Temporal Reasoning Over Event Knowledge Graphs

Ali Sadeghian*
University of Florida
Gainesville, FL
asadeghian@ufl.edu

Miguel Rodriguez*
University of Florida
Gainesville, FL
mer@cise.ufl.edu

Daisy Zhe Wang
University of Florida
Gainesville, FL
daisyw@cise.ufl.edu

Anthony Colas
University of Florida
Gainesville, FL
acolas1@ufl.edu

ABSTRACT

Many advances in the computer science field, such as semantic search, recommendation systems, question-answering, natural language processing, are drawn-out using the help of large scale knowledge bases (e.g., YAGO, NELL, DBPedia). However, many of these knowledge bases are static representations of knowledge and do not model time on its own dimension or do it only for a small portion of the graph. In contrast, projects such as GDELT and ICEWS have constructed large temporally annotated knowledge graphs of events collected from news hubs. In this paper, we study the problem of reasoning over such graphs. In particular, transpose two well-known techniques from knowledge base reasoning to utilize the temporal dimension: rule mining and graph embeddings. We mine temporally constrained first-order inference rules using the state-of-the-art relational knowledge base model. We interpret the learned rules as event sequence rules. We also use simple embedding methods to jointly learn a universal representation of entities and time-specific representations of the knowledge graph. We present the first set of temporal rules mined over event knowledge graphs and preliminary results on using the learned embeddings in the temporal link prediction task.

KEYWORDS

Sequential Rule Mining, Temporal Reasoning, Knowledge Graph Embedding

1 INTRODUCTION

Recent advances in scalability and accuracy of information extraction pipelines have increased the attention in automatically constructed Knowledge Bases (KBs) such as DBPedia[3], Yago[4], Google Knowledge Graph[1], etc. These knowledge bases contain multi-relational information between real-world entities such as the team membership of professional sports players or the government head of a country. Regardless of the ever increasing size of KBs, they remain largely incomplete. This has motivated research in link prediction to use information from existing knowledge bases to fill-in possibly missing relations between known entities. The two most popular statistical models used for the Knowledge Base Completion (KBC) task [22] are *latent feature* models or embeddings, that learn mappings from entities and relations to vectors and *observed feature* models that learn weights for existing paths between entities that are linked in the KB.

While link prediction pipelines continue to show increased average precision results, we note the data model they use is time-invariant, in contrast with most of human knowledge that is inherently dynamic. For instance, if a link prediction pipeline is used to complete the triple (*USA*, *president*, **?*) its unclear which US presidents are missing in the KB or which one is been specifically requested. This small motivational example suggests that link prediction should incorporate the temporal dimension to play a key role. One big problem in shifting from time-invariant link prediction to temporal link prediction is the lack of temporally annotated data in general purpose KBs. In Yago2 [13], one of the leading KBs in temporal annotation, only 9 relations are temporally annotated corresponding to 25.3% of its base triples. In contrast, knowledge graphs extracted from news hubs such as ICEWS [18] and GDELT [19] create a structured daily digest of events where every edge is properly timestamped. Despite the importance of temporal information and attention to temporal information extraction, there has been little research on inference from Temporal Knowledge Bases (TKB).

In this paper we present temporally-aware observed and latent features models learned over event knowledge graphs. Particularly,

- We customize Ontological Pathfinding, a large scale rule mining framework to mine event sequence rules, a type of temporally constrained first order horn clauses.
- We create the first set of event sequence rules automatically mined from temporal knowledge graphs.
- We show that embedding methods can be used for inference in Temporal Knowledge Bases and introduce ChronoTranslate.

[1]http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

*These two authors contributed equally.

KBCOM'18, Feb 2018, Los Angeles, CA USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 NOTATIONS

In this paper, we model a *temporal knowledge base* as a knowledge base where each triple has been augmented with a time stamp indicating the triple's occurrence at that time. Formally, a temporal knowledge base is $TKB = \{(s, r, o, t) \mid s, o \in \mathcal{E},\ r \in \mathcal{R},\ t \in \mathcal{T}\}$, where $\mathcal{E}$ denotes the set of entities, $\mathcal{R}$ denotes the set of possible relations in the ontology, and $\mathcal{T}$ is the set of all timestamps in the knowledge base. In this model, when TKB is sliced by the $\mathcal{T}$ dimension, each slice is the state of the time-invariant KB at t_i. This model fits well on event knowledge bases since the extraction model used is guided towards creating a digest of daily news instead of creating a global view of events.

