Unsupervised Context-Driven Question Answering Based on Link Grammar

Vignav Ramesh[1,2] and Anton Kolonin[2,3,4]

[1] Saratoga High School, Saratoga, USA
[2] SingularityNET Foundation, Amsterdam, Netherlands
[3] Aigents, Novosibirsk, Russian Federation
[4] Novosibirsk State University, Russian Federation
{rvignav, akolonin}@gmail.com

Abstract. While general conversational intelligence (GCI) can be considered one of the core aspects of artificial general intelligence (AGI), there currently exists minimal overlap between the disciplines of AGI and natural language processing (NLP). Only a few AGI architectures can comprehend and generate natural language, and most NLP systems rely either on hardcoded, specialized rules and frameworks that cannot generalize to the various complex domains of human language or on heavily trained deep neural network models that cannot be interpreted, controlled, or made sense of. In this paper, we propose an interpretable "Contextual Generator" architecture for question answering (QA), built as an extension of the recently published "Generator" algorithm for sentence generation, that produces grammatically valid answers to queries structured as lists of seed words. We demonstrate the potential for this architecture to perform automated, closed-domain QA by detailing results on queries from SingularityNET's "small world" POC-English corpus and from the Stanford Question Answering Dataset. Overall, our work may bring a greater degree of GCI to proto-AGI NLP pipelines. The proposed QA architecture is open-source and can be found on GitHub under the MIT License at https://github.com/aigents/aigents-java-nlp.

Keywords: General Conversational Intelligence, Interpretable Natural Language Processing, Natural Language Generation, Question Answering, Link Grammar.

1 Introduction

General conversational intelligence (GCI) can be considered one of the core aspects of artificial general intelligence (AGI); however, there currently exists minimal overlap between the disciplines of AGI and natural language processing (NLP). Only a few AGI architectures can comprehend and generate natural language, and most NLP systems rely on either hardcoded, specialized rules and frameworks that cannot generalize to the various complex domains of human language or on heavily trained deep neural network models that cannot be interpreted, controlled, or made sense of [1]. Moreover, the majority of AGI frameworks that do possess some level of natural language comprehension (NLC) or natural language understanding (NLU) cannot convey such

knowledge (e.g., a response to a query posed by a human) in natural language without template-based customization or similar manual, labor-intensive procedures (Goertzel et al., 2010; Goertzel and Yu, 2014).

In this paper, we propose a question answering (QA) architecture, founded upon an extension of the sentence generation system described in Section 2.1, that serves as an interpretable natural language processing (INLP) method. INLP, as proposed in [2], is an extension of interpretable AI (IAI)—which expands upon explainable AI (XAI) by calling for an interpretable model/knowledge base as well as explainable results—to NLP; INLP enables the acquisition of natural language, comprehension of text-based messages, and production of linguistic content in a reasonable and transparent manner [3]. The proposed method of QA satisfies the criteria for interpretability by providing both explainable results as well as an interpretable model for NLP in general and QA in particular, since we rely on Link Grammar (LG) as our formal grammar which itself is comprehensible [4]. To this end, our QA architecture may bring a greater degree of GCI to proto-AGI NLP pipelines.

1.1 Natural Language Generation

Natural language generation (NLG) is the task of producing linguistic content (most often in the form of grammatically and morphologically valid text) from semantic and/or non-linguistic data [5]. Even the process of producing simple sentences—a sub-problem of NLG known as sentence generation, which we will treat as synonymous with NLG for the purposes of this paper—requires significant grammatical, syntactical, morphological, and phonological knowledge. While the integration of semantic knowledge could improve our results and warrants future research, the NLG component of our current QA pipeline focuses on the use of solely grammatical knowledge.

Within the sentence generation process is the task of surface realization, which is concerned with the construction of sentences from the underlying content of a text, usually structured as an unordered set of tokens (words, punctuation, etc.) [6]. As we explain in Section 2.2, the same tokens can often be arranged into multiple grammatically and morphologically valid sentential forms that an NLG system must disambiguate in context, a process known as semantic disambiguation [7].

1.2 Question Answering

QA, a branch of computer science integrating information retrieval and NLP, refers to the ability for machines to automatically answer questions posed in the form of human language. As discussed in [8], an explainable QA pipeline involves components for both NLC and NLG. During NLC, the question, or input query, is parsed and then semantically interpreted (the pipeline determines the underlying “themes” or “topics” of the question and then computes relationships between each component of the parsed query). During NLG, the query is then semantically executed (the semantic relationships obtained during NLC are used to find the answer to the question), and finally, a formal grammar is utilized to construct a grammatically valid sentence from words associated with the non-linguistic answer obtained in the query execution step. Below is a diagram of an interpretable QA pipeline.

