ParsSimpleQA: The Persian Simple Question Answering Dataset and System over Knowledge Graph

Hamed Babaei Giglou*, **Niloufar Beyranvand***, **Reza Moradi***,
Amir Mohammad Salehoof, **Saeed Bibak**
Natural Language Processing Department, Part AI Research Center, Tehran, Iran
{hamedbabaeigiglou,nilou.beyranvand,rezymo,
a.m.salehoof}@gmail.com
saeed.bibak@partdp.ai

Abstract

The simple question answering over the knowledge graph concerns answering single-relation questions by querying the facts in the knowledge graph. This task has drawn significant attention in recent years. However, there is a demand for a simple question dataset in the Persian language to study open-domain simple question answering. In this paper, we present the first Persian single-relation question answering dataset and a model that uses a knowledge graph as a source of knowledge to answer questions. We create the ParsSimpleQA dataset semi-automatically in two steps. First, we build single-relation question templates. Next, we automatically create simple questions and answers using templates, entities, and relations from Farsbase. To present the reliability of the presented dataset, we proposed a simple question-answering system that receives questions and uses deep learning and information retrieval techniques for answering questions. The experimental results presented in this paper show that the ParsSimpleQA dataset is very promising for the Persian simple question-answering task.

1 Introduction

A knowledge graph (KG) represents a network of real-world entities, with a massive semantic net that integrates various, inconsistent and heterogeneous information resources to represent knowledge about different domains (Stroh and Mathur, 2016). Some KGs contain information from multiple domains to permit machine learning applications to operate on various tasks such as question answering (QA), recommender systems, and search systems by allowing them to retrieve and reuse comprehensive answers for a given query over KG. Many studies used well-structured KGs as external resources to support open-domain QA, whereas the Knowledge Graph-based Question Answering (KGQA) system aims to answer Natural Language Questions (NLQs) automatically.

KG as a data structure underpins digital information systems, assists users in finding and retrieving resources, and serves navigation and visualization purposes. In the humanities, knowledge graphs are usually rooted in knowledge organization systems that have a centuries-old tradition and have undergone a digital transformation with the advent of web-connected data (Haslhofer et al., 2018). This work addresses the Persian language, which in general is underrepresented in NLP and also within digital humanities. Considering open-domain QA in low resources languages such as Persian, this work certainly benefits also the research on digital humanities. Generally, knowledge graphs and applications could be vehicles for formalizing and connecting findings and insights derived from the analysis of possibly large-scale corpora in the digital humanities domain. Where with help of such applications we can digitize archive collections for librarians or social science research.

In the English language, there are more valuable research works, but there is limited work that has been carried out for the Persian KGQA. The Farsbase (Asgari-Bidhendi et al., 2019) is the first Persian KGs that uses hybrid techniques to extract knowledge from various sources, such as Wikipedia, Web tables, and unstructured texts. The Farsbase was published in 2018; since then, only a few research works have been published to incorporate the Farsbase in NLP tasks due to the unavailability of datasets that refer to the KG. This also limited research on digitalizing datasets for open-domain QA tasks, since, human readers often rely on a certain amount of broad background knowledge obtained from sources outside of the text. It is perhaps not surprising then, that machine readers also require knowledge external to the text itself to perform well on QA tasks where KGs are the best source of such information. To overcome these

*These authors contributed equally to this work.

The 2nd International Workshop on Natural Language Processing for Digital Humanities (NLP4DH), pages 59–68
November 20, 2022.

concerns, (Etezadi and Shamsfard, 2020) were the first researchers who proposed the PeCoQ dataset as the first dataset for Persian complex QA over KG. Although the existing dataset is very well developed for complex QA, many general-purpose KGQA systems in other languages (e.g English) are designed to deal with complex QA by decomposing complex questions into simple questions. The Knowledge Graph Simple Question Answering (KGSQA) is a key building block for complex QA, and its performance depends on KGSQA (Yani and Krisnadhi, 2021). So putting more emphasis on KGSQA is necessary for KGQA. To facilitate research on Persian KGQA, the first simple QA dataset and system have been introduced in this work. To the best of our knowledge, this is the first step toward Persian KGSQA. In this work, we first proposed the ParsSimpleQA dataset, a simple QA dataset in Persian. Next, we proposed a simple QA framework for the Persian KGSQA.