3 EVENT SEQUENCE MINING

First-order horn clauses are one of the most common observable feature models built on top of Knowledge bases. Given a time-invariant KB of facts (subject, predicate, object), a first-order horn clause rules is a formula of the form

$$(\mathbf{w}, \mathbf{B_1}(.,.) \wedge \ldots \wedge \mathbf{B_n}(.,.) \Rightarrow H(x,y))) \tag{1}$$

Where the body of the rule $B_i(.,.)$ is a conjunction of relations from the KB, the elements in parenthesis are place holders for subjects and objects of instances of the relation in the KB, H is the head predicate of the rule and $\mathbf{w}$ is the likelihood of the rule being true. Previous work on mining inference rules from KBs [7, 12] assume horn clauses to be *connected*, predicates in the rule are all connected by transitivity and *closed*, every variable appears at least twice and in separate predicates. The problem of mining First-order rules resembles the association rule mining problem from transaction databases. The difference between the two is the presence of variables in the rule atoms, making the first-order rule mining problem more challenging.

First-order rules are specially useful in reasoning about the topology of the knowledge base. For example, one would expect generic rules of the form $r_1(x,y) \Rightarrow r_2(y,x)$ for example, $ParentOf(x,y) \Rightarrow ChildrenOf(y,x)$ to be true for (relation, inverse_relation) pairs because the inverse of a relation usually holds. Also $r_1(x,y) \Rightarrow r_2(x,y)$ rules would hold when the relation in the head is a generalization of the body. To add temporal constraints into first-order horn clauses, we resort to the concept of sequential rules from sequence databases. A sequence database [2] is a set of sequences $S = s_1, s_2, ..., s_n$ where each sequence s_i is an ordered list of transactions and each transaction is composed of itemsets. A sequential rule, is defined as an association rule $X \Rightarrow Y$ where X and Y are disjoint itemsets and the items in X occur *before* the items in Y in transactions of the same sequence. A TKB is a sequence of events ordered in the temporal dimension. An event sequence rule is a first-order horn clause where the predicates in the rule are ordered with respect to the temporal dimension. Even sequence rules have the form

$$\begin{aligned}&(\mathbf{w}, \mathbf{B_1}(.,.) : \mathbf{t_1} \wedge \ldots \wedge \mathbf{B_n}(.,.) : \mathbf{t_n} \Rightarrow H(x,y) : t_h),\\ &\quad t_1 <= \ldots <= t_n <= t_h\end{aligned} \tag{2}$$

The main elements of an event sequence rule are the same as first-order rules shown in equation 1. In addition, t_i is the associated time of the grounded atom in the knowledge base. For example the event sequence rule, $Threaten\_with\_sanctions(x,y) \Rightarrow Criticize(y,x)$ mined by our system, is interpreted as a sanction threat followed by denounce from the threatened entity.

There are two main approaches to mining sequential rules. CMRules [10] uses association rules mined from the corresponding transaction database and then prunes based on the sequence database. RuleGrowth [11] defines a series of expansion operations the can iteratively grow smaller sequential rules. Our approach is in the same spirit as CMRules; we rely on first-order horn clauses as rule templates and then apply sequential constraints.