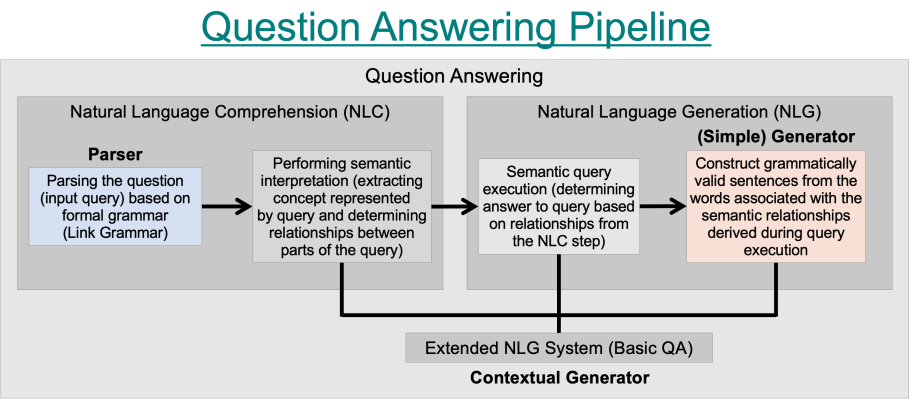

Fig. 1. QA workflow

Our current work is concerned with a more basic version of QA, in which the query is proposed as a list of seed words (in natural language) and the answer is a grammatically valid sentence that correctly captures the relationships between the seed words. In this sense, our work replaces the semantic interpretation and query execution components of the described QA pipeline with an extended version of sentence generation based on the context of the question and scope of textual data to search for an answer. That is, we treat the context of the question as a constraint for the generation phase of QA; in the perspective of the above pipeline, it is as if the words needed to answer the query are already known based on prior semantic query execution and must simply be arranged into a grammatically valid response. Given that most modern search queries online (e.g., Google, Bing, Yahoo) are performed only with keywords rather than properly structured sentences, we anticipate that our work would be practical for human-computer interactions.

1.3 Link Grammar

LG is founded upon the fact that each word is defined by a set of connectors and disjuncts associating those connectors. Connectors serve as either the left or right half of a grammatical link of a given type, and disjuncts are sets of connectors that define the valid grammatical context of a given word [1]. Rules—which represent lexical entries or grammatical categories—describe the sets of defining disjuncts for clusters of grammatically equivalent words. From our perspective, NLG is the process of following rules to construct sentences by matching connectors between words to form links.

LG also imposes two additional constraints: the planarity metarule, which specifies that links must not cross, and the connectivity metarule, which dictates that all links and tokens in a sentence must form a connected graph. Furthermore, unlike most alternative grammar rule dictionaries and APIs (spaCy[1], Universal Dependencies[2], etc.), LG does not require grammar rules to be hardcoded into client-side architectures; LG rules

[1] `https://spacy.io`
[2] `https://universaldependencies.org`

can also be learned dynamically as has been shown in our previous work [9]. Overall, the human-readable and editable nature of LG allows our grammar induction algorithm to better serve as an INLP method for the purposes of the QA task.

1.4 Prior Work

Sentence Generation. The proposed NLG and QA methods can be called unsupervised since they do not require prior training on supervisedly prepared corpora. However, the majority of published natural language generation methods are supervised and/or rely on deep learning models that require extensive training on labeled data; as such, they are "black box" algorithms that are neither explainable nor interpretable (Ratnaparkhi, 2000; Wen et al., 2015; Dathathri et al., 2020). There are only a few notable unsupervised (yet often not interpretable) NLG systems.

Lian et al. proposed SegSim, an approach based on the OpenCog NLGen software that constructs sentences by satisfying constraints posed by inverse relations of hypergraph homomorphisms. SegSim performs surface realization by matching the subsets of an Atom set in need of linguistic expression against a datastore of (sentence, link parse, RelEx relationship set, Atom set) tuples produced by OpenCog's NLC software. This matching allows SegSim to determine the syntactic structures that have previously been used to generate relevant Atom subsets, and these structures are then pieced together to form overall syntactic structures corresponding to one or more sentences. The sentence is solved for as a constraint satisfaction problem from the Atom set semantics [1]. SegSim constructs simple sentences unproblematically but becomes unreliable for more syntactically complex sentences (e.g., those involving conjunctions).

