Deriving Validity Time in Knowledge Graph

Julien Leblay*
Artificial Intelligence Research Center
AIST Tokyo Waterfront
Tokyo, Japan
firstname.lastname@aist.go.jp

Melisachew Wudage Chekol
Data and Web Science Group
University of Mannheim
Mannheim, Germany
mel@informatik.uni-mannheim.de

ABSTRACT

Knowledge Graphs (KGs) are a popular means to represent knowledge on the Web, typically in the form of node/edge labelled directed graphs. We consider temporal KGs, in which edges are further annotated with time intervals, reflecting when the relationship between entities held in time. In this paper, we focus on the task of predicting time validity for unannotated edges. We introduce the problem as a variation of relational embedding. We adapt existing approaches, and explore the importance example selection and the incorporation of side information in the learning process. We present our experimental evaluation in details.

CCS CONCEPTS

• **Computing methodologies** → *Temporal reasoning*; *Supervised learning*;

KEYWORDS

Temporal Knowledge Graph, Factorization Machines

ACM Reference Format:
Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving Validity Time in Knowledge Graph. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3184558.3191639

1 INTRODUCTION

Knowledge Graphs (KGs) encompass a class knowledge representation models, in which nodes correspond to entities, and directed labelled edges the relationships between them. Some well-known examples of KGs include Google's Knowledge Vault [5], NELL [4], YAGO [6], and DBpedia [1]. Whether the data is generated and maintained by users or computer programs, mistakes and omissions can easily proliferate, and the data can quickly become outdated. To make matters worse, some of the most popular formats used for data publishing, including RDF, JSON or CSV, do not provide built-in mechanisms to easily capture and retain information as the data changes over time. As an example, consider the following facts extracted from the DBpedia (http://dbpedia.org/page/Grover_Cleveland) dataset about Grover Cleveland, the 22^{th} and 24^{th} president of the USA.

```
(GCleveland, birthPlace, Caldwell),
(GCleveland, office, POTUS),
(GCleveland, office, NewYork_Governor)
```

The lack of temporal information is problematic in this example for several reasons. None of these facts is independently false, yet Grover Cleveland could not have been president and governor at the same time. Moreover, this information is missing since Grover Cleveland has been president twice, during two non consecutive periods. So, clearly temporal metadata would lift some ambiguity, yet not all facts typically need such metadata. For instance, his birth place is not expected to change over time.

Many KGs do not contain the validity period of facts, i.e., the period during which the fact is considered to hold. Notable exceptions include Wikidata [20] and YAGO, in which some facts that are endowed with temporal information. Our goal is to learn temporal meta-data on a knowledge graph where such information is incomplete. For the above example, we want to derive annotations of the following form:

```
(GCleveland, office, POTUS):[1885-1889;1893-1897]
(GCleveland, office, NewYork_Governor):[1883-1885]
```

Note that Grover Cleveland was president during two distinct, non consecutive terms.

In the following section, we provide some formal background and review the related work. In Section 3, we first attempt to carry over techniques from relational embedding models, and study the limitations of these approaches. Then, we proceed to show that factorization machines are particularly well-suited for our temporal scope prediction task, allowing to take valuable side-information into account. In Section 4, we report early experimental results.

2 PRELIMINARIES

In the following, we introduce temporal knowledge graphs formally, as well as the problem addressed in this paper. We present possible extensions of relational embedding approaches and factorization machines.

2.1 Temporal Knowledge Graphs

We are considering KGs of the form $G = (\mathcal{E}, \mathcal{R})$, where $\mathcal{E}$ is a set of labeled nodes known as entities, and R is a set of labeled edges known as relations. Alternatively, we can refer to G, as a set of triples of the form $(subject, predicate, object)$, where *subject* and *object* are node-labels, and *predicate* is an edge label. Labels act as unique identifiers for subjects and predicates, and either as identifier or literal value for objects. Hence, the presence of an edge p between two nodes s and o indicates that the fact (s, p, o) holds. In practice, knowledge is not static in time, thus we would like to capture when a given fact held over time. Thus, we assume a set of discrete time points T, and an additional labeling scheme on edges,

*Dr. Leblay's work is supported by the KAKENHI grant number 17K12786.

