Fine and Ultra-Fine Entity Type Embeddings for Question Answering

Sai Vallurupalli, Jennifer Sleeman, and Tim Finin

University of Maryland at Baltimore County, Baltimore, MD 21250 USA
kolli,jsleem1,finin@umbc.edu

Abstract. We describe our system for the SeMantic AnsweR (SMART) Type prediction task 2020 for both the DBpedia and Wikidata Question Answer Type datasets. The SMART task challenge introduced fine-grained and ultra-fine entity typing to question answering by releasing two datasets for question classification using DBpedia and Wikidata classes. We propose a flexible approach for both entity types using paragraph vectors and word embeddings to obtain high quality contextualized question representations. We augment the document similarity provided by paragraph vectors with semantic modeling and sentence alignment using word embeddings. For the answer category prediction, we achieved a maximum accuracy score of 85% for Wikidata and 88.5% for DBpedia. For the answer types prediction, we achieved a maximum MRR of 40% for Wikidata and a maximum nDCG@5 of 54.8% for DBpedia datasets.

Keywords: Word Embedding · Document Embedding · Paragraph Vectors · Fine-Grained Entity Typing · Ultra-Fine Entity Typing.

1 Introduction

To further research in the area of question answering using entity types, the 19th International Semantic Web Conference (ISWC) 2020 has put forward the Semantic Answer Type (SMART) prediction task challenge [18]. Two new datasets were released with the goal of classifying the questions into hundreds of fine and thousands of ultra-fine types using DBpedia and Wikidata ontologies. The two SMART challenge datasets include a total of 44,786 questions which are more varied than the short and single sentence factoid questions from the UIUC and the TREC QA datasets [14, 25]. The task goal is a dual classification where each question is assigned a single answer category, and an unknown number of answer types of the expected answer. While this task is considered a short-text classification, what makes the classification challenging is a few unique characteristics of the datasets which contribute to data sparsity.

Word Frequencies and Distribution. The questions are composed mainly of simple high frequency words, some of which are considered to be stop words such as: *the*, *and*, *of*, etc. And, these simple high frequency words are distributed uniformly among all questions. For example, all three question categories contain 90% of the top 100 words and 80% of the top 500 frequent words.

Answer Type Label Distribution. Answer type labels are not uniformly distributed in the training set, with two-thirds of the labels having less than five training samples. These long tailed distributions for both the datasets are shown in Figure 1. For two-thirds of the data, when no external resources are available for training, the classification task presents similar challenges to that of a low resource setting, needing data generalizations and augmentation.

