of similarity with the facts for the question, and the more probable answer is considered to be the one for which this similarity of facts is higher on average.

At the same time, triplets from a relevant subgraph to a query can be directly fed to the input of the model within the framework of pretraining on a par with text tokens using special embeddings to indicate the token type, as shown in [31]. In this regard, when implementing the attention mechanism in the model, a mask matrix is used to restrict the interaction of unrelated vertices in the subgraph. In [32], it was proposed to improve the approach of the ERNIE model by modifying the representation of entities by taking into account their relationships in the corresponding subgraph, and using the attention mechanism to filter out potentially irrelevant context for a query.

Another way of using knowledge bases in the pretraining phase of the model can be to build new QA datasets on their basis, with which the system also improves its ability to find correct answers in a certain way. This approach is used in [33], and in [34] it is developed by conceptualization: specific facts are considered in a more general way, so that more situations can be covered and the ability to distinguish between similar variants can be improved. For example, using the ATOMIC framework, playing soccer can be represented as a tedious event.

The concept of Self-supervised Bidirectional Encoder Representation Learning of Commonsense (elBERto) model [35] is more emphasized on quantitative expansion of the number of self-supervised learning tasks. In order to improve the system's ability to process difficult queries, three more tasks were added to the BERT self-supervised learning tasks: the first one is aimed at distinguishing contexts with opposite meanings; the second one requires putting in order several jumbled sentences taken from the same paragraph; the third one extends the learning of contextual relationships through entity masking. According to the authors, this approach will also allow the system to better capture linguistic patterns and provide more universal applications.

Another concept of knowledge injection implies as an additional step the fine-tuning on the basis of existing datasets corresponding to the practical task. For example, the use of the SQuAD dataset [36] from the field of QA systems has gained some popularity in this regard. Its key features include a relatively large size (more than 100000 queries), while each query is accompanied by a corresponding context taken from Wikipedia. Thus, as a result of training on this dataset, the model better adapts to the problem formulation and format, and in addition processes a rather significant amount of data, thus increasing the amount of learned factual information.

As a relevant and characteristic example in this regard, we can mention the UnifiedQA model [37], the development of which was based on training the language model on 8 QA datasets of different types. It allows the existing benchmark formats to be adapted and provides an increase in the accuracy of the model on unseen questions in the training process, opening also new opportunities for its further fine-tuning. The feasibility of such an approach was also confirmed for the Unicorn model from [38], but in this case, the scope of the study was limited exclusively to CommonsenseQA datasets.

The methods considered above can be referred to the class of approaches without explicit involvement

of knowledge bases, since the use of the corresponding systems does not imply the direct extraction of the context to the query exactly from knowledge bases, and the emphasis is created on the knowledge that was obtained in the process of training. In fact, the large language models developed in the last few years are essentially based on a similar idea: training on a large amount of qualitative data, taking into account different specificities and human preferences, can increase the versatility of the systems and the evaluation of the results obtained with their help, if this process is sufficiently scaled up.

The advantages of this class include, in a sense, greater versatility due to its independence from the use of knowledge bases. In addition, the significant reliance of the QA system architecture on pretrained and fine-tuned models allows us to simplify its development, subsequent use and adaptation to specific tasks.

At the same time, this class of approaches can be considered to a certain extent limited in its possibilities for further development. The point is that, in general, the increase in the efficiency of models here is associated with targeted and point improvement, expansion of the amount of learned information, while no fundamentally new mechanisms that improve the system's reasoning abilities are introduced. In addition, this direction does not practically solve the problem of the lack of interpretability of the received answers and their justification by the system, as well as the problem of knowledge obsolescence.

METHODS OF KNOWLEDGE INJECTION WITH THE KNOWLEDGE BASES

The basic unit of information in knowledge bases can be considered in terms of entity–relation–entity triplets, while more complex relationships can be conveyed by a set of triplets or paths. Knowledge base subgraphs used in addition to or instead of paths can be considered as a set of paths having common elements. As a result, in terms of information processing, a QA system may have 3 types of attributes in some combination:

1) features obtained by processing the query context by a language model;
2) features based on extracted paths;
3) features associated with subgraphs of the knowledge base.

Thus, one direction for research is how to effectively process these different types of features and how to combine the corresponding results.

One approach to injecting features into the system is based on the fact that triplets and their aggregates can often be translated quite easily into natural language sentences, and in this form, it is possible to feed them into the input of the language model as an auxiliary context. It should be noted, however, that in this form they can also serve as a justification for the resulting answer. An example of the implementation of such an approach is the Knowledge-Augmented language model PromptING (KAPING) [39]. In [40], its effectiveness is investigated in the context of using language models pretrained on auxiliary datasets. In the DEscriptive Knowledge for COmmonsense question answering (DEKCOR) model [41], in addition to the triplets extracted from ConceptNet, dictionary definitions of the corresponding entities are input, whereas in Knowledgeable External Attention for commonsense Reasoning (KEAR) [42] (Fig. 4) the context of the query is extended by including additional information from a number of QA datasets, which enables the use of more specific information.

In particular, under the KEAR architecture, to the concatenation of a question with one of the answer choices (Question & Candidate) relevant extracted auxiliary data (Knowledge Retrieval) from ConceptNet knowledge base, Wiktionary dictionary (Definition) and additional datasets (Training Data) are added to the model input. Embeddings of the query tokens ($E_{[CLS]}$, E_0, ..., E_N) to which a segment indicator (S_0) and the auxiliary context (E_0^k, ..., $E_{N_k}^k$) to which a segment indicator (S_1) is added are fed to the Transformer input. The answer probability (Score Prediction) is defined based on the final embedding of the auxiliary token $E_{[CLS]}$ obtained by the attention mechanism (Self-Attention / External Attention).

The advantage of knowledge injection through language models is the possibility to rely heavily on the performance of pretrained models, while in some cases even avoiding the need to change their weights (e.g., the KAPING model). Also, the computational cost of acquiring additional features can be considered relatively small. At the same time, since most pretrained models can only efficiently utilize a fixed amount of information from the input data, there is a need to limit the amount of less relevant information.

In the simplest case, a restriction on the number of triplets (e.g., no more than 3 consecutive triplets) or a set of heuristics that take into account the peculiarities of a particular knowledge base can be used. In [43], in order to include only potentially relevant metadata from Wikidata in the model's input, the increase in the probability of a correct answer is estimated taking into account the corresponding Information Gain:

$$P(y \mid k_m) = 2^{\mathrm{pmi}(y,k_m)} P(y), \qquad (8)$$

where y is the correct answer; k_m is a particular pattern containing m metadata; $\mathrm{pmi} = \log\left(\frac{P(y,k_m)}{P(y)P(k_m)}\right)$.

