Relaxing Unanswerable Geographic Questions Using A Spatially Explicit Knowledge Graph Embedding Model

Gengchen Mai, Bo Yan, Krzysztof Janowicz, and Rui Zhu

Abstract Recent years have witnessed a rapid increase in Question Answering (QA) research and products in both academic and industry. However, geographic question answering remained nearly untouched although geographic questions account for a substantial part of daily communication. Compared to general QA systems, geographic QA has its own uniqueness, one of which can be seen during the process of handling unanswerable questions. Since users typically focus on the geographic constraints when they ask questions, if the question is unanswerable based on the knowledge base used by a QA system, users should be provided with a relaxed query which takes distance decay into account during the query relaxation and rewriting process. In this work, we present a spatially explicit translational knowledge graph embedding model called TransGeo which utilizes an edge-weighted PageRank and sampling strategy to encode the distance decay into the embedding model training process. This embedding model is further applied to relax and rewrite unanswerable geographic questions. We carry out two evaluation tasks: link prediction as well as query relaxation/rewriting for an approximate answer prediction task. A geographic knowledge graph training/testing dataset, *DB18*, as well as an unanswerable geographic query dataset, *GeoUQ*, are constructed. Compared to four other baseline models, our TransGeo model shows substantial advantages in both tasks.

Keywords: Geographic Question Answering · Query Relaxation · Knowledge Graph Embedding · Spatially Explicit Model

Gengchen Mai
STKO Lab, UC Santa Barbara
e-mail: gengchen_mai@geog.ucsb.edu

Bo Yan
STKO Lab, UC Santa Barbara
e-mail: boyan@geog.ucsb.edu

Krzysztof Janowicz
STKO Lab, UC Santa Barbara
e-mail: jano@geog.ucsb.edu

Rui Zhu
STKO Lab, UC Santa Barbara
e-mail: ruizhu@geog.ucsb.edu

1 Introduction

In the field of natural language processing, Question Answering (QA) refers to the methods, processes, and systems which allow users to ask questions in the form of natural language sentences and receive one or more answers, often in the form of sentences (Laurent et al. 2006). In the past decades, researchers from both academia and industry have been competing to provide better models for various subtasks of QA. Nowadays, many commercial QA systems are widely used in our daily life such as Apple Siri and Amazon Alexa.

Although QA systems have been studied and developed for a long time, geographic question answering remained nearly untouched. Although geographic questions account for a large part of the query sets in several QA datasets and are frequently used as illustrative examples (Yih et al. 2016, Liang et al. 2017), they are treated equally to other questions even though geographic questions are fundamentally different in several ways. First, many geographic questions are highly context-dependent and subjective. Although some geographic questions can be answered objectively and context independently such as *what is the location of the California Science Center*, the answers to many geographic questions vary according to when and where these questions are asked, and who asks them. Examples include *nightclubs near me that are 18+* (location-dependent), *how expensive is a ride from Stanford University to Googleplex* (time-dependent), and *how safe is Isla Vista* (subjective). Second, another characteristic of geographic questions is that the answers are typically derived from a sequence of spatial operations rather than extracted from a piece of unstructured text or retrieved from Knowledge Graphs (KG) which are the normal procedures for current QA systems. For example, the answer to the question *what is the shortest route from California Science Center to LAX* should be computed by a shortest path algorithm on a route dataset rather than searching in a text corpus. The third difference is that geographic questions are often affected by vagueness and uncertainty at the conceptual level (Bennett et al. 2008), thereby making questions such as *how many lakes are there in Michigan* difficult to answer.[1].

Due to the previously mentioned reasons, it is likely to receive no answer given a geographic question. In the field of general QA such cases are handled by so-called (query) relaxation and rewriting techniques (Elbassuoni et al. 2011). We believe that geographic questions will benefit from *spatially-explicit relaxation methods* in which the spatial adjacency and time continuity should be taken into account during relaxation and rewriting. Interestingly, only a few researchers have been working on geographic question answering (Chen et al. 2013, Pulla et al. 2013, Scheider et al. 2018). In this paper, we will mainly focus on how to include spatial adjacency (distance decay effect) into the geographic query relaxation/rewriting framework.

The necessity of query relaxation/rewriting arises from the problem of *unanswerable questions* (Rajpurkar et al. 2018). Almost all QA systems answer a given question based on their internal knowledge bases (KB). According to the nature of such knowledge bases, current QA research can be classified into three categories:

[1] Where the answer can vary between 63,000 and 10 depending on the conceptualization of `Lake`.

unstructured data-based QA (Rajpurkar et al. 2016, Miller et al. 2016, Yang et al. 2017, Chen et al. 2017, Mai, Janowicz, He, Liu & Lao 2018), semi-structure table-based QA (Pasupat & Liang 2015), and structured-KB-based QA (so-called semantic parsing) (Yih et al. 2016, Liang et al. 2017, Berant et al. 2013, Liang et al. 2018, Yih et al. 2016). If the answer to a given question cannot be retrieved from these sources, this question will be called an *unanswerable question*. There are different reasons for unanswerable questions. The first reason is that the information this question focuses on is missing from the current KB. For example, if the question is *what is the weather like in Creston, California* (Question A) and if the weather information of Creston is missing in the current KB, the QA system will fail to answer it. Another reason may stem from logical inconsistencies of a given question. The question *which city spans Texas and Colorado* (Question B) is unanswerable no matter which KBs is used because these states are disjoint.

In order to handle these cases, the initial questions need to be relaxed or rewritten to answerable questions and spatial adjacency need to be considered in this process. A relaxed question to Question A can be *what is the weather like in San Luis Obispo County* because Creston is part of San Luis Obispo County. Another option is to rewrite Question A to a similar question: *what is the weather like in San Luis Obispo (City)* because San Luis Obispo is near to Creston. Which option to consider depends on the nature of the given geographic question. As for Question B, a relaxation solution would be to delete one of the contradictory conditions. Sensible query relaxation/rewriting should be based on both the similarity/relatedness among geographic entities (the distance decay effect) and the nature of the question. However, current relaxation/rewriting techniques (Elbassuoni et al. 2011, Fokou et al. 2017, Wang et al. 2018) do not consider spatial adjacency when handling unanswerable questions, and, thus, often return surprising and counter-intuitive results.