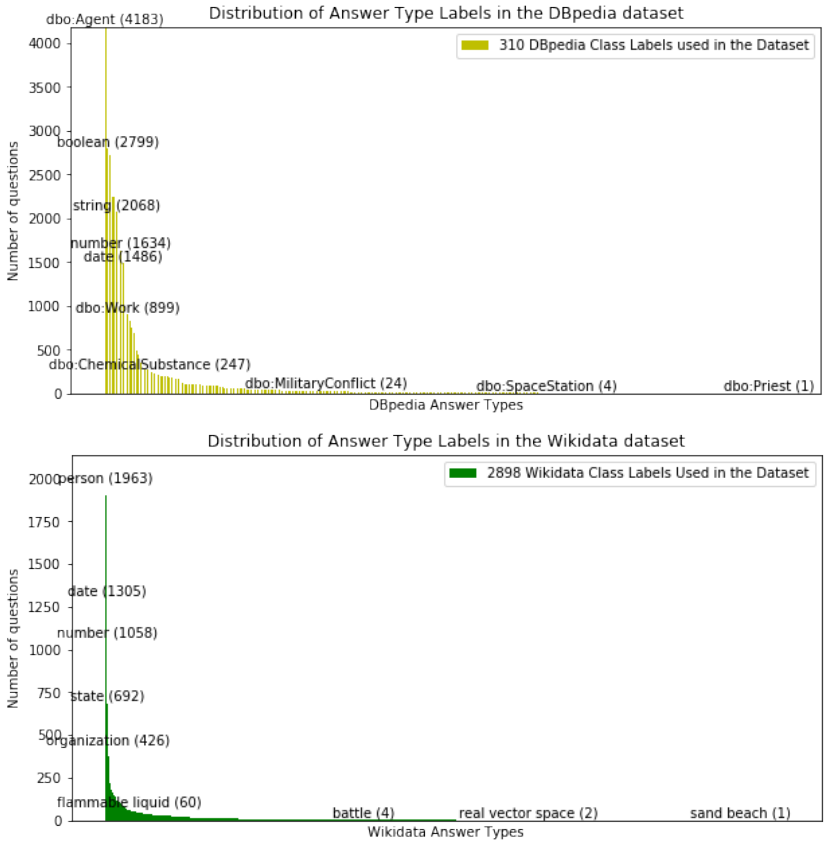

Fig. 1: Answer Type Label distribution - DBpedia Dataset (top) and Wikidata Dataset (bottom)

Question Structure and Label Assignment. Sentence grammar and structure are varied, resulting in noisy constituency and dependency parses. The gold

labels in the DBpedia dataset do not always include all the labels in the hierarchy. In addition, a question can be labeled with multiple answer types, with the number of labels being unknown apriori. Furthermore, the challenge evaluation is dependent on the correct number and ordering of these labels.

In this paper, we describe, a novel and flexible approach, that builds upon the seminal work of distributed representations of paragraphs, sentences, phrases and words [19, 11]. The methodology we use in this approach includes training a Doc2vec [11] and Word2vec [19] model on the questions in the training datasets. These models enable the extraction of novel syntactic, grammatical and distributed representations from the question semantics. Using these semantic representations, nested filtering is applied to the top-N documents inferred by the trained Doc2vec model, with the overall goal of improving the accuracy of the dual classification. The completely unsupervised training of the models, and the semantic representations, leverage the unique aspects of the data to deal with data sparsity. The nested filtering, and our novel method of computing subject similarity proved to be highly effective in handling sparse data. Our method generalizes well, such that it can be applied to both datasets. It is flexible and can be used for both DBpedia and Wikidata class labels. It offers an additional benefit, in that, it can incorporate external data for further refinement.

2 Background

Early question classification systems were rule based; a question was matched to a category, based on hand crafted rules. Rule based systems are resource intensive, and do not scale well. They break when a single question is reformulated using different words and sentence structures [14]. This led to supervised machine learning methods using feature sets and statistical techniques. SVM classifiers were used to classify semantic features such as bag-of-words, ngrams, wh-words, head words, and hypernyms extracted from WordNet [16, 7, 21]. Supervised learning requires expert knowledge to extract features and create curated datasets. Unsupervised learning addresses this with neural models, which automatically extract useful features, when trained on large amounts of data.

Distributed Representation of words and phrases (word embeddings) [19] and distributed representations of documents and paragraphs (Paragraph Vectors) [11] are unsupervised learning models called Word2vec and Doc2vec respectively. These shallow neural models are trained to represent words or documents as n-dimensional vectors (aka embeddings). The embeddings produced by these models are effective in a wide variety of applications [2, 6, 27, 8, 3]. In this section, we briefly provide the theoretical basis for these models.

2.1 Word Embeddings.

Word embeddings are used as a neural network classifier trained to learn the surrounding words within a fixed window on either side of a word, aka the word's context. There are two models based on whether the given word predicts

the context as in the skipgram model, or the context predicts the given word as in the continuous bag of words (CBOW) model. In the latter model, the context is obtained by averaging the vectors of all the words in the context. While this averaging is thought to potentially derive better representations, in practice, the skipgram model performs better on most word similarity tasks [13, 15, 26]. We use the skipgram model which works as follows:

Every word is assigned a vector of dimension D. For a training corpus consisting of C contexts, given a word w, and its context c, the model objective is to maximize the conditional log probability for the corpus:

$$argmax_\theta \sum_{(w,c)\in C} logp(c|w,\theta) \tag{1}$$

If we denote the vector for w as v_w, and the vector for the context as v_c, the conditional probability is represented as:

$$p(c|w,\theta) = \frac{e^{v_c.v_w}}{\sum_{c'\in C} e^{v_{c'}.v_w}} \tag{2}$$

Instead of training on all contexts C, negative sampling is used to train on a subset sampled from C. While this changes the objective function (similar to eq. 3), essentially, after training on a number of words and contexts, the model increases the quantity $v_c.v_w$ for words that share contexts, and decreases it for words that do not share contexts. Any two words sharing similar contexts, and any two contexts sharing many words will end up with similar vectors, resulting in a high cosine similarity between the vectors.