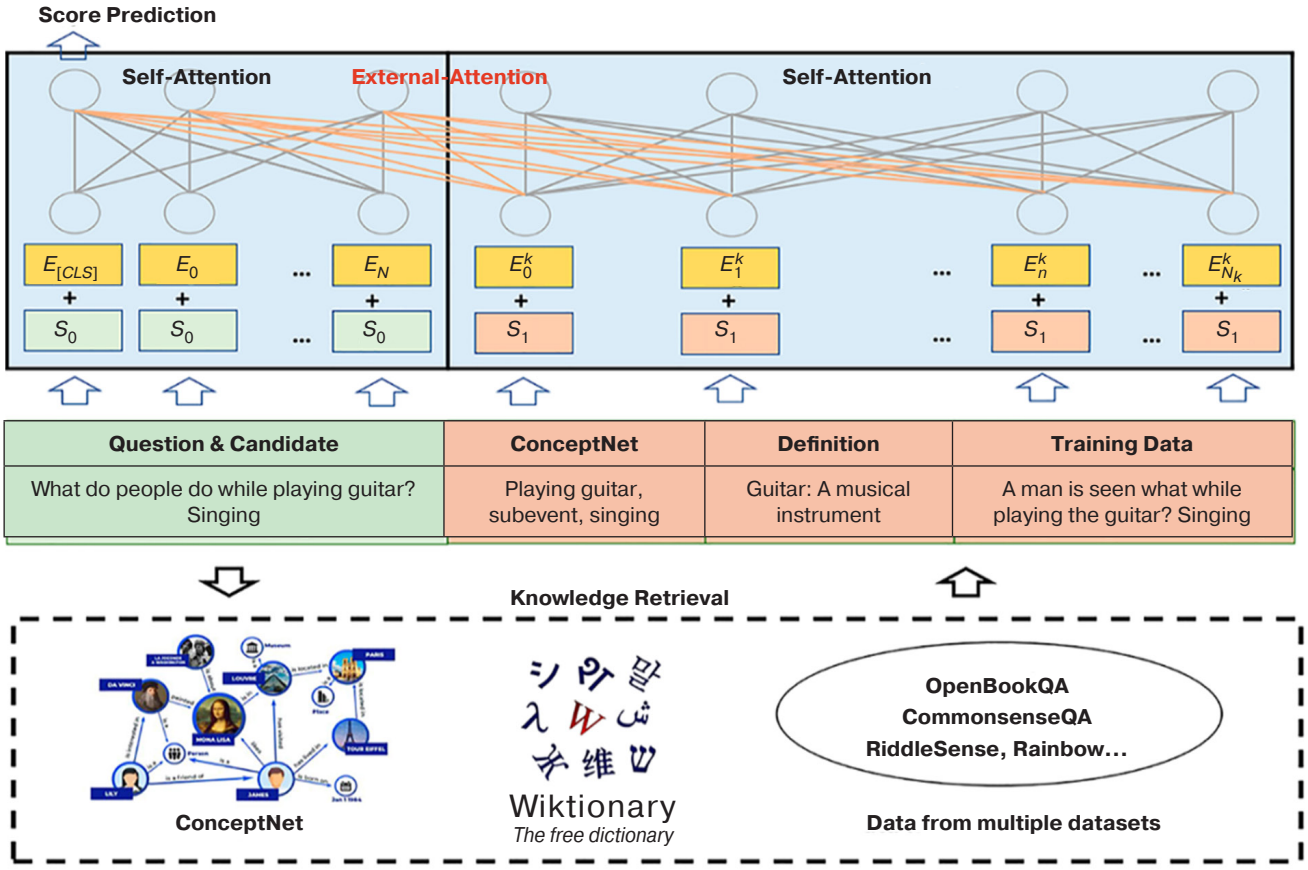

Fig. 4. KEAR model architecture [42]

The study [44] shows the feasibility of ranking the extracted additional information based on its importance estimation through an auxiliary pretrained model, while [45] shows the positive effect of using weighted summation of knowledge embeddings when injecting them, in addition to ranking, to emphasize more salient facts.

Also, in order to inject heterogeneous data, an attention mechanism can be applied to aggregate features based on their relevance to the task. In other words, the extracted knowledge that has more semantic relationship with the query, which is determined based on operations on embeddings, will be considered more relevant. Most often in practice, attention weights are derived based on operations on embeddings, which enable updating the relevant embeddings with respect to the specific task and its context. In particular, a similar approach is presented in works [46–48], and in the article [49] auxiliary knowledge is also filtered based on the frequency of occurrence of entities and relevant paths, while for knowledge injection a sigmoid function is additionally used to adjust how much they will affect the context update for the query. In addition to the attention mechanism for filtering out irrelevant data, the study [50] proposed to use graph-based approaches to determine the importance of individual nodes in the extracted subgraph: node closeness calculation, PageRank[12] and its modification, which enables only the most informative paths to be considered.

In addition, a disadvantage of knowledge injection through language models is the limited use of structured knowledge bases, which may reduce the potential efficiency of the final implementation. In order to preserve the effect of considering the relationships when translating triplets into text and to prevent information mixing in the K-BERT model [51], the positional encoding is included at the stage of generating embeddings, and in the subsequent computations, a specially introduced Visible Matrix is adopted, the elements of which determine what tokens a particular token should interact with in a given context.

In this regard, it is necessary to mention one of the main tools for processing structured knowledge—graph neural networks. This tool allows us to obtain and update embeddings of graph vertices using the concept of message passing:

$$\mathbf{h}_u = \phi(x_u, \bigoplus_{v \in N_u} \psi(x_u, x_v, e_{uv})), \quad (9)$$

where $\mathbf{h}_u$ is the embedding of the vertex u; x_u and x_v are the features of the vertices u and v; e_{uv} is the feature of

[12] A ranking algorithm that evaluates the number and quality of links leading to web pages.

the edge between the vertices u and v; ϕ and ψ are the set differentiable functions; $\bigoplus_{v \in N_u}$ is a permutation invariant aggregation operator acting on the neighbors of the vertex u.

Due to this, in practice, the embedding of each entity can use different contextual data obtained from the knowledge base, taking into account in a certain way the information from its neighbors in the graph. An example of such a model used in the context of knowledge injection is the Graph Convolutional Network [52].