The research contributions of our work are as follows:

1. We propose a spatially explicit knowledge graph embedding model, TransGeo, which explicitly models the distance decay effect.
2. This spatially explicit embedding model is utilized to relax/rewrite unanswerable geographic queries. To the best of our knowledge, we are the first to consider the spatial adjacency between geographic entities in this process.
3. We present a benchmark dataset to evaluate the performance of the unanswerable geographic question handling framework. The evaluation results show that our spatially explicit embedding model outperforms non-spatial models.

The remainder of this work is structured as follows. In Sec. 2, several works about unanswerable question handling are discussed. Next, we present our spatially explicit KG embedding model, TransGeo, and show how to use this model to do unanswerable geographic question relaxation/rewriting in Sec. 3. Then, in Sec. 4 we empirically evaluate TransGeo against 4 other baseline models in two tasks: link predication task, unanswerable geographic question relaxation/rewriting and approximate answer prediction task. Then we conclude our work in Sec. 5 and point out the future research directions.

2 Related Work

The *unanswerable question* problem was recently prominently featured in the open domain question answering research field by Rajpurkar et al. (2018). The authors constructs a benchmark dataset, SQuADRUn, by combining the existing Stanford Question Answering Dataset (SQuAD) with over 50,000 unanswerable questions. These new unanswerable questions are adversarially written by crowd-workers to *look similar to* the original answerable questions. In their paper, the unanswerable questions are used as negative samples to train a better QA model to discriminate unanswerable questions from answerable ones. In our work, we assume the question has already been parsed (e.g. to a SPARQL query) by a semantic parser and resulted in an empty answer set. The task is to relax or rewrite this question/SPARQL query and to generate a related query with its corresponding answer. In the Semantic Web field, SPARQL query relaxation aims to reformulate queries with too few or even no results such that the intention of the original query is preserved while a sufficient number of potential answers are generated (Elbassuoni et al. 2011).

Query relaxation models can be classified into four categories: similarity-based, rule-based, user-preference-based, and cooperative techniques-based models. Elbassuoni et al. (2011) proposed a similarity-based SPARQL query relaxation method by defining a similarity metric on entities in a knowledge graph. The similarity metric are defined based on a statistic language model over the context of entities. The relaxed queries are then generated and ranked based on this metric. This query relaxation method is defined purely based on the similarity between SPARQL queries. In contrast, our model jointly considers the similarity between queries and the probability that a selected answer to the *relaxed* query is, indeed, the answer to *original* query. This is possible due to the so-called Open World Assumption (OWA) commonly used by Web-scale KG by which statements/triples missing from the knowledge graph can still be true unless they are explicitly declared to be false within the knowledge graph. Our model aims at relaxing or rewriting a query such that the top ranked rewritten queries are more likely to generate the correct answer to the original one if it would be known.

With the increasing popularity of machine learning models in question answering and the Semantic Web, knowledge graph embedding models have been used to either predict answers for failed SPARQL queries (Hamilton et al. 2018) or recommend similar queries (Zhang et al. 2018, Wang et al. 2018). KG embedding models aim to learn distributional representations for components of a knowledge graph. Entities are usually represented as continuous vectors while relations, i.e., object properties, are typically represented as vectors (such as in TransE (Bordes et al. 2013), TransH (Wang et al. 2014), and TransRW (Mai, Janowicz & Yan 2018)), matrices (e.g. TransR (Lin et al. 2015)), or tensors. For a comprehensive explanation of different KG embedding models, readers are referred to a recent survey by Wang et al. (2017).

Hamilton et al. (2018) proposes a graph query embeddings model (GQEs) to predict answers for conjunctive graph queries in incomplete knowledge graphs. GQEs first embeds graph nodes (entities) in a low-dimensional space and represents logical operators as learned geometric operations (e.g., translation, rotation) in the embedding space. Based on the learned node embeddings and geometric operations, each conjunctive graph query can be converted into an embedding in a same embedding space. Then cosine similarity is used to compare the query embeddings and node embeddings, and subsequently rank the corresponding entities as potential answers to the current query. While GQEs have been successfully applied to representing conjunctive graph queries and entities in the same embedding space, they have some limitations. For instance, GQEs can only handle conjunctive graph queries, a subset of SPARQL queries. Additionally, the predicted answer to a conjunctive graph query is not associated with a relaxed/rewritten query as an explanation for the answer.

Wang et al. (2018) proposed an entity context preserving translational KG embedding model to represent each entity as a low-dimensional embedding and each predicate as a translation operation between entities. The authors show that compared with TransE (Bordes et al. 2013), the most popular and straightforward KG embedding model, their embedding model performs better in terms of approximating answers to empty answer SPARQL queries. They also present an algorithm to compute *similar* queries to the original SPARQL queries based on the approximated answers. Our work is developed based on this work by overcoming some limitations and including distance decay in the embedding model training process.

3 Method

Before introducing our spatially explicit KG embedding model, we briefly outline concepts relevant to our work.

Definition 1 Knowledge Graph: A knowledge graph (KG) is a data repository, which is typically organized as a directed multi-relational graph. Let $G = \langle E, R \rangle$ be a knowledge graph where E is a set of entities (nodes) and R is a set of relations (labeled edges). A triple $T_i = (h_i, r_i, t_i)$ can be interpreted as an edge connecting the head entity h_i (subject) with the tail entity t_i (object) by relation r_i (predicate). [2]

Definition 2 Entity Context: Given an entity $e \in E$ in the knowledge graph G, the context of e is defined as $C(e) = \{(r_c, e_c) | (e, r_c, e_c) \in G \vee (e_c, r_c, e) \in G\}$.

Definition 3 Basic Graph Pattern (BGP): Let V be a set of query variables in a SPARQL query (e.g., ?place). A basic graph pattern in a SPARQL query is a set of triple patterns (s_i, p_i, o_i) where $s_i, o_i \in E \cup V$ and $p_i \in R$. Put differently, we restrict triple patterns and thus BGP to cases where the variables are in the subject or object position.

[2] Note that in many knowledge graphs, a triple can include a datatype property as the relation where the tail is a literal. In our work, we do not consider these kind of triple as they are not used in any major current KG embedding model. We will use head (h), relation (r), and tail(t) when discussing embeddings and subject (s), predicate (p), object (o) when discussing Semantic Web knowledge graphs to stay in line with the literature from both fields.

Definition 4 SPARQL select query: For the purpose of this work, a SPARQL select[3] query Q_j is defined as the form: $Q_j = \texttt{SELECT}\ V_j\ \texttt{FROM}\ KG\ \texttt{WHERE}\ GP$ where $V_j \subseteq V$ and KG is the studied knowledge graph and GP is a BGP.