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3191639

which takes a set of time intervals over T, denoting the periods within which a fact was considered true. This yields a *temporal KG*.

2.2 Problem statement

Our goal is to learn associations between facts of a KG and one or more time points in T. This gives us the ability to tackle the following tasks:

Time prediction: given a query of the form $(s, p, o, ?)$, predict the time point(s) for which the fact is consider valid/true.

Time-dependent query answering: given a point in time and a fact with missing subject, predicate or object, predict the most likely label.

2.3 Related Work

We present the related work from three different angles: (i) temporal scoping of knowledge graph facts, (ii) relational embedding for link prediction, and (iii) factorization machines for triple classification.

2.3.1 Temporal scoping of KG facts. The study of deriving the temporal scopes of KG facts has recently gained momentum. The most recent of which is Know-Evolve [19]. A temporal KG in Know-Evolve is a set of facts where each fact has a timestamped relation. For embedding entities and timestamped relations, they use a bilinear model (RESCAL) and employ a deep recurrent neural network in order to learn non-linearly evolving entities. The learning phase espouses a point-process, by which the estimation of whether a fact hold at time t is based on the state at time $t-1$. That said, they do not exploit side information as we do in this work. Another closely related work is the time-aware KG embedding model of Jiang et al. [7]. They focus on the prediction of an entity or relation given a time point in which the fact is supposed to be valid. Both Know-Evolve and time-aware KG completion methods use relational embedding models which are discussed below. Furthermore, in [18], the authors use tensor decomposition to assign validity scopes for KG facts. However, as reported in the paper, their models do not perform sufficiently well. Nonetheless, this can be improved by including side information as we did here.

In contrast, Rula et al. [14] extract time information contained in Web pages using syntactic rules. This process has three phases whereby candidate intervals for facts are matched, selected and then merged according to temporal consistency rules. YAGO [6] is another earlier example, in which both time and space scopes were extracted using linguistic extraction rules, followed but conflict resolving post-processing.

In [21], the authors formulate the temporal scoping problem as a state change detection problem. In doing so, they enrich temporal profiles of entities with relevant contextual information (these are unigrams and bigrams surrounding mentions of an entity, for instance, for the entity Barack Obama relevant unigrams include 'elect', 'senator' and so on). From there, they learn vectors that reflect change patterns in the contexts. For example, after 'becoming president', US presidents often see a drop in mentions of their previous job title state such as 'senator' or 'governor' in favor of 'president'.

Another temporal scoping system developed by [15] relies on a language model consisting of patterns automatically derived from Wikipedia sentences that contain the main entity of a page and temporal slot-fillers extracted from the corresponding infoboxes.

Talukdar et al. [17] use frequency counts of fact mentions to define temporal profiles (basically a time-series of the occurrences of facts over time in a corpus of historical documents) of facts and analyze how the mentions of those facts rise and fall over time. They identify temporal scope over input facts, using a 3-phase procedure.

Yet, the approach is rather brittle in that it does not automatically adapt to new relations, and requires human experts at several steps in the process.

Bader et al. [2] used matrix decomposition on the Enron email dataset, to estimate relationship among the scandal's stakeholders over time. Unlike in our settings, the relationships were not labeled.

2.3.2 Relational Embedding approaches. Our problem is more generally related to relational embedding models, a paradigm of relational learning in low dimensional vector space, which has been widely used for tasks such as link prediction and fact classification. Such embeddings can be viewed as a special case of graph embedding, a very active research topic, which we omitted here for conciseness. We can broadly divide the models into three categories based on: (i) translational distance, (ii) tensor factorization (bilinear models), and more recently, (iii) neural networks. Vectors are used to learn entity and relation embeddings in translational models, whereas additional matrices are used in the case of bilinear models and neural networks. While the translational models use a distance metric to measure the plausability of facts, bilinear models rely on the dot product of entity and relational embeddings. One of the most well known translational models is TransE [3]. Its simplicity allows for straightforward extensions [9]. The translation embedding of a triple (s, p, o) corresponds to $\mathbf{s} + \mathbf{p} \approx \mathbf{o}$. A scoring function $score(\mathbf{s}, \mathbf{p}, \mathbf{o})$, either the ℓ_1 or ℓ_2 norm, is used to measure the distance (i.e., similarity) as:

$$score(\mathbf{s}, \mathbf{p}, \mathbf{o}) = -||\mathbf{s} + \mathbf{p} - \mathbf{o}||_{\ell_{1/2}} \tag{1}$$