The rest of the article is organized as follows. We first describe the problem statements and definitions in section 2. The KGQA studies are studied in section 3. Section 4 presents the ParsSimpleQA dataset. The first Persian simple QA model is presented in section 5. Section 6 describes experimental setups, and results are covered in section 7. Finally, in section 8 we conclude the article.

2 Problem Statement and Definitions

Our study aims to design the first simple QA dataset and system for the Persian language that can map a simple NLQ q to a matching query Q consisting of the subject and relation to be executed in the KG G to retrieve answers. KG G comprises triples in the form of (s, r, o) where s, r, and o denote the head entity, predicate/relation, and the tail entity, respectively. In this work, G is Farsbase KG, the first Persian multi-source KG. A simple question is a question that contains a single relation that can be queried through G to extract facts. For example, the question "*who is the director of Alone in Berlin?*" contains a director relation which can be answered using G fact that "*Vincent Perez*" is the director.

The KGSQA task is defined as (Buzaaba and Amagasa, 2021): given a KG $G = \{(s_i, r_i, o_i)\}$ that represents a set of triples, and a NLQ $q = \{w_1, w_2, \ldots, w_T\}$, where $w_i \in q$ is a sequence of words, the simple QA task is to find a triple $(s', r', o') \in G$, such that o' is the answer to the question.

3 Related Works

The KGQA has attracted a considerable body of research in recent years, and increasingly, researchers are building end-to-end neural network models for this task. A straightforward decomposition of the KGQA pipeline is entity recognition, relation prediction, entity linking, and evidence integration. This work explored relationship prediction, entity linking, and evidence integration to transform natural language into queries to extract simple factoid question answers from KGs.

3.1 Relation Prediction

Relation prediction (RP) can be considered a classification task since the simple QA assumes only a single relation is mentioned in the question. (Mohammed et al., 2018) and (Buzaaba and Amagasa, 2021) investigated various models including BiLSTM, BiGRU, CNN, and logistic regression for RP. Similarly, (Li et al., 2021) tried the BiGRU model for the RP task. Overall, (Mohammed et al., 2018) and (Li et al., 2021) concluded that BiGRU is the best model for RP in their KGSQA pipelines. Moreover, (Sidiropoulos et al., 2020) used a combination of word2vec (Mikolov et al., 2013) with LSTM to solve the RP task. However, (Lukovnikov et al., 2019) took advantage of pre-trained language models and fine-tuned BERT (Devlin et al., 2019) for the RP task.

3.2 Entity Linking

Entity linking (EL) is the task of linking a set of entities mentioned in a text to a KG. (Buzaaba and Amagasa, 2021; Lukovnikov et al., 2019; Mohammed et al., 2018) used an inverted index to retrieve entity mentions from KG and then ranked mentions using *fuzzywuzzy*, a string-based similarity method. Additionally, (Fu et al., 2020) proposed a low-resource cross-lingual EL (XEL) that supports 25 languages, including Persian. They proposed a simple yet effective zero-shot XEL system, *QuEL*, that utilizes the search engine's query logs. (Asgari-Bidhendi et al., 2020) proposed the ParsEL-Social, the first Persian EL dataset which is constructed from social media contents. Next, they utilize context-dependent and context-independent features to propose the first EL model called ParsEL 1.0 in Persian using Farsbase KG. Moreover, in (Asgari-Bidhendi et al.) they proposed the ParsEL 1.1, which is an improved version of the previously proposed EL model, by adding

graph-based features. In the latest work in EL at Persian, (Asgari-Bidhendi et al., 2021) introduced an unsupervised language-independent entity disambiguation (ULIED), which uses disambiguate and linked named entities. The proposed entity disambiguation uses different similarity measurements for candidate entity weighting and aggregation. The ULIED showed promising results in languages other than English, such as Persian.

3.3 Evidence Integration

Evidence integration (EI) is the final task to integrate evidence to reach a single (entity, relation) prediction. (Sidiropoulos et al., 2020) used a heuristic based on popularity, that chooses entities that appear among the facts in KG either as a subject or as an object. (Mohammed et al., 2018) used the top m entities and r relations to generate $m * r$ tuples where scores are the product of their component scores. After pruning meaningless combinations, they used graph-based features such as popularity nodes to select the final answers. (Lukovnikov et al., 2019) ranked the given entity-relation pairs by considering string-based similarity for entity and higher prediction probability using the BERT language model (Devlin et al., 2019) for relation. Next, they took top-scored pairs, which can easily generate a query to retrieve the answer from the KG.