Freitag and Roy proposed an unsupervised NLG system in which denoising autoencoders are used to construct sentences from structured data interpreted as "corrupt representations" of the desired output. The denoising autoencoders can also generalize to unstructured training samples to which noise has already been introduced [10].

Question Answering. There exist a variety of well-known, large, supervised language understanding models that have been applied to the QA task. BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers), a pre-trained language representation model that learns bidirectional representations from unlabeled text, can be fine-tuned on token sequences representing labeled question-answer pairs to perform the downstream task of QA [11]. DistilBERT, a distilled version of BERT, implements knowledge distillation during the pre-training phase to reduce the size of BERT by 40% while retaining 97% of its language understanding capabilities and being 60% faster [12]. LUKE (**L**anguage **U**nderstanding with **K**nowledge-based **E**mbeddings), a multilayer bidirectional transformer utilizing a novel entity-aware self-attention mechanism, is trained by predicting randomly masked words and entities in the input corpora. When fine-tuned to perform both cloze-style and extractive QA, LUKE achieves state-of-the-art results [13]. ELECTRA, RoBERTa, and BART are among other popular machine reading comprehension models [14–16].

However, all such models are supervised and uninterpretable; there are only two notable published methods for unsupervised QA. Lewis et al. proposed an unsupervised QA method whereby (context, question, answer) triples are generated unsupervisedly and then used to synthesize extractive QA training data in "cloze" format [17]. Perez et

al. developed an unsupervised QA approach focused on decomposing a single hard, multi-hop question into several simpler, single-hop sub-questions that are answered with an off-the-shelf QA model and recomposed into a final answer [18].

2 Methodology

2.1 Generator Architecture for NLG

To perform QA in an unsupervised and interpretable manner, we adopt and extend the Generator architecture proposed in [2]. After calling the Loader, utility infrastructure as shown in Fig. 2, to store the LG dictionary in memory, the Generator performs surface realization as follows:

1. Given a list of words, computes a subset of all orderings of those words that satisfies initial checks of the planarity and connectivity meta-rules.
2. Determines if each ordering is grammatically valid; that is, ensures that every pair of consecutive tokens can be linked via a pair of "connectable" disjuncts.
3. Returns all grammatically valid orderings as sentences.

2.2 Contextual Generator Architecture for QA

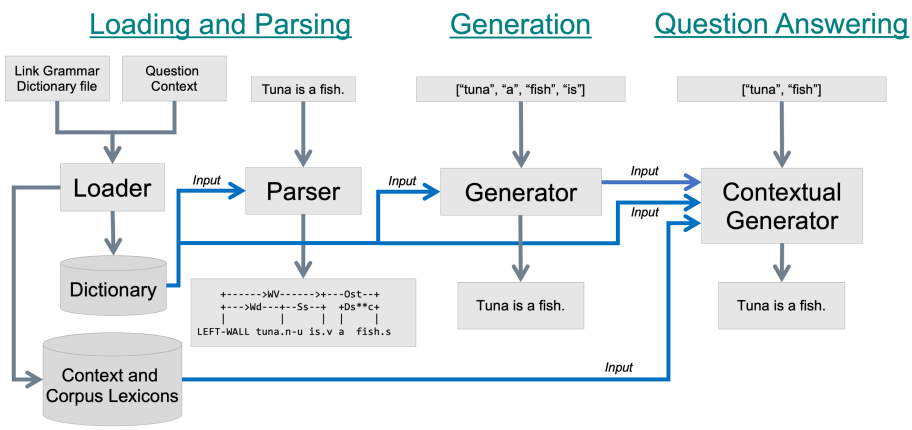

Fig. 2. INLP architecture involving a sentence parsing algorithm (the "Parser") and QA framework (the "Contextual Generator") built upon an NLG algorithm (the "Generator"), all relying on the same Loader infrastructure.

In this paper, we propose a Contextual Generator (CG) architecture which extends our prior NLG architecture in two major ways:

1. The surface realization model described in Section 2.1 is expanded recursively to build grammatically and morphologically valid sentences from two or more seed words that comprise only a subset of the total list of words needed to construct the final sentence (see "Contextual Generator").