2.2 Paragraph Vectors.

Paragraph vectors are used as a neural classifier for question sentences to learn words in a question. Of the two paragraph vector models, distributed memory (DM) and distributed bag of words (DBOW), we used the DBOW model. It was shown to perform better in semantic similarity tasks [10] and does not suffer the curse of dimensionality as much as the DM model. This model is similar to the skipgram Word2vec model described above, where a document, instead of a word, is used to predict the context. The model objective is to maximize the conditional log probability $p(c|d,\theta)$ for a corpus C, where v_c is the context vector and v_d is the document vector:

$$argmax_\theta \sum_{(d,c)\in C} log\frac{1}{1+e^{v_c.v_d}} \tag{3}$$

3 Related Work

Most of the related work identify entity types for the given entity mentions to improve a downstream task of question answering. We highlight a few of these

methods that offer incremental improvements and we describe how our method is different from these previous approaches. Sun et al. [24] used Freebase entity types to rank answer candidates for a given question. However, they used a search engine to retrieve sentences related to a question, for which they applied entity linking to extract entities. They used the answer candidates and Freebase for entity typing. Since our method uses low-dimensional embedding models, we are able to achieve a richer understanding of the context of the questions. Dong et al. [4] used 22 different types from DBpedia to classify entity mentions in questions with two methods. They combined the context representations obtained from a multilayer perceptron model, and the vector representations of entity mentions obtained using a recurrent neural network, to predict type information. Yavuz et al. [29] built upon the work of [4] using type information to improve semantic parsing for question answering. The semantic parsing component maps the natural form of a question to an abstract semantic representation of the question by replacing entity mentions with type information. They train a bidirectional LSTM on the abstract forms of questions to infer answer types. In a recent work, Choi et al. [1] proposed a bidirectional LSTM for predicting natural language phrases describing the entity mentions in a given sentence. We believe we can achieve sufficient answer typing with our approach which includes both a word and a document embedding. Not only does our method offer contextual understanding similar to previous work, our unique combination of the two embedding models offers more flexibility, and is able to work on more varied types of sentences.

4 Methodology

The Distributional Hypothesis [5] proposes the grouping of entities that share similar distributional properties. Such entities include "representations of how words are used in natural context" [9]. We extend this idea of similar representations to the distributional representation of questions. i.e., similar question sentence representations share similar labels. For example, questions of the form "Who is the president of United States?", and "Who is the president of Canada?" share a similar sentential representation, and similar distributional representations. We posit that such similar questions tend to share category and type labels.

Our method is based on a unique collaborative combination of both Word2vec and Doc2vec models [11, 19], designed to achieve better contextual understanding of questions. Our trained Doc2vec[1] model builds distributional representations of questions. Our trained Word2vec model[2] helps with contextualizing the expected answer category and type. We train both models using the python gensim package. Training in both models is unsupervised. Both models are trained on the training questions without the gold labels. Gold labels are used later, during inference. The Doc2vec model is trained with each question treated as

[1] doc2vec parameters: $vector\_size = 300, hs = 0, negative = 5, epochs = 50$

[2] word2vec parameters: $vector\_size = 300, window = 10, negative = 5, iter = 50$

a document. The trained model is used to map a given test question to the document embedding space to find similar training questions. Effective training of neural models requires a large amount of training data. With limited data, semantically similar documents tend to be noisy. High variability in sentence structures, and sizes adds to the noise. To help filter the noise, the top N similar training questions in the embedding space are filtered using syntactic and grammatical modelling applied with the Word2vec model.