The features obtained by graph neural networks can also be subsequently injected into the system operation using an attention mechanism. Among the implementations of this kind is the model architecture [53] depicted in Fig. 5. Here, the embedding of each vertex of the auxiliary subgraph is adjusted for relevance with respect to the existing embedding of the query before it is directly used to obtain an answer:

$$\alpha_i = \frac{\mathbf{h}^c \sigma(\mathbf{W}\mathbf{h}_i)}{\sum_{j \in N} \mathbf{h}^c \sigma(\mathbf{W}\mathbf{h}_j)}, \quad (10)$$

where α_i is the relevance degree of the vertex i; $\mathbf{h}^c$ is the embedding of the query context; $\mathbf{W}$ is the weight matrix; $\mathbf{h}_i$ is the embedding of the vertex i; N is the set of vertex indices neighboring the vertex i.

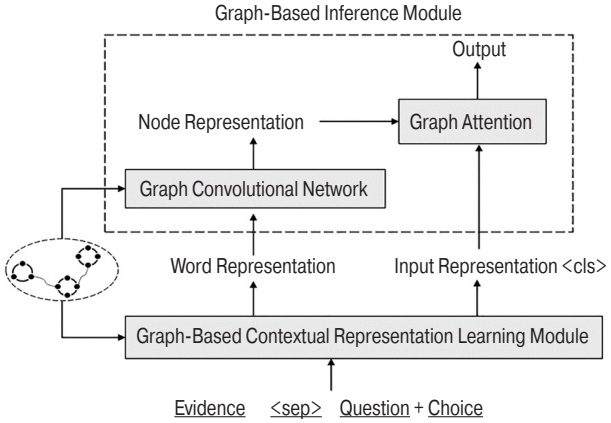

Fig. 5. Model architecture from [53]

In general, the model is organized as follows: first, a subgraph with auxiliary information is extracted from an existing query consisting of a question and one of the answer choices. This information in the form of evidence is appended to the query and fed to the input of the language model (Graph-Based Contextual Representation Learning Module). The Word Representation token embeddings obtained at the output of the model are fed to the Graph-Based Inference Module, where they are used to initialize the corresponding nodes of the auxiliary graph, which are subsequently updated using the Graph Convolutional Network. The resulting Node Representation embeddings of the graph nodes are then aggregated using the Graph Attention mechanism with importance relative to the Input Representation embedding of the textual context, and the resulting graph embedding along with the textual embedding are directly used to predict the answer probability using a multilayer perceptron.

Similarly, the Multi-Hop Graph Relation Networks (MHGRN) model [54] is organized in a similar way, but its key difference is the consideration of the auxiliary subgraph as a set of paths connecting vertices, according to which the embedding of each vertex is updated based on the given length of paths from it. To aggregate information along the paths, special attention weights are introduced, which are defined as the conditional probability of a given sequence of triplets given the available context for a query, whereas to calculate the probability of a particular answer, the resulting embeddings of entities from the answer are aggregated using the attention mechanism and, together with the embedding of the query context, are processed by the multilayer perceptron. Thus, this approach also takes into account the importance of relations between entities. In the Joint reasoning with Language models and Knowledge graphs (JointLK) model [55], the least relevant nodes of the auxiliary subgraph are cut off and a new representation of the query context is additionally introduced, which takes into account the degree of its importance with respect to the subgraph and is the third component for obtaining the answer score along with the original context representation and the embedding of the subgraph. In the study [56], the message passing mechanism implements consistent updating of both entity and relation embeddings, which in this case are also used to estimate the answer probability. In this case, a modified adjacency matrix, whose elements are the corresponding attention weights, is used to formalize the relevance of relations between vertices under a given query context. In the Knowledge-Aware Graph Network (KagNet) module [57], the embeddings of the vertices of the auxiliary graph updated with the help of message passing mechanism are considered as elements of paths connecting entities from the question and one of the answer options. As a result, for each such pair of entities, a vector of structured features is generated as an average of the embeddings of the paths connecting them, and a vector of textual features obtained as the result of applying a multilayer perceptron to the concatenation of the embeddings of the query and each entity from the pair. To estimate

the probability of a particular answer, averaging over all pairs of entities from the query and response is implemented. Furthermore, in addition, instead of averaging, the authors also propose to utilize the attention mechanism for feature aggregation.

One of the certain disadvantages of using graph neural networks is the increase in the number of parameters in the model and, consequently, in the resources for its training and use. In this regard, [58] proposes a simplified algorithm for obtaining triplet embeddings based on one-hot vectors indicating the type of entity in the graph and a certain relation within the ConceptNet database. To calculate the final answer probability, the model uses two scores: for textual and graph features. The former is based on the processing of the query embedding by the multilayer perceptron, while the latter is based on a weighted sum of path embeddings that takes into account their frequency of occurrence.

The process of knowledge injection may be somewhat more difficult when there are multiple sources of information and training on different types of tasks. In such conditions, it is necessary to solve the problems associated with the need to retrain the model weights and the displacement of learned facts by new ones, which can lead to unstable results. One possible solution is the use of adapters [59]—special modules oriented for a specific data source or task, which allows us not to change the weights of the main model and to train only a relatively small number of adapter weights, and thus avoid knowledge mixing. In practice, several different adapters are usually trained independently and then used together to solve a particular task. Thus, the model [60] employs two types of adapters: the first one is focused on learning general facts from knowledge bases, while the second one is focused on linguistic information. Within the architecture, each output from the transformer model layer is fed to the input of the corresponding adapter layer, resulting in the formation of certain auxiliary features on the last adapter layer, which can be used to predict the answer together with the outputs of the last transformer layer. In [61] (Fig. 6), a slightly different approach is implemented where the weights of the adapters pre-trained on data from ATOMIC, ConceptNet, WikiData, and WordNet knowledge bases are also not changed when training the model on a specific task, but instead knowledge injection is performed by the attention mechanism (formula), where the adapters form Value and Key, and the pretrained transformer forms Query:

At each model layer, input data passes through the transformer layer and enters the Zero-shot Fusion knowledge integration block both directly (circle 4) and after interaction with adapter models (circles 1, 2, and 3). In this block, embeddings interact within the attention mechanism (formula (1)): the output representation from the transformer are used as query, while the outputs from the adapters act as Value and Key. Subsequently, the result of the knowledge integration block is summed with the output from the Multi-Head Attention block of the transformer and normalized (Add & Norm). The goal