2. To allow for semantic disambiguation—determining which of multiple sentential answers to a given query is most appropriate in the given context—the revised Loader architecture, besides loading the LG dictionary and corpus lexicon, also loads context lexicons. Each word in the context lexicon is supplied with a weight denoting its contribution to the context in accordance with Zipf's Law (see "Zipfian Calculations"). Note that the Loader creates the corpus lexicon only once, but builds a new lexicon for each unique context.

Two sample question-answer pairs, with answers generated by our CG architecture, are shown below. The CG generates correct answers but, as seen in the first question-answer pair, succumbs to the grammatical ambiguity problem described in Section 3.

Context: Identity and Relationships
A **mom** is a **human**. A **dad is a human**. A **mom** is a parent. A **dad** is a parent. A son is a child. A **daughter** is a child. A son is a **human**. A **daughter** is a **human**. **Mom** is a **human** now. **Dad** is a **human** now. **Mom** is a parent now. **Dad** is a parent now. Son is a child now. **Daughter** is a child now. Son is a **human** now. **Daughter** is a **human** now. **Mom was a daughter before. Dad** was a son before. **Mom** was not a parent before. **Dad** was not a parent before.

Question: mom daughter
Answer: [**Mom was a daughter before.**, **Daughter was a mom before.**]

Question: dad human
Answer: [**Dad is a human.**]

Contextual Generator. The CG performs closed-domain QA on queries structured as lists of seed words. It operates as follows:

1. The CG calls the Loader to store the LG dictionary and corpus lexicon as well as extract a context lexicon from a given document as described above.
2. The CG then recursively calls the Generator to determine valid sentence constructions from a list containing all seed words and n additional words from a subset of words in the context lexicon that satisfy initial checks of the planarity and connectivity meta-rules (e.g., one partial connectivity check confirms that the first and last tokens in a potential sentence can form links to the right and left, respectively). The n-addition permutations are tested in order of decreasing Zipfian frequency corresponding to the current context. If a valid sentence is found, the CG returns that sentence and stops running. When step 2 is first executed, $n = 1$, imposing the constraint that the answer is of minimum length.
3. If no sentences have been generated after testing all valid subsets and the runtime has not exceeded a limit specified to avoid combinatorial explosions (3 minutes in our experiments), the CG increases n by 1 and repeats step 2.

Zipfian Calculations. In the previous subsection, it was specified that the CG checks n-addition permutations in order of decreasing Zipfian frequency. To motivate this,

consider the sample query “mom cake” along with a context specifying the food preferences of family members. If the CG were to use context word frequencies without modification rather than Zipfian frequencies, it would perceive the sentence “Cake was a mom” as more contextually appropriate than “Mom likes cake,” which is the ground truth answer. This is due to a phenomenon known as Zipf’s law [19], which states that the rank-frequency distribution of words in a given lexicon is an inverse relation; determiners and linking verbs (“a,” “the,” “was,” “is,” etc.) are more common in any corpus, regardless of context, than more semantically appropriate words (such as “likes” in the context of food preferences as in the motivating example above). To account for Zipf’s law, we use the Zipfian frequency, which is calculated as follows:

$$Z_w = \frac{\log(1 + F_X(w))}{\log(1 + F_C(w))}, \tag{1}$$

where w is the given word, $F_X(w)$ is the frequency of w in the context document, and $F_C(w)$ is the frequency of w in the corpus. Because the Zipfian frequency divides the logarithm of the context frequency by that of the corpus frequency, more contextually appropriate words receive higher scores than do determiners and similarly common words. For multiple-word additions, we sum individual Zipfian frequencies.

3 Results

Our algorithm was primarily tested on 60 queries with words all part of SingularityNET’s “small world” POC-English corpus.[3] For this purpose, we have used a corresponding “small world” LG dictionary (automatically inferred from high quality LG parses created by SingularityNET’s ULL pipeline).[4] To evaluate the proposed architecture, we report four scores: the bigram variant of BLEU (**Bil**ingual **E**valuation **U**nderstudy), a measure of the number of matching bigrams in two sentences [20]; WVCS (**W**ord2**V**ec **c**osine **s**imilarity), calculated as the cosine of the angle between the vector encodings of the candidate and reference sentences [21]; WER (**W**ord **E**rror **R**ate), a measure of the edit distance between two sentences [22]; and TER (**T**ranslation **E**dit **R**ate), another measure of edit distance [23]. Note that higher BLEU and WVCS scores as well as lower WER and TER scores indicate more accurate language models. Each metric is calculated for each (answer, ground truth answer) pair and then averaged over all 60 queries.

