A Conversational Framework for Semantic Question Answering in Customer Services with Machine Learning on Knowledge Graph *

Lazzarinetti Giorgio[1][0000−0003−0326−8742] and Massarenti Nicola[1][0000−0002−8882−4252]

Noovle S.p.A, Milan, Italy https://www.noovle.com/en/

Abstract. Despite the recent advances in Natural Language Processing (NLP) techniques many issues and inefficiencies arise when it comes to creating a system capable of interacting with the users by means of text conversations. The current techniques rely on the development of chatbots that, however, require designing the conversation flow, defining training questions and associating the expected responses.

Even though this process allows the creation of effective question-answering systems, this methodology is not scalable, especially when the answers are to be found in documents. Other approaches, instead, rely on graph embedding techniques and graph neural networks to define the best answer given a question. These methods, however, require set-up training routines and to dispose of the ground truth that, in general, is difficult to retrieve or create for real industrial applications.

In this paper we introduce a conversational framework for semantic question answering. Our work relies on knowledge graphs and the use of machine learning for determining the best answer given a question associated with the content of the knowledge graph. In addition, by leveraging text mining techniques we are able to identify the best set of answers that suit the question that are further filtered by means of deep learning algorithms.

Keywords: Semantic Question Answering · Knowledge Graph · Machine Learning.

1 Overview

In recent years the advances in chatbot development have given rise to efficient frameworks for virtual assistants creation. Even though it's possible to

* Activities were partially funded by Italian "Ministero dello Sviluppo Economico", Fondo per la Crescita Sostenibile, Bando "Agenda Digitale", D.M. Oct. 15th, 2014 - Project n. F/020012/02/X27 - "Smart District 4.0".

design complex conversational flows, these frameworks have not yet solved the core aspect of the technology: the question/answer pair definition. It's indeed mandatory not only to design the conversational flow, but also to define a set of questions associated with the response that the virtual assistant, i.e. the chatbot, has to return. This process exposes the technology to inefficiencies mainly due to the manual process of analysing the use case, defining the conversational flow, defining the conversational steps, associating the training examples and the answers to each conversational steps.

In industrial contexts it's often the case where the set of questions and the respective answers are already defined into documents that, however, are often unstructured because designed for human readers. In these situations, especially if the number of documents is non negligible, to retrieve the questions and answers pair becomes impossible, requiring the business owners to define manually a subset of the original questions or to avoid adopting this technology.

The framework we introduce in this paper proposes to solve the aforementioned problem. Thanks to a proprietary technology we map the content of arbitrarily shaped documents into a knowledge graph and by means of the synergy between text mining techniques and deep learning algorithms we are able to identify the best answer without any manual setup required nor training of custom models.

This research is driven by the business needs of a partner company that aims at creating an application to continuously improve an existing conversational customer service agent able to assist customers to manage and maintain the machines and software they produce. Indeed, they produce two main kinds of machines and a management software. For each product, they carefully write an user manual that contains all the necessary information to understand, use and maintain that product. However, at every new release of the products they need to update the manual, thus possibly changing also the intents, the training phrases and the answers of the chatbot system designed for customer service. For this reason we designed the framework presented in this paper, which can manage this issue by directly creating the query-answer engine using the product's manuals.

The rest of this paper is organized as follows. Chapter 2 presents an overview of the state of the art, with a focus on the methodologies for graph question answering, in Chapter 3 the knowledge graph schema and the framework logic is defined and in Chapter 4 the experimental results obtained are shown. Finally, Chapter 5 draws some considerations about the infrastructural setup and in Chapter 6 some conclusions and future directions are mentioned.

2 State of the art

For a long time has been aimed the use of systems capable of assisting users by providing instant responses to their questions. Indeed such systems, today called chatbots, have been studied and investigated since 1960. Many proposals have been given to birth, such as Eliza [1], Parry [2], and Alice [3], but only in recent years AI chatbots have become more reliable and accurate (such as Google DialogFlow [28], Amazon Lex [29], Azure Bot Service [30]).The latest advances in this field of study have produced frameworks that are easily configurable but the design of the conversational flow needed to train such chatbot engines is complex, often error-prone and incomplete due to the lack of a context and semantic aware approach of structuring information (intent, entity and training phrases). In recent years, to take into account context-aware applications with a semantic perspective many researchers focus on the study of knowledge graph-based chatbot systems.