3.1 Mining Algorithm

In order to mine event sequence rules, we use customize Ontological Pathfinding (OP), the state-of-the-art rule mining framework [7]. OP uses a relational knowledge base model to store KB facts and first-order rules in database tables. The intuition behind the data model is to group equivalent rules (e.g. rules where variables connect predicates equally for all rules) and store only the predicates that compound them. OP defines the following set of equivalent classes of rules

$$p(x,y) \leftarrow q(x,y) \tag{3}$$
$$p(x,y) \leftarrow q(y,x) \tag{4}$$
$$p(x,y) \leftarrow q(z,x), r(z,y) \tag{5}$$
$$p(x,y) \leftarrow q(x,z), r(z,y) \tag{6}$$
$$p(x,y) \leftarrow q(z,x), r(y,z) \tag{7}$$
$$p(x,y) \leftarrow q(x,z), r(y,z) \tag{8}$$

OP first generates candidate rules by traversing the schema graph of the knowledge base and obtain semantically sound rules. Instances of each rule type are stored in separate tables. For example, the table *Type-1* will store *head* and *body* predicates of candidate rules in separate columns. The grounded version of the rules is obtained by the 2-way join on head and body columns with the facts table and the corresponding subject and object as conditions shown in equation 3. The same process is repeated for all equivalence class table.

The grounded version of the rules table is used to calculate the interestingness scoring metrics for each rule. OP defines support as the number of times the body and head of the rule can be grounded.

$$\mathrm{supp}(\mathbf{B} \Rightarrow H(x,y)) := \#(x,y) : \exists z_1, \ldots, z_m : \mathbf{B} \wedge H(x,y). \tag{9}$$

Confidence is defined as the support of the rule over the number of times only the body of the rule is grounded

$$\mathrm{conf}(\mathbf{B} \Rightarrow H(x,y)) := \frac{\mathrm{supp}(\mathbf{B} \rightarrow H(x,y))}{\#(x,y) : \exists z_1, \ldots, z_m : \mathbf{B}}. \tag{10}$$

OP relies heavily on joining operations, known to be expensive in databases but it also allows rules to be evaluated in batches. In order to speed up the processing, OP uses a number of optimizations and parallelizations that can be found in the original manuscript.

In event sequence mining, we aim to find rules where the atoms in the rule occur in sequential snapshots of the temporal knowledge base. Imposing an order, can be achieved by adding an ordering constraint to the original join condition. For *Type-1* rules we add $TKB1.t <= TKB2.t$, where $TKB1$ and $TKB2$ are the temporal facts tables used in the 2-way join. The original join query contains the topology conditions for grounding the equivalence class with the

	SeqConf	Conf	Rule
(1)	0.75	0.68	Obstruct_passage$(x, y) \Rightarrow$ Demonstrate_or_Rally(x, y)
(2)	1.00	1.00	Provide_military_protection_or_peacekeeping$(y, x) \Rightarrow$ Receive_deployment_of_peacekeeping(x, y)
(3)	0.79	0.21	Increase_police_power$(z, x) \wedge$ Bring_lawsuit_against$(z, y) \Rightarrow$ Return,_release_person(x, y)
(4)	0.71	0.17	Threaten_with_sanctions$(x, z) \wedge$ Threaten_to_reduce_stop_aid$(z, y) \rightarrow$ Express_intent_to_cooperate(x, y)
(5)	0.70	0.13	Confiscate_property$(z, x) \wedge$, Refuse_to_release_person_property$(y, z) \rightarrow$ Complain_officially(x, y)
(6)	0.70	0.12	Employ_aerial_weapons$(x, z) \wedge$, Use_as_human_shield$(y, z) \rightarrow$ Employ_aerial_weapons(x, y)

Figure 1: Top event sequence rules mined for equivalence class 1-6 from ICEWS-2014 . $MinSeqSup = 100$, $MinSeqConf = 0.5$

rules table. The main problem of grounding rules with temporal constraint is that instances of the body and the head of the rule often exist multiple times in different snapshots of the knowledge base. For example, the atom $q(x, y) : t_p$ can exist in more than one snapshot, therefore any other atom $p(x, y) : t_p$ that appear with q in a rule with grounded $t_q <= t_p$ will make a valid instance. Therefore, we define the sequential support (seqSupp) similarly to equation 9 but with time constraint $t_1 <= \ldots <= t_n <= t_h$. This is the count of all grounded occurrences of the sequence in the body that contain a closing head afterwards. The latest occurrence of the head ground will then close all sequences in the body that occur before it and mark the rest as unclosed rules. The sequential confidence (seqConf) is defined as the ratio between the number of closed and unclosed rules. This definition of sequential support and confidence make the calculations easy to incorporate into the OP distributed join framework.