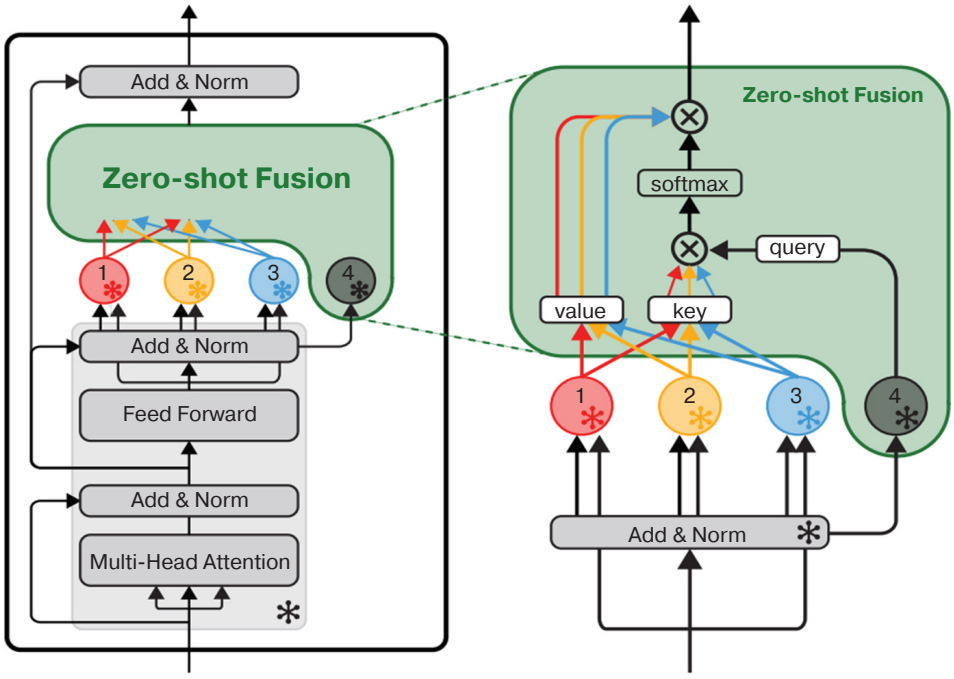

Fig. 6. Knowledge injection scheme using adapters from [61]

of model training under this architecture is to be able to address a more relevant adapter, which to some extent resembles the concept of mixture of experts [62].

Separately, in the context of knowledge injection, we can distinguish a group of approaches based on the use of so-called Interaction Tokens. The concept of interaction tokens is largely similar to the idea of using a special token [CLS] in language models, which can serve to classify a whole text fragment. Similarly, interaction tokens in the case of textual information or Interaction Nodes in the case of graphs can act as an intermediate container of necessary information for combining heterogeneous data. An example of the corresponding QA architecture can be seen in Fig. 7: within the Graph REASoning Enhanced Language Model (GreaseLM) [63] textual and structured information are processed independently, and their integration is realized by updating the vector representations of the interaction token and the interaction node by applying a bilayer perceptron to their concatenation:

$$[\mathbf{h}_{\text{int}};\mathbf{e}_{\text{int}}] = \text{MInt}([\tilde{\mathbf{h}}_{\text{int}};\tilde{\mathbf{e}}_{\text{int}}]) = \text{MLP}([\tilde{\mathbf{h}}_{\text{int}};\tilde{\mathbf{e}}_{\text{int}}]), \quad (11)$$

where $\tilde{\mathbf{h}}_{\text{int}}$ is the embedding of the interaction token before the knowledge integration, $\tilde{\mathbf{e}}_{\text{int}}$ is the embedding of the interaction vertex before the knowledge integration, MInt is the modality interaction layer.

In GreaseLM training, the concatenation of a question with one of the answer choices is processed using N layers of the model-encoder (Uni-modal Encoder) and together with the auxiliary graph (KG Retrieval) passes through M layers (GreaseLM Layer) of the knowledge integration block (Cross-modal Fuser). At each layer of this block, the embeddings of the text tokens (h_1, …, h_T, h_{int}) and graph vertices (e_{int}, e_1, …, e_J) are processed by the language model layer (LM Layer) and graph neural network layer (GNN Layer), respectively, and the integration process itself is carried out through the interaction (MInt, formula (11)) of the embeddings of the special tokens ($\tilde{h}_{\text{int}}$ and $\tilde{e}_{\text{int}}$). After the knowledge integration process is completed, the embeddings of the special tokens with the graph embedding (Pooling) obtained by means of the attention mechanism are used for Answer Selection by the perceptron (MLP).

Within the DRAGON[13] model [64], the GreaseLM architecture was considered in the context of self-supervised learning: after a knowledge integration layer, the obtained textual features are used to predict masked text tokens, while the graph features are used for the Link Prediction task, which involves establishing probability of a link between vertices in a graph using scoring functions similar to (7).

The Question Answer Graph Neural Network (QA-GNN) model [65] uses only an interaction node initialized by a embedding of the textual context from the query, based on similarity with which, determined using a pretrained model, the relevance of other nodes is estimated. These evaluations, together with features representing the types of vertices and relations in the form of one-hot encoding, are used to compute attention weights, which are used to implement the message passing between vertices and the corresponding update of their embeddings. The answer selection process is also essentially formulated similarly to the GreaseLM model. In PipeNet [66], compared to QA-GNN, the computation of the relevance of the vertices of the auxiliary graph to the query context is, in a sense, replaced by an algorithm for cutting off irrelevant vertices, based on determining the shortest distance between entities within the language dependency graph corresponding to the query:

$$D(c_{\text{q}}) = -\frac{\sum_{i=1}^{|V_{\text{a}}|} Dist(c_{\text{q}}, c_{\text{a}})}{|V_{\text{a}}|}, \quad (12)$$

where $D(c_{\text{q}})$ is the relevance of the entity c_{q} from the query, $Dist(c_{\text{q}}, c_{\text{a}})$ is the shortest distance between the entity c_{q} from the query and the entity c_{a} from the corresponding answer choice, V_{a} is the set of entities from the answer choice for the query.

The rest of the QA-GNN architecture is essentially the same, except for the use of vertex relevance scores in the calculation of attention weight.

To summarize the methods of knowledge injection with graph models, it can be stated that they are characterized by the greatest variety of ideas used, which demonstrate a wide range of possibilities for taking into account the features of structured knowledge and their inclusion in the work of QA systems. A positive aspect can also be considered the possibility to increase the interpretability of the model due to the formation of fact chains with the help of knowledge bases, which can be updated separately in a timely manner depending on the current events. At the same time, the full-fledged integration of graph features leads to a significant complication of model architectures and, depending on the implementation, may require certain additional computational resources, as a result of which the benefit of knowledge injection becomes more ambiguous.