Human knowledge provides a formal understanding of the world. Knowledge graphs have become an increasingly popular research direction towards cognition and human-level intelligence. A knowledge graph is a representation of the knowledge contained in an ontology, a structured approach to represent concepts and relations that belong to a shared conceptualization of a specific domain [4, 9]. More specifically, it can be defined as the union of concepts, relations, attributes and hierarchies that belong to a domain. Indeed, the formalization proposed for the Web Ontology Language (OWL) [5] allows to map the ontology into a knowledge graph by considering the nodes as the entities of the domain, whereas the edges represent the relations among two or more entities [6].

The main research topics related to knowledge graphs cover four main aspects: knowledge graph representation learning, knowledge acquisition and completion, temporal knowledge graph and knowledge-aware applications. Among knowledge-aware applications, an extremely active area of research is represented by knowledge-graph-based question answering that answers natural language questions with facts from knowledge graphs. There exist several proposals for question answering systems based on knowledge graphs. For example, in [7] the authors propose a knowledge embedding based question answering system that relies on a low-dimensional embedding representation of the graph, on the training of two models and on the definition of a joint distance metric that takes into account the embeddings representations. Also other knowledge graph based question answering systems have been proposed: in [10] the author proposes to project the question representation and the candidate responses into the same high dimensional space by learning the latent representations for words, predicates and entities with respect to the training questions. In [11], instead, the candidates have been ranked by means of a bidirectional gated recurrent units based neural network, whereas a character-level convolutional neural network has been proposed in [12] to match the questions and the predicates. A character-level and attention-based long-short-term-memory is used in [13] to encode and decode questions. In [14] the author manually defines several constraints used to perform constraint learning with the aim of handling complex questions.

Instead in [8] the authors present a hierarchical graph network for multi-hop question answering framework that is based on the separation of the information in different levels of granularity: questions, paragraphs, sentences, entities. They leverage the use of pre-trained contextual encoders to update the hierarchical graph so as to reason across multiple documents or paragraphs. The authors of [15] decompose the question into simpler sub-questions and take advantage of the single-hop NLP models to answer the questions, whereas in [16] is proposed a neural modular network dynamically composed for more advanced multi-hop reasoning.

Other approaches make use of graph neural networks, such as [17–19], to reason over the constructed graph. In [20] the authors propose a method that is based on the definition of three types of edge. Instead in [21] graph attention (or self-attention) on entity graphs is used.

3 Methodology

The framework that is presented in this paper has the objective of being a reliable system to associate user questions to the content of arbitrarily structured documents. With this objective in mind the design choices led to a solution that is general enough to map the content of documents and that requests no training because, in general, the business applications for which this framework has been developed almost never comes with the ground truth labels. For such reasons, the main drivers that led the design choices are:

- The definition of a knowledge graph schema that is able of mapping the content of arbitrarily structured documents
- The adoption of text mining techniques and the use of pre-trained deep learning models on relevant content in order to avoid further training
- The execution of iterative searches with increasingly less search constraints and the definition of acceptance thresholds

In section 3.1 will be presented the graph schema as well as will be described the entities and relations involved. In section 3.2 will be presented the search logic, the text mining techniques, the deep learning algorithms adopted and the iterative constraint relaxation process that has the objective of increasing the number of candidate answers to be associated with the user question.

3.1 Knowledge graph schema

The knowledge graph is defined as $G = (A, E, R)$ where $A = \{a_1, \ldots, a_{\bar{A}}\}$ is the set of attributes, $E = \{e_1, \ldots, e_{\bar{E}}\}$ is the set of entities and $R = \{r_1, \ldots, r_{\bar{R}}\}$ is the set of relations. Its schema has been designed with the aim of mapping the content of arbitrarily structured documents. Knowledge graph schema is depicted in Figure 1.

