Resumé

Denne artikel omhandler **embedding metoder for temporal knowledge grafer og undersøger hvorvidt forskellige karakteristika af et query har en påvirkning på præcisionen** af link prediction. De undersøgte grafer er ICEWS, WikiData og YAGO3 og de undersøgte metoder er DE-TransE, DE-SimplE, DE-Dismult, ATiSE, TeRo og TimePlex. Dette bliver gjordt ved først at opstille tre overordnede hypoteser som omhandler udvalgte karakterstika. Hypoteserne bliver herefter enten be- eller afkræftet baseret på vores resultater. Disse resultater bliver derefter brugt til at lave en ensemble voting model der favoriserer specifikke modeller afhængigt af de karakteristika der findes i det pågældende query. Til sammenligning laves også en naiv ensemble voting model som ikke favoriserer nogen modeller.

Den første hypotese undersøger hvorvidt **koncentrationen af de kendte fakta igennem tidsperioder** har en påvirkning på ydeevnen af de forskellige modeller. Både samlet ydeevne og ydeevne for tidsforudsigelse bliver undersøgt her, og vi forventer at der er bedre ydeevne når der er en høj koncentration af kendte faktum i den tidsperiode som et givent query omhandler. Evalueringen af denne hypotese begynder med at opdele vores dataset i tre partitioner, en partition for tidsperioder med højest koncentration af kendte faktum, en med lav koncentration af kendte faktum, og en med mellem koncentration, som ikke bliver brugt. Dette sikrer os en stor forskel i koncentrationen af data i mellem det høje og det lave koncentrationssæt. Metoderne evalueres over disse to datasæt. Det viste sig at på ICEWS og WikiData er der bedre ydevne på datasættet med den lave koncentration, men på YAGO er der en stor forbedring på datasættet med den høje koncentration. Siden vores forventede resultat kun kunne observeres i én af de tre dataset betyder det at at hypotesen er afkræftet. Den samme undersøgelse blev også gjort for tidsforudsigelser specifikt. Her kan man se en stor forbedring i ydeevne på datasættet med høj koncentration i forhold til datasættet med lav koncentration, hvilket betyder at underhypotesen om tidsforudsigelser er bekræftet.

Den anden hypotese undersøger hvorvidt en samlet **gennemsnitsforudsigelse for tidsforudsigelser mellem flere modeller** generelt giver bedre resultater end tidsforudsigelser lavet af hver model individuelt. Dette kan lade sig gøre da tid er en kontinuerlig værdi og det derfor er muligt at finde et gennemsnit, i modsætning til andre elementer af grafen. Tidsforudsigelser af hver individuel model på ICEWS har nogenlunde samme middelafvigelse fra den forudsagte tid til den rigtige tid. Her fik den gennemsnitslige tid et bedre resultat end hver af de andre metoder. På de to andre datasæt har tre af modellerne betydelig værre middelafvigelse i forudsigelserne, så den gennemsnitlige forudsigelse får værre resultater af at inkludere dem i gennemsnittet. Dette betyder at denne hypotese er sand i nogle situationer, men at forudsigelserne er stærkt afhængeige af middelafvigelsen i forudsigelserne af de individuelle modeller.

Den tredje hypotese undersøger relationer og den omkringliggende struktur i grafen for at **kategorisere relationerne som symmetriske, antisymmetriske og inverse, og hvordan det påvirker ydeeven** af modeller som teoretisk ikke burde være i stand til at modellere de relationer, sammenlignet med modeller som kan. Til dette formål laves der seks nye datasæt, to for symmetriske og ikke-symmetriske, to for antisymmetriske og ikke-antisymmetriske, og to for inverse og ikke-inverse. Modellerne bliver evalueret på disse datasæt, og sammenlignet for deres ydeevne. Hvad vi fandt frem til var at DE-TransE, som ikke burde kunne håndtere symmetri, har betydeligt værre ydeevne på symmetriske relationer end de sammenlignede modeller. Udover det fandt vi at DE-DistMult, som hverken burdte kunne modellere antisymmetriske eller inverse relationer, kan modellere antisymmetriske og inverse relationer omtrent lige så godt som DE-SimplE. Generelt kan vi se at der er en forbindelse mellem kategorien af relationer, egenskaberne af modellerne og deres resultater med datasæt af forskellige relationskategorier. Dog har vi også resultater der viser at modeller kan få gode resultater på trods af deres teoretiske begrænsninger.

Resultaterne fra hypoteserne bliver brugt til at lave ORB-E som er vores **regelbaserede ensemblemodel**. I ensemblemodellen har metoder en vægt baseret på karakteristika af et query og deres ydeevne for det karakteriska. Vægten bestemmer hvor stor en påvirkning den metode har for ensemblemodellens evaluering af det pågældende query. Til at sammenligne med ORB-E laver vi også en naiv ensemblemodel, som giver hver model den samme vægt, så alle modeller har samme indflydelse for alle queries. Resultaterne viser at ORB-E er bedre end alle de andre metoder på ICEWS, YAGO, og WikiData. Den naive model har en ydeevne som er på cirka samme niveau som de andre individuelle modeller. Dette viser at de forskellige karakteristika af et query kan bruges til at forbedre ensemble metoder ved dynamisk tildeling af vægte.

Til sidst lavede vi også nogle **ablationsstudier** for at nærmere undersøge effekten af de forskellige karakteristika på ORB-E. De 12 forskellige studier er delt op i tre dele. Syv studier hvor et enkelt karakteristika er fjernet, fire hvor alle karakteristika er fjernet undtagen et og et enkelt hvor vægten af et enkelt karakteristika er blevet justeret. Forskellen i disse ablationsstudier er lille, men generelt kan vi se at forudsigelsesmålet som karakteristika har den bedste positive indflydelse på ORB-E. Udover det kan man se at det er på YAGO der er størst forskel på resultaterne af ablationsstudierne, hvilket antyder at her er de forskellige query-karakteristika vigtigere.

ORB-E: Ensemble Method with Query-Specific Weight Assignment Depending on Query Characteristics

Astrid Ipsen
aipsen18@student.aau.dk
Aalborg University
Aalborg Øst, Denmark

Jeppe W. Lindberg
jwli21@student.aau.dk
Aalborg University
Aalborg Øst, Denmark

Jonas C. Lindberg
jlindb18@student.aau.dk
Aalborg University
Aalborg Øst, Denmark

Abstract

The performance of link prediction on Temporal Knowledge Graphs (TKGs) has improved in the last decade via development of several new and diverse Knowledge Graph Embedding (KGE) methods. In this paper, the strengths and weaknesses of several temporal KGE methods are examined, with focus on the temporal data density of the overall Knowledge Graph (KG) and temporal properties of relations determined by the structure of the surrounding KG. The results of this analysis are used to create a new ensemble voting method ORB-E with query-specific model weights based on the characteristics of the query and model performance relative to that characteristic. The characteristics are target of prediction, temporal data density, properties of relations and overall scores of models. The rules that determine the weights of each model reflect which query characteristics have the most importance for different methods, datasets, and overall. We find that prediction target is the most influential characteristic, and that query-specific weight assignment has better performance than a static weight assignment. Link prediction preformance is increased for time prediction on temporally dense data but not other prediction targets and the theoretical expressivity of methods does not always reflect the actual performance of the models. Additionally, the domain of time predictions is analyzed to determine their accuracy under different circumstances and we find that most methods are not capable of predicting time within an acceptable error margin. A strategy that utilizes the continuous nature of time information and combines predictions of multiple models is presented, and can be used to improve time predictions when models have equal precision.

Code: https://github.com/cs-23-mi-10-01
Date: 15 June, 2023

1 Introduction

A Knowledge Graph (KG) is a data structure that stores multi-relational data, typically in the form of a directed, edge-labelled graph, where nodes represent entities and edges represent relations. Information is stored as facts of format (h, r, t), where elements h and t are entities and element r is a relation. A Temporal Knowledge Graph (TKG) is a KG where the facts have been extended with a time information element τ, either as a singular timestamp or as a timespan. A query is a fact with a missing element. A Knowledge Graph Embedding (KGE) is a low-dimensional, numerical representation of a KG, and facilitates link prediction, which aims to find the missing fact element of a query. This makes up the core of Question Answering (QA) systems, as it is the main way to infer facts in a KG. The result of link prediction on a query is a ranked list of most probable answers and the quality of an embedding can be determined by the average rank of the correct answers [Hogan et al. 2021].

The goal of this paper is to analyze the characteristics of link prediction queries to find which characteristics suit which types of models, enabling the construction of an ensemble method where models are prioritized according to the analysis findings and characteristics in the query. These characteristics depend on the information available in the elements of the query, as well as the structure of the KG and the available temporal information.

This work is a continuation of our previous work [Ipsen et al. 2022], where we investigated characteristics of selected KGE methods and their influence on link prediction. In the previous work, we found indications that the quality of the predictions is highly dependent on the target of the prediction, which we in this work examine more thoroughly as a preliminary experiment by testing on multiple datasets and splits of those datasets.