3.4 Datasets

In (Bordes et al., 2015), the SimpleQuestions benchmark was first introduced, and this benchmark consists of 108,442 simple questions annotated with the correct Freebase knowledge base fact, where facts have exactly one relation. This allowed a significant improvement in English KGSQA research, where (Petrochuk and Zettlemoyer, 2018) reported an empirical analysis and concluded the SimpleQuestions dataset is nearly solved. However, researchers continued to analyze this benchmark further. Since neural network models require appropriate data for end-to-end training, the demand for a dataset in a new language is increased. Due to this concern, for the Persian language, the first KGQA dataset, which supports the complex QA, was introduced by (Etezadi and Shamsfard, 2020).

4 ParsSimpleQA Dataset

To construct a KGQA system in the practical environment, we should solve the following tasks: entity detection, EL, RP, and EI where each task can be addressed in a supervised or unsupervised learning fashion using an appropriate dataset. The proposed dataset is suited for training and evaluation of models for suitable and optimal queries to extract answers from KG G. For automated QA dataset generation, we had to deal with two challenges: creating high-quality templates and creating logical/correct QAs which the ParsSimpleQA datasets consider these challenges properly. Figure 1 depicts the process of creating the dataset. In the following, we discussed the details for ParsSimpleQA creation.

1. **Relationship Selection:** Farsbase consists of many relationships and considering all of them for dataset creation is computationally costly. For this reason, we filtered almost 100 most common relationships from G. Next, we analyzed chosen relationships based on two criteria: (1) whether they are meaningful in Persian or not, and (2) whether they meet Persian NLP dataset creation needs. In the final, we obtained 35 relationships for dataset creation.

2. **Generating Templates:** In this step, we used three annotators who were familiar with KGQA techniques for building templates and asked them to build templates for each relationship collaboratively. Templates are questions that only require an entity to be complete. By using specified entities, we were able to generate QAs. For example, for the template *Who is the author of* $< ENT >$*?*, we may consider the following samples: *Who is the author of Harry Potter?* and *Who is the author of Blindness?* where *Harry Potter* and *Blindness* are entities.

3. **Template Evaluations and Cleaning:** For creating high-quality templates, we asked three NLP researchers with at least a master's degree to check templates and find the inappropriate templates for relationships. During this process, we kept evaluations blind for researchers. To obtain high-quality templates, we applied hard voting for researcher decisions about accepting or rejecting templates. Next, the rejected template was removed from the final list to obtain 149 templates. Then, we asked annotators to analyze templates to see whether there are relationships that share the same information or not. Thereby, we

Figure 1: The ParsSimpleQA dataset generation procedure

combined multiple relationships to obtain 32 relationships in total. On average, 4.65 templates per relationship were constructed for the ParsSimpleQA dataset.

4. **Entity Selection:** Two ranges of data sources have been used to build the Farsbase knowledge base. The first source involves extracting rule-based information from dumped *Wikipedia* data, and the other source includes other knowledge bases such as *Yago* and *Wikidata*. Rule-based information extracted from the *Wikipedia* dumps naturally contains a lot of noise, including meaningless words, as well as information from other knowledge bases, including entities from different languages such as Chinese and Korean. None of them are appropriate for this task. Therefore, we considered the following three conditions in the entity selection for dataset creation: 1) The entity must have a URI field (a unique ID in KG), 2) Ignoring entities from *dbpedia*, *yago-knowledge.org*, *ecowlim.tfri.gov.tw*, *data.linkedmbd.org*, *data.linkedopendata.it*, and 3) Considering only entities with Persian letters.

5. **Automatic QA Generation:** All selected KG entities are inserted into template slots with condition that they have that template relation. So, we can say that template markings such as author/actor allow for natural sentences to be generated. We used all entities as subject entities and relationships for each template to run a Cypher query (Francis et al., 2018) – a Neo4j query language – over KG to generate object entities. Subsequently, we constructed (s, r, o) triples. Next, we constructed the samples using subject s and relationship templates where we know that answer is an object entity o. The constructed samples contain information such as relation type, template, question entity (subject), question (combination of template and question entity), question entity URI (identified entity in KG), answer entity (object), and answer entities URIs. At the end, the $n_{templates} * n_{relationships} * n_{entities}$ samples were obtained (n is number of items).