The entities (purple rounded rectangles) represent the domain concepts and are characterized by one or more attributes (blue ellipses). The relations (green hexagons) link two or more entities and can have one or more attributes.

As it may be noticed, some *contains* relations own the attribute *index*. This has been added to reconstruct the content of each *chapter/paragraph* by ordering the contained elements. In addition, any paragraph may contain an arbitrary number of paragraphs to allow replicating the structure of complex documents,

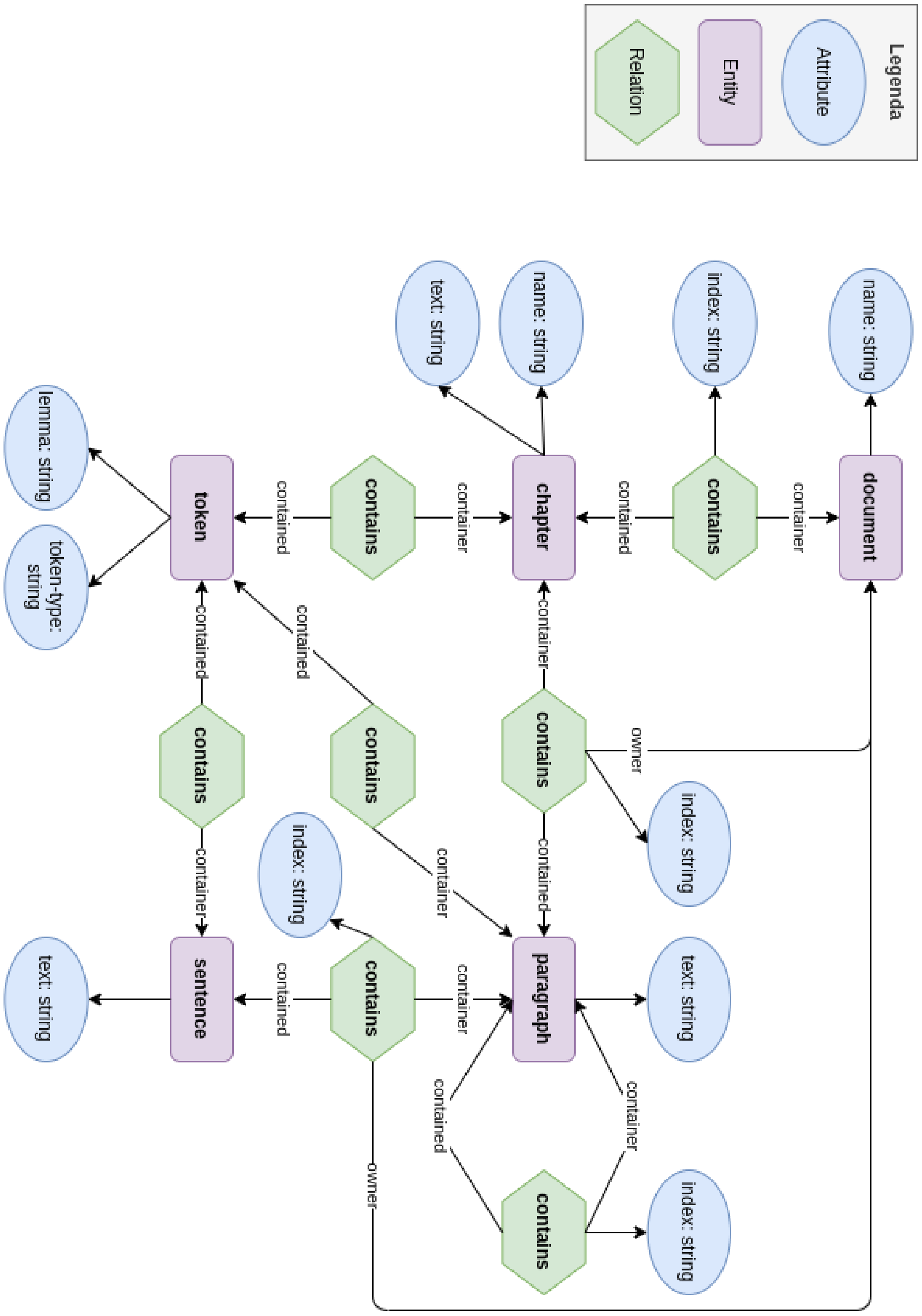


Fig. 1. Knowledge graph schema. The purple rounded rectangle represents the entities, the blue ellipses represent the attributes and the green hexagons represent the relations that link two or more entities.