The training set contains positive examples (G), and negative examples (G') generated as follows:

$$G'_{(s,p,o)\in G} = \{(s', p, o) \mid s' \in \mathcal{E}, (s', p, o) \notin G\} \cup \{(s, p, o') \mid o' \in \mathcal{E}, (s, p, o') \notin G\}.$$

Hence, G' contains triples with either s or o replaced by a random entity from the set $\mathcal{E}$.

RESCAL [11], also referred to as *bilinear model*, uses a tensor factorization model by representing triples in a tensor. That is, for each triple $x_{ijk} = (s_i, p_k, o_j)$, $y_{ijk} = \{0, 1\}$ denotes its existence or nonexistence in a tensor $\mathbf{Y} \in \{0, 1\}^{|\mathcal{E}|\times|\mathcal{E}|\times|\mathcal{R}|}$. RESCAL learns vector embeddings of entities and a matrix $\mathbf{W}_p \in \mathbb{R}^{d\times d}$ for each relation $r \in \mathcal{R}$ where each slice $\mathbf{Y}$ is factorized as: $\mathbf{Y} \approx \mathbf{s}^\top \mathbf{W}_p \mathbf{o}$. Hence, the scoring function for the bilinear model is:

$$score(s, p, o) = \mathbf{s}^\top \mathbf{W}_p \mathbf{o}. \tag{2}$$

Other notable relational embedding models are HolE [10] and Neural Tensor Networks (NTN) [16]. HolE improves the efficiency of RESCAL by using a circular correlation operation (it compresses the interaction between two entities) for scoring triples.

Almost all relational embedding approaches minimize a margin-based ranking loss function $\mathcal{L}$ over some training dataset. $\mathcal{L}$ is

given by the following equation:

$$\mathcal{L} = \sum_{(s,p,o)\in G} \sum_{(s,p,o)'\in G'_{(s,p,o)}} [\gamma + score((\mathbf{s},\mathbf{p},\mathbf{o})) - score((\mathbf{s},\mathbf{p},\mathbf{o})')]_+, \tag{3}$$

where $[x]_+$ denotes the positive part of x, $\gamma > 0$ is a margin hyperparameter. Different optimization functions such as stochastic gradient descent are used to minimize $\mathcal{L}$.

2.3.3 Factorization Machines. Unlike vector space embedding models, Factorization Machines (FMs) allow us to incorporate contextual information which improves prediction performance. Rendle [12] introduced FMs to model the interaction between features using factorized parameters. One big advantage of FMs is that they allow to estimate all interactions between features even with very sparse data. In addition, FMs can mimic many different matrix factorization models such as biased matrix factorization, Singular Value Decomposition (SVD++) [8], and Pairwise Interaction Tensor Factorization (PITF) [13]. FMs provide flexibility in feature engineering as well as high prediction accuracy. Moreover, FMs can be applied to the following tasks: regression, binary classification, and ranking. The model of a factorization machine is given by the following equation:

$$score(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j,$$

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle := \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f},$$

where $score : \mathbb{R}^n \to T$ is a prediction function from a real valued feature vector $\mathbf{x} \in \mathbb{R}^n$ to a target domain, $T = \mathbb{R}$ for regression, $T = \{+,-\}$ for classification and so on. The model parameters: w_0 denotes the global bias; w_i within $\mathbf{w} \in \mathbb{R}^n$ indicates the strength of the i-th variable with n being the size of the feature vector; $\langle \mathbf{v}_i, \mathbf{v}_j \rangle$ models the interaction between the i-th and j-th variables. $\langle ., . \rangle$ is the dot product of two vectors of size k.