We select syntactic and semantic feature words which aid in aligning questions of similar category and types. We group these words into three types: *Q-word*, *Action words* and *Anchor words*. Using word vectors which we obtain from training a Word2vec model with the question sentences, we assign a Q-word and a similarity vector to every question. For a test question, we filter the top N similarity matches obtained from the Doc2vec model into two groups. The first group consists of questions with the same Q-word as the test question. The second group consists of questions where the subject vector of the question has a high cosine similarity with the subject vector of the test question. The first group is used to find the answer category, and the second group to find the answer types. We describe our approach as a dual classification framework as shown in Fig. 2, and applicable to both the datasets, except for, inducing a hierarchy for the gold answer type labels.

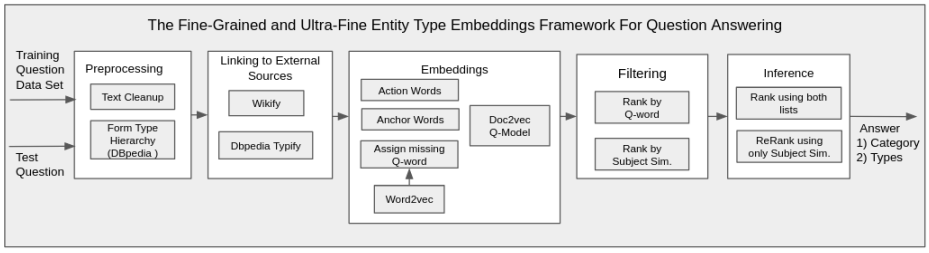

Fig. 2: The Generalized Fine and Ultra-Fine Entity Type Embeddings Framework.

4.1 Semantic Processing and Induced Type Hierarchy.

The first step in the framework is the pre-processing step which encompasses a pipeline constructed using Stanza [20], and an inducing of DBpedia type hierarchy. The pipeline includes a tokenizer, lemmatizer, POS tagger and dependency parser. We select a few syntactic and parts of speech (POS) tags for use in our filtering. From the dependency parse of a question we obtain root and subject/object (*nsub*, *nsubj*:*pass*, *csubj*, *csubj*:*pass* and *obj*) words, which contribute most to the answer type of a question (aka Action words). We select words from the Universal POS categories of ADP, AUX, DET and PART, which contribute most to the basic sentential structure of a question (aka Anchor words). Since DBpedia types form a hierarchical tree, we induce a hierarchy from the

gold types for each question. A hierarchical path helps include any missing types in the gold labels. Since type labels used in the wikidata dataset are not part of a hierarchy, we did not induce a hierarchy from the gold answer type labels.

4.2 Linking to External Sources.

An important component of this framework is identifying entity mentions and replacing them with a type hierarchy. This normalizes questions into abstract forms aiding the Doc2vec model embed similar types of questions closer in the document embedding space. For identifying entity mentions, we collected n-grams (1 to 12) from question sentences and linked to two external sources: pretrained wikipedia2vec word embeddings [28] and a collection of 3.6 million DBpedia entity names with their associated types, collected from DBpedia using SPARQL queries. We performed *wikification*, the process of identifying entity mentions in text by checking against the titles of wikipedia entries [17]. We also performed *typification*, our novel contribution, similar to wikification, where entity mentions with an associated type in the DBpedia hierarchy are normalized to a generic form. We build on ideas from our previous work which used the identification of fine-grained entity types for improved entity coreference resolution [22, 23]. This normalization to a smaller set of types instead of a large number of noun forms abstracts the question into a more generalized form which helps reduce some of the data sparsity. For example, after typification, a question sentence "Who is the president of United States?" is transformed to "who is the _Thing_Agent_Person of _Thing_Place_PopulatedPlace_Country _Thing_Agent_Person ? where *president* and *united_states*, are replaced by their hierarchical DBpedia types. This example also illustrates the noise introduced by DBpedia types: the entity *united_states* is assigned the types: Thing, Place, PopulatedPlace, Country and Person, which resulted in the two induced types where the _Thing_Agent_Person is the noise. DBpedia entities were assigned types through distance learning and these tend to be noisy. While the generalization helps, the noise does not. By inducing a hierarchy, we account for missing types such as Agent in this example, and reduce the type count from 5 to 2.