often characterized by several nested levels. It's also worth mentioning the role of *owner* played by the entity *document* in relations *contains*. This has been added because if there exists more than a sentence or paragraph with the same text associated to more than one document, without the *owner* role it would be impossible to distinguish which document contains the aforementioned entity and which does not.

Finally, entity *token* owns attributes *lemma* and *token-type*. Each token corresponds to one of the results of the tokenization process and is further characterized by its lemma and its token type, i.e. whether it is a verb, a noun, an adjective, etc.

3.2 Answers selection and constraints relaxation

Given the populated graph G and a user query q, the objective of identifying the best answer is reached, in some cases, with an iterative process that involves the relaxation of the constraints involved in candidate selection.

Intuitively, user query q is analyzed to extract the set of tokens, each composed of its lemma and the type (e.g. verb, name, adjective). Then the iterative process of searching the best answer begins by selecting all and only the sentences that contains all the tokens extracted from query q. If exist any sentences that contain all the tokens, for each of them and for the query is computed the embedding using Universal Sentence Encoder Multilingual QA [22, 23], then the similarity between each pair query-sentence is computed. If the maximum similarity between question and answer embeddings is higher than a given threshold, the related answer is selected as best answer, otherwise the process is repeated by considering only the lemma of each token (relaxing the constraints over the token type). Again embedding is computed and if similarity measure doesn't exceed the threshold, all the tokens combinations without repetition are retrieved and the sentences linked to the tokens of each combination are searched. If there are results the similarity measure of each sentence embedding with respect to question embedding is calculated. If the threshold isn't high enough, last iteration involves relaxing the tokens of the aforementioned combinations by removing the token type constraint, then executing the search with each set of tokens of the combinations.

More specifically, first of all the candidate answers without constraint relaxation (relaxation case I) are searched. The user query q is used to extract and select a subset, that depends on the *token-type*, of the corresponding set of tokens:

$$T^{q,I} = \{t_1^q, \dots, t_{\bar{T}^{q,I}}^q\} : t_i^q = (t_i^{q,lemma}, t_i^{q,type}) \forall i = \{1, \dots, \bar{T}^{q,I}\} \tag{1}$$

where

$$t_i^q = (t_i^{q,lemma}, t_i^{q,type}) \tag{2}$$

is the query dependent pair composed of the token lemma and the token type obtained with respect to the user question. Then, given the set of all the sentence entities

$$S = \{s_1, \dots, s_{\bar{S}}\} \in E \tag{3}$$

given the set of the *contains* relations that must exist between a candidate answer and the tokens selected from user query

$$C^{q,I}(s) = \{(s, t_1^q), \ldots, (s, t_{\bar{T}^{q,I}}^q)\}, s \in S, t_i^q \in T^{q,I}, \forall i = \{1, \ldots, \bar{T}^{q,I}\} \tag{4}$$

given the set of all the relations of type *contains* that belong to the knowledge graph and that involve entities *token* and *sentence*

$$C = \{(s_1, t_1), (s_1, t_2) \ldots, (s_{\bar{S}}, t_{\bar{T}})\}, s_i \in S, t_j \in T, \forall i = \{1, \ldots, \bar{S}\} \forall j = \{1, \ldots, \bar{T}\} \tag{5}$$

is then retrieved the set of candidate answers for constraint relaxation case I:

$$S^I(C^{q,I}) = \{s_1, \ldots, s_{\bar{S}}\} \in S : \forall s_i \in S^I(C^{q,I}) \Rightarrow \forall c^q \in C^{q,I}(s_i) \exists \hat{c} \in C : c^q = \hat{c} \tag{6}$$