3.2 Interesting Event Sequence Rules

We applied our event sequence rule mining method on the ICEWS news dataset [18], a daily digest of news from around the world. In our experiments, we used subset of ICEWS corresponding to all events extracted during 2014. Figure 3.2 shows the top event sequence learned by our system. We compare sequential confidence with standard confidence for each of the equivalent classes. We compute the standard confidence from the time-invariant version of the dataset by removing time and deleting duplicate facts.

We note that the confidence for the first two equivalent classes is very close since they represent the inverse and class subsumption or predicates and therefore time does not change the top ranked rules. Nevertheless, we find other lower ranked rules that who interesting insights such as $Use\_tactics\_of\_violent\_repression(y, x) \Rightarrow Protest\_violently, \_riot(x, y)$ this rule suggests that the use of violence and repression leads to violent protests, but the inverse do not pass the minSup threshold.

We also note that the difference of confidence score of larger rules is much larger. The increase suggest that event sequences are able to find interesting sequences that would otherwise be ignored. For example, rule (6) from Figure 3.2 has very interesting instances such as $(Israeli\_Defense\_Forces, Citizen\_(Palestina), Hamas)$.The instance of this rule suggests along with other found examples that a group or country knowing that its enemy uses its civilians as human shield, they still use aerial weapons against them. The complete set of event sequence rules is available online[2].

[2]https://dsr.cise.ufl.edu/projects/sigmakb/

4 TEMPORAL EMBEDDING

In this section we will examine the potential of embedding methods for inference on Temporal Knowledge Bases. By introducing a temporal embedding method similar to the current state-of-the-art embedding methods used for static Knowledge Bases, we show that the same techniques can be generalized for use on Temporal data.

4.1 Background

Among methods used for KBC, e.g., path ranking [17], rule mining [7], or Markov random fields [14], *embedding models* have shown to be scalable to very large knowledge bases and proven to achieve state of the art performance in triple classification tasks [21] and link prediction [6].

One of the simplest and earliest embedding methods, TransE [6], is based on the idea that if two entities s and o are in a relation r, then their embedded vectors $\mathbf{s}$ and $\mathbf{o}$ should be close when translated by the relation vector $\mathbf{r}$, i.e., $\mathbf{s} + \mathbf{r} \approx \mathbf{o}$. TransE achieves high hits@10 on two of the well known datasets: FB15k a subset of Freebase [5] and WN18 derived from WordNet [20], used as link prediction benchmarks.

Bishan Yang, et al. show that many of the previous models can be generalized under a unified structure [25]. Under this generalized model, each method can be modeled as a combination of a basic linear transformation with a bilinear transformation and trained using a specific cost function. They also introduce DistMult, where each relation is modeled by a diagonal matrix and follows a bilinear scoring function $g(s, r, o) = \mathbf{s}\mathbf{W}_r\mathbf{o}^\intercal$. In addition they show that these latent representations can be used to mine first order logical rules with comparable performance to traditional rule mining methods.

For a recent review, please consult [24] by Quan Wang et al., who give a general overview of many recent models and training methods.

In spite of recent advances of many novel models, it has been shown that a vanilla base line with a lesser number of parameters can outperform most of the complex models [15]. This can be done by slightly modifying and good hyper parameter tuning of a model similar to DistMult.

Although there is some prior work done on using embeddings for temporal evolutions and time patterns in clinical and sensory data [8, 9], there has been little work on temporal knowledge bases. Inference in the presence of dynamic facts is not well understood. To the best of our knowledge the only related study is [23]. Know-evolve learns entity representations, a.k.a embeddings, for entities and relations and uses the score of each fact to modulate the intensity function of a point process which models each temporal fact.

They achieve better performance than other methods which don't take into account the dynamic structure of the datasets.

4.2 Chrono-Translation

In this paper we want to propose the simple idea of adapting similar embedding methods, which have been proven to be very effective on static knowledge bases, to learn representations for temporal knowledge bases.