The SPARQL query 1 shows an example which corresponds to the natural language question: *In which computer hardware company located in Cupertino is/was Steve Jobs a board member*. The answer should be `dbr:Apple_Inc`. If the triple `(dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California)`, however, is missing from current KG, this question would become an unanswerable geographic question. Compared to the full SPARQL 1.1 language standard, two limitations of the given definition of a SPARQL query should be clarified:

1. Predicates in a SPARQL 1.1 BGP can also be a variables. Hence, Def. 3 presents a subset of all triple patterns, which can appear in a standard SPARQL query.
2. SPARQL 1.1 also contains other operations (`UNION, OPTION, FILTER, LIMIT`, etc.) not considered here and in related state-of-the-art work (Wang et al. 2018, Hamilton et al. 2018) .

```
SELECT ?v
WHERE {
?v dbo:locationCity dbr:Cupertino,_California .
?v dbo:industry dbr:Computer_hardware .
dbr:Steve_Jobs dbo:board ?v .}
```

Listing 1: An example SPARQL query generated by a semantic parser.

Given a SPARQL query Q_j parsed from a natural language geographic question, if executing Q_j on the current KG yields an empty answer set, our goal is: **1)** learn a spatially explicit KG embedding model for the current KG which takes distance decay into account; **2)** use the embedding model to infer a ranked list of approximated answers to this question; and **3)** generate a relaxed/related SPARQL query for each approximate answer as an explanation for the query relaxation/rewriting process.

3.1 Modeling Geographic Entity Context in Knowledge Graphs

Based on the examples about relaxing or rewriting Question A and Question B in the introduction, we observe that a suitable query relaxation/rewriting for an unanswerable geographic question should consider both the similarity/relatedness among geographic entities (e.g., the distance decay effect) as well as the nature of the question. In terms of measuring semantic similarities among (geographic) entities in a knowledge graph, we borrow the assumption of distributional semantics from computational linguistic that *you shall know a word by the company it keeps* (Firth 1957). In analogy, the semantic similarity among (geographic) entities can be measured based on their contexts (Yan et al. 2017).

[3] We ignore `ASK`, `CONSTRUCT`, and `DESCRIBE` queries here as they are not typically used for question answering, and, thus, also not considered in related work.

With regards to measuring the similarity/relatedness between general entities in a knowledge graph, both Elbassuoni et al. (2011) and Wang et al. (2018) consider the one degree neighborhood of the current entity as its context, which is shown in Def. 2. However, **this entity context modeling falls apart when geographic entities are considered in two ways**. First, this geographic entity context modeling does not fully reflect *Tobler's first law of geography*, which indicates that *near things are more related than distant things*. Since Def. 2 only considers object property triples as the entity context and disregard all datatype properties, all positional information, e.g., geographic coordinates, would not be considered in the context modeling. Although the place hierarchy is encoded as object property triples in most KG, e.g., GeoNames, GNIS-LD, and DBpedia, and these triples can also indirectly introduce distance decay effects into the context modeling, such contextual information is far too coarse. For example, Santa Barbara County, Los Angeles County, and Humboldt County are all subdivisions of California. From a place hierarchy perspective, all three should have the same relatedness to each other. But Santa Barbara County is more related to Los Angeles County rather than Humboldt County.

The second reason is due to the way geographic knowledge is represented in Web-scale knowledge graphs. For any given populated place, the place hierarchy of administrative units is modeled using the same canonical predicates. Put differently, even if no other triples are known about a small settlement, the KG will still contain at least a triple about a higher-order unit the place belongs to, e.g., a county. Consequently, all populated places in, say, Coconino County, Arizona, will share a common predicate (e.g., `dbo: isPartOf`) and object (e.g., `dbr:Coconino_County,_Arizona`) . For tiny deserted settlements such as Two Guns, AZ this may also be the sole triple known about them. In contrast, major cities in the same county or state, e.g., Flagstaff, will only have a small percentage of their total object property triples be about geographic statements. This will result in places about which not much is known to have an artificially increased similarity.

These aforementioned two reasons demonstrate the necessity to model geographic entity context in a different way rather than Def. 2. In this work, we redefine Def. 2 by combing an edge-weighted PageRank and a sampling procedure. The underlying idea is to assign larger weights to geographic triples in an entity context where the weights are modeled from a distance decay function.

To provide a final and illustrative example of the problems that arise form embedding models that are not spatially explicit, consider the work by Wang et al. (2018). Their query example is *which actor is born in New York and starred in a United States drama film directed by Time Burton*. After passing the SPARQL version of this question to their query relaxation/rewriting model, the model suggests to change the birthplace from New York to Kentucky which is certainly a surprising relaxation from the original query. Although Kentucky is also a place as New York, it is too far away from the brithplace, New York, the QA system user is interested in. A more reasonable relaxed/rewritten query should replace New York City with its nearby places, e.g. New Jersey.

3.2 Spatially Explicit KG Embedding Model

Given a knowledge graph $G = \langle E, R \rangle$, a set of geographic entities $P \subseteq E$, and a triple $T_i = (h_i, r_i, t_i) \in G$, we treat G as an undirected, unlabeled, edge-weighted multigraph MG, which means that we ignore the direction and label (predicate) for each triple in G. The weight $w(T_i)$ for triple T_i is defined in Equ. 1, where D is the longest (simplified) earth surface distance which is half of the length of the equator measured in kilometer; $dis(h_i, t_i)$ is the geodesic distance between geographic entity h_i and t_i on the surface of an ellipsoidal model of the earth measured in kilometer. The ε is a hyperparameter to handle the cases where h_i and t_i are collocated; and l is the lowest edge weight we allow for each triple. If the head place and tail place of a geographic triple are too far apart, we set its weight as the lower bound l, indicating that we do not expect strong spatial interaction at this distance. [4]

$$w(T_i) = \begin{cases} \max(\ln \frac{D}{dis(h_i,t_i)+\varepsilon}, l) & \text{if } h_i \in P \land t_i \in P \\ l & \text{otherwise} \end{cases} \tag{1}$$

The location of h_i and t_i are represented as their geographic coordinates stored in a knowledge graph, which are usually points. In this work, we use the `geo:geometry` property to get the coordinates of all geographic entities in DBpedia.