We formulate hypotheses on the role of temporal data density in link prediction results, potential of combining time predictions of models and methods' ability to model relations with different properties. Temporal density refers to the amount of facts within a timespan, and datasets are split into sparse and dense partitions, enabling us to examine model performance depending on the temporal density. The possible relation properties are symmetric, anti-symmetric and inverse, and each relation can have more than one assigned property, based on the structure and temporal information of the surrounding graph. Finally, the findings are utilized in the construction of an ensemble model ORB-E with dynamic weights across different models, depending on the characteristics of the query. It is evaluated against a naive version that assigns the same weight to each included model. Additionally, we make use of the continuous nature of timestamps to examine the degree of precision in time predictions to determine if they are sufficiently accurate to be used in QA systems, and present an approach of how to improve precision using joint selection of most probable timestamps over several models.

Our contributions are an in-depth analysis of query characteristics, and how they affect the performance of link prediction for each KGE method. We perform link prediction on both entities, relations and timestamps, of which only entities are standard practice. Time and relation predictions are both a focus in this work, which is not widely studied in other articles. We evaluate accuracy of time predictions, which is also not widely studied. Furthermore, the results gained from this analysis are utilized to create a new ensemble method called ORB-E, which yield better results than all individual models across all datasets.

The notation used is documented in section 2 and the related work is examined in section 3. Section 4 contains an in-depth explanation of our hypotheses, the intuition behind them and how we intend to examine them. Our selection of methods and datasets are explained in section 5 and section 6 respectively. The experiments not conducted as part of a hypothesis are covered in section 7 and the experiments conducted to evaluate hypotheses are covered in section 8. Details about the ensemble method and how it uses the results are explained in section 9. The discussion, conclusion and future prospects of our work are detailed in section 10, section 11, and section 12 respectively.

2 Background and Notation

Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ represent the set of all **entities**, **relations** and **timestamps** respectively. For simplicity, $\mathcal{T}$ is discretized to a suitable granularity (days for ICEWS, years for WikiData and YAGO), and represented by a continuous and unbroken ordered sequence of unique integers. In addition, the *missing timestamp* $\varnothing$ is also contained in $\mathcal{T}$. **Facts** f are represented as (h, r, t, τ), where $h, t \in \mathcal{E}$, $r \in \mathcal{R}$ and $\tau \in \mathcal{T}^2$. τ is a **timespan** consisting of two timestamps, where $\overleftarrow{\tau}$ refers to the first timestamp, and $\vec{\tau}$ refers to the last. f_h is the head element, f_r is the relation, f_t is the tail, and f_τ is the timespan of fact f. Let ζ represent all true facts in a world, and ζ' represent all false facts. Let $\mathcal{G}$ be a **temporal knowledge graph**, which is a subset of ζ, and contains facts $f \in \mathcal{G}$. Let $\eta_r \subset \mathcal{G}$ represent all facts in the KG where r is the relation in the fact.

A relation r is **temporally symmetric** if $(e_1, r, e_2, \tau) \in \zeta \Leftrightarrow (e_2, r, e_1, \tau) \in \zeta$ for some entities $e_1, e_2 \in \mathcal{E}$ and timespans $\tau \in \mathcal{T}^2$. A relation r is **temporally antisymmetric** if $(e_1, r, e_2, \tau) \in \zeta \Rightarrow (e_2, r, e_1, \tau) \in \zeta'$ for some entities $e_1, e_2 \in \mathcal{E}$ and timespans $\tau \in \mathcal{T}^2$. A relation r^{-1} is the **temporally inverse** of relation r if $(e_1, r, e_2, \tau) \in \zeta \Leftrightarrow (e_2, r^{-1}, e_1, \tau) \in \zeta$ for some entities $e_1, e_2 \in \mathcal{E}$ and timespans $\tau \in \mathcal{T}^2$.

Each KG is split into a number of **train** sets R_i, **validation** sets V_i and **test** sets T_i. No facts are shared among the sets within the same split $R_i \cap V_i = \emptyset$, $V_i \cap T_i = \emptyset$, $R_i \cap T_i = \emptyset$, but all facts are contained in the sets $R_i \cup V_i \cup T_i = \mathcal{G}$.

The quality of models is evaulated using the Mean Reciprocal Rank (MRR) metric, which is a real number between 0 and 1, and higher is better. We explicitly define significant differences in MRR scores, for use in hypothesis evaluation. We define the MRR score b to be significantly higher than the MRR score a if $b > a + (1-a) \cdot 0.08$. Conversely, we define the MRR score b to be significantly lower than the MRR score a if $b < a - (1-a) \cdot 0.08$.

3 Related Work

This paper is a continuation of [Ipsen et al. 2022], where different embedding methods are analyzed through an exploration of the quality of results when the models make link predictions. The main observation made from that analysis is that the most influential contributing factor that influence the quality of the results is the relations and the structure of the KG that surrounds them. This is the reasoning for what this paper concerns, and this paper serves to further examine the structure of the KG surrounding the relations, increase reliability of the findings from the previous paper, and find other contributing factors that can influence the quality of the results.

As this paper is a direct continuation of [Ipsen et al. 2022], the related work of that paper is also applicable here. It covers the different categories of TKG embedding methods, namely transformation, tensor decomposition, and neural network. Transformational methods use geometric functions to score facts in the embedding. Tensor decomposition methods use tensor and eigenvector products to decompose tensors into low-dimentional representations that is used to score the facts. Neural network methods use a number of learned layers to score facts based on the numerical representation of the elements of the facts. We have been unable identify an influential, recent, and temporal neural network method, which we could replicate the results of and therefore neural network methods are not included in this study. Influential transformational methods include RotatE [Sun et al. 2019], TeRo [Xu et al. 2020], and ChronoR [Sadeghian et al. 2021]. Influential tensor decomposition methods include TNTComplEx [Lacroix et al. 2020], TimePlex [Jain et al. 2020], and ATiSE [Xu et al. 2019]. Influential neural network methods include TFLEX [Lin et al. 2022] and Re-Net [Jin et al. 2019]. A temporal model agnostic method takes an existing non-temporal embedding method and modifies it, such that temporal information is added to the embeddings. Influential model agnostic methods include diachronic embeddings [Goel et al. 2019] and time aware representations [García-Durán et al. 2018].

Temporal accuracy metrics have previously been defined for scoring functions using a measure that compares absolute distances between elements of intervals [Surdeanu 2013], and using bounding boxes to evaluate the intersections and hulls between intervals [Jain et al. 2020]. These metrics are mostly made for scoring functions and not statistical evaluation of time predictions, as they represent the scores as a measure that cannot be evaluated directly by humans, and it is difficult to interpret if the results are usable in a QA context from these metrics alone.

Relation properties such as symmetry, antisymmetry, and inversion have been defined in non-temporal contexts [Schmidt 2010]. These properties define the structure of the KG surrounding the relation, and some methods are incompatible with some of these properties [Chami et al. 2020; Gregucci et al. 2023].

4 Hypotheses

We formulate the following three hypotheses on properties of TKG queries. They are constructed such that the results quantify the strengths and weaknesses of each method such that they can be used in the construction of an ensemble method. The hypotheses concern prediction quality, which is measured using MRR score. For simplicity's sake only MRR is used as evaluation metric for the hypotheses.

4.1 Hypothesis on Time Density

As a dataset has varying amounts of data in different time periods, we considered whether the prediction quality is affected by the temporal density of the data. This led to the hypothesis:

> **H1:** Prediction quality is significantly higher on queries where the timestamps are in a temporally

dense partition of the training dataset, compared to a sparse partition.

The significance of results refer to differences in MRR scores that are above a certain threshold, as defined in section 2.

H1 will be examined by creating two non-continuous partitions that contain test facts, a sparse partition where the temporal density is low and a dense partition where the temporal density is high. The two partitions should have approximately the same number of facts in order to make fair comparisons between them. These partitions are illustrated for YAGO in figure 1. Intuitively, we expect the MRR score to be higher in the dense part of the dataset, as the amount of available data at each point in time is larger.

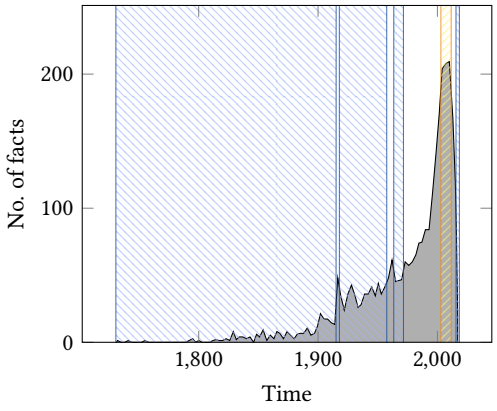

Figure 1: Dense and sparse partitions on YAGO. Dense partition marked in yellow, sparse in blue. Facts from before year 1700 are not included.