6. **Post-Processing:** The goal of this step is to create a final dataset for KGSQA by performing grouping, processing, and doing train-validation-test set splits. The following steps were performed for post-processing where the train/validation/test split with respect to relationship type and entities to avoid leakage.

   (a) Grouping data based on relation types. So, we clustered the samples for relation templates for each relation group.

   (b) Aggregation of groups based on question entity URIs. For each entity, we may have multiple samples with different answers (multiple answers for a question). In this step, we list the answers to each question.

   (c) Combining similar relations that annotators identified.

   (d) Train, validation, and test splits. We performed a train-validation-test split on question entity URIs and templates separately for each relation group. Next, the final train-validation-test sets for relation types were generated using divided questions entity URI and templates. This process was done by looking for pair entities and templates in the divided lists. After obtaining train-validation-test sets individually for relations, we merged them to form the final train, validation, and test sets.

We employed 60% train, 20% validation, and 20% test split rates while creating datasets. Table 1 presents the stats of the proposed datasets. Overall 36,122 samples for 32 relationship types using 149 templates and 16,772 unique entities were created.

5 Method

Our methodology for the Persian KGSQA uses the ParsSimpleQA dataset and comprises the following components: relation prediction (RP), entity linking (EL), and evidence integration (EI). Figure 2 depicts a proposed framework. In the proposed

Sets	# of samples	# of templates	# of question entities
Train	29,360	78	10,945
Validation	2,261	34	1,898
Test	4,501	37	3,929
Overall	36,122	149	16,772

Table 1: ParsSimpleQA dataset stats

KGSQA, a deep learning model is presented to identify the relation type of the questions. Next, the EL module uses a hybrid method for candidate entity generation and rankings for URI identification of entities from KG. Finally, the EI module uses question relation type, obtained URIs from entity linker, and KG to generate the answers using a Cypher query.

5.1 Relation Prediction

This module's goal is to identify the question relation type r' using a supervised approach. In recent years, transformer-based language models, such as BERT, have achieved state-of-the-art performance on many tasks (Min et al., 2021). Transformers encode context bidirectionally and require minimal architecture changes for a wide range of NLP tasks. Since the nature of KGSQA is open-domain specific, and to solve RP, a general-purpose representation such as BERT is a logical choice due to the advantages of contextualized representations. For relation type identification, we used ParsBERT (Farahani et al., 2021) a BERT variant for the Persian language as a pre-trained language model. To modify the output of ParsBERT for the RP task, we added an extra layer for fine-tuning. In ParsSimpleQA, samples appear to be imbalanced, which affects the RP fine-tuning since during the training, weights flow toward the majority class. To overcome this issue, we applied the focal loss function (Lin et al., 2017) which is an improved version of cross-entropy loss by focusing on hard learning of misclassified examples.

5.2 Entity Linking

The EL aims to link a set of entities mentioned in a text to a KG. EL consists of candidate generation (CG) and candidate ranking (CR). For CG, we applied a keyword-based search engine to retrieve the entities, where it uses a string-based BM25 scoring function as the similarity scoring function. For CR, most of the approaches tried to use node context information to perform CR. However, the Farsbase (Asgari-Bidhendi et al., 2019) contains only 14% of abstract context information which can be encoded in the form of contextualized representation for EL using transformers. However, in this way, we may lose nodes that do not contain context information but are the key object nodes for the question to form the answer triples. To overcome this limitation, we proposed a graph-based features ranking mechanism as a second ranking function that considers node connections instead of context information. Assessing EL with graph-based and context-based rankings boosts the performance of the EL rankings, regardless of the question itself. We acknowledge that the question itself must be taken into consideration since it plays an essential role in candidate entity generation. So, CR contains information about string-based features. Finally, a string-based ranking method that has been used for CG is incorporated with graph-based and context-based rankings as a hybrid CR for EL. The proposed hybrid CR takes advantage of each other for the final appropriate candidate ranking. The final CR has been calculated in the following manner:

String-based ranking: Entities in KG are indexed into the search engine using an inverted index data structure. Next, with BM25 scoring function (Robertson and Zaragoza, 2009), relative entities are retrieved as a CG for CR. BM25 is based on the bag-of-words approach. The score of a subject s' given a query q which contains the words $w_1, w_2, ..., w_T$ is given by:

$$R_{bm25}(s', q) = \sum_{i=1}^{n} IDF(w_i) \cdot QT(s', w_i)$$

$$QT(s', w_i) = \frac{f(w_i, s') \cdot (k_1 + 1)}{f(w_i, s') + k_1 \cdot (1 - b + b \cdot \frac{|s'|}{avgdl})}$$

where $f(w_i, s')$ is q's term frequency in the s', $|s'|$ is the length of the s' in words, and $avgdl$ is the average length in the text collection from KG. k_1 and b are free parameters.

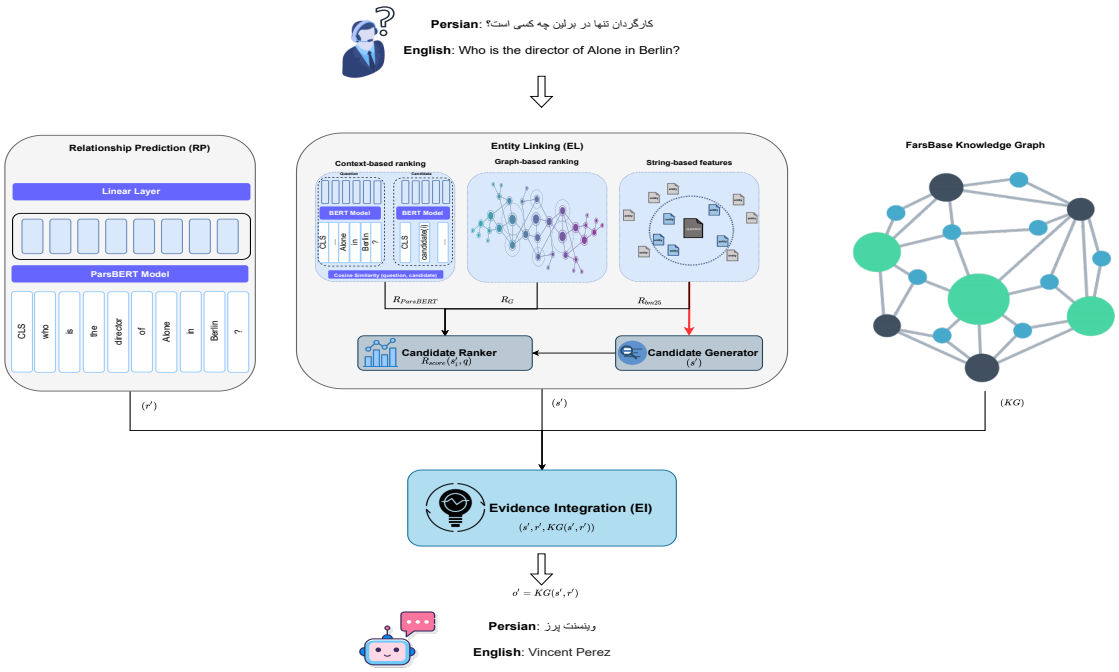

Figure 2: The workflow of SimpleQA model

Context-based ranking: Since the question conveys a better level of meaning, first, the embeddings of the question q and candidates s' are obtained using the ParsBERT language model. Next, we calculated cosine similarity between $V_{s'}$ and V_q vectors to obtain $R_{ParsBERT}(V_{s'}, V_q)$, where $V_{s'}$ and V_q are embeddings of s' and V_q, respectively.

$$R_{ParsBERT}(s', q) = Similarity(V_{s'}, V_q)$$

Graph-based ranking: Assuming that a node with more connections is also more popular, the degree of each node is used as its popularity score to calculate $R_G(s')$.

$$R_G(s', q) = degree(s')$$

Finally, for each candidate s'_i where $s'_i \in s'$, we calculate the probability $P(R, k, q)$ for ranking outputs $R \in \{R_{bm25}, R_{ParsBERT}, R_G\}$ using the following formula:

$$P(R, s'_i, q) = \frac{R(s'_i, q)}{\sum_{j=1}^{n_{s'}} R(s'_j, q)}$$

where $n_{s'}$ is the number of candidates s'_i. In the final, the average of ranked probabilities was calculated to obtain the final $R_{score}(s'_i, q)$ ranking score for candidates, where we pick the top_e ELs with the highest probability score.