After we compute weights for each triple in MG, an edge-weighted PageRank is applied to this weighted multigraph, where edge weights are treated as the transition probability of the random walker from one entity node to its neighboring entity node. In order to prevent the random walker to get stuck at one *sinking node*, the PageRank algorithm also defines a teleport probability, which allows the random walker to jump to a random node in MG with a certain probability at each time step. Let $PR(e_i)$ be the PageRank score for each entity e_i in the knowledge graph, then $PR(e_i) \in (0, 1)$ represents the probability of a random walker to arrive at entity e_i after n time steps. If e_i had a lot of one degree triples (i.e., $|C(e_i)|$ is large), then e_i would have a larger $PR(e_i)$. Since $\sum_i PR(e_i) = 1$ and $|C(e_i)|$ have a long tail distribution, $PR(e_i)$ will also have a long tail distribution with few very large values but many small values. This skewed distribution would affect the later sampling process. In order to normalize $PR(e_i)$, we apply a *damping* function (Equ. 2). In Equ. 2, ln is the natural log function; N is the number of entities in the knowledge graph G. This function has the nice property that $w(e_i)$ increases monotonically w.r.t. $PR(e_i)$ and the distribution of $w(e_i)$ is more normalized than $w(e_i)$. Therefore, $w(e_i)$ encodes the structural information of the original knowledge graph and the distance decay effect on interaction (and similarity/relatedness more broadly) among geographic entities. The more incoming and outgoing triples one entity e_i has, the larger its $w(e_i)$ will be. Also, the closer two geographic entities $e_i, e_j \in P$ are, the larger $w(e_i)$ and $w(e_j)$ would be.

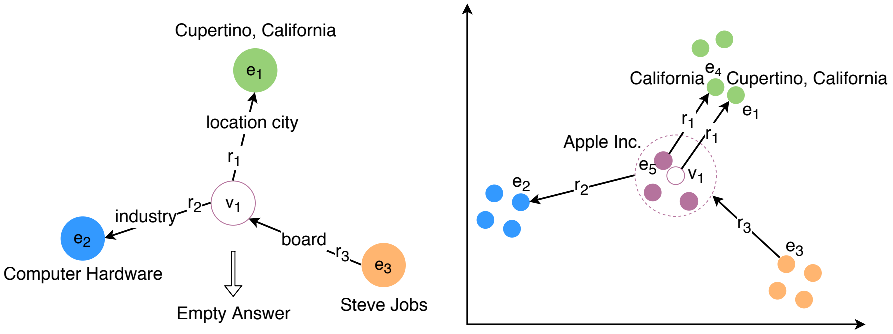
[4] We leave the fact that interaction depends on the travel mode and related issues for further work. Similarity, due to the nature of existing knowledge graphs, we use point data to represent places despite the problems this may introduce. Work on effectively integrating linestrings, polygons, and topology into Web-scale knowledge graphs is ongoing (Regalia et al. 2017).

$$w(e_i) = N \cdot \frac{\frac{1}{-\ln PR(e_i)}}{\sum_i \frac{1}{-\ln PR(e_i)}} \tag{2}$$

Next, we introduce the knowledge graph embedding model, which utilizes $w(e_i)$ as the distribution from which the entity context is sampled. Since $w(e_i)$ directly encodes the distance decay information among geographic entities, we call our model spatially explicit KG embedding model, denoted here as TransGeo.

Translation-based KG embedding models embed entities into low-dimensional vector spaces while relations are treated as translation operations in either the original embedding space (TransE) or relation-specific embedding space (TransH, TransR). This geometric interpretation provides us with a useful way to understand the embedding-based query relaxation/rewritten process.

Fig. 1 shows the basic graph pattern of Query 1 and their vector representations in KG embedding space. If triple `(dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California)` is missing from the current KG, this query becomes an unanswerable query. However, if we already obtained the learned embeddings for e_1, e_2, e_3, r_1, r_2, and r_3, we could compute the embedding of the query variable $?v$ with each triple pattern. Next, we can compute the weighted average of these embeddings to get the final embedding of $?v$, which is denoted as $\mathbf{v}$. Next, the k-nearest neighbor entities of $\mathbf{v}$ can be obtained based on the cosine similarity between their embeddings. These k-nearest neighbor entities are treated as approximated answers to the original query 1. Based on each of these candidate answers, we cycle through each triple pattern in the original Query 1 to see whether they need to be relaxed or not, which is the major procedure for embedding-based query relaxation/rewritten.



Incomplete knowledge graph yields an empty answer

Relaxation in the embedding space

Fig. 1: An unanswerable geographic query example and its corresponding KG embedding

In order to make the embedding-based query relaxation/rewriting process work well, the KG embedding model should be an entity context preserving model. However, one problem for the original TransE model is that each triple is treated independently in the training process which does not guarantee its context preservation. Inspired by the Continuous-Bag-of-word (CBOW) word embedding model (Mikolov et al. 2013), Wang et al. (2018) proposed an entity context preserved KG embedding model which predicts the *center* entity based on the entity context (Def. 2). However, as we discussed in Sec. 3.1, the geographic entity context can not be fully captured by using Def. 2 and we need another method to capture the distance decay effect, where $w(e_i)$ plays a role. Another shortcoming of the embedding model proposed in Wang et al. (2018) is that the size of entity context $|C(e_i)|$ varies among different entities which will make the number of triples trained in each batch different. This will have a negative effect on the model optimization process. Some entities may have thousands of incoming and outgoing triples, e.g., `dbr:United_States` has 232,573 context triples. This will imply that the model parameters will only update once all these triples are processed which is not a good optimization technique.

Based on this observation, we define a hyperparameter d as the context sampling size for each entity. If $|C(e_i)| > d$, then the context $C(e_i)$ of entity e_i would not be fully used in each KG embedding training step. Instead, the training context $C_{samp}(e_i)$ is sampled from $C(e_i)$ ($C_{samp}(e_i) \subseteq C(e_i)$) while the sampling probability of each context item (r_{ci}, e_{ci}) is calculated based on the damped PageRank value $w(e_{ci})$. If $|C(e_i)| > d$, the training context $C_{samp}(e_i)$ is sampled without replacement. If $|C(e_i)| < d$, $C_{samp}(e_i)$ is sampled with replacement. After a certain number of epochs t_{freq}, $C_{samp}(e_i)$ will be resampled for each entity. Because of this sampling strategy, a context item (r_{ci}, e_{ci}) of e_i would have a higher chance to be sampled if $e_i \in P \wedge e_{ci} \in P$, and e_i is close enough to e_{ci} in geographic space.