Furthermore, the model parameter $\mathbf{v}_i$ in $V \in \mathbb{R}^{n \times k}$ describes the i-th variable with k factors. k is a hyperparameter that defines the dimension of the factorization.

In this work, since we need to predict the validity of facts of (possible many) time points, we use factorization machine for classification rather than regression or ranking.

3 TEMPORAL SCOPE PREDICTION

In the following we consider relational embedding models and factorization machines for temporal scope prediction.

3.1 Relational Embedding Models for Temporal KGs

We propose various approaches for representing temporal knowledge graphs in vector space. In particular, we investigate several extensions of existing relational embedding approaches.

3.1.1 TTransE. Short for Temporal TransE, this is an extension of the well known embedding model TransE [3], by substituting its scoring function.

(a) Naive-TTransE: time is encoded by way of synthetic relations. For each relation r in the vocabulary and each time point $t \in T$, we assume a synthetic relation $r{:}t$. For instance, the temporal fact `(GCleveland, office, POTUS):1888`, is encoding as `(GCleveland, office:1888, POTUS)`. The scoring function is unchanged (as in equation (1)):

$$score(\mathbf{s}, \mathbf{p}{:}\ \mathbf{t}, \mathbf{o}) = -||\mathbf{s} + \mathbf{p} : \mathbf{t} - \mathbf{o}||_{\ell 1/2} \tag{4}$$

While this model is simple, it is not scalable. Besides the link prediction does not distinguish between two consecutive timepoints, for instance, for the task (GCleveland, ?, POTUS), office:1988 and office:1989 are equally possible links.

(b) Vector-based TTransE: in this approach, time is represented in the same vector space as entities and relations. The scoring function becomes:

$$score(\mathbf{s}, \mathbf{p}, \mathbf{o}, \mathbf{t}) = -||\mathbf{s} + \mathbf{p} + \mathbf{t} - \mathbf{o}||_{\ell 1/2} \tag{5}$$

In this approach, time points have embedding representations, just like entities and relations. The rationale behind this scoring function is to drive a (subject, predicate)-pair close to the correct object, relative to any valid point in time.

(c) Coefficient-based TTransE: time points (or rather a normalization thereof) are used as a coefficient affecting the subject and relation embeddings of a triple.

$$score(s, p, o, t) = -||t * (\mathbf{s} + \mathbf{p}) - \mathbf{o}||_{\ell 1/2} \tag{6}$$

As a variant of this, only the relation is affected by time:

$$score(s, p, o, t) = -||\mathbf{s} + t * \mathbf{p} - \mathbf{o}||_{\ell 1/2} \tag{7}$$

Unlike Vector-based TTransE, time points are represented as real values in $(0, 1]$, and thus are not directly affected by the optimization.

3.1.2 TRESCAL. TRESCAL is a temporal extension of RESCAL. We extend its bilinear temporal scoring function as follows. As in Naive-TTransE, time is encoded by means of synthetic relations just like Naive-TTransE.

$$score(s, p, o, t) = \mathbf{s}^\top \mathbf{W}_{\mathbf{p}:\mathbf{t}} \mathbf{o} \tag{8}$$

This model is straight forward extension of the bilinear model. Despite its simplicity, it does not scale well, besides, the prediction results are quite poor.

3.2 Factorization Machines for Temporal KGs

Among the approaches described so far, the naive ones do not scale well with time domains of increasing size or resolution. Although the vector-based TTransE approach performs overall better than the other techniques, it did not show good enough performance to solve our problem in practice. In the following, we show how we used factorization machines to solve both or scability and performance issues.