[13] DRAGON— Deep Bidirectional Language-Knowledge Graph Pretraining.

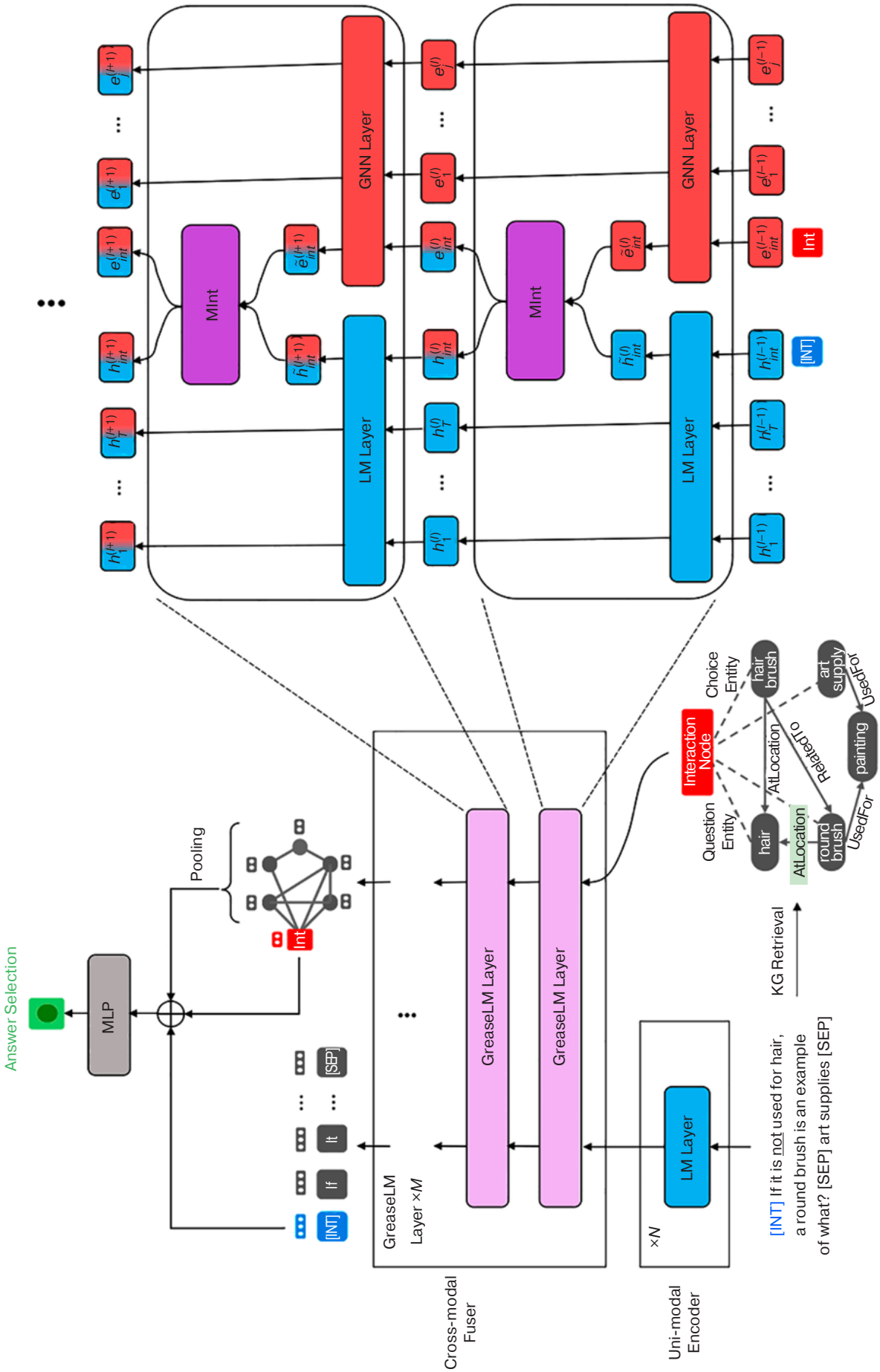

Fig. 7. GreaseLM model architecture [63]

COMPARATIVE ANALYSIS

The described approaches also differ from the point of view of setting up the experimental part. Thus, first of all, different benchmarks could be used to test the efficiency of implementations. As a result, it was decided to perform the comparative analysis with respect to the CommonsenseQA dataset that appeared most frequently in the considered works [67].

CommonsenseQA consists of 12102 questions offering five answer choices, one of which is correct. The choice in favor of this dataset can be justified by its higher complexity in terms of the relatively poor results of QA systems on it compared to its counterparts. In this case, the higher complexity is due to the focus of the questions on social and psychological aspects and the need to establish causal relationships, as well as the lack of any additional context for the questions. While this complicates the effective implementation of pretrained language models due to the smaller number of inputs, such a formulation of the problem favors the formation of such a context through external knowledge bases.

Table 2 summarizes the results of the models without Ensemble on the CommonsenseQA dataset test sample. For practical comparison of implementations in the context of this dataset, accuracy (the percentage of questions that were answered correctly) is used as a metric. It should also be noted that one of the most frequently used language encoder models, RoBERTa [68], was chosen as the baseline benchmark.

The results show that any of the considered approaches can increase the accuracy of the QA system with respect to the base solution using a pretrained language model, thus confirming the promising avenue of this line of

Table 2. Comparison of the effectiveness of knowledge injection methods

Model	Injection method	Accuracy on CommonsenseQA test set, %
RoBERTa [68] (2019)	–	68.7
Model from [15] (2020)	Self-supervised learning	75.6
Model from [23] (2022)	Self-supervised learning	78.5
UnifiedQA [37] (2020)	Fine-tuning	79.1
Model from [44] (2023)	Text embeddings and attention mechanism	75.0
Model from [47] (2020)	Text embeddings and attention mechanism	80.3
DEKCOR [41] (2021)	Text embeddings and attention mechanism	80.7
KEAR [42] (2022)	Text embeddings and attention mechanism	86.1
JointLK [55] (2022)	Graph embeddings and attention mechanism	74.4
Modelfrom [53] (2020)	Graph embeddings and attention mechanism	75.3
MHGRN [54] (2020)	Graph embeddings and attention mechanism	75.4
QA-GNN [65] (2021)	Interaction tokens	73.4
GreaseLM [63] (2022)	Interaction tokens	74.2
DRAGON [64] (2022)	Interaction tokens	76.0

research. At the same time, the models using graph-based embeddings demonstrate noticeably lower accuracy, while the best result on the CommonsenseQA dataset is obtained using the KEAR model with knowledge base information injection via text-based embeddings.