$$P(r_{ci}, e_{ci}) = \frac{w(e_{ci})}{\sum_{(r_{cj}, e_{cj}) \in C(e_i)} w(e_{cj})}, \text{ where } (e_i, r_{ci}, e_{ci}) \in G \vee (e_{ci}, r_{ci}, e_i) \in G \qquad (3)$$

Based on the definition of entity training context $C_{samp}(e_i)$, a compatibility score between $C_{samp}(e_i)$ and an arbitrary entity e_k can be computed as Equ. 4, in which $\phi(e_k, r_{cj}, e_{cj})$ is the plausibility score function between (r_{cj}, e_{cj}) and e_k. In Equ. 5, $|| \cdot ||$ represents the $L1$-norm of the embedding vector; $\mathbf{e_k}, \mathbf{e_{cj}}$ represent the KG embeddings for the corresponding entity e_k, e_{cj}, and $\mathbf{r_{cj}}$ is the relation embedding of r_{cj}.

$$f(e_k, C_{samp}(e_i)) = \frac{1}{|C_{samp}(e_i)|} \cdot \sum_{(r_{cj}, e_{cj}) \in C_{samp}(e_i)} \phi(e_k, r_{cj}, e_{cj}) \qquad (4)$$

$$\phi(e_k, r_{cj}, e_{cj}) = \begin{cases} ||\mathbf{e_k} + \mathbf{r_{cj}} - \mathbf{e_{cj}}|| & \text{if } (e_i, r_{cj}, e_{cj}) \in G \\ ||\mathbf{e_{cj}} + \mathbf{r_{cj}} - \mathbf{e_k}|| & \text{if } (e_{cj}, r_{cj}, e_i) \in G \end{cases} \qquad (5)$$

The same assumption has been used here as TransE, which is that, in the *perfect* situation, if $(h_i, r_i, t_i) \in G$, $||\mathbf{h_i} + \mathbf{r_i} - , \mathbf{t_i}|| = 0$. Based on Equ. 4 and 5, if $e_k = e_i$, each $\phi(e_k, r_{cj}, e_{cj})$ would be small and close to zero, thus $f(e_i, C_{samp}(e_i))$ would be also small and close to zero. In contrast, if $C(e_k) \cap C(e_i) = \emptyset$, each $\phi(e_k, r_{cj}, e_{cj})$ would be very large and $f(e_i, C_{samp}(e_i))$ would also also large.

In order to set up the learning task, the pairwise ranking loss function has been used as the objective function like most KG embedding models do. Specifically, for each entity e_i, we randomly sample K entities as the negative sampling set $Neg(e_i)$ for e_i. Equ. 6 shows the objective function of TransGeo, where γ is the margin and $max()$ is the maximum function.

$$\mathcal{L} = \sum\nolimits_{e_i \in G} \sum\nolimits_{e_i' \in Neg(e_i)} max\Big(\gamma + f(e_i, C_{samp}(e_i)) - f(e_i', C_{samp}(e_i)), 0\Big) \tag{6}$$

3.3 KG Embedding Model Based Query Relaxation and Rewriting

After obtaining the learned TransGeomodel, we adopt the same procedure as Wang et al. (2018) to relax/rewrite the query. We briefly summarize the process below. We assume a SPARQL query Q with two variables $?v_1$ and $?v_2$, which are targets to be relaxed/rewritten in order to find approximated answers.

1. Given an empty answer SPARQL query Q, we partition the basic graph pattern into several groups such that all triple patterns in one group only contain one variable. Triples who have two variables $?v_1$ and $?v_2$ (connected triples) as its subject and object respectively are treated differently;
2. For each triple pattern group which contains variable $?v$, the embedding of $?v$ is first computed by each triple pattern based on the translation operations from the entity node to the variable node. Then the final embedding of $?v$ is computed as the weighted average of previous computed variable embeddings. The weight is calculated based on the number of matched triples of each triple patterns in the KG;
3. If Q has any connection triples, the embeddings of variables computed from each triple pattern group are refined based on the predicate of the connection edges. Then these embeddings will be treated as the final embeddings for each variable;
4. The approximate answers to each variable are determined by using their computed variable embeddings to search for the k-nearest embeddings of entities based on their cosine similarity. Each variable will have a ranked list of entities, e.g. $A(?v_1)$, $A(?v_2)$, as their approximated answers;
5. If Q has any connection triples, e.g. $(?v_1, r, ?v_2)$, we need to first use beam search to get top-K answer tuples for $?v_1$ and $?v_2$. And then each answer tuple (e_{1i}, e_{2j}) is checked for the condition $(e_{1i}, r, e_{2j}) \in G$. The answer tuples which satisfy this condition will be returned as a ranking list $Ans(Q)$ of approximated answers;
6. For each answer tuple $(e_{1i}, e_{2j}) \in Ans(Q)$, we enumerate each triple pattern to check the satisfaction. As for triple $(?v_1, r, e)$, if $(e_{1i}, r, e) \in G$, we do not perform any relaxation. If $(e_{1i}, r, e) \notin G$, then $(?v_1, r, e)$ will be relaxed based on Equ. 7. However, if e_{1i} does not have any outgoing triples, this triple pattern could not be relaxed and we would delete this triple pattern from the query relaxation/rewriting result. But the similarity score of this relaxation result will be set to 0;
7. The ranked list of answer tuples as well as the relaxed queries associated with them are returned to the users.

$$(e_{1i}, r_k, e_k) = \arg\max\Big(\frac{\mathbf{r} \cdot \mathbf{r_k}}{||\mathbf{r}|| \cdot ||\mathbf{r_k}||} + \frac{\mathbf{e} \cdot \mathbf{e_k}}{||\mathbf{e}|| \cdot ||\mathbf{e_k}||}\Big) \tag{7}$$

4 Experiment

Since almost all the established knowledge graph training dataset for KG embedding models, e.g., FB15K, WN18, do not contain enough geographic entities, we collect a new KG embedding training dataset, *DB18*[5], which is a subgraph of DBpedia. The dataset construction procedure is as follow: 1) We first selected all geographic entities which are part of (`dbo:isPartOf`) `dbr:California` with type (`rdf:type`) `dbo:City` which yields 462 geographic entities; 2) We use these entities as seeds to get their 1-degree and 2-degree object property triples and filter out triples with no `dbo:` properties; 3) we delete the entities and their associated triples whose node degree is less than 10; 4) we split the triple set into training and testing set and make sure that every entity and relation in the testing dataset will appear in training dataset. The statistic of *DB18* is listed in Tab. 1. 'Geographic entities' here means entities with a `geo:geometry` property.