4.3 Word Embeddings.

Instead of using lexical understanding, we use similarity measures obtained from word embeddings trained on the questions, to aid in filtering. The answer category is highly dependent on the question words. For example, questions starting with a *Is* or *Does* always expect a *boolean* answer category, whereas, questions starting with *when* expect either a number or a date, a *literal* answer category. To reduce noise, we assign a single lemmatized question word, a Q-word, to each question, if it falls into our predefined list of question words shown in x-axis of Fig. 3. About 10-15% of questions do not start with a question word. For these questions, we infer a Q-word using our trained Word2vec model. For example, given the sentence "After what is marathon named and what is the current record?", since the first word is not a question word, we use our word2vec model

to infer the Q-word "what". The goal is to group questions into bins based on Q-word. The Q-word assigned to a question is used as a filter to predict the answer category.

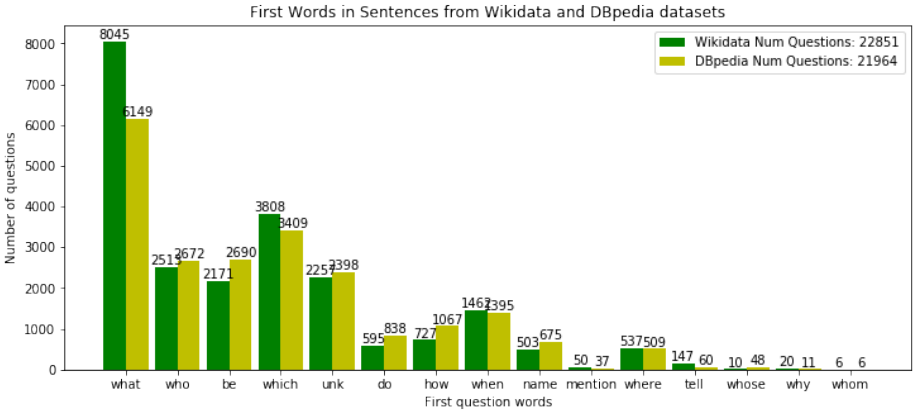

Fig. 3: The distribution of Question words.

To study the efficacy of Q-word inference with our word2vec model, we trained the model on question sentences from the training set. Using the model, we inferred the Q-word for two groups of questions from the test set: a) questions which start with a question word b) questions that do not start with a question word, but contain one or more question words. During inference, the entire question is used as the context; the question word with the highest cosine similarity (only those question words with a frequency > 500 were considered), with respect to the context, is assigned as the Q-word. This assignment is assumed to be correct if it is the same as the question word in the test question. With this model, the prediction accuracy on questions which start with a question word was 80%, and 60% for the questions which contain one or more question words. These results were the same for both datasets (tested individually).

The answer type is highly dependent on the Action and Anchor words. We leverage the trained Word2vec model for Q-words to obtain word vectors for the Action and Anchor words. In addition to the Anchor words belonging to the selected POS categories, we added two manually constructed word categories: a list of words that refer to a date and a list of words that refer to a number. We average the word vectors of Action and Anchor words to obtain a Subject vector for the question. This Subject vector aids in predicting answer types.

4.4 Q-Word and Subject Similarity Filtering.

Q-words and subject similarity are used for filtering the set of similar questions found by Doc2vec for a test question, into two lists: questions with the same Q-word as the test question, and questions with a high subject similarity. Subject similarity is the cosine similarity between the subject vector of the similar

question and the test question, weighted by a question sentence length measure. Paragraph Vectors are biased towards shorter documents [10]; weighting based on sentence length eliminates this bias. A higher weight is assigned to questions similar in length to the test question, and a lower weight to shorter and longer questions. After filtering, we attach the gold answer labels for the answer category, and the answer types to these two lists. The first list is sorted by the document similarity, and second list is sorted by the length weighted document and subject similarity. For both the lists, we only consider the top 10% to reduce noise.