$$R_{score}(s'_i, q) = \frac{P_{R_{bm25}} + P_{R_{ParsBERT}} + P_{R_G}}{3}$$

5.3 Evidence Integration

In evidence integration (EI), once we have a list of candidate entities, each candidate node is used as a starting point to reach candidate answers o'. We limit our search to a single hop and retrieve all nodes that are reachable from the candidate node s' where the relation path is consistent with the predicted relation r'. Due to the high performance of the EL model, we used EL final scoring R_{score} as a sorting function for answers o'. During EI, since EL and RP models are independent, the triples may appear to be meaningless because EL can produce s' that does not have the r' that the RP model predicted. In this case, the (s', r') pairs are ignored from EI.

6 Experimental Setups

Metrics: Commonly used performance measures include accuracy, precision, recall, and F-measure (Diefenbach et al., 2018). Since For each question there is an expected set of correct answers, these are called the gold standard answers. we can define the metrics as followings (n is a number of samples):

$$accuracy(q) = \frac{n_{correct\ predictions}}{n_{dataset\ samples}}$$

$$precision(q) = \frac{n_{correct\ system\ answers\ for\ q}}{n_{system\ answers\ for\ q}}$$

$$recall(q) = \frac{n_{correct\ system\ answers\ for\ q}}{n_{gold\ standard\ answers\ for\ q}}$$

$$F - measure = 2 * \frac{precision(q) * recall(q)}{precision(q) + recall(q)}$$

Where precision indicates how many of the answers are correct, recall indicates how many of the returned answers are in the gold standard. F-measure is the weighted average between the (macro) precision and recall.

Training Setups: We have used the ParsSimpleQA dataset for training and evaluating the first Persian KGSQA model. We imported Farsbase KG into the Neo4j database. Next, after tuning several hyperparameter models, the final model was trained. Training and hyperparameter tuning was done using the NVIDIA Tesla V100 GPU machine.

7 Results

7.1 Results and Hyperparameter Tunings

Regarding imbalanced nature of the data, results for RP, EL, and EI tasks are demonstrated in Tables 2, 3, and 4, respectively. In the following, we presented a more detailed analysis.

Relation Prediction: For comparison of the proposed method for RP, we implemented the BiGRU baseline, which was proposed in (Mohammed et al., 2018). We did the hyperparameter tuning for BiGRU and ParsBERT models using the validation set. The optimal values for the BiGRU are $\alpha = 1e-4$, $batch - size = 16$, $optimizer = adam$, and $loss = cross - entropy$. The optimal values for ParsBERT are $\alpha = 1e-5$, $batch - size = 4$, and $optimizer = adam$. For loss function in fine-tuning ParsBERT, we examined cross-entropy, Inverse of Square Root of Number of Samples (ISNS) (Mahajan et al., 2018), and FL loss functions which experimental results showed the superiority of the focal loss function for RP task. Table 2 presents the experimental results over validation and final results over the test set. The experiment with three loss functions showed that hard learning of misclassified classes using the focal loss function is an appropriate choice for this task.

Entity Linking: The results for the proposed EL method over the validation and test set are presented in Table 3, and the optimal top_e parameter is the number of top candidates for EL prediction and it is set into 1.

Evidence Integration: After entering RP, EL, and KG into EI, the final hyperparameters needed to be tuned for the final system. The final parameters are 1) top_a, the number of top answers in the final system, 2) top_r, the number of top relations, and 3) top_e, the number of top ELs. We run the two-step tuning by considering F1-score as a judgment metric for optimal parameters. In the first step we tried to find optimal values for top_r and top_e since both top_r and top_e are effects the final system response. As a result, according to the Figure 3, the optimal values for (top_r, top_e) pair are (2, 5). Next, we used optimal top_r and top_e to tune the top_a. Figure 4 shows the tuning results for top_a, where the results after $top_a = 5$ remains unchanged. In the end, we used optimal values for the final system evaluation, where the results are presented in Table 4 for the test set.