Additionally, we hypothesize that temporal density greatly impacts time predictions:

> **H1-A:** Prediction quality of time predictions is significantly higher on dense data partitions, than on sparse data partitions.

This subhypothesis is created from an intuition that the possibility of making a time prediction that is closer to the correct answer is higher when the data is spread over a shorter period of time.

Construction of the datasets requires the function *range* and *num*. The function *range* maps a timespan to a set of timestamps $\mathcal{T}^2 \rightarrow \mathcal{P}(\mathcal{T})$, where $\mathcal{P}(\mathcal{T})$ is the power set of $\mathcal{T}$. It is defined as

$$range(\tau) = \{i \in \mathcal{T} \mid \overleftarrow{\tau} \leq i \wedge i < \vec{\tau}\} \tag{1}$$

where τ is a timespan, $\overleftarrow{\tau}$ is the beginning timestamp and $\vec{\tau}$ is the ending timestamp. The result of $range(\tau)$ is a set of timestamps that contain all timestamps between the beginning timestamp and the end timestamp of τ. The beginning timestamp and all intermediate timestamps are included, but not the end timestamp.

The *range* function is then used to define the function *num*, which maps a timespan to a natural number $\mathcal{T}^2 \rightarrow \mathbb{N}$. This function describes how many facts in $\mathcal{G}$ that are contained in the timespan. The function is defined as

$$\begin{gathered} num(\tau) = |N| \\ N = \{f \in \mathcal{G} \mid range(\tau) \cap range(f_\tau) \neq \emptyset\} \end{gathered} \tag{2}$$

where N is the set of all facts where the timestamp overlaps with τ.

For every fact $f \in \mathcal{G}$ we find the number of facts $num(f_\tau)$ in the timespan f_τ of f, and arrange the values in a sorted sequence of numbers O. The sparse partition T_P of test set T is defined as the subset of T where the number of facts that falls within the timespan of f_τ is lower than p for all $f \in T$:

$$T_P = \{t \in T \mid num(t_\tau) < p\} \tag{3}$$

where p is the 25^{th} percentile value of O. The dense partition T_D is similarly defined as a subset of T where the number of facts that falls within the timespan of f_τ is higher than d for all $f \in T$:

$$T_D = \{t \in T \mid num(t_\tau) > d\} \tag{4}$$

where d is the 75^{th} percentile value of O. Note that by these definitions the partitions are not continuous.

The MRR score of each method is calculated over the T_D and T_P test sets, on several datasets. For both hypotheses, if the score is significantly higher in T_D than T_P, then the hypothesis is deemed true.

4.2 Hypothesis on Joint Timestamp Selection

We observed that timestamps are a continuous value and therefore has varying degrees of error which can be measured using Mean Average Error (MAE). As such it is possible to find the average between a number of different timestamps enabling us to combine the answers of models, where the result is an average of the answers. From this we have created the hypothesis:

> **H2:** Time predictions produced by taking the mean average of all models have a lower MAE than time predictions produced by each individual model.

To evaluate the models individually, as well as to compare them with the average results of all models, we use MAE between the predicted timestamp and the correct timestamp, over all time prediction tasks. It is defined as

$$MAE(T, m) = \frac{1}{|T|} \sum_{f \in T} |\overleftarrow{p_\tau} - \overleftarrow{f_\tau}| \tag{5}$$

where T is a test set containing only prediction tasks targeting the first timestamp, m is a model, p_τ is the timespan of the predicted fact of model m when evaluating f, and f_τ is the timespan of f.

We evaluate the MAE of the individual models, to see how precise they are at making time predictions. To our knowledge, this way of evaluating time predictions has no precedent in other papers. As such, we will instead evaluate whether the results are within an acceptable range for potential use in QA systems.

If the MAE of the joint timestamp results is lower than the MAE of each individual model, the hypothesis is deemed true.

4.3 Hypothesis on Relation Properties

This hypothesis is inspired by the fact that some embedding methods are unable to handle specific properties.

> **H3:** Prediction quality of queries with certain temporally constrained relation properties is significantly higher on methods, which theoretically can model those properties, than methods which cannot.

For this hypothesis we compare DE-TransE, DE-DistMult and DE-SimplE as they resemble one another and have varied combinations of relation properties that they can and cannot model. For more information on the selected methods see section 5.

The relation properties selected for analysis are symmetry, antisymmetry, and inversion. Reflection, composition and hierarchy were considered but disregarded. Reflection was disregarded because all considered methods are capable of modeling it and because we found no relation with that property in any of the datasets. Composition was disregarded because composition is made up of multiple facts, and as such it is more affected by incomplete data than other relation properties. Additionally, we found no occurences of composition in the data, possibly due to aforementioned vulnerability. Hierarchy was disregarded as we were unable to identify a KGE method which was both capable of modeling that relation property and temporal.

This hypothesis is divided into several subhypotheses specific to each relation property.

> **H3-A:** DE-TransE has worse performance than DE-DistMult and DE-SimplE on link prediction tasks in which the relation in the queries has the **symmetric** property.

As TransE is a straightforward translational method in Euclidian space, it is incapable of modelling symmetry [Goel et al. 2019]. Theoretically, DE-TransE has the same limitation with temporal symmetric relations, which are relations that happen simultaniously between two entities. An example of such a relation could be *is_married*. The problem is illustrated in figure 2. The purpose of H3-A is to empirically analyze the practical impact of this theoretical limitation.

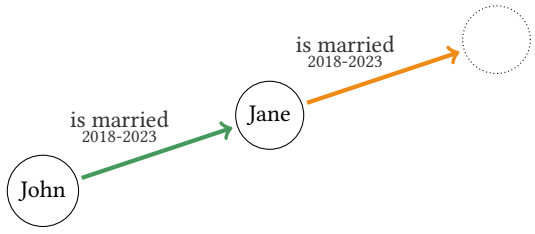

Figure 2: Illustration of DE-TransE embedding of the relation *is_married*. John is married to Jane, which is modelled with a translation from John to Jane (green), but when the same translation is applied to Jane the result is unknown (yellow).

> **H3-B:** DE-DistMult has worse performance than DE-TransE and DE-SimplE on link prediction tasks in which the relation in the queries has the **antisymmetric** property.

DistMult uses pairwise interactions in diagonal matrices, and as such cannot model edge direction, which in turn means it cannot model antisymmetry [Goel et al. 2019]. Theoretically, DE-DistMult has the same problem, but for temporal antisymmetric relations, which are relations where an entity performs an action to another entity and that other entity cannot do the same action back in the same timespan. An example of such a relation is *owes money*, as two people cannot owe eachother money as one debt will cancel out the other. The problem is illustrated in figure 3. The purpose of H3-B is to empirically analyze the practical impact of this theoretical limitation.

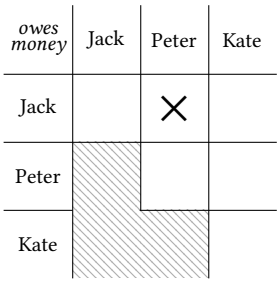

Figure 3: Illustration of DE-DistMult embedding of the relation *owes money*. The relation exists between Jack and Peter, but it is unknown who owes who money, as DistMult cannot model the hatched cells.

> **H3-C:** DE-DistMult has worse performance than DE-TransE and DE-SimplE on link prediction tasks in which the relation in the queries has the **inversion** property.

As DistMult cannot model edge direction, it also cannot model inversion [Goel et al. 2019]. Theoretically, DE-DistMult has the same problem, but for temporal inverse relations, which are pairs of relations, where when one entity relates to another with one type, they are also related in the opposite direction with the other type, both relations happening in the same timespan. An example of such a pair is the relations *host_a_visit* and *make_a_visit*. The purpose of H3-C is to empirically analyze the practical impact of this theoretical limitation.

To analyze these subhypotheses, the test sets are divided into sets of facts that contain only relations of that type and sets of facts that contain all but that relation. The first step is to assign a number of soft labels to each relation, using a function for each relation property that maps each relation to a real number $\mathcal{R} \longrightarrow [0, 1]$. We define a threshold for each of the relation properties, and classify each relation as being symmetric, antisymmetric, and/or inverse if the soft label for that relation is higher than or equal to the threshold. Alternative approaches are discussed in subsection 10.1. This function describes how many facts fulfill the requirements for

that relation. The reason soft labels are used is that TKGs are not complete, and some relations might be missing from the graphs.

The symmetry soft label for relation r is

$$sym(r) = \frac{|S|}{|\eta_r|}$$
$$S = \{(e_1, r, e_2, \tau) \in \eta_r \mid (e_2, r, e_1, \tau) \in \eta_r\} \tag{6}$$

where $\eta_r \subset \mathcal{G}$ is the set of all facts in $\mathcal{G}$ where the relation is r, $e_1, e_2 \in \mathcal{E}$, and τ is a timespan.