4.5 Inference.

To infer answer category and types, the document vector for a given test question from the Doc2vec model is used to obtain similar questions from the embedding space. These are filtered to get the Q-word and subject similarity lists. The inferred category is the gold answer category of the top ranked question in the Q-word list. Inferred types are collected from the gold answer types of questions in the subject similarity list belonging to the inferred category. For Wikidata questions we collect up to 50 answer types, and for DBpedia we collect up to 10 answer types for achieving the best evaluation score. For DBpedia types, we unroll the hierarchy, listing types in the higher levels only once. Reranking with only subject similarity is applied when there are no inferred types. This could happen when action words are used in a different context. For these questions, we change the inferred category to the gold answer type category of the top ranked question in the subject similarity list, and proceed to infer answer types based on the newly inferred category.

4.6 Examples:

The following illustrate the output of the Doc2vec model before and after filtering.

Example 1. Similar questions for a test question before filtering. (Shows the generalized questions)	
Question	***Generalized Question***
What did sub-orbital spaceflight mean for the mission that the crew member Alan Shepard was a part of ?	what did sub-orbital spaceflight mean for the mission that the crew member _thing_agent_person_astronaut was a part of ?
Top 3 Similar Questions found by the Doc2vec model (before filtering)	
1. What is the human spaceflight mission that Neil Armstrong was part of? 2. Gordon Cooper was the crew member for which space launch? 3. When was Dennis Lillee a member of the Tasmanian cricket team?	what is the human spaceflight mission that _thing_agent_person_astronaut was part of ? _thing_agent_person_astronaut was the crew member for which space launch? when was _thing_agent_person_athlete_cricketer a member of the tasmanian cricket team ?

Example 2. Similar questions for a test question after filtering is applied. (actual questions instead of the generalized questions are listed for readability)
Test Question
For what work did Poul Anderson receive the Prometheus Award - Hall of Fame?
***T*op 3 Similar Questions found by the Doc2vec model (before filtering)**
1. What is the NCL ID of Cao Xueqin? 2. For what work did W.H. Auden receive the Pulitzer Prize for Poetry? 3. For what work did François Mauriac receive the award Grand Prix du roman de l'Académie française?
***Q*-Word Filtering reduces the list to:**
1. For what work did W.H. Auden receive the Pulitzer Prize for Poetry?
***S*ubject Similarity Filtering reduces the list to :**
1. For what work did François Mauriac receive the award Grand Prix du roman de l'Académie française?

From the unfiltered list, although, question 2 appears to be similar, the generalized question is of a different length when compared to the test question

5 Experiments and Evaluation

The SMART task challenge consists of two separate datasets – one for assigning Wikidata type classes and another for assigning DBpedia type classes [12]; each of these datasets consists of a training set and a test set. The training set consists of natural language questions with their corresponding answer category and answer type. The number of training questions for Wikidata is 18,251 and the number for DBpedia is 17,571. The number of test questions for Wikidata was 4,571 and the number of test questions for DBpedia was 4,381.

We compared several methods to study the effects of various parameters on inference. For both datasets, we compared the results with and without the use of the external data sources. The baseline for comparison is our Doc2vec model without any additional filtering. For this baseline, both the Q-word and Subject similarity filtered lists are the same. We compared the performance of the filtering with three different settings, with and without using external source data. The three settings are: 1) using subject similarity calculated from average word vectors of the action words only, 2) using subject similarity calculated from average word vectors of the action words and the anchor words, and 3) re-adjusting the answer category when no answer types can be inferred.

5.1 Results.

Accuracy is used for evaluating the answer category. MRR is used for evaluating answer types from the Wikidata classes. Lenient NDCG@5 and NDCG@10 with a linear decay [17] are used for evaluating the answer types from the DBpedia classes. Accuracy, MRR and NDCG values for the various settings for both the datasets are listed in Table 1.

Experimental Settings	Wikidata		Dbpedia		
	Accuracy	MRR	Accuracy	NDCG@5	NDCG@10
No External Sources	.85	.26	.831	.275	.281
1) Subject Sim. using Action words	.85	.39	.883	.542	.519
2) Subj Sim. using Action & Anchor	.85	.39	.881	.535	.514
3) Rerank with Subject Sim.	.84	.39	.823	.523	.500
Using External Sources	.85	.26	.811	.266	.271
1) Subject Sim. using Action words*	.85	.38	.870	.531	.508
2) Subj Sim. using Action & Anchor*	.85	.39	.873	.527	.505
3) Rerank with Subject Sim.*	.85	.40	.812	.517	.494
	*Results after eliminating the UNKNOWN type				
1) Subject Sim. using Action words	.85	.38	.885	.548	.527
2) Subj Sim. using Action & Anchor	.85	.39	.881	.544	.525
3) Rerank with Subject Sim.	.85	.40	.839	.532	.513

Table 1: Prediction Results for various experimental settings of the framework.