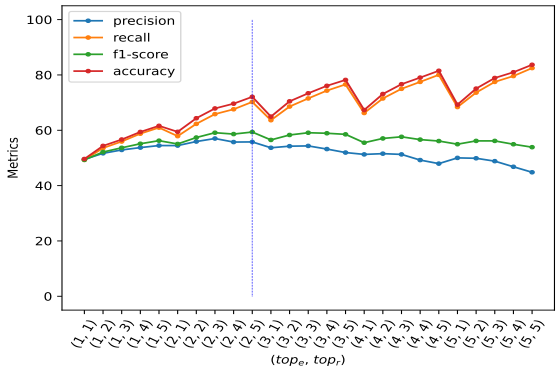

Figure 3: Hyperparameter tuning for (top_e, top_r) set using validation set

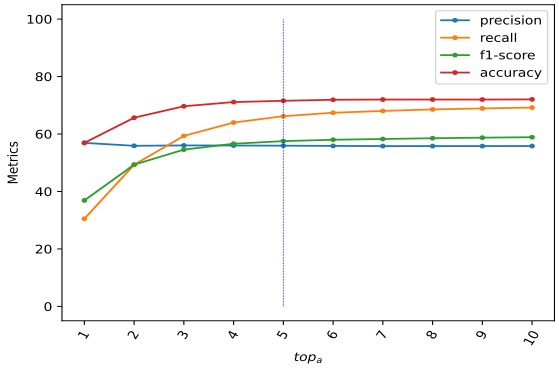

Figure 4: Hyperparameter tuning for top_a using validation set

7.2 Analysis

All models under comparison have all their components fixed, except RP. Therefore, any improvement observed is due to RP.

Effect of RP: RP uses the ParsBERT model, and in the experimental result, ParsBERT recognizes the relations more accurately than the baseline. However, we observed two concerns:

Model	Loss Function	Precision	Recall	F1-score	Accuracy	Dataset
BiGRU	cross entropy	34.33	36.36	33.61	60.15	Validation
ParsBERT	cross entropy	60.8	64.56	60.87	77.62	Validation
ParsBERT	ISNS	62.97	61.79	55.85	75.36	Validation
ParsBERT	FL	60.18	65.88	61.14	77.39	Validation
ParsBERT	FL	53.99	59.22	54.25	54.61	Test

Table 2: Relation prediction results

top_e Entities	Precision	Recall	F1-score	Accuracy	Dataset
1	68.02	67.18	67.38	68.02	Validation
1	68.23	66.77	67.18	68.23	Test

Table 3: Entity linking results

top_e	top_r	top_a	Precision	Recall	F1-score	Accuracy	Dataset
2	5	5	55.94	66.20	57.52	71.56	Validation
2	5	5	48.56	63.93	52.27	68.30	Test

Table 4: Final system results

- *Imbalanced Relations:* The weighting technique solved this issue somehow, but the issue isn't completely solved. Still, we can see the model does not work accurately for some relations.
- *Relations Similarity:* This happened due to the similarity in templates, and asking annotators to find similar relations wasn't enough since the experimental analysis showed that some templates might have common words in templates that lead to misclassification.

However, regarding both weaknesses, the model performs promisingly regarding the baseline model and positively helps the rest of the pipeline. This shows that producing this kind of data could help better identification of relation types.

KGSQA Performance Analysis:

- According to Table 3, the proposed EL strongly generalized well on unseen questions.
- Figures 3 and 4 for hyperparameter tunning showed the effect of low performance on RP affect on the other tasks such as ELs in the system. This is obvious in (top_e, top_r) pairs, e.g $(4, 1)$, $(3, 1)$. But applying $top_r > 1$ solved this issue positively.
- In terms of accuracy, the final system achieved an accuracy of 68.30% on the test set. This means the proposed KGSQA produced answer sets (top_a answers for each sample) in the test set which 68.30% (2671 out of 3929 samples) of samples contained answers (gold answers exist in answer sets). Regarding this, the presented model disables producing a correct answer (in the answer set) for 1258 samples in the test set.
- We obtained the optimal value of $top_a = 5$ for the number of output answers. Because most of the questions contained only one or two answers, the number of correct answers for each question may be ended up with 1 or 2 true answers, which results in low precision. One way of solving this issue is decreasing top_a or considering a separate ranker from EL for top_a answer selections.
- The high recall alarms the number of outputs that are well intersected with gold answers. So, answers are mostly among the top_a answers.