Data/Feature Representation. We consider a knowledge graph $G = G_t \cup G_c$ where G_t is a set of quadruples or timestamped triples, and G_c is a set of atemporal triples that we refer to as a *context graph*. For instance, the following is a temporal graph G_t:

```
(GCleveland, office, POTUS):1888,
(GCleveland, office, POTUS):1895,
```

Approach	LR	M	D	E	MR (p)	Hits@1 (p)	MR (o)	Hits@10 (o)	MR (t)	Hits@10 (t)	Cost Red.
Eq. 4	0.1	2	100	1000	537.51	0.6	2578.4	11.0	59.2	10.3	99.75%
Eq. 5	0.01	1	200	1000	141.67	22.69	1295.54	13.59	58.44	7.76	45.32%
Eq. 6	0.1	10	100	500	835.22	0.55	9884.69	0.91	58.50	8.62	0.13%
Eq. 7	0.01	2	50	500	796.65	0.18	9374.92	0.19	58.50	8.62	0.45%
Eq. 8	0.01	2	100	1000	483.32	3.1	6588.6	1.9	58.5	12.1	99.99%

Table 1: Mean Rank (MR), Hits@{1,10} and cost reduction for our temporal embeddings methods on the FreeBase dataset.

and its context graph G_c is given below:

`(GCleveland, birthPlace, Caldwell)`.

An input to an FM is a feature vector representation of the pair (G_t, G_c). The feature vector encoding can be constructed in several ways such as one-hot encoding, bag-of-words (representing KG entities and relations in a bag or multiset) and so on [12]. The features associated with a fact of the form (s, p, o) are $\{bow(s), p, bow(o)\}$, where $bow(x)$ returns the bag of words of all the literals in relations with subject x.

Example Generation. To generate positive examples, we used temporal sampling, guiding by input parameter TS, which consists in sampling uniformly s_t time points within the fact's validity intervals. A second parameter, NS, guides negative sampling, producing s_n for each positive time-point-based fact/example, using the same random corruption techniques as in [3].

4 EXPERIMENTS

We implemented our approach based on the scikit-kge library of RESCAL and TransE[1], and libFM/pywFM[2].

4.1 Datasets

We originally experimented with theFreebase database often used in the related work (our first set of experiments). However, the facts on those dataset not having temporal information, we randomly generate such metadata for a subset of them, by picking two random years and using them as start and end validity dates. For this reason, it is hard to compare our results with corresponding work in the non-temporal relational embedding scenarios. Freebase has approximately 14K entities, et 1000 relations, with 60k examples. We later decided to switch to Wikidata, a knowledge base with reasonably high quality time information. More over Wikidata is much larger and up-to-date. We only briefly present the result obtained in the former dataset, and results were largely negative. Besides, using the Freebase and WordNet data set with the factorization machines approach was not possible because of the lack of side-information to exploit; the data sets contain very little plain text.

Our process in preparing the Wikidata data set was the following. We extracted triples from a recent dump, and partitioned them into two sets: (i) temporal facts: facts having some temporal annotations, such as point-in-time, start time, end time or any sub-property thereof, (ii) atemporal facts: atemporal facts, having no such annotations. Temporal properties annotating temporal facts include "start time", "inception", "demolition time", etc. In this work, we only consider years, and thus normalize all years to the Gregorian calendar and discard information of finer granularity. Facts annotated with a single point-in-time are associated with that time-point as start and end time.

During the learning phase, temporal facts are used to generate positive and negative examples and atemporal facts are used to collect side information. The complete data has 4.2M temporal facts. Out of approximately 3600 distinct properties, 2770 are strictly atemporal, i.e., none of their corresponding triples are temporal annotation. Out of the remaining properties, 17 are strictly temporal, i.e., all their corresponding triples have temporal annotations, while for the remaining 813 properties, only some triples are annotated. We partition the triples into two sets, respectively with and without temporal annotations, the former being our original example set. From this example set (temporal facts), we exclude the strictly temporal ones (since they are not candidate for prediction), the fact featuring the most frequent single frequent property — covering nearly 1.2M examples —, and those with properties covering less than 10 examples (approximately 397 properties). Ultimately, our example set contains 2.5M examples, much more than most datasets used in related approaches (see for example [3, 10]). We also report our results on a reduced version of this data set, containing 180K temporally annotated facts (i.e., approx. 5% of the overall data). Our dataset can be found online for reproducibility[3].