5.2 Analysis - DBpedia Dataset.

The dual filtering improved the accuracy of the model by 5% to 6%. It also doubled the NDCG@5 values and almost doubled the NDCG@10 values, This shows that both our subject similarity and Q-word filtering are effective in identifying the correct answer category and types. Using Anchor words in the subject similarity calculation did not improve the model performance as expected, instead reduced the performance slightly. This could be because the Anchor words cause the model to overfit to the sentential structure.

Using external sources did not improve model performance as expected. Upon closer examination of the predictions, we found entities consisting of long phrases which essentially reduced a question to one or two phrases. This made it difficult to find similar sentences creating more sparsity instead of alleviating it. In addition, for entities where we did not find DBpedia types we used the type UNKNOWN. About 46% of the questions contained at least one entity with no DBpedia types, and 14% of the the questions contained two or more entities with no DBpedia types. The use of the UNKNOWN type reduced the performance. By not replacing entities with UNKNOWN we were able to improve the performance, and notice that generalization with abstract forms helped.

Readjusting the answer category using the subject similarity list did not improve model performance. This implies the answer category inferred originally with the Q-word list is a better fit than the one found through the subject similarity. This shows that Q-word is a good indicator for predicting answer category. We believe inducing a hierarchy is useful especially when labels in the hierarchy are missing from the gold labels. However, we feel that having these additional type labels, missing in the dataset, resulted in a lower evaluation score.

5.3 Analysis - Wikidata Dataset.

The dual filtering improved model prediction for the answer types. The MRR for predicting answer types improved by 1.5 times. This shows that subject sim-

ilarity filtering is effective in identifying the correct answer types. Using Anchor words in subject similarity did not improve model performance. This could be because the questions in this dataset are more of the factoid kind with a simpler sentence structure, and the sentence weighting already contributed to finding similar sentential structures. Sentence generalization with abstract forms (Using External Sources) slightly reduced the performance. However, this reduction was corrected by using Anchor words. Setting entities with no DBpedia types to UNKNOWN did not negatively affect performance, as only 0.6% of the questions contained an entity with no DBpedia types. Readjusting the category using subject similarity improved MRR.

6 Conclusions and Future Work

In this work we present a novel methodology to perform Question Classification that is both flexible and generalizes to other question classification datasets, using a zero shot learning approach. We show that an unsupervised model, namely paragraph vectors, can be used effectively in Question classification. Our model performs well on the limited sized datasets, and long tailed label distributions. Our use of semantic modeling combined with word embeddings helps capture contextual information. Inducing a hierarchy in the Gold Labels for the DBpedia dataset did not improve performance, as the missing labels added were treated as extraneous labels. However, this does appear to be the best way to represent hierarchical labels. Our results show that semantic modeling can improve the performance of shallow neural models. With the use of Word2vec, we show a collaborative approach to semantic modeling and model training. This modular approach is generalized that it can be applied to any question classification dataset. Although, we did see some change in performance with the use of Anchor words, more experimentation is necessary to realize their contribution. Future work will investigate how we can leverage them to learn better sentential structures. In addition, the incorporation of external sources showed promise in this work. We will explore other ways of incorporating external sources in our future work.