One can think of time as transformations of the embedding space. At every time step we learn knew information about each entity e_t, this knew information can be represented by a transformation of each embedding $\mathbf{e}_{t+1} = \phi_t(\mathbf{e}_t)$. For simplicity we model ϕ_t as a linear transformation: $\mathbf{e}_{t+1} = \mathbf{e}_t \cdot \phi_t$. By unrolling the transformation back in time, we can model $\mathbf{e}_t = \mathbf{e}\mathbf{M}_t$. Although this may cause some loss of information, it makes the model much easier and faster to train and eliminates any need of recurrent models.

Following other embedding methods, we can compute a score for each quadruple as

$$g(s, r, o, t) = \mathbf{s}_t \mathbf{W}_r \mathbf{o}_t^\top = \mathbf{s}\,\mathbf{M}_t \mathbf{W}_r \mathbf{M}_t^\top \mathbf{o}^\top$$

where $\mathbf{W}_r \in \mathbb{R}^{d \times d}$ is a diagonal matrix representing relation r, $\mathbf{M}_t \in \mathbb{R}^{d \times d}$ is the linear transformation that projects each entity into the embedding space at time t, and $\mathbf{s}, \mathbf{o} \in \mathbb{R}^d$ are embedding representations of the subject and object of the fact, modeling the static information about each entity.

Our goal is to train the model such that the score of a valid quadruple is higher than an invalid one. We use a max-margin objective function with negative sampling:

$$\underset{\mathbf{M},\mathbf{W},\mathbf{E}}{\text{minimize}} \sum_{i=1}^{|\text{TKB}|} \sum_{n=1}^{N} \max\left(0,\ \lambda - g(q_i) + g\big(q_i'^{(n)}\big)\right) + \alpha \mathrm{L}_2(\mathbf{M}, \mathbf{W}, \mathbf{E})^2$$

where $\lambda, \alpha \in \mathbb{R}$ are constant hyperparameters, $\mathbf{M}, \mathbf{W}, \mathbf{E}$ represent the tensors corresponding to time translation, relation embeddings, and entity embeddings. q_i is a quadruple in TKB and $q_i'^{(n)}$ is the same quadruple where one of its entities is corrupted by replacing it with an entity is suitable at that position with respect to the relation, but does not appear anywhere in the TKB with the same other entity for any other time. N is the number of negative examples per quadruple.

Table 1: Relationship of Ukraine and European Union.

Subject	Relation	Object	Time
Ukraine	Reject plan to settle dispute*	E.U.	2014-01-28
Ukraine	Make pessimistic comment	E.U.	2014-02-19
Ukraine	Threaten	E.U.	2014-10-09
Ukraine	Praise or endorse	E.U.	2014-10-17
Ukraine	Appeal for economic aid	E.U.	2014-10-23
Ukraine	Express intent to negotiate*	E.U.	2014-10-27
Ukraine	Sign formal agreement	E.U.	2014-10-30
Ukraine	Diplomatic cooperation*	E.U.	2014-12-17
Ukraine	Host a visit	E.U.	2014-12-17

* Relations slightly shortened to fit.

Ukraine and the E.U are in a hostile relation, Ukraine then proposes to negotiate and the two entities interactions becomes friendly.

As the example in Table 1 illustrates, the data is inherently temporal and each entity in the KB may have different characteristics as time passes, unlike TransE and DistMult, our model can adapt to this changes by having a global embedding for each entity to capture the global information about the entity and allowing transformation with M_t to capture local dynamics over time.

4.3 Preliminary Results

Datasets We use the facts involving top 500 most frequent entities from year 2014 of ICEWS [18] as the dataset for our rudimentary experiments. We remove all duplicate entries and self-loops from the dataset. Table 2 provides the statistics regarding the data.

Table 2: Dataset statistics.

	$\lvert\text{TKB}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	#Train	#Dev	#Test
ICEWS	265K	500	247	212K	26K	26K

Setup Since our method is an embedding method and because of time and space limitations, we compare our method with two other purely embedding methods, TransE [6] and DistMult [25]. For a fair comparison, we tune each model to the new dataset. We perform hyperparameter search for all tested models: $d \in \{50, 75, 100\}$, $N \in \{2, 4, 8\}$, $\alpha \in \{0.5, 0.05, 0.005\}$, $\lambda \in \{1, 2, 4\}$. All models are trained using Adam Optimizer [16] with fixed learning rate of 0.005, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. To prevent overfitting we stop training when hits@10 stops improving on the development set.