The set of candidates of Equation 6 is then served to the pre-trained deep learning model Universal Sentence Encoder Multilingual QA [22, 23] to obtain the embeddings and then to evaluate the semantic similarity between the user question embedding and the ones of each candidate sentence. The similarity score associated with each sentence is the evaluation metric used for ordering the sentences and therefore to determine the best answer to the user question. In particular, if the maximum similarity score is higher than an empirically chosen threshold, the sentences are ordered with respect to the similarity score and the first is elected as the best answer. Otherwise, if the maximum similarity score doesn't exceed the threshold, a series of analogous searches with relaxed constraints is executed. In particular, given case *I* reported in equation 1, the constraints relaxation involve the following changes:

II All the tokens selected from the user query are used but is only considered attribute *lemma*

III The combination of all the tokens selected from the user query is used considering both attributes: *lemma* and *token-type*

IV The combination of all the tokens selected from the user query is used, considering only attribute *lemma*

In particular, the set of tokens for cases *II*, *III* and *IV* is:

$$T^{q,II} = \{t_1^q, \ldots, t_{\bar{T}^{q,II}}^q\} : t_i^q = (t_i^{q,lemma}) \forall i = \{1, \ldots, \bar{T}^{q,II}\} \tag{7}$$

$$T^{q,III} = \{t_1^q, \ldots, t_{\bar{T}^{q,III}}^q\} : \exists i \in \{1, \ldots, \bar{T}^{q,III}\} : t_i^q = (t_i^{q,lemma}, t_i^{q,type}) \tag{8}$$

$$T^{q,IV} = \{t_1^q, \ldots, t_{\bar{T}^{q,IV}}^q\} : \exists i \in \{1, \ldots, \bar{T}^{q,IV}\} : t_i^q = (t_i^{q,lemma}) \tag{9}$$

In addition, for cases *III* and *IV* that involve the combinations of tokens, the set of candidate answers corresponds to the union of candidate answers obtained with respect to each combination.

Finally, given the results of the constraint relaxation case *II* and the maximum similarity score, if the value does not exceed the threshold, then the search with constraint relaxation case *III* is executed. An analogous logic happens after case *III* that continues with constraint relaxation case *IV* if the threshold is not exceeded.

4 Experimental results

The knowledge graph content corresponds to the content of three technical manuals about industrial machines produced for the business need of the partner company. The main topics concern document management, the resolution of problems encountered with machinery, contract management and restoration operations.

The content of such documents has been first extracted by means of a proprietary technology, then has been ingested into the database in accordance with the schema described in Section 3.1. The graph database chosen for such tests is Vaticle [24].

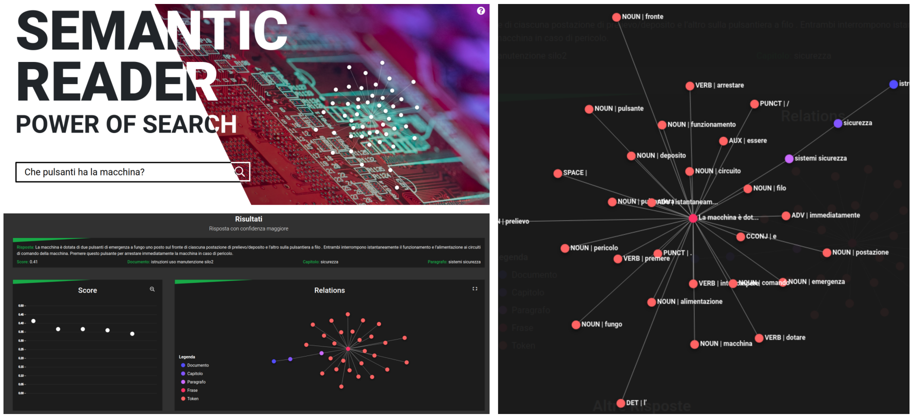
The same dataset has also been used for the development of a virtual agent structured in intents, according to the business need of the partner company. The dataset is composed of 1297 phrases, used by the conversational engine for training the model, representative of 140 intents with an average number of training phrases for each intent equal to 9.26.