The second set of triples (atemporal facts) is used for generating features. We also remove the set of triples with low semantic content such as those mapping a Wikidata entity ID to that of other datasets.

4.2 Temporal relational embeddings

For this experiment, we use the modified Freebase dataset, and evaluated the approaches with a slightly modified version of that from the related work, which evaluate using query triples, i.e., facts in which one item is omitted and need to be predicted by the models. For query answering, s or o is omitted, while p is omitted in link prediction. The evaluation metrics are the Mean Rank of the correct answers among all answers order by their predicating probability. The lower, the better. Metrics also include the "Hits@K", i.e., the percentage of case in which the correct answer is in the top K results. Hits@10 is a popular metric, yet for small domain (such as in link prediction), Hits@1 is usually preferred. In our setting, we deal with quadruple, therefore we extend the process to time prediction in which, time is omitted, and will evaluate how often a predicted validity time point is *within* the actual validity interval of the fact.

In Table 1, we only report the best results obtained with each approaches. We ran the approaches with learning rates (LR) among

[1]https://github.com/mnick/scikit-kge
[2]https://github.com/srendle/libfm

[3]http://staff.aist.go.jp/julien.leblay/datasets/

	OM	TS	Precision	Recall	F1	Accuracy
WD_180K	ALS	1	58.44%	71.27%	64.22%	60.23%
WD_180K	ALS	10	67.94%	88.95%	77.04%	73.48%
WD_180K	ALS	100	74.56%	92.47%	82.56%	80.45%
WD_2.5M	ALS	10	78.15%	97.64%	86.81%	85.16%
WD_180K	MCMC	1	64.98%	81.07%	72.14%	68.64%
WD_180K	MCMC	10	69.55%	89.69%	78.35%	75.21%
WD_180K	MCMC	100	79.28%	92.28%	85.28%	84.07%
WD_2.5M	MCMC	10	85.41%	97.64%	91.12%	90.48%

Table 2: Precision, recall, F1-measure and accuracy on the WD_180K dataset with varying temporal sampling at 100 iterations (OM**: Optimization Method,** TS**: temporal sample size).**

{.01, .1}, margins (M) among {2, 10}, dimensionalities of the vector space (D) among {20, 50, 100, 200}, and learning over 500 or 1000 epoqs (E). It is clear from the table that the performs are not satisfying. However, we can distinguish two general problems. For the naive methods (Eq. 4-8), the space explodes from the multiplication of "virtual relations" entailed by the methods. This is why performance are poor despite significant cost reductions achieved through the learning process. The other methods however do not achieve much cost reduction all together. Our best explanation for this is that learning time validity simple from the *structure* of the graph (i.e., using no other external information) is simply to hard. This conclusion led us to turn to the Factorization Machine approach, more akin to the incorporation of side information.

4.3 Classification task on FM

For the classification task, the learning is done on quadruples of the form $(s, p, o, t) = \pm 1$, modeling whether the triple (s, p, o) held at time t or not. After the sampling, the effective number of examples increase. For instance with TS = 3, `(GCleveland, office, POTUS):[1885, 1889]`, will generate positive examples for the time points 1885, 1887, 1889. The evaluation, in turn, is performed on *time points* rather than time intervals. We use the standard definition of prediction, recall, F-measure and accuracy. The definitions of these measures are given below:

$$precision = \frac{\#true\ positives}{\#positive\ predictions}$$

$$recall = \frac{\#true\ positives}{\#ground\ truth\ positives}$$

$$F{-}measure = 2 \times \frac{precision \times recall}{precision + recall}$$

$$accuracy = \frac{\#correct\ predictions}{\#all\ predictions}$$

We used the optimization functions Alternating Least Square (ALS), and Markov Chain Monte Carlo (MCMC). We report the precision, recall, F-measure and accuracy in Table 2, which shows the results for experiments run on a Wikidata data set of 180K and 2.5M examples, using bag-of-words as side information, with increasing temporal sampling size. The results for high NS are omitted since the greater number of negative examples tends to biases the model towards negative predictions, resulting in high accuracy, despite poor precision. With a balanced set of positive and negative examples, precision is positively correlated with TS. Using a temporal sampling of 100, with our smaller dataset, precision and recall peak at 74.5% and 92% respectively after 100 iterations, with an F1-measure and accuracy around 82%. Using a temporal sampling size of 10, with our bigger dataset, the F1-measure and accuracy reach 90%. Increasing the sample size, also improves performance, yet producing positive examples for all time points within a time interval degrades the performance, probably due to over-fitting. Our result also shows that a precision of around 70% can be achieved with only 10 iterations.