The antisymmetry soft label for relation r is

$$asym(r) = \frac{|A|}{|\eta_r|}$$
$$A = \{(e_1, r, e_2, \tau) \in \eta_r \mid (e_2, r, e_1, \tau) \notin \eta_r\} \tag{7}$$

The inversion soft label for relation r is

$$inv(r) = \max_{r^n \in \mathcal{R}\setminus\{r\}} \frac{|I_{r^n}|}{|\eta_{r^n}|}$$
$$I_{r^n} = \{(e_2, r^n, e_1, \tau) \in \eta_{r^n} \mid (e_1, r, e_2, \tau) \in \eta_r\} \tag{8}$$

where $\mathcal{R}\setminus\{r\}$ is $\mathcal{R}$ without the element r. $inv(r)$ finds the r^n with the highest percentage of instances where there for a fact (e_1, r, e_2, τ) exists a different fact (e_2, r^n, e_1, τ).

Any relation can be classified with any number of relation properties. The threshold for symmetry is 0.8, for antisymmetry is 1.0, and for inverse is 0.8. The set of facts where the relation is symmetric (T_S), as well as the set of test facts where the relation is non-symmetric (T'_S), on test set T is defined

$$T_S = \{f \in T \mid sym(r) \geq 0.8\}$$
$$T'_S = \{f \in T \mid sym(r) < 0.8\} \tag{9}$$

Similarly, the antisymmetric test facts T_A, the non-antisymmetric test facts T'_A, the inverse test facts T_I, and the non-inverse test facts T'_I are defined, using their respective thresholds.

The models are evaluated over these property specific test sets for each dataset and compared to one another. The difference between the score of the test set of a relation property and the test set without that relation property is worse for a model that cannot express that relation property than the difference between those scores for models that can express that property. If this is the case the hypothesis is deemed true.

5 Methods

The selected methods are of DE-TransE, DE-DistMult, DE-Simple, ATiSE, TeRo, and TimePlex. In table 1, an overview of what each method is designed for is provided. Methods have been selected such that they are diverse in the relation properties that they are theoretically capable of modelling and overall approach. In this section, a general outline of the method approaches are given.

DE is a model-agnostic method, meaning it adapts existing non-temporal methods to the use of temporal information. Here we use it on the methods TransE, DistMult and SimplE.

DE-TransE is based on the TransE [Bordes et al. 2013] method, a method used as a benchmark for many embedding approaches. TransE embeds entities and relations as vectors in Euclidian space, models relations as a translation between head and tail entity and scores the facts by Euclidian distance calculation. TransE is unable to model symmetry as that would require embedding both entities with the same vector and the symmetric relation to be a 0 distance vector. This would make it impossible to differentiate between the entities in all other relations.

Table 1: Overview of method and characteristics. T: Transformation, TD: Tensor decomposition.

Method	Category	Symmetry	Antisymmetry	Inversion
DE-TransE	T	×	✓	✓
DE-DistMult	TD	✓	×	×
DE-SimplE	TD	✓	✓	✓
ATiSE	TD	✓	✓	✓
TeRo	T	✓	✓	✓
TimePlex	TD	✓	✓	✓

DE-DistMult is based on the DistMult [Yang et al. 2015] method, a method that like TransE uses a single vector to embed each relation and entity. Unlike TransE it scores facts as summations of element wise products of the embedding vectors. As the ordering of elements in the element-wise product does not affect the result, DistMult cannot model edge direction, which in turn means that it cannot model antisymmetry or inversion.

DE-SimplE is based on SimplE [Kazemi and Poole 2018] which in turn is based on Canonical Polyadic [Hitchcock 1927] embedding. SimplE embeds relations and entities using two vectors each. Relation embeddings have a vector r for forward relations and a vector r^{-1} for reverse relations. Entity embeddings have a vector h for head entities, and a vector t for tail entities. The score function is then defined as the average of the element wise product of the embeddings of the elements in two facts, namely (h, r, t) and (t, r^{-1}, h).

Methods modified with DE [Goel et al. 2019] are extended such that the base method includes temporal information by making the entity embeddings dependent on time. Each model has a part of their entity embedding vector values that are impacted by timestamp. DE adds two learnable embedding vectors for each entity, one for dependency on time, and one for a bias. The resulting entity embedding values are found by multiplying the time-dependent vector values with the timestamp, adding them to a bias value, using an activation function on them, and then multiplying them with the non time-dependent embedding. The part of embedding vector values that are not dependent on the timestamp are simply the unaltered original embedding vector values.

ATiSE [Xu et al. 2019] is a method that uses additive time series decomposition to model changes and uncertainty in the data over time. The additive time series is a combination of a linear function, a sine function and a random function, to describe trends, seasonal changes and noise respectively. Each entity and relation has a seperate time series to represent how those elements change over time, following the patterns that those elements demonstrate. Entities and relations are embedded as vectors, and they follow Gaussian probability distributions. The score function considers the difference between the probability distribution of entities in head and tail positions of a fact, and their similarity to the probability distribution of the relation of the fact at a certain time. To compare the probability distributions, ATiSE uses Kullback–Leibler divergence.

TeRo [Xu et al. 2020] embeds entities and relations as vectors in complex space and uses element-wise rotations in complex space to model entities at specific timestamps. The scoring function is defined as the distance between the time-specific head entity plus the relation, and the conjugate of the time-specific tail entity.

TimePlex [Jain et al. 2020] is a model that embeds entities and timestamps in complex vector space, and each relation with three vectors in complex space. The first relation embedding vector represents a relation that is true for a head entity at a specific timestamp, the second relation embedding represents a relation that is true for a head entity and a tail entity, the third relation represents a relation that is true for a tail entity and a specific timestamp. The score function uses element-wise products between the head, relations, tail and timestamps, the latter embeddings in the products being the conjugate of those vectors. In addition, the score function also includes two additional scores, where one of them considers recurrences of events, the other considers time constraints between pairs of facts with certain relations.

6 Datasets

When selecting datasets for model evaluation we require the data to be at least partly temporal. Prevalence in related work is also valued highly, as it enables us to compare our implementation with the original implementation when both are evaluated over the same data.

Table 2: Statistics of datasets. Incomplete time refers to facts where at least one timestamp is missing. There are no facts where all timestamps are missing.

Dataset	ICEWS	WikiData	YAGO
# Facts	96730	40621	20507
# Entities	7128	12554	10623
# Relations	230	24	10
Time Period	2014	19–2020	-431–2844
# Incomplete Time	0	6134	8813

The selected graphs are ICEWS [Boschee et al. 2015], WikiData [Vrandečić and Krötzsch 2014], and YAGO [Mahdisoltani et al. 2015; Pellissier Tanon et al. 2020]. ICEWS is a number of fully temporal, event-based KGs specific to the domain of crisis alerts, where each graph contains facts from a single year. The granularity of time steps in ICEWS is one day, and the dataset contains one timestamp per fact. These timestamps are treated as timespans with the same beginning and end time. ICEWS is particularly noted for its consistency and uniform temporal distribution of data. WikiData and YAGO are partly temporal, general knowledge KGs and are not constrained to any certain time span. WikiData has a time step granularity of one year and YAGO has a granularity of one day. Both represent time as a timespan with a beginning and an end. WikiData is particularly noted for its size and YAGO for its focus on temporal data and ability to represent uncertainty in timestamps.

Specifically we use the datasets ICEWS14[1] , WikiData12K [Dasgupta et al. 2018], and YAGO11K [Dasgupta et al. 2018], henceforth referred to simply as ICEWS, WikiData and YAGO. Statistics of the datasets can be found in table 2. Date and month information has been removed from YAGO for the sake of uniformity, making the granularity of time steps one year for both WikiData and YAGO. All datasets have at least one timestamp for each fact. 3% of facts in WikiData are missing the beginning timestamp and 12% are missing the end timestamp. No facts in YAGO are missing the beginning timestamp and 43% of facts are missing the end timestamp.

[1]ICEWS14 exists in at least two versions: One with 7128 entities [García-Durán et al. 2018] and one with 12498 [Trivedi et al. 2017]. We use the first version.

7 Preliminary Experiments

As a preliminary step, the overall MRR scores of each model on every dataset are evaluated, to determine the general performance of the models. These scores are compared with the scores from the original papers to verify that our implementations have similar performance to the original implementations. These results are presented in Appendix G. TimePlex uses both beginning and end timestamp. As ICEWS only has one timestamp, TimePlex is trained on ICEWS with the available timestamp as the beginning timestamp and a blank timestamp as the end timestamp. As can be seen in Appendix F, the performance of TimePlex is significantly lower for time predictions on ICEWS than on other datasets. This might have been mitigated if the model was trained using the available timestamp as both the beginning and the end timestamp.

Additionally, some of the models have been evaluated using different splits of the datasets, to support the reliability of the observations from our previous work [Ipsen et al. 2022]. We discovered that for models trained on ICEWS, tail predictions are most accurate, followed by the head, relation, and then timestamp predictions. To investigate whether this is a coincidence, we created three new splits of ICEWS, trained DE-TransE, DE-SimplE, ATiSE and TeRo on those splits, and checked the quality of predictions of these models on the new splits. The same pattern emerged, confirming that the predictions on ICEWS follows this expected result on prediction targets.