However, from a practical point of view, the existence of other important factors should be taken into account when comparing approaches. For example, models based on self-supervised learning and fine-tuning, despite their lower accuracy, require less additional computations to obtain an answer to a query. At the same time, the very process of pretraining such models implies a rather significant expenditure of computational resources. In addition, not all implementations use the same language models, which in itself may result in differences in the final accuracy. The amount of time the model needs to obtain an answer it can also be considered as a relevant factor.

If proceeding solely from the results on the CommonsenseQA benchmark, it can be stated that the use of more architecturally complex models in general does not have a significant enough effect to compete with more established approaches that focus solely on the use of language models. Nevertheless, it continues to be worthwhile to continue the comparative analysis using other benchmarks as a means of better assessing the real state of art.

CONCLUSIONS

The presented review forms a basis to argue for the effectiveness of knowledge injection techniques in the field of QA system design. Already existing solutions experimentally confirm the possibility of simultaneously achieving several main goals of knowledge injection in this context.

However, there is still considerable room for further improvement in multiple aspects of the process. Firstly, currently relatively basic and well-established in the field of natural language processing methods for extracting data from knowledge bases for a query prevail. Only a few works propose ways to improve this process, such as paraphrasing additional knowledge from the database to simplify its processing by the system. In this context, given the potential importance of extracting relevant information in terms of further implementation, specific approaches can be considered along with their impact on the result.

Secondly, it is of interest to analyze the potential impact of choosing a particular graph model for processing structured information, since in existing works the main emphasis is shifted to comparison according to the criterion of the used language model. At the same time, over the last few years, many new promising models of knowledge graphs embeddings and graph neural networks have appeared, whose capabilities in the framework of practical tasks of this kind have yet to be established, but can significantly affect the results of the system as a whole.

Thirdly, there is currently a lack of systematic studies comparing methods for combining data from different modalities in the context of QA system design. This issue can also be considered relevant due to the possibility of generalizing to a wider range of tasks.

Finally, within the current vector of development of the field of QA system design leading to the prevalence of universal generative language models such as ChatGPT in applications, it makes sense to emphasize the study of the peculiarities of knowledge injection methods in this type of model.

REFERENCES

1. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2019;1:4171–4186. https://doi.org/10.18653/v1/N19-1423
2. Petroni F., Rocktäschel T., Lewis P., et al. Language Models as Knowledge Bases? *Processing* (*EMNLP-IJCNLP*). 2019. P. 2463–2473. https://doi.org/10.18653/v1/D19-1250
3. Sap M., Le Bras R., Allaway E., et al. ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2019;33(1):3027–3035. https://doi.org/10.1609/aaai.v33i01.33013027
4. Niven T., Kao H.-Y. Probing Neural Network Comprehension of Natural Language Arguments. *arXiv preprint arXiv:1907.07355*. 2019. https://doi.org/10.48550/arXiv.1907.07355
5. McCoy R. T., Pavlick E., Linzen T. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019. P. 3428–3448. http://doi.org/10.18653/v1/P19-1334
6. Li J., Chen J., Ren R., et al. The Dawn After the Dark: An Empirical Study on Factuality Hallucination in Large Language Models. *arXiv preprint arXiv:2401.03205*. 2024. https://doi.org/10.48550/arXiv.2401.03205
7. Wei J., Wang X., Schuurmans D., et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In: *36th Conference on Neural Information Processing Systems*. 2022;35:24824–24837. https://doi.org/10.48550/arXiv.2201.11903
8. Lewis P., Perez E. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems*. 2020;33:9459–9474. https://doi.org/10.48550/arXiv.2005.11401
9. Ye Zhi-Xiu, Chen Q., Wang W., Ling Zhen-Hua. Align, Mask and Select: A Simple Method for Incorporating Commonsense Knowledge into Language Representation Models. *arXiv preprint arXiv:1908.06725v5*. 2020. https://doi.org/10.48550/arXiv.1908.06725
10. Vaswani A., Shazeer N., Parmar N., et al. Attention Is All You Need. *Advances in Neural Information Processing Systems 30*. 2018. https://doi.org/10.48550/arXiv.1706.03762
11. Liu J., Shen D., Zhang Y., et al. What Makes Good In-Context Examples for GPT-3? *arXiv preprint arXiv:2101.06804*. 2021. https://doi.org/10.48550/arXiv.2101.06804
12. Gao T., Fisch A., Chen D. Making Pre-trained Language Models Better Few-shot Learners. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing* (*Volume 1: Long Papers*). 2021. P. 3816–3830. http://doi.org/10.18653/v1/2021.acl-long.295
13. Shwartz V., West P., Le Bras R., et al. Unsupervised Commonsense Question Answering with Self-Talk. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*). 2020. P. 4615–4629. http://doi.org/10.18653/v1/2020.emnlp-main.373
14. Wang J., Zhao H. ArT: All-round Thinker for Unsupervised Commonsense Question-Answering. In: *Proceedings of the 29th International Conference on Computational Linguistics*. 2022. P. 1490–1501. https://doi.org/10.48550/arXiv.2112.13428
15. Wang P., Peng N., Ilievski F., et al. Connecting the Dots: A Knowledgeable Path Generator for Commonsense Question Answering. *arXiv preprint arXiv:2005.00691*. 2020. https://doi.org/10.48550/arXiv.2005.00691
16. Raffel C., Shazeer N., Roberts A., et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*. 2020;21(140):1–67. https://doi.org/10.48550/arXiv.1910.10683
17. Zhang Z., Han X., Liu Z., et al. ERNIE: Enhanced Language Representation with Informative Entities. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019. P. 1441–1451. https://doi.org/10.18653/v1/P19-1139
18. Peters M.E., Neumann M., Logan IV R.L., et al. Knowledge enhanced contextual word representations. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*). 2019. P. 43–54. https://doi.org/10.18653/V1/D19-1005
19. He L., Zheng S., Yang T., Zhang F. KLMo: Knowledge Graph Enhanced Pretrained Language Model with Fine-Grained Relationships. In: *Findings of the Association for Computational Linguistics: EMNLP*. 2021. P. 4536–4542. https://doi.org/10.18653/v1/2021.findings-emnlp.384
20. Xiong W., Du J., Wang W.Y., Stoyanov V. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. *arXiv preprint arXiv:1912.09637*. 2019. https://doi.org/10.48550/arXiv.1912.09637
21. Sun Y., Wang S., Li Y., et al. ERNIE: Enhanced Representation through Knowledge Integration. *arXiv preprint arXiv:1904.09223*. 2019. https://doi.org/10.48550/arXiv.1904.09223
22. Zhang D., Yuan Z., Liu Y., et al. E-BERT: A Phrase and Product Knowledge Enhanced Language Model for E-commerce. *arXiv preprint arXiv:2009.02835*. 2020. https://doi.org/10.48550/arXiv.2009.02835
23. Chen Q., Li F.-L., Xu G., et al. DictBERT: Dictionary Description Knowledge Enhanced Language Model Pre-training via Contrastive Learning. *arXiv preprint arXiv:2208.00635*. 2022. https://doi.org/10.48550/arXiv.2208.00635