Table 1: Summary statistic for *DB18*

DB18	Total	Training	Testing
# of triples	139155	138155	1000
# of entities	22061	-	-
# of relations	281	-	-
# of geographic entities	1681 (7.62%)	-	-

Following the method we describe in Sec. 3.2, we compute the edge weights for each triple in *DB18* and an edge-weighted PageRank algorithm is applied on this undirected unlabeled multigraph. Here we set l to 1 and ε to 1. We select four models as the baseline models to compare with TransGeo: **1)** *TransE*; **2)** the context preserving translational KG embedding (Wang et al. 2018); **3)** a simplified version of TransGeo in which the entity context items are randomly sampled from a uniform distribution, denoted as $TransGeo_{unweighted}$; **4)** another simplified version of our model in which the PageRank are applied to unweighted multigraph, denoted as $TransGeo_{regular}$. We implement *TransE*, $TransGeo_{unweighted}$, $TransGeo_{regular}$, and TransGeo, in Tensorflow. We use the original Java implementation of (Wang et al. 2018)[6]. For all five models, we train them for 1000 epochs with the margin $\gamma = 1.0$ and learning rate $\alpha = 0.001$. As for $TransGeo_{unweighted}$, $TransGeo_{regular}$, and TransGeo, we use 30 as the entity context sampling size d and 1000 as batch size. We resample the entity context every 100 epochs. As for the context preserving transla-

[5] `https://github.com/gengchenmai/TransGeo`

[6] `https://github.com/wangmengsd/re`

tional KG embedding (Wang et al. 2018), we use 10 as the entity context size cut-off value. The embedding dimension of all these five embedding models is 50.

In order to demonstrate the effectiveness of our spatially explicit KG embedding model, TransGeo, over the other four baseline models, we evaluate these five KG embedding models in two task: the standard link predication task and an relaxation/rewriting task to predict answers to the otherwise unanswerable geographic questions. The evaluation results are listed in Tab. 2.

The common link prediction task is used to validate the translation preserving characteristic of different models. The set up of the link prediction task follows the evaluation protocol of Bordes et al. (2013). Given a correct triple $T_k = (h_k, r_k, t_k)$ from the testing dataset of DB18, we replace the head entity h_k (or tail t_k) with all other entities from the dictionary of DB18. The plausibility scores for each of those n triples are computed based on the plausibility score functions of *TransE* ($\| \mathbf{h} + \mathbf{r} - \mathbf{t} \|$). Then these triples are ranked in ascending order according to this score. The higher the correct triple ranks in this list, the better this learned model. Note that some of the corrupted triples may also appear in the KG. For example, as for triple (`dbr:Santa_Barbara,_California`, `dbo:isPartOf`, `dbr:California`), if we replace the head `dbr:Santa_Barbara,_California` with `dbr:San_Francisco`, the result corrupted triple (`dbr:San_Francisco`, `dbo:isPartOf`, `dbr:California`) is still in the DBpedia KG. These false negative samples need to be filtered out. Mean reciprocal rank (*MRR*) and *HIT@10* are used as evaluation matrics where *Raw* and *Filter* indicate the evaluation results on the original ranking of triples or the filtered list which filters out the false negative samples. According to Tab. 2, TransGeo performs the best in most of the metrics and the only metric TransGeo cannot outperform is *MRR* in the raw setting. This evaluation shows that our spatially explicit model does indeed hold the translation preserving characteristic.

Table 2: Two evaluation tasks for different KG embedding models

	Link Prediction				SPARQL Relaxation	
	MRR		HIT@10		MRR	HIT@10
	Raw	Filter	Raw	Filter		
TransE Model	**0.122**	0.149	30.00%	34.00%	0.008	5% (1 out of 20)
Wang et al. (2018)	0.113	0.154	27.20%	30.50%	0.000	0% (0 out of 20)
$TransGeo_{regular}$	0.094	0.129	28.50%	33.40%	0.098	25% (5 out of 20)
$TransGeo_{unweighted}$	0.108	0.152	30.80%	37.80%	0.043	15% (3 out of 20)
TransGeo	0.104	**0.159**	**32.40%**	**42.10%**	**0.109**	**30% (6 out of 20)**

For the quality of the unanswerable geographic query relaxation/rewriting results, we evaluate the results based on the ranking of the approximate answers (Hamilton et al. 2018). Let's take Question 1 as an example. One reason which causes an empty answer is that some triples were missing from the KG, e.g., (`dbr:Apple_Inc,dbo:locationCity,dbr:Cupertino,_California`), and the current SPARQL query is overly restrictive. However, based on the KG embedding model, we can approximate the embeddings of the variables in the current query. This variable embeddings can be used to search for the most *probable* answers/entities to each variable in the embedding space. These *k-nearest* entities are assumed to be more probable to be the correct answer of the original question. The correct answer (based on the Open World Assumption) to Question 1 is `dbr:Apple_Inc`. If the KG embedding is good at preserving the context of entities, the embedding of `dbr:Apple_Inc` will appear close to the computed variable embedding (See Fig. 1). So the performance of the query relaxation/rewriting algorithm can be evaluated by checking the rank of the correct answer in the returned ranking list of the approximate answers.

Based on the above discussion, we construct another evaluation dataset, *GeoUQ*, which is composed of 20 unanswerable geographic questions. Let G_{train} be a knowledge graph which is composed of all the training triples of DB18[7] and G_{all} be a knowledge graph containing all training and testing triples in DB18[8]. Both G_{train} and G_{all} can be accessed through the SPARQL endpoint. These queries satisfy 2 conditions: 1) each query Q will yield empty answer set when executing Q on G_{train}; 2) Q will return only one answer when executing Q on G_{all}. The reason for making Q a one-answer query in G_{all} is that the user also expects one answer from the QA system to the question (s)he poses. One-answer queries are also the common setup for many QA benchmark datasets, e.g. WikiMovie (Miller et al. 2016), WebQuestionsSP (Yih et al. 2016). *MRR* and *HIT@10* are used as evaluation metrics for this task.

All five KG embedding models are evaluated based on the same query relaxation/rewriting implementation. The evaluation results are shown in Tab. 2. From Tab. 2, we can conclude that TransGeo outperform all the other baselines models both on *MRR* and *HIT@10*.