Our most demanding experiment took sightly over 6 hours to complete on a regular laptop, with 16GB of RAM, and a 2.8 GHz Intel Core i5 processor.

We have excluded experimental results for TTransE and TRESCAL as our result showed the methods were not competitive.

5 CONCLUSION

In this work, we studied the problem of temporal scope prediction. We adapted several existing relational embedding approaches in which our experimental results have shown that they suffer from either scalability or accuracy. Factorization machines overcome these shortcomings as they provide a way to incorporate side information which improves prediction performance. We designed a new dataset by carefully analyzing Wikidata and carried out several experiments. We believed our experimental results are quite promising. Next, we plan to turn our attention to neural network-based approaches, extend our current framework to support time-aware link prediction and query answering, and applies our finding to other types are context prediction, such as space or provenance. We all plan to apply the approach in an open information extraction setting.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[2] Brett W Bader, Richard A Harshman, and Tamara G Kolda. 2007. Temporal analysis of semantic graphs using ASALSAN. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*. IEEE, 33–42.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*. 2787–2795.

[4] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning.. In *AAAI*, Vol. 5. 3.

[5] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *SIGKDD*. 601–610.

[6] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard De Melo, and Gerhard Weikum. 2011. YAGO2: exploring and querying world knowledge in time, space, context, and many languages. In *Proceedings of the 20th international conference companion on World wide web*. ACM, 229–232.

[7] Tingsong Jiang, Tianyu Liu, Tao Ge, Lei Sha, Baobao Chang, Sujian Li, and Zhifang Sui. 2016. Towards Time-Aware Knowledge Graph Completion.. In *COLING*. 1715–1724.

[8] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 426–434.

[9] Dat Quoc Nguyen. 2017. An overview of embedding models of entities and relationships for knowledge base completion. *arXiv preprint arXiv:1703.08098* (2017).

[10] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, and others. 2016. Holographic Embeddings of Knowledge Graphs.. In *AAAI*. 1955–1961.

[11] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th*

international conference on machine learning (ICML-11). 809–816.

[12] Steffen Rendle. 2012. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 3 (2012), 57.

[13] Steffen Rendle and Lars Schmidt-Thieme. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 81–90.

[14] Anisa Rula, Matteo Palmonari, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, Jens Lehmann, and Lorenz Bühmann. 2014. Hybrid acquisition of temporal scopes for rdf data. In *European Semantic Web Conference*. Springer, 488–503.

[15] Avirup Sil and Silviu Cucerzan. 2014. Temporal scoping of relational facts based on Wikipedia data. *CoNLL-2014* (2014), 109.

[16] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*. 926–934.

[17] Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. 2012. Coupled temporal scoping of relational facts. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 73–82.

[18] Volker Tresp, Yunpu Ma, Stephan Baier, and Yinchong Yang. 2017. *Embedding Learning for Declarative Memories*. Springer International Publishing, Cham, 202–216. DOI: https://doi.org/10.1007/978-3-319-58068-5_13

[19] Rakshit Trivedi, Mehrdad Farajtabar, Yichen Wang, Hanjun Dai, Hongyuan Zha, and Le Song. 2017. Know-Evolve: Deep Reasoning in Temporal Knowledge Graphs. *arXiv preprint arXiv:1705.05742* (2017).

[20] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[21] Derry Tanti Wijaya, Ndapandula Nakashole, and Tom M Mitchell. 2014. CTPs: Contextual Temporal Profiles for Time Scoping Facts using State Change Detection.. In *EMNLP*. 1930–1936.