The same experiment was conducted using the datasets WikiData and YAGO. These datasets and new splits all exhibited a different pattern from ICEWS for prediction targets. In both new datasets and all their splits, the relation prediction is most accurate, followed by the tail, head and then time predictions. This is likely because Wikidata and YAGO have considerably fewer relations than ICEWS, and it is therefore less likely to be incorrect. The results indicate that the performance is generally best when making tail predictions, followed by head, relation and then time, however if some fact element has a considerably lower amount of possible answers, that might increase the expected performance for predictions on that fact element.

The overall performance, as well as the performance when predicting on specific fact elements have been illustrated for each model and the original split of each dataset in Appendix F. The alternative splits have similar results.

8 Hypothesis Evaluation

This section contains the evaluation of hypotheses presented in section 4. We determine differences in MRR results to be significant according to the definition in section 2.

8.1 Time Density

Hypothesis H1 concerns the temporal density of datasets and we expect the results to be more accurate the more dense the data is. To evaluate it we consider prediction quality of models on dense T_D and sparse T_P partitions of test sets. These two partitions contain approximately the same number of facts, but the sparse partition is spread over a longer period of time than the dense partition. Overview and statistics of test sets can be found in Appendix B and Appendix C. Results can be seen in table 3.

Table 3: MRR scores on dense (T_D) and sparse (T_P) partitions of test sets. Significant results marked with ▲ or ▼.

	ICEWS			WikiData			YAGO		
	T_P	T_D	Diff.	T_P	T_D	Diff.	T_P	T_D	Diff.
DE-T	0.28	0.27	−.01	0.47	0.43	−.04	0.39	0.39	–
DE-D	0.38	0.37	−.01	0.43	0.40	−.03	0.28	0.34	+.06▲
DE-S	0.43	0.41	−.02	0.45	0.41	−.04	0.29	0.36	+.07▲
TR	0.45	0.44	−.01	0.53	0.46	−.07▼	0.28	0.34	+.06▲
AT	0.43	0.42	−.01	0.52	0.45	−.07▼	0.31	0.41	+.10▲
TP	0.38	0.35	−.03	0.29	0.22	−.07▼	0.17	0.15	−.02

In the results, we see a trend that predictions on ICEWS and WikiData are slightly better in the sparse partition compared to the dense partition, which contradicts the hypothesis. The difference is not considered significant for any results on ICEWS, but three results on WikiData are significant. It is also noted that the trend is consistent throughout the results for these datasets. It is worth noting that ICEWS has a particularly consistent density throughout the dataset causing the dense and sparse partitions to be more similar than in other datasets and therefore we did expect the difference in prediction quality to be smallest on this dataset. On YAGO results from two of the models contradict the hypothesis, and the results from the other four models support the hypothesis by a significant amount. These results resemble the expected results much better. As YAGO is the dataset with the largest temporal range it might also be the dataset that benefits the most from higher data density. Overall, the results indicate that **H1 is false**. However, as the results on one dataset do support the hypothesis and we have an indication that temporal density is more important the larger the temporal range then it is possible that a more detailed analysis would lead to a different conclusion.

The hypothesis H1-A concerns the temporal density of datasets specifically for queries where the prediction target is a timestamp. Once again, we expect the results to be more accurate the more dense the dataset is, but we also expect the difference to be more pronounced when only evaluating timestamp predictions. The results can be seen in table 4.

Once again we expect the differences to be smallest on ICEWS due to the similarities in temporal density between dense and sparse partitions. This is confirmed by the data where we see 0.01 as the biggest difference between sparse and dense partitions.

The DE methods have very low accuracy of timestamp predictions on the WikiData and YAGO datasets in general. While it is noted that the sparse partition yields a better accuracy than the dense one in half the cases, all scores are so low that it is not possible to determine which factors affected them and how, making them unsuitable for a detailed analysis.

Table 4: MRR scores for timestamp predictions on dense (T_D) and sparse (T_P) partitions of test sets. Significant results marked with ▲ or ▼.

	ICEWS			WikiData			YAGO		
	T_P	T_D	Diff.	T_P	T_D	Diff.	T_P	T_D	Diff.
DE-T	0.10	0.09	−.01	0.01	0.00	−.01	0.01	0.00	−.01
DE-D	0.09	0.08	−.01	0.00	0.00	–	0.00	0.00	–
DE-S	0.09	0.08	−.01	0.00	0.00	–	0.01	0.00	−.01
TR	0.17	0.18	+.01	0.26	0.29	+.03	0.18	0.25	+.07▲
AT	0.15	0.16	+.01	0.19	0.24	+.05	0.08	0.16	+.08▲
TP	0.02	0.02	–	0.17	0.25	+.07▲	0.05	0.12	+.07

TeRo, ATiSE and TimePlex yield moderately or significantly better results on dense than sparse partitions of WikiData and YAGO as expected. Similarly to the results for predictions on all prediction targets YAGO seems to be particularly affected by data density, but it is worth noting that results in general are better on WikiData.

As such, these results indicate that **H1-A is true** and that the quality of time predictions is greatly impacted by the temporal density of the data.

8.2 Joint Timestamp Selection

Hypothesis H2 concerns making time predictions by using the average of the best prediction of several models.

To evaulate this, the models are compared using the MAE score achieved when making time predictions. The MAE is in days for ICEWS, and years for WikiData and YAGO. The scores are also compared to the MAE score of predictions achieved when all models jointly select a timestamp answer. The results of this evaluation can be seen in table 5. Error distributions of the individual methods can be found in Appendix E.

Table 5: MAE of top time predictions of each model, and joint selection. The most and least accurate time in the interval prediction of TeRo and ATiSE is noted.

Model	ICEWS (days)	WikiData (years)	YAGO (years)
DE-TransE	90.51	1003.50	863.69
DE-DistMult	87.90	978.24	791.51
DE-SimplE	93.66	966.97	814.93
TeRo	129.75	76.93–102.53	358.14–642.29
ATiSE	137.39	85.39–153.25	94.57–579.43
TimePlex	122.87	**21.90**	**148.27**
Selection	**84.71**	513.83	389.76

When TeRo and ATiSE make time predictions, they attempt to predict both the beginning and end timestamp of the query, resulting in a time interval prediction. When contributing to the joint timestamp selection, a timestamp in the middle of the time interval is used instead.

The results show that the MAE score is lower in predictions on ICEWS when jointly selecting the answer timestamp. An average

error of 90+ days is very high in ICEWS, as this dataset only spans a year. This means the average error range is 6 months, which is half of the dataset. For TeRo, ATiSE and TimePlex, the result indicates that the predicted timestamp is random, which is supported by their error distribution in Appendix E. Joint timestamp selection seems to average out wrong answers in both directions for each method, and thereby achieves a higher score, supporting H2.

On WikiData TimePlex has the highest precision, and the difference between the MAE of the most precise model and the least precise model DE-TransE is very high at ~1000 years. The results on WikiData indicate that this dataset is difficult for the DE models to make time predictions on, and the MAE scores of these three models seem to indicate that the predicted answer is random, which is again supported by the error distribution in Appendix E. Joint selection achieves a score that is in the middle between the two extremes, and as it is decided using the mean average of the time predictions of all contributing models, the DE models make the results significantly worse.

For YAGO the same pattern as WikiData emerges. However TimePlex is significantly worse on this dataset and ATiSE has the best score when comparing the closest timestamp in its time span. The joint time prediction is once again impaired by the worst models.

The results show that the overall error of time predictions is high. The best MAE result on ICEWS is 84.71 days, which is an unacceptable average error in a QA context, as the purpose of queries on ICEWS is to make early predictions on critical events like military operations or civilian unrest. These need to be highly accurate to be useful. Similarly, having an average error of 94+ years on YAGO is unacceptable, as queries like birth dates and war periods need to be somewhat precise to be useful. On the other hand, TimePlex achieves an MAE score of 21.90, which is accurate enough to be useful in some contexts that do not require high accuracy, e.g. when asking about when technological ages like the industrial age began and ended. TimePlex is the only examined model that attempts to optimize time predictions, and as such it is encouraging that it achieves the only useful result, but it still only achieves it on one of the three examined datasets.

Overall, this indicates that **H2 is true** if the base models have approximately equal precision. A more complex voting mechanism when jointly selecting timestamps might yield better results, such as giving less weight to less accurate models, or using more than just the top scoring prediction for each model.

8.3 Relation Properties

Hypothesis H3 concerns relations with certain properties and their connection to model performance.

To evaluate this, the methods have been evaluated on test sets divided into a number of different relation properties, each test set containing no relation predictions.

For an overview of what each test set contains, see Appendix B. In Appendix C the number of facts in each test set is detailed, as well as the number of types of relations.