Tab. 4 show the top 3 query relaxation/rewriting results of Question 1 from all the 5 KG embedding models. For each query, the highlighted part in the BGP is the part where the query is changed from the original Query 1. Note that some of the relaxation/rewriting results have less triple patterns than the original Query 1. This is because the current approximate answer/entity does not have any outgoing or incoming triples to be set as the alternative to the original triple pattern. Hence, we delete this triple pattern. This has been described in Step 6 in Sec. 3.3. From Tab. 4, we can see that the correct answer `dbr:Apple_Inc` has been listed as the second approximate answer for TransGeo. However, all the 4 baseline models fail to predict this correct answer in their top 10 approximate answers list. Besides the perspective of predicting the correct answers, we can also evaluate the models by inspecting the quality of the relaxed/rewritten queries. For example, the top 1 relaxed query from TransGeo changes `dbr:Cupertino,_California` to `dbr:Redwood_City,_California` which is a nearby city of `dbr:Cupertino,_California`. Although the predicted answer is `dbr:NeXT` rather than `dbe:Apple_Inc`, this query relaxation/rewriting makes sense and is meaningful for the user. The 2nd relaxed result from TransGeo changes `dbr:Cupertino,_California` to `dbr:California`

[7] `http://stko-testing.geog.ucsb.edu:3080/dataset.html?tab=query&ds=/GeoQA-Train`

[8] `http://stko-testing.geog.ucsb.edu:3080/dataset.html?tab=query&ds=/GeoQA-All`

which is a superdivision of `dbr:Cupertino,_California`. This is indeed a real *query relaxation* which relaxes the geographic constraint to its superdivision. In short, our spatially explicit KG embedding model, TransGeo, produces better result than all baseline models.

5 Conclusion

In this work, we discussed why geographic question answering differs from general QA in general, and what this implies for relaxation and rewriting of empty queries specifically. We demonstrated why distance decay has to be included explicitly in the training of knowledge graph embeddings and showed cases of neglecting to do so. As a result, we propose a spatially explicit KG embedding models, TransGeo, which utilizes an edge-weighted PageRank and sampling strategy to include the distance decay effect into the KG embedding model training. We constructed a geographic knowledge graph training dataset, *DB18* and evaluated TransGeo as well as four baseline models. We also created an unanswerable geographic question dataset (*GeoUQ*) for two evaluation tasks: link prediction and answer prediction by relaxation/rewriting. Empirical experiments show that our spatially explicit embedding model, TransGeo, can outperform all the other 4 baseline methods on both task. As for the link prediction task, in the filter setting, our model outperforms the other baselines by at least 3.2% at *MRR* and 11.4% at *HIT@10*. In terms of the unanswerable geographic question approximate answer prediction task, our model outperform the other 4 baselines by at least 11.2% at *MRR* and 20% at *HIT@10*.

In terms of future work, firstly, the distance decay information is explicitly encoded into our KG embedding model which gives up on flexibility, e. g., to model modes of transportation. In the future, we want to explore ways to only consider distance decay during query relaxation rather than the model training step. Secondly, as for the method to compute the edge weights of the knowledge graph, we used point geometries which may yield misleading results for larger geographic areas such as states. This limitation is due to the availability of existing knowledge graphs. Work to support more complex geometries and topology is under way.

References

Bennett, B., Mallenby, D. & Third, A. (2008), An ontology for grounding vague geographic terms., *in* 'FOIS', Vol. 183, pp. 280–293.

Berant, J., Chou, A., Frostig, R. & Liang, P. (2013), Semantic parsing on freebase from question-answer pairs, *in* 'Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing', pp. 1533–1544.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J. & Yakhnenko, O. (2013), Translating embeddings for modeling multi-relational data, *in* 'Advances in neural information processing systems', pp. 2787–2795.

Table 3: Query relaxation/rewriting results of different KG embedding models for Query 1

	Relaxation 1	Relaxation 2	Relaxation 3

TransE	**Query:**
	SELECT ?v WHERE {
	?v dbo:locationCity **dbr:Fountain_Valley,_California** .
	?v dbo:industry **dbr:Data_storage_device** .
	dbr:John_Tu dbo:knownFor ?v .
	}
	Answer: dbr:Kingston_Technology
		Query:
		SELECT ?v WHERE {
		?v dbo:locationCity **dbr:Agoura_Hills,_California** .
		?v dbo:industry **dbr:Interactive_entertainment** .
		dbr:Heavy_Iron_Studios dbo:owningCompany ?v .
		}
		Answer: dbr:THQ
			Query:
			SELECT ?v WHERE {
			?v dbo:locationCity **dbr:Pasadena,_California** .
			?v dbo:industry **dbr:Entertainment** .
			}
			Answer: dbr:Landmark_Entertainment_Group

Wang et al. (2018)	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Irvine,_California** .
	?v dbo:industry **dbr:Video_game_industry** .
	dbr:Activision_Blizzard dbo:division ?v .
	}
	Answer: dbr:Blizzard_Entertainment
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:Mountain_View,_California** .
		?v dbo:industry **dbr:Interactive_entertainment** .
		}
		Answer: dbr:Paragon_Studios
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:San_Mateo,_California** .
			?v dbo:industry **dbr:Video_game** .
			}
			Answer: dbr:Digital_Pictures

$TransGeo_{unweighted}$	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Palo_Alto,_California** .
	?v dbo:industry dbr:Computer_hardware .
	dbr:William_Redington_Hewlett dbo:knownFor ?v .
	}
	Answer: dbr:Hewlett-Packard
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:California** .
		?v **dbo:keyPerson dbr:Elon_Musk** .
		dbr:Elon_Musk dbo:knownFor ?v .
		}
		Answer: dbr:SolarCity
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:Santa_Clara,_California** .
			?v dbo:industry dbr:Computer_hardware .
			}
			Answer: dbr:Console_Inc

$TransGeo_{regular}$	**Query:**
	SELECT ?v
	WHERE {
	?v **dbo:foundationPlace dbr:Santa_Clara,_California** .
	?v dbo:industry dbr:Computer_hardware .
	}
	Answer: dbr:Jasomi_Networks
		Query:
		SELECT ?v
		WHERE {
		?v dbo:location **dbr:San_Carlos,_California** .
		?v dbo:industry dbr:Computer_hardware .
		}
		Answer: dbr:Check_Point
			Query:
			SELECT ?v
			WHERE {
			?v dbo:locationCity **dbr:Lake_Forest,_California** .
			?v dbo:industry dbr:Computer_hardware .
			}
			Answer: dbr:PSSC_Labs