8 Conclusion

This paper introduced ParsSimpleQA, the first KGSQA dataset for the Persian language that contains questions with corresponding entities, entity

links, relation types, and answers. The machine-generated questions using human-generated templates are preprocessed and divided into train, validation, and test sets for training and analyzing machine learning techniques. Next, we introduced the first Persian KGSQA to perform EL, RP, and EI. Our experimental results on the ParsSimpleQA dataset show that our proposed framework is robust and generalized well. This framework will play a baseline model that opens many works in Persian KGSQA. Moreover, our generated ParsSimpleQA dataset is available to the research community at the GitHub[1] repository.

Acknowledgements

The authors would like to give big thanks to Part AI Research Center[2] (the biggest AI company in Iran) for supporting and funding this research for digital humanities.

References

Majid Asgari-Bidhendi, Farzane Fakhrian, and Behrouz Minaei-Bidgoli. A graph-based approach for persian entity linking.

Majid Asgari-Bidhendi, Farzane Fakhrian, and Behrouz Minaei-Bidgoli. 2020. Parsel 1.0: Unsupervised entity linking in persian social media texts. In *2020 11th International Conference on Information and Knowledge Technology (IKT)*, pages 148–153. IEEE.

Majid Asgari-Bidhendi, Ali Hadian, and Behrouz Minaei-Bidgoli. 2019. Farsbase: The persian knowledge graph. *Semantic Web*, 10(6):1169–1196.

Majid Asgari-Bidhendi, Behrooz Janfada, Amir Havangi, Sayyed Ali Hossayni, and Behrouz Minaei-Bidgoli. 2021. An unsupervised language-independent entity disambiguation method and its evaluation on the english and persian languages. *arXiv preprint arXiv:2102.00395*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Happy Buzaaba and Toshiyuki Amagasa. 2021. Question answering over knowledge base: A scheme for integrating subject and the identified relation to answer simple questions. *SN Comput. Sci.*, 2:25.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Dennis Diefenbach, Vanessa Lopez, Kamal Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowledge and Information systems*, 55(3):529–569.

Romina Etezadi and Mehrnoush Shamsfard. 2020. PeCoQ: A dataset for persian complex question answering over knowledge graph. In *2020 11th International Conference on Information and Knowledge Technology (IKT)*. IEEE.

Mehrdad Farahani, Mohammad Gharachorloo, Marzieh Farahani, and Mohammad Manthouri. 2021. Parsbert: Transformer-based model for persian language understanding. *Neural Processing Letters*, 53(6):3831–3847.

Nadime Francis, Alastair Green, Paolo Guagliardo, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Stefan Plantikow, Mats Rydberg, Petra Selmer, and Andrés Taylor. 2018. Cypher: An evolving query language for property graphs. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, page 1433–1445, New York, NY, USA. Association for Computing Machinery.

Xingyu Fu, Weijia Shi, Xiaodong Yu, Zian Zhao, and Dan Roth. 2020. Design Challenges in Low-resource Cross-lingual Entity Linking. In *Proc. of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Bernhard Haslhofer, Antoine Isaac, and Rainer Simon. 2018. *Knowledge Graphs in the Libraries and Digital Humanities Domain*, pages 1–8. Springer International Publishing, Cham.

Lin Li, Mengjing Zhang, Zhaohui Chao, and Jianwen Xiang. 2021. Using context information to enhance simple question answering. *World Wide Web*, 24(1):249–277.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. pages 2980–2988.

Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2019. Pretrained transformers for simple question answering over knowledge graphs. In *International Semantic Web Conference*, pages 470–486. Springer.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[1]https://github.com/partdpai/ParsSimpleQA
[2]https://www.partdp.ai

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heinz, and Dan Roth. 2021. Recent advances in natural language processing via large pre-trained language models: A survey.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 291–296.

Michael Petrochuk and Luke Zettlemoyer. 2018. Simplequestions nearly solved: A new upperbound and baseline approach. *arXiv preprint arXiv:1804.08798*.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Georgios Sidiropoulos, Nikos Voskarides, and Evangelos Kanoulas. 2020. Knowledge graph simple question answering for unseen domains. In *Automated Knowledge Base Construction*.

Eylon Stroh and Priyank Mathur. 2016. Question answering using deep learning. *SCPD*, pages 1–2.

Mohammad Yani and Adila Alfa Krisnadhi. 2021. Challenges, techniques, and trends of simple knowledge graph question answering: A survey. *Information*, 12(7):271.