ICEWS is the dataset best suited for analysis of this hypothesis, as there is a higher number of relations, with more varied relation properties. All three sub-hypotheses of H3 refer to specific methods, and what they can and cannot represent. They are all diachronic embedding methods, and they will be compared to the other diachronic embedding methods, as they share the most characteristics with each other, but differentiate in the relations they can and cannot model.

To evaluate each subhypothesis we have analyzed and compared the performance of DE-TransE, DE-DistMult, and DE-SimplE on the different test sets, as can be seen in table 6. Some datasets have no relations with some properties and those results are therefore blank.

H3-A concerns symmetric relations. As DE-TransE cannot model symmetry but DE-DistMult and DE-SimplE can, we expect DE-TransE to achieve worse results on symmetric than non-symmetric relations, in relation to the results for DE-DistMult and DE-SimplE on those relations. On WikiData no relations met the requirements of the soft label and therefore there is no data for that dataset. On both ICEWS and YAGO, the symmetric relations seem to be simpler than the non-symmetric relations for the embedding methods to model – They all achieve a better performance on the symmetric dataset than the non-symmetric dataset. What can also be observed is that DE-TransE has less of an improvement between the symmetric test sets and the non-symmetric test sets compared to DE-DistMult and DE-SimplE, as was theorised in H3-A. This is illustrated for ICEWS in figure 4a. On ICEWS the performance of DE-TransE is similar across the symmetric test set and the non-symmetric, whereas the performance of the compared models is significantly better on the symmetric test set. On YAGO DE-TransE performs significantly better on symmetric than non-symmetric relations, but the performance of the compared models is improved

Table 6: MRR scores on partitions of test sets with (T_S, T_A, T_I) and without (T'_S, T'_A, T'_I) certain properties. Significant results marked with ▲ or ▼.

		Symmetry			Antisymmetry			Inversion		
	Method	T'_S	T_S	Diff.	T'_A	T_A	Diff.	T'_I	T_I	Diff.
ICEWS	DE-TransE	0.24	0.26	+0.02	0.23	0.32	+0.09 ▲	0.24	0.34	+0.11 ▲
	DE-DistMult	0.33	0.48	+0.15 ▲	0.35	0.38	+0.03	0.35	0.44	+0.10 ▲
	DE-SimplE	0.34	0.48	+0.14 ▲	0.36	0.39	+0.03	0.36	0.49	+0.13 ▲
WD	DE-TransE	–	–	–	0.11	0.14	+0.03	–	–	–
	DE-DistMult	–	–	–	0.12	0.14	+0.02	–	–	–
	DE-SimplE	–	–	–	0.12	0.14	+0.03	–	–	–
YAGO	DE-TransE	0.06	0.32	+0.26 ▲	0.32	0.06	−0.26 ▼	–	–	–
	DE-DistMult	0.03	0.63	+0.60 ▲	0.63	0.03	−0.60 ▼	–	–	–
	DE-SimplE	0.04	0.62	+0.59 ▲	0.62	0.04	−0.58 ▼	–	–	–

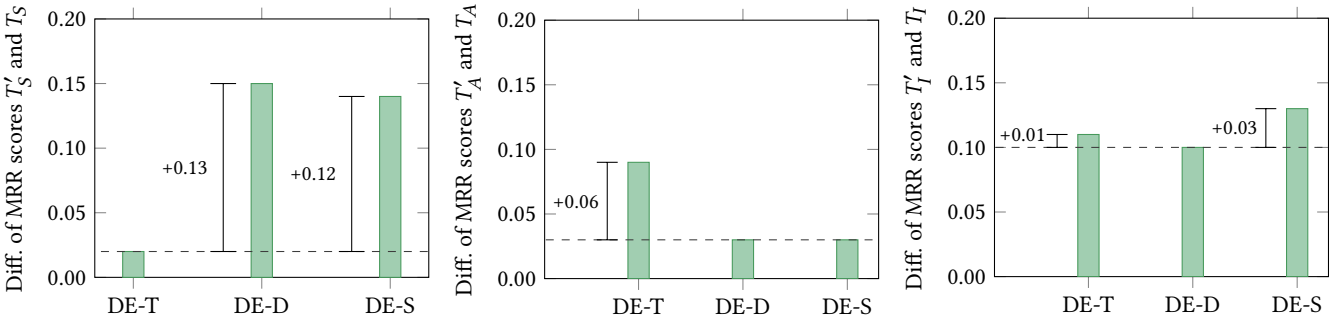

Figure 4: Comparison of differences in performance of models between test sets that contain a given relation property, and test set that do not contain that property in ICEWS.

by a larger margin. The results are not as strong as we initially anticipated, but they still indicate that DE-TransE performs worse on symmetric relations than other models do, compared to their performance on non-symmetric relations. This indicates that **H3-A is true**.

H3-B concerns antisymmetric relations. As DE-DistMult cannot model antisymmetry but DE-TransE and DE-SimplE can, we expect DE-DistMult to achieve worse results on antisymmetric than non-antisymmetric relations, in relation to the results for DE-TransE and DE-SimplE on those relations. On ICEWS only DE-TransE has significantly better performance on the antisymmetric relations than non-antisymmetric relations. On WikiData all models perform similarly and on YAGO all models perform worse on the antisymmetric relations. The differences in improvement for antisymmetry on ICEWS are illustrated in figure 4b. As the figure shows, only DE-TransE performs significantly better than DE-DistMult, and the performance of DE-DistMult is almost the same as DE-SimplE. We expected both DE-TransE and DE-SimplE to perform better across all datasets, but this is evidently not the case. The same pattern is evident on YAGO, but there are no differences in improvement on WikiData. Therefore, the results indicate that DE-DistMult perfoms similarly on antisymmetric relations and non-antisymmetric relations as other methods despite the theoretical disadvantage that DE-DistMult has, which indicates that **H3-B is false**.

H3-C concerns inverse relations. As DE-DistMult cannot model inversion but DE-TransE and DE-SimplE can, we expect DE-DistMult to achieve worse results on inverse than non-inverse relations, in relation to the results for DE-TransE and DE-SimplE on those relations. On WikiData and YAGO no relations met the requirements of the soft label and therefore there is no data for those datasets. The differences in improvement for ICEWS are illustrated in figure 4c. The comparison shows that DE-DistMult has higher performance on the inverse test set than the non-inverse test set, but that the other models improve by a larger margin, which is what we expected to see. The difference is however not very pronouced, and the performance is overall similar to the performance of DE-SimplE. These results are not remarkable enough to confirm the hypothesis, and therefore the results indicate that **H3-C is false**, however this result is only based on a single dataset, and therefore not very well supported.

Overall these results indicate that **H3 is true** for some methods and relation properties. This indicates that the performance of a model can in some cases be predicted by the theoretical limitations of the methods, but the model can also achieve good results despite them.

9 Ensemble Model

Using our findings from the hypotheses, we create Our Rule-Based Ensemble method ORB-E. The architecture is based off of the voting ensemble method presented by [Otte et al. 2022]. Additionally, a naive ensemble is also created as a baseline for performance comparisons with ORB-E. The models of both methods have been previously trained indidually, before they are included in the ensemble methods. The naive ensemble method uses voting [Mohammed and Kora 2023] where each method has the same weight when deciding the answer to a query. ORB-E uses voting as well, but uses rules to assign different weights to each model based on our findings and the information contained in a query. This way each method has a different level of importance, depending on the query.

9.1 Naive Voting Ensemble

The naive voting ensemble method assigns the same weight to each model. The queries in the test set are evaluated by each model. The answers from each model are assigned points based on the reciprocal rank, such that the highest scoring prediction has a score of 1/1, the second highest 1/2 and so on. The scores are then added together for each possible element that can be an answer to the query across all of the models. The element with the highest combined score is the answer of the naive ensemble method.

9.2 Rule-Based Voting Ensemble

The rule-based voting ensemble method ORB-E is similar to the naive one, except a specific weight distribution for the models is found for each query, assigning the most weight to the model that is most suited to answer that query.

Table 7: Evaluation over all prediction targets. Naive-E is our naive voting ensemble method and ORB-E is our rule-based. Best results are highlighted in bold.