All the experiments are performed on a Tesla K40c, using TensorFlow [1] and Python.

Results Since TransE and DistMult cannot take advantage of the temporal information, we convert each fact to a triplet by keeping the subject, relation and object. We then evaluate the performance of TransE and DistMult. Our results are shown in Table 3.

Table 3: Summary of experimental results.

	Hits@10
TransE	43.1
DistMult	54.8
ChronoTranslate	**63.2**

Our model achieves significantly better performance than both TransE and DistMult. While our model uses more parameters, increasing the number of parameters for other models did not improve their performance (it even slightly degraded).

5 CONCLUSIONS

We presented a relatively simple method to adapt current embedding methods and a new set of event sequence rules to handle temporal facts. Our methods takes into consideration the dynamic behavior of entities through time. First-order rules are temporally constrained to represent sequences and the embeddings use a linear transformation per time snapshot to reflect the changes of an entity onto its embeddings. The event sequence rules learn observable interesting patterns that the time-invariant version of the KB omits.

The embedding method shows significant improvements compared to plain models that can not use time. In future work we would use a larger dataset to learn rules and constraint the events to time spans such as in the same year, in the same month, etc. We like to also propose a more generic embedding model and use the learned embeddings to collaborate in the mining of event sequence rules.

ACKNOWLEDGMENTS

This work is supported by DARPA under FA8750-12-2-0348-2 (DEFT / CUBISM), and NSF IIS Award #1526753. This work is partially funded by a graduate fellowship from Fulbright.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on.* IEEE, 3–14.

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

[4] Joanna Biega, Erdal Kuzey, and Fabian M Suchanek. 2013. Inside YAGO2s: A transparent information extraction architecture. In *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 325–328.

[5] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data.* ACM, 1247–1250.

[6] 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.

[7] Yang Chen, Sean Goldberg, Daisy Zhe Wang, and Soumitra Siddharth Johri. 2016. Ontological Pathfinding. In *Proceedings of the 2016 International Conference on Management of Data.* ACM, 835–846.

[8] Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *Advances in Neural Information Processing Systems.* 3504–3512.

[9] Cristóbal Esteban, Oliver Staeck, Stephan Baier, Yinchong Yang, and Volker Tresp. 2016. Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *Healthcare Informatics (ICHI), 2016 IEEE International Conference on.* IEEE.

[10] Philippe Fournier-Viger, Usef Faghihi, Roger Nkambou, and Engelbert Mephu Nguifo. 2012. CMRules: Mining sequential rules common to several sequences. *Knowledge-Based Systems* 25, 1 (2012), 63–76.

[11] Philippe Fournier-Viger, Roger Nkambou, and Vincent Shin-Mu Tseng. 2011. RuleGrowth: mining sequential rules common to several sequences by pattern-growth. In *Proceedings of the 2011 ACM symposium on applied computing.* ACM, 956–961.

[12] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web.* ACM, 413–422.

[13] Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. *Artificial Intelligence* (2013).

[14] Shangpu Jiang, Daniel Lowd, and Dejing Dou. 2012. Learning to refine an automatically extracted knowledge base using markov logic. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on.* IEEE, 912–917.

[15] Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge Base Completion: Baselines Strike Back. *arXiv preprint arXiv:1705.10744* (2017).

[16] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Ni Lao, Tom Mitchell, and William W Cohen. 2011. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, 529–539.

[18] Jennifer Lautenschlager, Steve Shellman, and Michael Ward. 2015. ICEWS Event Aggregations. (2015). https://doi.org/10.7910/DVN/28117

[19] Kalev Leetaru and Philip A Schrodt. 2013. GDELT: Global data on events, location, and tone. In *ISA Annual Convention.* Citeseer.

[20] George A Miller. 1995. WordNet: a lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[21] 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.

[22] Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality.* 57–66.

[23] 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).

[24] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Transactions on Knowledge and Data Engineering* (2017).

[25] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR.* 12.