TransGeo	**Query:**
	SELECT ?v
	WHERE {
	?v dbo:locationCity **dbr:Redwood_City,_California** .
	?v dbo:industry dbr:Computer_hardware .
	dbr:Steve_Jobs **dbo:occupation** ?v .
	}
	Answer: dbr:NeXT
		Query:
		SELECT ?v
		WHERE {
		?v dbo:locationCity **dbr:California** .
		?v dbo:industry dbr:Computer_hardware .
		dbr:Steve_Jobs **dbo:board** ?v .
		}
		Answer: dbr:Apple_Inc
			Query:
			SELECT ?v
			WHERE {
			?v **dbo:foundationPlace dbr:Sioux_City,_Iowa** .
			?v dbo:industry dbr:Computer_hardware .
			dbr:EMachines dbo:owningCompany ?v .
			}
			Answer: dbr:Gateway,_Inc

Chen, D., Fisch, A., Weston, J. & Bordes, A. (2017), Reading wikipedia to answer open-domain questions, *in* 'Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Vol. 1, pp. 1870–1879.

Chen, W., Fosler-Lussier, E., Xiao, N., Raje, S., Ramnath, R. & Sui, D. (2013), A synergistic framework for geographic question answering, *in* 'Semantic Computing (ICSC), 2013 IEEE Seventh International Conference on', IEEE, pp. 94–99.

Elbassuoni, S., Ramanath, M. & Weikum, G. (2011), Query relaxation for entity-relationship search, *in* 'Extended Semantic Web Conference', Springer, pp. 62–76.

Firth, J. R. (1957), 'A synopsis of linguistic theory, 1930-1955', *Studies in linguistic analysis* .

Fokou, G., Jean, S., Hadjali, A. & Baron, M. (2017), 'Handling failing rdf queries: from diagnosis to relaxation', *Knowledge and Information Systems* **50**(1), 167–195.

Hamilton, W., Bajaj, P., Zitnik, M., Jurafsky, D. & Leskovec, J. (2018), Embedding logical queries on knowledge graphs, *in* 'Advances in Neural Information Processing Systems', pp. 2027–2038.

Laurent, D., Séguéla, P. & Nègre, S. (2006), QA better than IR?, *in* 'Proceedings of the Workshop on Multilingual Question Answering', Association for Computational Linguistics, pp. 1–8.

Liang, C., Berant, J., Le, Q., Forbus, K. D. & Lao, N. (2017), Neural symbolic machines: Learning semantic parsers on freebase with weak supervision, *in* 'Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', Vol. 1, pp. 23–33.

Liang, C., Norouzi, M., Berant, J., Le, Q. V. & Lao, N. (2018), Memory augmented policy optimization for program synthesis and semantic parsing, *in* 'Advances in Neural Information Processing Systems', pp. 10014–10026.

Lin, Y., Liu, Z., Sun, M., Liu, Y. & Zhu, X. (2015), Learning entity and relation embeddings for knowledge graph completion., *in* 'AAAI', Vol. 15, pp. 2181–2187.

Mai, G., Janowicz, K., He, C., Liu, S. & Lao, N. (2018), POIReviewQA: A semantically enriched POI retrieval and question answering dataset, *in* 'Proceedings of the 12th Workshop on Geographic Information Retrieval', ACM, p. 5.

Mai, G., Janowicz, K. & Yan, B. (2018), Support and centrality: Learning weights for knowledge graph embedding models, *in* 'International Conference on Knowledge Engineering and Knowledge Management', Springer, pp. 212–227.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. (2013), Distributed representations of words and phrases and their compositionality, *in* 'Advances in neural information processing systems', pp. 3111–3119.

Miller, A., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A. & Weston, J. (2016), Key-value memory networks for directly reading documents, *in* 'Empirical Methods in Natural Language Processing (EMNLP)', pp. 1400–1409.

Pasupat, P. & Liang, P. (2015), Compositional semantic parsing on semi-structured tables, *in* 'Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)', Vol. 1, pp. 1470–1480.

Pulla, V. S., Jammi, C. S., Tiwari, P., Gjoka, M. & Markopoulou, A. (2013), Questcrowd: A location-based question answering system with participation incentives, *in* '2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)', IEEE, pp. 75–76.

Rajpurkar, P., Jia, R. & Liang, P. (2018), 'Know what you don't know: Unanswerable questions for SQuAD', *arXiv preprint arXiv:1806.03822* .

Rajpurkar, P., Zhang, J., Lopyrev, K. & Liang, P. (2016), SQuAD: 100,000+ questions for machine comprehension of text, *in* 'Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing', pp. 2383–2392.

Regalia, B., Janowicz, K. & McKenzie, G. (2017), Revisiting the representation of and need for raw geometries on the linked data web., *in* 'LDOW@ WWW'.

Scheider, S., Ballatore, A. & Lemmens, R. (2018), 'Finding and sharing GIS methods based on the questions they answer', *International Journal of Digital Earth* pp. 1–20.

Wang, M., Wang, R., Liu, J., Chen, Y., Zhang, L. & Qi, G. (2018), Towards empty answers in sparql: Approximating querying with rdf embedding, *in* 'International Semantic Web Conference', Springer, pp. 513–529.

Wang, Q., Mao, Z., Wang, B. & Guo, L. (2017), 'Knowledge graph embedding: A survey of approaches and applications', *IEEE Transactions on Knowledge and Data Engineering* **29**(12), 2724–2743.

Wang, Z., Zhang, J., Feng, J. & Chen, Z. (2014), Knowledge graph embedding by translating on hyperplanes., *in* 'AAAI', Vol. 14, pp. 1112–1119.

Yan, B., Janowicz, K., Mai, G. & Gao, S. (2017), From itdl to place2vec: Reasoning about place type similarity and relatedness by learning embeddings from augmented spatial contexts, *in* 'Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems', ACM, p. 35.

Yang, F., Nie, J., Cohen, W. W. & Lao, N. (2017), 'Learning to organize knowledge with n-gram machines', *arXiv preprint arXiv:1711.06744* .

Yih, W.-t., Richardson, M., Meek, C., Chang, M.-W. & Suh, J. (2016), The value of semantic parse labeling for knowledge base question answering, *in* 'Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)', Vol. 2, pp. 201–206.

Zhang, L., Zhang, X. & Feng, Z. (2018), 'TrQuery: An embedding-based framework for recommanding sparql queries', *arXiv preprint arXiv:1806.06205* .