	ICEWS				WikiData				YAGO			
Method	MRR	hits@1	hits@3	hits@10	MRR	hits@1	hits@3	hits@10	MRR	hits@1	hits@3	hits@10
DE-TransE	0.23	0.10	0.31	0.47	0.35	0.28	0.36	0.47	0.30	0.23	0.33	0.40
DE-DistMult	0.31	0.21	0.38	0.52	0.32	0.26	0.34	0.43	0.24	0.19	0.25	0.36
DE-SimplE	0.34	0.24	0.38	0.52	0.33	0.27	0.35	0.45	0.25	0.21	0.25	0.36
ATiSE	0.37	0.25	0.43	0.60	0.41	0.31	0.45	0.60	0.32	0.26	0.35	0.45
TeRo	0.41	0.30	**0.47**	0.62	**0.43**	0.33	**0.46**	0.62	0.30	0.19	0.36	**0.51**
TimePlex	0.25	0.17	0.28	0.40	0.25	0.14	0.27	0.49	0.18	0.11	0.19	0.31
Naive-E	0.34	0.26	0.37	0.50	0.42	0.33	0.45	0.59	0.29	0.24	0.32	0.39
ORB-E	**0.43**	**0.32**	**0.47**	**0.63**	**0.43**	**0.34**	**0.46**	**0.64**	**0.38**	**0.31**	**0.40**	**0.51**

The characteristics that affect the weight distribution are based on the results of the hypotheses presented in section 4 and evaluated in section 8, as well as the previous work by [Ipsen et al. 2022]. The score for each characteristic is calculated individually for each dataset. These characteristics are as follows:

- The overall model score, i.e. the MRR score a model achieves across all prediction targets.
- The density of the data in that timespan i.e. sparse or dense.
- The properties of the relation i.e. symmetry, antisymmetry and inversion, where each property is an independent characteristic.
- The false properties of the relation i.e. non-symmetry, non-antisymmetry and non-inversion. If a relation is not symmetric then it has the non-symmetric characteristic and vice-versa.
- The prediction target i.e. head, relation, tail, or timestamp.

For a total of 13 possible characteristics.

The formula for weight distribution is defined as:

$$weight(m, q) = \sum_{c \in C} \begin{cases} norm(s^c)_m \cdot d_{s^c} \cdot \psi_c & \text{if } c \text{ is a characteristic of } q \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

where $weight(m, q)$ is the weight of a model m for a query q. C is the set of all characteristics, s^c is a list of real numbers that contains the score for each model for characteristic c on a specific dataset, $norm$ is a min-max normalization function, $norm(s^c)_m$ returns the normalized score for model m, d_{s^c} is the absolute difference between the minimum and maximum value of s^c, and ψ_c is a constant variable that modifies the importance of characteristic c, and is used for controlling the ablation studies. ψ_c can be configured to give certain characteristics higher impacts.

The difference value d_{s^c} is used as a way to account for the impact of a characteristic. If all models have similar results for that characteristic, then it does not have a large impact on the score, and d_{s^c} will have a low value, and thereby reduce the effect that characteristic has on the weight distribution. ψ_c is set to 1 for all characteristics in ORB-E.

To find the final weight distribution for a query q, the value of $weight(m, q)$ for each model is normalized with the softmax function, in order to get a distribution that totals 1.

9.3 Ensemble results

The results of the ensemble models can be seen in table 7, along with the overall scores of other models. The naive method has rather average scores across all datasets. ORB-E has the best results in all metrics across all datasets. TeRo has the same score as ORB-E in a few cases like MRR on WikiData.

This indicates that the query characteristics investigated in this paper are able to use the advantages of several different models to improve the overall results of a link prediction task.

9.4 Ablation study

The ablation study is split into 11 different studies: Six where a single characteristic is disabled, four where only a single characteristic is enabled, and one where each relation property only gets one third of the weight. An example of the six studies where a characteristic is disabled, is the *no overall ablation* study disables the overall score when calculating the weight distribution. An example of the four single charateristic studies is the *only overall* study, where every characteristic other than the overall scores is disabled. For *only time density* and *only property* disabling all other characteristics means that some queries have no applicable characteristic e.g. 50% of the facts are not in either dense or sparse time partitions. In these cases the naive weight distribution is used. The study where each relation property is reduced to one third of the weight is conducted to balance out the impact of relation properties on the weight distribution, as there are three possible relation properties.

The scores for all the ablation studies can be seen in table 8. On ICEWS the *no property* study has the shared highest score. Here all the relation properties are disabled when calculating the weight, which indicates that relation properties do not contain useful information in this dataset. Additionally, when only the properties were used in the weight calculation in the *only property* study, the score is the second lowest among the ablation studies, which further supports this. However the other shared highest score is *no false property* ablation score, which means that it could be the false properties specifically that decrease the performance.

On WikiData we can see that *only property* and *only target* have the best performance. This trend can also be seen in that *no property* is worse than both *no true property* and *no false property*, and *no target* is worse than the original ORB-E. The two ablation studies with the worst performance are *only overall* and *only time density*. The *only overall* study is similar to the naive method, except the

static weights are assigned based on the overall score of each model. It has worse performance than ORB-E but better than Naive-E, indicating that dynamic weights improve performance. The *only time density* study is the only one across all datasets that is worse than its respective naive solution, which suggests that incorporating time density lowers the performance for WikiData, which supports the result of hypothesis H1, which indicated that time density and performance were not connected.

Lastly on YAGO the study *only target* has the best performance. This ablation study is consistently one of the top performing studies across the datasets, which suggests that this characteristic contains the most information. On YAGO we see the biggest differences in score, which indicates that the question characteristics are more important on this dataset than the others.

Table 8: Ablation MRR scores. Comparison scores relative to ORB-E.

Method	ICEWS		WikiData		YAGO	
Naive-E	0.343		0.419		0.295	
ORB-E	0.426		0.434		0.378	
no overall	0.425	−.001	0.435	+.001	0.383	+.005
no time density	0.426	–	0.435	+.001	0.379	+.001
no true property	0.426	–	0.433	−.001	0.378	–
no false property	**0.427**	+.001	0.431	−.003	0.375	−.003
no property	**0.427**	+.001	0.428	−.006	0.372	−.006
no target	0.425	−.001	0.433	−.001	0.368	−.010
only overall	0.426	–	0.421	−.013	0.343	−.045
only time density	0.420	−.006	0.417	−.018	0.356	−.022
only property	0.421	−.005	**0.436**	+.002	0.375	−.003
only target	0.426	–	**0.436**	+.002	**0.391**	+.013
1/3 property	0.426	–	0.431	−.003	0.375	−.003

The scores of these ablation studies are very similar to each other and ORB-E, so it is difficult to make any definitive conclusions about what characteristics of the query influence link prediction the most. As some ablation studies achieve better performance than ORB-E on different datasets, ORB-E should not be considered a static configuration, but should be adapted to a specific dataset by configuring the hyperparameters. We can however conclude that the target characteristic seems to be the most impactful as the performance is consistently decreased when this characteristic is excluded, and the study *only target* is consistently either better or as good as ORB-E. We also conclude that dynamic weight assignment improves performance as the static *only overall* study performs considerably worse than ORB-E on WikiData and YAGO and *no overall* study performs better than ORB-E on the same datasets, whilst barely impacting performance on ICEWS. Additionally, relation properties seem difficult to use as the study *only properties* achieves a better score than ORB-E on WikiData, but a worse score on the other two datasets.

10 Discussion

In this section we present some of the considerations made throughout the project, detail alternatives and explain the reasoning behind our choices.

10.1 Alternative Approaches to Relation Properties

Relation properties are determined using a soft label approach. This section details alternative approaches and their advantages and disadvantages.

Ideally, we would extract the properties of relations directly from metadata about the source graphs, as this would accurately depict the properties of the relations. However, most KGs do not contain this metadata, and none of the KGs used in this paper has complete metadata information, making this approach unfeasible.

Another alternative approach is manual assignment of properties based on notions about what each relation models. This approach risks misinterpreting the purpose of the relations, resulting in property assignment that do not reflect the data, and involves additional work to include each dataset.

We chose to do soft label assignment, as it depicts the relevant data, does not require a complete KG unlike hard label assignment, and makes it easy to add new datasets.

Based on the labels assigned to each relation, we considered using data materialization, to create more complete KGs. This would include modifying the dataset with new facts that can be inferred from the relation properties e.g. adding (e_1, r, e_2, τ) if (e_2, r, e_1, τ) exists and r has the symmetric property. We chose not to do this, as we still could not guarantee a complete KG after data materialization, and it would make it impossible to compare our results to other works that was learned on the original dataset.

The soft label thresholds were selected empirically. We identified some relations that we expect to have certain properties, and the thresholds were selected such that those relations were assigned the expected relation properties.

10.2 Errors in WikiData Dataset

During this project we discovered that the WikiData dataset contains multiple errors.

Timestamps in WikiData are formatted as yyyy-mm-dd and uses # in place of numbers where values are unknown. However, there are 54 instances where the timestamps contain only one or two characters in the year value. The data appears to be correct in most of these cases, but we have identified at least two instances of factual errors in these timestamps.

As the scale of the errors appears to be negligible and because identifying and correcting the errors would be both time consuming and prevent accurate comparison between the results of this and other papers, we elected to make no changes.

10.3 Time Predictions of DE Methods

The DE methods split a timestamp into three parts, year, month and day. This could be detrimental when doing time predictions, as the three values are predicted individually, however single month have the same importance as 31 days and one year is equal to 12 months. This decreases performance as if the month is predicted wrong, it gets a much larger error than if the day is predicted wrong. The error distributions in Appendix E indicate that this is the case. This could be alleviated by predicting time as a continuous value, such that each year is equal to 365. This would eliminate year and month having more importance than day.