The objective of the virtual agent is to answer questions about the content of the documents. For such reasons, the answers of the chatbot are similar to those that are retrieved from the framework proposed in this paper. Therefore, the evaluation process consists in a qualitative comparison of the two kinds of answers. These, in general, may differ because the answers retrieved from the framework correspond to the sentences of the document whereas the virtual assistant answers have been specifically designed to be coherent with the conversational flow.

For such reasons, before proceeding with the evaluation analysis it has been executed a preprocessing and filtering operation of the dataset. More specifically, the chatbot intents have been reduced from 140 to 24, namely those that return as an answer a string similar to some sentences of the documents, i.e. that have not been specifically created as a result of a synthesis or rework of document content. In addition, a non negligible number of intents with answers like *"yes", "no", "indeed"* or that refer to customer service emails or phone numbers have been removed. After the aforementioned reduction of the dataset, the number of training phrases used to test the performance of the proposed framework is 205.

The qualitative evaluation process consisted in using each virtual assistant training phrase as a user query and to propose it to the presented framework to compare the responses. In particular, the service that implements the framework returns the top 10 answers that, in general, may be less than 10 depending on the number of candidate sentences eligible as answers. Each sentence has been qualitatively compared with the chatbot response and if the content of a candidate answer corresponded to the chatbot response (by means of a qualitative analysis, focusing on the semantic of the sentences), then the candidate answer was considered as correct with respect to the user query. The *precision@1* and *precision@3* have been computed: the results show that the *precision@1* is 0.42 whereas the *precision*@3 is 0.89. Even though the *precision@1* is not high, we can see that the *precision@3* reaches really good results, especially considering that this framework does not require any kind of training. Moreover, each answer provided by the framework is associated with the paragraph and the title of the document that contains it. This is extremely useful in detecting the best answer, because many paragraphs of the same document or even of different documents contain similar phrases, that could match with the input question, especially if the question is not that specific (as an example, the question ”what do I have to do if the machine suddenly stop working?” does not specify which machine and given that there are two different manuals related to different machines the ranking of the answers will not take into account this information, thus the results could not be properly ranked based on the machine type but the fact that the answer is associated with the title of the manual that contains that phrase allows to easily detect the proper answer.). Thus, thanks to the fact that the framework shows all the answers detected with some informative metadata, it is easy to properly select the best answer even though this is not the first in the ranking. This is the reason why *precision@3* is a good indicator of the performance of the framework.

In order to enhance its usability, the framework is also provided with a user-friendly interface for interaction, as depicted in Figure 2, that allows to insert the query, to consult the answers and the graph associated to the answers.



Risposta: La macchina è dotata di due pulsanti di emergenza a fungo uno posto sul fronte di ciascuna postazione di prelievo/deposito e l'altro sulla pulsantiera a filo . Entrambi interrompono istantaneamente il funzionamento e l'alimentazione ai circuiti di comando della macchina. Premere questo pulsante per arrestare immediatamente la macchina in caso di pericolo.

Score: 0.41 Documento: istruzioni uso manutenzione silo2 Capitolo: sicurezza Paragrafo: sistemi sicurezza

Fig. 2. Example of the user interface to interact with the framework

5 Infrastructural considerations

To conclude, we present some considerations related to the infrastructure needed to run such framework. We deployed the services using Google Cloud Platform [25] infrastructure.

As depicted in Figure 3, we served the front-end web application using Google App Engine [26], a fully managed, serverless platform for developing and hosting web applications at scale. Vaticle [24] database and the other services have been hosted on Google Kubernetes Engine (GKE) [27], a fully managed Kubernetes service. In particular, the back-end handles the requests coming from the front-end and, whether it is for getting predictions to a user query or to get the graph associated to a sentence for dysplaying it (as shown in Figure 2), it redirects the requests to the framework service or to the graph extraction service. Indeed the framework service implements the methodology explained in Section 3.2, whereas the graph extraction service is responsible for retrieving the graph associated to a sentence that belongs to the database. The pre-trained deep learning Universal Sentence Encoder Multilingual QA [22, 23] has been deployed on AI Platform Prediction [31], a fully managed infrastructure for predictions.