References

1. Choi, E., Levy, O., Choi, Y., Zettlemoyer, L.: Ultra-fine entity typing. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 87–96 (2018)
2. Clark, K., Manning, C.D.: Improving coreference resolution by learning entity-level distributed representations. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics Volume 1. pp. 643–653 (2016)
3. Dai, A.M., Olah, C., Le, Q.V.: Document embedding with paragraph vectors. In: NIPS Deep Learning Workshop (2014)
4. Dong, L., Wei, F., Sun, H., Zhou, M., Xu, K.: A hybrid neural model for type classification of entity mentions. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)

5. Harris, Z.: Mathematical structures of language. In: Interscience tracts in pure and applied mathematics (1968)
6. Hill, F., Cho, K., Korhonen, A.: Learning distributed representations of sentences from unlabelled data. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1367–1377 (2016)
7. Huang, Z., Thint, M., Qin, Z.: Question classification using head words and their hypernyms. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing. pp. 927–936. Association for Computational Linguistics
8. Kim, H.K., Kim, H., Cho, S.: Bag-of-concepts: Comprehending document representation through clustering words in distributed representation. Neurocomputing **266**, 336–352 (2017)
9. Landauer, T., Dumais, S.T.: A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review **104**, 211–240 (1997)
10. Lau, J.H., Baldwin, T.: An empirical evaluation of doc2vec with practical insights into document embedding generation pp. 78–86 (Aug 2016), https://www.aclweb.org/anthology/W16-1609
11. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196 (2014)
12. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al.: Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. Semantic web **6**(2) (2015)
13. Levy, O., Goldberg, Y., Dagan, I.: Improving distributional similarity with lessons learned from word embeddings. Transactions of the Association for Computational Linguistics **3**, 211–225 (2015)
14. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th International Conference on Computational Linguistics - Vol. 1. p. 1–7 (2002)
15. Melamud, O., Dagan, I., Goldberger, J., Szpektor, I., Yuret, D.: Probabilistic modeling of joint-context in distributional similarity. In: CoNLL (2014)
16. Metzler, D., Croft, W.B.: Analysis of statistical question classification for fact-based questions. Information Retrieval **8**(3), 481–504 (2005)
17. Mihalcea, R., Csomai, A.: Wikify! linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management. p. 233–242. CIKM '07, New York, NY, USA (2007)
18. Mihindukulasooriya, N., Dubey, M., Gliozzo, A., Lehmann, J., Ngomo, A.C.N., Usbeck, R.: SeMantic AnsweR Type prediction task (SMART) at ISWC 2020 Semantic Web Challenge. CoRR/arXiv **abs/2012.00555** (2020), https://arxiv.org/abs/2012.00555
19. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
20. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C.D.: Stanza: A python natural language processing toolkit for many human languages. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations. pp. 101–108. Online (Jul 2020)
21. da Silva, J.P.C.G., Coheur, L., Mendes, A., Wichert, A.: From symbolic to subsymbolic information in question classification. Artificial Intelligence Review **35**, 137–154 (2010)

22. Sleeman, J., Finin, T.: Type prediction for efficient coreference resolution in heterogeneous semantic graphs. In: 2013 IEEE Seventh International Conference on Semantic Computing. pp. 78–85. IEEE (2013)
23. Sleeman, J., Finin, T., Joshi, A.: Entity type recognition for heterogeneous semantic graphs. AI Magazine **36**(1), 75–86 (2015)
24. Sun, H., Ma, H., Yih, W.t., Tsai, C.T., Liu, J., Chang, M.W.: Open domain question answering via semantic enrichment. In: Proceedings of the 24th International Conference on World Wide Web. pp. 1045–1055 (2015)
25. Voorhees, E.M.: Overview of the trec 2001 question answering track. In: In Proceedings of the Tenth Text REtrieval Conference (TREC. pp. 42–51 (2001)
26. Wang, B., Wang, A., Chen, F., Wang, Y., Kuo, C.C.: Evaluating word embedding models: methods and experimental results. APSIPA Transactions on Signal and Information Processing **8** (01 2019). https://doi.org/10.1017/ATSIP.2019.12
27. Wang, S., Tang, J., Aggarwal, C., Liu, H.: Linked document embedding for classification. In: Proceedings of the 25th ACM international on conference on information and knowledge management. pp. 115–124 (2016)
28. Yamada, I., Asai, A., Sakuma, J., Shindo, H., Takeda, H., Takefuji, Y., Matsumoto, Y.: Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. arXiv preprint 1812.06280v3 (2020)
29. Yavuz, S., Gur, I., Su, Y., Srivatsa, M., Yan, X.: Improving semantic parsing via answer type inference. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 149–159 (2016)