Our QA architecture significantly outperforms prior state-of-the-art models on the POC-English dataset for the task of QA from short lists of seed words. As baselines, we implemented BERT, ELECTRA, DistilBERT, and RoBERTa models that were pretrained on the **S**tanford **Qu**estion **A**nswering **D**ataset (SQuAD2.0) and then fine-tuned on the POC-English corpus [24]. Our QA system attains superior BLEU, WVCS, and TER scores and competitive WER scores, demonstrating an 0.11 increase in BLEU, 0.15 increase in WVCS, and 0.08 decrease in TER from the best baseline results.

---

[3] `http://langlearn.singularitynet.io/data/poc-english/poc_english.txt`

[4] `http://langlearn.singularitynet.io/test/nlp/poc-english_5C_2018-06-06_0004.4.0.dict.txt`

Table 1. Results when tested on 60 queries from SingularityNET's POC-English corpus.

Metric	Results				
	Ours	**BERT**	**ELECTRA**	**DistilBERT**	**RoBERTa**
BLEU	**0.878**	0.639	0.712	0.604	0.767
WVCS	**0.944**	0.606	0.741	0.595	0.799
WER	0.645	0.924	0.550	1.095	**0.150**
TER	**0.166**	0.381	0.342	0.457	0.245

Our results were mainly affected by the issue of grammatical ambiguity, a problem whereby the same word can take on different roles in a sentence (e.g., subject-object ambiguity, where a noun can be either the subject or object of a sentence). While both sentences are grammatically valid, only one is semantically correct; implementing semantic disambiguation as part of the NLG component of our pipeline beyond that described in Section 2.2 will be part of our future work.

We additionally tested our algorithm on three samples randomly obtained from SQuAD2.0 using the complete LG; all samples were cleaned (questions were restated as sets of seed words and proper nouns/phrases not present in LG were removed):

Sample 1

Context: Imperialism

Imperialism is defined as "A policy of extending the power and influence of a country through diplomacy or military force." **Imperialism** is particularly **focused on** the **control** that one group, often a state power, has **on** another group of people. There is "formal" and "informal" **imperialism**. "Formal **imperialism**" is defined as "physical control or colonial rule". "Informal **imperialism**" is less direct; however, it is still a powerful form of dominance.

Question: imperialism focused on

Answer: [**Imperialism focused on control.**]

Sample 2

Context: Pharmacy

Pharmacists are healthcare professionals with specialized education and training who perform various roles to ensure optimal health outcomes for their patients through the quality use of medicines. **Pharmacists** may also be small business **proprietors**, owning the pharmacy in which they practice. Since **pharmacists** know about the mode of action of a particular drug, and its metabolism and physiological effects on the human body in great detail, they play an important role in optimization of a drug treatment for an individual.

Question: pharmacists are

Answer: [**Pharmacists are proprietors.**]

Sample 3

Context: Normans

A **tradition** of singing had developed and the choir achieved fame in Normandy. Under the Norman abbot Robert, several **monks fled to** southern Italy, where they were patronized by Robert and established a Latin monastery. There they continued the **tradition** of singing.

Question: monks fled to

Answer: [**Monks fled to tradition.**]

Samples 1, 2, and 3 show the benefits and drawbacks of QA using Zipfian frequencies. While Samples 1 and 2 correctly and partially correctly identify the additional word needed to answer the question in context, respectively, Sample 3 incorrectly chooses the word “tradition” over “Italy” since “tradition” appears twice in the context document while “Italy” appears once. Fine-tuning our QA architecture to account for such inconsistencies between Zipfian frequency and contextual appropriateness will be part of our future work.

4 Conclusion

We propose a novel CG architecture to solve the problem of basic QA from lists of seed words. Our algorithm serves as an INLP method and largely outperforms current state-of-the-art QA models on the POC-English corpus.

Our QA architecture will primarily be applied to the Aigents Social Media Intelligence Platform [25]. If integrated into the Aigents cognitive architecture—which currently depends on artificially controlled language similar to oversimplified “pidgin” English—our algorithm could provide GCI to Aigents chatbots.