24. Lauscher A., Vulić I., Ponti E.M., et al. Informing Unsupervised Pretraining with External Linguistic Knowledge. *arXiv preprint arXiv:1909.02339v1*. 2019. https://doi.org/10.48550/arXiv.1909.02339
25. Levine Y., Lenz B., Dagan O., et al. SenseBERT: Driving Some Sense into BERT. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020. P. 4656–4667. https://doi.org/10.18653/v1/2020.acl-main.423
26. Wang X., Gao T., Zhu Z., et al. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Trans. Assoc. Comput. Linguis*. 2021;9:176–194. https://doi.org/10.1162/tacl_a_00360
27. Bordes A., Usunier N., Garcia-Durán A., et al. Translating Embeddings for Modeling Multi-relational Data. *Advances in Neural Information Processing Systems*. 2013. P. 2787–2795.
28. He B., Zhou D., Xiao J., et al. BERT-MK: Integrating Graph Contextualized Knowledge into Pre-trained Language Models. *Findings of the Association for Computational Linguistics: EMNLP*. 2020. P. 2281–2290. https://doi.org/10.18653/v1/2020.findings-emnlp.207
29. Banerjee P., Baral C. Self-Supervised Knowledge Triplet Learning for Zero-shot Question Answering. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*). 2020. P. 151–162. https://doi.org/10.18653/v1/2020.emnlp-main.11
30. Zhong W., Tang D., Duan N., et al. Improving Question Answering by Commonsense-Based Pre-training. In: Tang J., Kan M.Y., Zhao D., Li S., Zan H. (Eds.). *Natural Language Processing and Chinese Computing. NLPCC 2019. Lecture Notes in Computer Science*. Springer; 2019. V. 11838. P. 16–28. https://doi.org/10.1007/978-3-030-32233-5_2
31. Sun T., Shao Y., Qiu X., et al. CoLAKE: Contextualized Language and Knowledge Embedding. *arXiv preprint arXiv:2010.00309v1*. 2020. https://doi.org/10.48550/arXiv.2010.00309
32. Su Y., Han X., Zhang Z., et al. CokeBERT: Contextual knowledge selection and embedding towards enhanced pre-trained language models. *AI Open*. 2021;2:127–134. https://doi.org/10.1016/j.aiopen.2021.06.004
33. Ma K., Ilievski F., Francis J., et al. Knowledge-driven Data Construction for Zero-shot Evaluation in Commonsense Question Answering. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2021;35(15):13507–13515. https://doi.org/10.1609/aaai.v35i15.17593
34. Wang W., Fang T., Ding W., et al. CAR: Conceptualization-Augmented Reasoner for Zero-Shot Commonsense Question Answering. *Findings of the Association for Computational Linguistics: EMNLP 2023*. 2023. P. 13520–13545. https://doi.org/10.18653/v1/2023.findings-emnlp.902
35. Zhan X., Li Y., Dong X., et al. elBERto: Self-supervised Commonsense Learning for Question Answering. *arXiv preprint arXiv:2203.09424v1*. 2022. https://doi.org/10.48550/arXiv.2203.09424
36. Rajpurkar P., Jia R., Liang P. Know What You Don't Know: Unanswerable Questions for SQuAD. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*. 2018;2(Short Papers):784–789. https://doi.org/10.18653/v1/P18-2124
37. Khashabi D., Min S., Khot T., et al. UnifiedQA: Crossing Format Boundaries with a Single QA System. In: *Findings of the Association for Computational Linguistics*. 2020. P. 1896–1907. https://doi.org/10.18653/v1/2020.findings-emnlp.171
38. Lourie N., Le Bras R., Bhagavatula C., Choi Y. UNICORN on RAINBOW: A Universal Commonsense Reasoning Model on a New Multitask Benchmark. *arXiv preprint arXiv:2103.13009v1*. 2021. https://doi.org/10.48550/arXiv.2103.13009
39. Baek J., Aji A.F., Saffari A. Knowledge-Augmented Language Model Prompting for Zero-Shot Knowledge Graph Question Answering. In: *Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations* (*NLRSE*). 2023. P. 78–106. https://doi.org/10.18653/v1/2023.nlrse-1.7
40. Pan X., Sun K., Yu D., et al. Improving Question Answering with External Knowledge. In: *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. 2019. P. 27–37. https://doi.org/10.18653/v1/D19-5804
41. Xu Y., Zhu C., Xu R., et al. Fusing Context Into Knowledge Graph for Commonsense Question Answering. *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 2021. P. 1201–1207. https://doi.org/10.18653/v1/2021.findings-acl.102
42. Xu Y., Zhu C., Wang S., et al. Human Parity on CommonsenseQA: Augmenting Self-Attention with External Attention. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence* (*IJCAI*). 2022. P. 2762–2768. https://doi.org/10.24963/ijcai.2022/383
43. Arora S., Wu S., Liu E., Ré C. Metadata Shaping: A Simple Approach for Knowledge-Enhanced Language Models. In: *Findings of the Association for Computational Linguistics: ACL 2022*. 2022. P. 1733–1745. https://doi.org/10.18653/v1/2022.findings-acl.137
44. Li S., Gao Y., Jiang H., et al. Graph Reasoning for Question Answering with Triplet Retrieval. In: *Findings of the Association for Computational Linguistics: ACL 2023*. 2023. P. 3366–3375. https://doi.org/10.18653/v1/2023.findings-acl.208
45. Mitra A., Banerjee P., Pal K.K., et al. How Additional Knowledge can Improve Natural Language Commonsense Question Answering? *arXiv preprint arXiv:1909.08855v3*. 2020. https://doi.org/10.48550/arXiv.1909.08855
46. Chen Q., Zhu X., Ling Z.-H., et al. Neural Natural Language Inference Models Enhanced with External Knowledge. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* 2018;1(Long Papers):2406–2417. https://doi.org/10.18653/v1/P18-1224