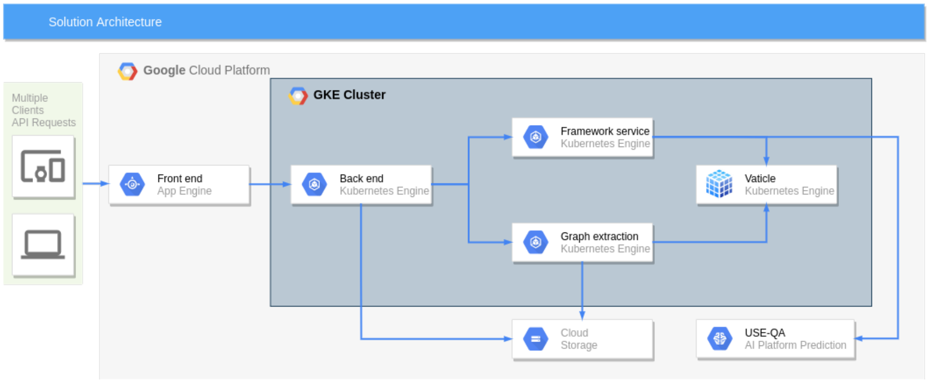


Fig. 3. Example of the user interface to interact with the framework

6 Conclusions

This study presents a framework based on knowledge graph and machine learning for semantic question answering. As mentioned, this research is driven by the business needs of a partner company, that aims at creating a question/answer system based on a corpus of documents that describe the usage and maintenance processes of machines and software they produce. After an extensive analysis of the state of the art and considering the main issues related to the creation of a chatbot system, we designed a framework that leverages on two pillars: an ontology schema able of mapping the content of arbitrarily structured documents and a methodology that uses text mining techniques and a pre-trained deep learning model for natural language processing.

One main value of the proposed work is that it doesn't require any training. Ultimately, the developed framework shows good performance when compared to the answer of a trained virtual assistant developed using the same dataset. In conclusion, the methodology is effective in answering questions related to the content of documents mapped into the knowledge graph. In general, it appears to be a performing system to be used in an industrial context, especially when the source of information is contained in documents not designed for question-answering tasks.

Some future works to enhance the performance of the proposed framework may involve investigating the use of graph machine learning techniques for node embeddings and link prediction as well as using synonyms of the tokens to augment the set of candidate sentences. Moreover, one of the most challenging tasks in the field of knowledge graphs is represented by knowledge graph acquisition, especially because this task enables all the knowledge-aware applications. Indeed, even though we have designed an ad hoc ontology to manage unstructured documents, this ontology does not take into account specific concepts contained in the documents. Thus, enriching the schema with concepts, relations and entities can enhance the performance of the proposed solution, especially when considering the use of graph machine learning techniques. For this reason, to enhance the framework, an important area of research to investigate is that of knowledge graph acquisition, especially knowledge graph completion, to be able to enrich the ontological schema with useful concepts, relations and entities to be leveraged for detecting the best answer.

References

1. Weizenbaum, J.: ELIZA—A computer program for the study of natural language communication between man and machine. In: Communications of the ACM, pp. 36–45. Association for Computing Machinery, New York (1966)
2. Colby, M.: Human-computer conversation in a cognitive therapy program. Machine conversations. Springer, Boston, MA, 1999
3. AbuShawar, B., Atwell, E.: ALICE chatbot: Trials and outputs. Computación y Sistemas **19**(4), pp. 625–632 (2015)
4. Gruber, R.T.: Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies **43**(5-6), pp. 907–928 (1995)
5. Grau, B., Horrocks, I., Motik, B., et al. OWL 2: The next step for OWL. Journal of Web **6**(4), pp. 309–322 (2008).
6. Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., Nardi, D.: The Description Logic Handbook: Theory, Implementation and Applications. 2nd edn. Cambridge university press, Cambridge (2003)
7. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., Zimmermann, A.: Knowledge graphs. ACM Computing Surveys **54**(4), pp. 1–37 (2021)
8. Fang, Y., Sun, S., Gan, Z., Pillai, R., Wang, S., Liu, J.: Hierarchical graph network for multi-hop question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8823–8838. Association for Computational Linguistics, Online (2019)
9. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15), pp. 2181–2187. AAAI Press, USA (2015)