Our future work will involve: 1) testing our algorithm on queries from arbitrary English documents using the complete LG dictionary; 2) implementing grammatical disambiguation; 3) expanding our algorithm’s QA capabilities by building an integrated LG schema containing semantic as well as grammatical knowledge; 4) implementing the full QA pipeline including semantic interpretation and query execution preceding response generation; and 5) adding support for languages other than English (including languages such as Russian that require heavy morphology usage).

References

1. Lian, R., et al.: Syntax-Semantic Mapping for General Intelligence: Language Comprehension as Hypergraph Homomorphism, Language Generation as Constraint Satisfaction. In: Bach, J., Goertzel, B., Iklé, M. (eds.) ARTIFICIAL GENERAL INTELLIGENCE 2012, LNCS, vol 7716, pp. 158–167. Springer, Heidelberg (2012).
2. Ramesh, V., Kolonin, A.: Natural Language Generation Using Link Grammar for General Conversational Intelligence. arXiv:2105.00830 [cs.CL] (2021).
3. https://link.springer.com/chapter/10.1007/978-3-030-27005-6_11
4. Sleator, D., Temperley, D.: Parsing English with a Link Grammar. In: Proceedings of the Third International Workshop on Parsing Technologies, pp. 277–292. Association for Computational Linguistics, Netherlands (1993).
5. Gatt, A., Krahmer, E.: Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. Journal of AI Research (JAIR) 61(1), 75–170 (2018).
6. Mellish, C., Dale, R.: Evaluation in the context of natural language generation. Computer Speech and Language 10(2), 349–373 (1998).
7. Weaver, W.: Translation. In: Locke, W., Booth, A. (eds.) MACHINE TRANSLATION OF LANGUAGES: FOURTEEN ESSAYS, Technology Press of the Massachusetts Institute of Technology (1955).

8. Ramesh, V., Kolonin, A.: Interpretable Natural Language Segmentation Based on Link Grammar. In: 2020 Science and Artificial Intelligence conference (S.A.I.ence), pp. 25–32. IEEE, Novosibirsk (2020).
9. Glushchenko, A., Suarez, A., Kolonin, A., Goertzel, B., Baskov, O.: Programmatic Link Grammar Induction for Unsupervised Language Learning. In: Hammer, P., Agrawal, P., Goertzel, B., Iklé, M. (eds.) ARTIFICIAL GENERAL INTELLIGENCE 2019, LNCS, vol 11654, pp. 111-120. Springer, Cham (2019).
10. Freitag, M., Roy, S.: Unsupervised Natural Language Generation with Denoising Autoencoders. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3922–3929. Association for Computational Linguistics, Belgium (2018).
11. Devlin, J., et al.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805v2 [cs.CL] (2019).
12. Sanh, V., et al.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv:1910.01108v4 [cs.CL] (2020).
13. Yamada, I., et al.: LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. arXiv:2010.01057 [cs.CL] (2020).
14. Clark, K., et al.: ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. arXiv.2003.10555 [cs.CL] (2020).
15. Liu, Y., et al.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL] (2019).
16. Lewis, M., et al.: BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. arXiv:1910.13461 [cs.CL] (2019).
17. Lewis, P., et al.: Unsupervised Question Answering by Cloze Translation. arXiv:1906.04980v2 [cs.CL] (2019).
18. Perez, E., et al.: Unsupervised Question Decomposition for Question Answering. arXiv:2002.09758v3 [cs.CL] (2020).
19. Powers, D.: Applications and Explanations of Zipf's Law. In: New Methods in Language Processing and Computational Natural Language Learning, pp. 151–160. Association for Computational Linguistics (1998).
20. Papineni, K., et al.: Bleu: A Method for Automatic Evaluation of Machine Translation. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics, USA (2002).
21. Sitikhu, P., et al.: A Comparison of Semantic Similarity Methods for Maximum Human Interpretability. arXiv:1910.09129 [cs.IR] (2019).
22. Klakow, D., Peters, J.: Testing the Correlation of Word Error Rate and Perplexity. Speech Communication 38(1), 19–28 (2002).
23. Snover, M., et al.: A Study of Translation Edit Rate with Targeted Human Annotation. In: Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers, pp. 223–231. Association for Machine Translation in the Americas , USA (2006).
24. Rajpurkar, P., et al.: SQuAD: 100,000+ Questions for Machine Comprehension of Text. arXiv:1606.05250 [cs.CL] (2016).
25. Kolonin, A.: Personal Analytics for Societies and Businesses: With Aigents Online Platform. In: 2017 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON), pp. 272–275. IEEE, Novosibirsk (2017).