47. Chen Q., Ji F., Chen H., Zhang Y. Improving Commonsense Question Answering by Graph-based Iterative Retrieval over Multiple Knowledge Sources. In: *Proceedings of the 28th International Conference on Computational Linguistics*. 2020. P. 2583–2594. https://doi.org/10.18653/v1/2020.coling-main.232
48. Ma K., Francis J., Lu Q., et al. Towards Generalizable Neuro-Symbolic Systems for Commonsense Question Answering. In: *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*. 2019. P. 22–32. https://doi.org/10.18653/v1/D19-6003
49. Bauer L., Wang Y., Bansal M. Commonsense for Generative Multi-Hop Question Answering Tasks. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018. P. 4220–4230. https://doi.org/10.18653/v1/D18-1454
50. Paul D., Frank A. Ranking and Selecting Multi-Hop Knowledge Paths to Better Predict Human Needs. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 2019;1(Long and Short Papers):3671–3681. https://doi.org/10.18653/v1/N19-1368
51. Liu W., Zhou P., Zhao Z., et al. K-BERT: Enabling Language Representation with Knowledge Graph. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. 2020;34(03):2901–2908. https://doi.org/10.1609/aaai.v34i03.5681
52. Kipf T.N., Welling M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv preprint arXiv:1609.02907*. 2017. https://doi.org/10.48550/arXiv.1609.02907
53. Lv S., Guo D., Xu J., et al. Graph-Based Reasoning over Heterogeneous External Knowledge for Commonsense Question Answering. In: *Proceedings of the AAAI Conference on Artificial Intelligence.* 2020;34(05):8449–8456. https://doi.org/10.1609/aaai.v34i05.6364
54. Feng Y., Chen Y., Lin B.Y., et al. Scalable Multi-Hop Relational Reasoning for Knowledge-Aware Question Answering. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing* (*EMNLP*). 2020. P. 1295–1309. https://doi.org/10.18653/v1/2020.emnlp-main.99
55. Sun Y., Shi Q., Qi L., Zhang Y. JointLK: Joint Reasoning with Language Models and Knowledge Graphs for Commonsense Question Answering. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2022. P. 5049–5060. https://doi.org/10.18653/v1/2022.naacl-main.372
56. Yan J., Raman M., Chan A., et al. Learning Contextualized Knowledge Structures for Commonsense Reasoning. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP*. 2021. P. 4038–4051. https://doi.org/10.18653/v1/2021.findings-acl.354
57. Lin B.Y., Chen X., Chen J., Ren X. KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing* (*EMNLP-IJCNLP*). 2019. P. 2829–2839. https://doi.org/10.18653/v1/D19-1282
58. Jiang J., Zhou K., Zhao W.X., Wen J.-R. Great Truths are Always Simple: A Rather Simple Knowledge Encoder for Enhancing the Commonsense Reasoning Capacity of Pre-Trained Models. In: *North American Chapter of the Association for Computational Linguistics-Findings*. 2022. https://doi.org/10.48550/arXiv.2205.01841
59. Houlsby N., Giurgiu A., Jastrzebski S., et al. Parameter-Efficient Transfer Learning for NLP. In: *Proceedings of Machine Learning Research*. 2019;97:2790–2799. https://doi.org/10.48550/arXiv.1902.00751
60. Wang R., Tang D., Duan N., et al. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP.* 2021. P. 1405–1418. https://doi.org/10.18653/v1/2021.findings-acl.121
61. Kim Y.J., Kwak B., Kim Y., et al. Modularized Transfer Learning with Multiple Knowledge Graphs for Zero-shot Commonsense Reasoning. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2022. P. 2244–2257. https://doi.org/10.18653/v1/2022.naacl-main.163
62. Jacobs R., Jordan M., Nowlan S., Hinton G. Adaptive Mixtures of Local Experts. *Neural Computation*. 1991;3(1):79–87. https://doi.org/10.1162/neco.1991.3.1.79
63. Zhang X., Bosselut A., Yasunaga M., et al. GreaseLM: Graph REASoning Enhanced Language Models for Question Answering. In: *The International Conference on Learning Representations* (*ICLR*). 2022. https://doi.org/10.48550/arXiv.2201.08860
64. Yasunaga M., Bosselut A., Ren H., et al. Deep Bidirectional Language-Knowledge Graph Pretraining. In: *36th Conference on Neural Information Processing Systems* (*NeurIPS*). 2022. https://doi.org/10.48550/arXiv.2210.09338
65. Yasunaga M., Ren H., Bosselut A., et al. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 2021. P. 535–546. https://doi.org/10.18653/v1/2021.naacl-main.45

66. Su Y., Zhang J., Song Y., Zhang T. PipeNet: Question Answering with Semantic Pruning over Knowledge Graphs. *arXiv preprint arXiv:2401.17536v2*. 2024. https://doi.org/10.48550/arXiv.2401.17536
67. Talmor A., Herzig J., Lourie N., Berant J. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In*: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2019;1(Long and Short Papers):4149–4158. https://doi.org/10.18653/v1/N19-1421
68. Liu Y., Ott M., Goyal N., et al. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*. 2019. https://doi.org/10.48550/arXiv.1907.11692
69. Robertson S.E., Walker S. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In: *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*. 1994. P. 232–241. https://doi.org/10.1007/978-1-4471-2099-5_24

About the Author

Daniil V. Radyush, Postgraduate Student, Faculty of Software Engineering and Computer Systems, ITMO University (49-A, Kronverkskii pr., Saint Petersburg, 197101 Russia). E-mail: daniil.radyush@gmail.com. Scopus Author ID 58234958500, https://orcid.org/0000-0001-8823-0609

Об авторе

Радюш Даниил Валентинович, аспирант, факультет программной инженерии и компьютерной техники, ФГАОУ ВО «Национальный исследовательский университет ИТМО» (197101, Россия, Санкт-Петербург, Кронверкский пр., д. 49, лит. А). E-mail: daniil.radyush@gmail.com. Scopus Author ID 58234958500, https://orcid.org/0000-0001-8823-0609

Translated from Russian into English by L. Bychkova
Edited for English language and spelling by Thomas A. Beavitt