10. Bordes, A., Usunier, N., Chopra, S., Weston, J.: LargeScale Simple Question Answering with Memory Networks. arXiv preprint, arXiv:1506.02075 (2015)
11. Dai, Z., Li, L., Xu, W.: CFO: Conditional Focused Neural Question Answering with Large-scale Knowledge Bases. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 800–810. Association for Computational Linguistics, Berlin (2016)
12. Yin, W., Yu, M., Xiang, B., Zhou, B., Schütze, H.: Simple Question Answering by Attentive Convolutional Neural Network. In: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 1746–1756. Osaka (2016)
13. Golub, D., He, X.: Character-Level Question Answering with Attention. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1598–1607. London (2016)
14. Bao, J., Duan, N., Yan, Z., Zhou, M., Zhao, T.: Constraint-based Question Answering with Knowledge Graph, In: Proceedings of COLING 2016, pp. 2503-2514. The COLING 2016 Organizing Committee, Osaka (2016)
15. Min, S., Zhong, V., Zettlemoyer, L., Hajishirzi, H: Multi-hop Reading Comprehension through Question Decomposition and Rescoring. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 6097–6109. Association for Computational Linguistics, Florence (2019)
16. Jiang, Y., Bansal, M.: Self-Assembling Modular Networks for Interpretable Multi-Hop Reasoning. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. pp. 4474–4484, Association for Computational Linguistics, Hong Kong (2019)
17. Kipf, T.N., Welling, M.: Semisupervised classification with graph convolutional networks. arXiv preprint, arXiv:1609.02907 (2016)
18. Song, L., Wang, Z., , Yu, M., Zhang, Y., Florian, R., Gildea, D.: Exploring graph-structured passage representation for multihop reading comprehension with graph neural networks. arXiv preprint, arXiv:1809.02040 (2018)
19. Dhingra, B., Jin, Q., Yang, Z., Cohen, W., Salakhutdinov, R.: Neural Models for Reasoning over Multiple Mentions Using Coreference. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 42–48. Association for Computational Linguistics, New Orleans (2018)
20. Tu, M., Huang, K., Wang, G., Huang, J., He, X., Zhou, B.: Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 9073–9080, AAAI Press, New York (2020)
21. Shao, N., Cui, Y., Liu, T., Wang, S., Hu, G.: Is graph structure necessary for multi-hop reasoning?, arXiv preprint, arXiv:2004.03096 (2020)
22. Yang, Y., Cer, D., Ahmad, A., Guo, M., Law, J., Constant, N., Kurzweil, R.: Multilingual Universal Sentence Encoder for Semantic Retrieval. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 87–94, Association for Computational Linguistics, San Francisco (2020)
23. Chidambaram, M., Yang, Y., Cer, D., Yuan, S., Sung, Y., Strope, B., Kurzweil, R.: Learning Cross-Lingual Sentence Representations via a Multi-task Dual-Encoder Model, In: Proceedings of the 4th Workshop on Representation Learning for NLP, pp. 250-259, Association for Computational Linguistics, Florence (2019)

24. Vaticle homepage, https://vaticle.com/. Last accessed: 27 Sep 2021
25. Google Cloud Platform homepage, https://cloud.google.com. Last accessed: 27 Sep 2021
26. Google Cloud AppEngine, https://cloud.google.com/appengine. Last accessed: 27 Sep 2021
27. Google Cloud Kubernetes Engine, https://cloud.google.com/kubernetes-engine. Last accessed: 27 Sep 2021
28. Google Dialogflow, https://dialogflow.cloud.google.com/. Last accessed: 27 Sep 2021
29. Amazon Lex, https://aws.amazon.com/lex/. Last accessed: 27 Sep 2021
30. Microsoft Bot Services, https://azure.microsoft.com/en-us/services/bot-services/. Last accessed: 27 Sep 2021
31. Google Cloud AI Platform Prediction documentation, https://cloud.google.com/ai-platform/prediction/docs. Last accessed: 27 Sep 2021