10.4 Extendability of ORB-E

ORB-E is a configurable method that uses results from multiple models. As such it is possible to adjust the importance of each characteristic and extend the method with additional models and characteristics without retraining any of the models. Extending ORB-E with additional methods requires training a model of that method on the selected datasets, modifying the output to have the same format as the other models in ORB-E, and evaluating it over all specified test sets. Additional adjustments like adjusting weights of other models in relation to the new model are automatically performed by ORB-E. The complexity of ORB-E scales linearly with the number of models. Extending ORB-E with additional characteristics requires defining those characteristics, creating suitable experiments and evaluating all models on those experiments. E.g. adding the characteristic time density required defining sparse and dense values of timestamps for all datasets, creating test sets corresponding to those values and evaluating all models over those test sets.

11 Conclusion

We have analyzed the TKG embedding methods DE-TransE, DE-DistMult, DE-SimplE, ATiSE, TeRo and TimePlex to evaluate their performance depending on the charactersitics of queries. The inspected characteristics are the temporal density of data, the relation properties symmetry, antisymmetry, and inversion, and prediction target which has been included and expanded upon from a previous set of findings.

We find that the overall score of predictions is not particularly impacted by the temporal density of the data, however time predictions specifically score higher in temporally dense partitions. Our findings also support that DE-TransE, which theoretically cannot model symmetry, is indeed worse at symmetric relations than other models, while DE-DistMult, which theoretically cannot model antisymmetry and inversion, is not worse at antisymmetric or inverse relations than other models.

The findings are used in a new ensemble model ORB-E where the results are used to calculate query-specific weights for each model. ORB-E achieves better performance than each individual model, as well as an ensemble model that assignes the same static weight to each model. This suggests that it is possible to achieve a better performance by taking a model's advantages and disadvantages into account.

Finally, we use the continuous nature of timestamps to examine the accuracy of time predictions and improve them. We find that the difference between the predicted and correct timestamp is generally too large to be useful in most QA systems. However, if all models are equally accurate in their top timestamp answer, the average of all predictions is more accurate than each indvidual model.

12 Future Work

In this paper the ensemble method ORB-E is presented. ORB-E is a rule-based voting ensemble method that does not use learning to calculate score values, and is instead driven by the results of link prediction on pretrained models. Alternative approaches could make better use of learning in ORB-E to increase performance. One possible alternative is to use the defined characteristics and create a learnable score vector for each characteristic that contains parameters, one parameter for each model per characteristic. The score vector for each characteristic is then trained to maximize the MRR score of the ensemble over a validation set of queries. With this implementation, it would be possible to inspect the learned score vectors to find what characteristics influence performance positively and negatively, and find the most important characteristics for each dataset. Another possible alternative is to not use predefined characteristics, and instead rely on the learned ensemble model to find the characteristics to score the models individually. In this version, a vector representation of the query could facilitate the learned model to infer information in the query, and assign weights from the inferred information. This version could theoretically get a higher score than the previous implementation, but analysis to find the influential characteristics would be difficult.

Accuracy of time predictions is too low to be useful for answering real queries. To mitigate this, it might be useful to optimize time predictions by creating a method which uses the error difference as a part of the scoring function. The accuracy of joint timestamp selection could be improved by prioritizing methods with better accuracy, using more than one prediction result form each model, and combining values with something other than the mean. This could result in a method that can more accurately answer temporal questions in a QA system.

A two-step prediction approach could be used to first estimate whether the timestamp is within a dense or sparse subset of the dataset and use the result for model weight assignment when predicting time could improve ensemble performance in time predictions.

More detailed and accurate data could enable additional temporal signals in the data, as temporal events can permanently change an entity, and therefore change the behaviour that entity will have in all following relations. A method that accounts for the state of the entity at a given point in time, depending on which relations that entity has been a part of earlier, could improve performance.

Acknowledgments

We thank Daniele Dell'Aglio and Huan Li for help and guidance. We also thank Anna Veibel Bonde for help with proofreading.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2* (Lake Tahoe, Nevada) *(NIPS'13)*. Curran Associates Inc., Red Hook, NY, USA, 2787–2795.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. *ICEWS Coded Event Data.* Harvard Dataverse. https://doi.org/10.7910/DVN/28075

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 6901–6914. https://doi.org/10.18653/v1/2020.acl-main.617

Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2001–2011. https://doi.org/10.18653/v1/D18-1225

Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. 2018. Learning Sequence Encoders for Temporal Knowledge Graph Completion. *CoRR* abs/1809.03202 (2018), 6 pages. arXiv:1809.03202 http://arxiv.org/abs/1809.03202

Rishab Goel, Seyed Mehran Kazemi, Marcus Brubaker, and Pascal Poupart. 2019. Diachronic Embedding for Temporal Knowledge Graph Completion. https://doi.org/10.48550/ARXIV.1907.03143

Cosimo Gregucci, Mojtaba Nayyeri, Daniel Hernandez, and Steffen Staab. 2023. Link Prediction with Attention Applied on Multiple Knowledge Graph Embedding Models. In *Proceedings of the ACM Web Conference*. Association for Computing Machinery, Austin, TX, USA.

Frank Lauren Hitchcock. 1927. The Expression of a Tensor or a Polyadic as a Sum of Products. *Journal of Mathematics and Physics* 6 (1927), 164–189.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia D'amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2021. Knowledge Graphs. *ACM Comput. Surv.* 54, 4, Article 71 (jul 2021), 37 pages. https://doi.org/10.1145/3447772

Astrid Ipsen, Jeppe Lindberg W., Jonas Lindberg C., and Ning An. 2022. *Investigating Properties of Selected Knowledge Graph Embedding Methods.* Aalborg University, Aalborg, Denmark. https://projekter.aau.dk/projekter/da/studentthesis/investigating-properties-of-selected-temporal-knowledge-graph-embedding-methods(4c090514-3055-461f-b357-863e2aed53f2).html

Prachi Jain, Sushant Rathi, Mausam, and Soumen Chakrabarti. 2020. Temporal Knowledge Base Completion: New Algorithms and Evaluation Protocols. *CoRR* abs/2005.05035 (2020), 15 pages. arXiv:2005.05035 https://arxiv.org/abs/2005.05035

Woojeong Jin, Changlin Zhang, Pedro A. Szekely, and Xiang Ren. 2019. Recurrent Event Network for Reasoning over Temporal Knowledge Graphs. *CoRR* abs/1904.05530 (2019), 15 pages. arXiv:1904.05530 http://arxiv.org/abs/1904.05530

Seyed Mehran Kazemi and David Poole. 2018. SimplE Embedding for Link Prediction in Knowledge Graphs. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc., Palais des Congrès de Montréal, Montréal CANADA, 12 pages. https://proceedings.neurips.cc/paper/2018/file/b2ab001909a8a6f04b51920306046ce5-Paper.pdf

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. 2020. Tensor Decompositions for Temporal Knowledge Base Completion. In *International Conference on Learning Representations*. ICLR, Addis Ababa, Ethiopia, 12 pages. https://openreview.net/forum?id=rke2P1BFwS

Xueyuan Lin, Chengjin Xu, Haihong E, Fenglong Su, Gengxian Zhou, Tianyi Hu, Ningyuan Li, Mingzhi Sun, and Haoran Luo. 2022. TFLEX: Temporal Feature-Logic Embedding Framework for Complex Reasoning over Temporal Knowledge Graph. https://doi.org/10.48550/ARXIV.2205.14307

Farzaneh Mahdisoltani, Joanna Asia Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Seventh Biennial Conference on Innovative Data Systems Research*. www.cidrdb.org, Asilomar, CA, USA, 11 pages.

Ammar Mohammed and Rania Kora. 2023. A comprehensive review on ensemble deep learning: Opportunities and challenges. *Journal of King Saud University - Computer and Information Sciences* 35, 2 (2023), 757–774. https://doi.org/10.1016/j.jksuci.2023.01.014

Kristian Otte, Kristian Simoni Vestermark, Huan Li, and Daniele Dell'Aglio. 2022. Towards A Question Answering System over Temporal Knowledge Graph Embeddings. In *Proceedings of the Workshop on Deep Learning for Knowledge Graphs (DL4KG 2022)*. CEUR, online, 10 pages.

Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. 2020. YAGO 4: A Reason-able Knowledge Base. In *The Semantic Web*, Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Anna Lisa Gentile, Peter Haase, and Michael Cochez (Eds.). Springer International Publishing, Cham, 583–596.

Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. 2021. ChronoR: Rotation Based Temporal Knowledge Graph Embedding. *CoRR* abs/2103.10379 (2021), 9 pages. arXiv:2103.10379 https://arxiv.org/abs/2103.10379

Gunther Schmidt. 2010. *Relational Mathematics.* Cambridge University Press, Cambridge.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *CoRR* abs/1902.10197 (2019), 18 pages. arXiv:1902.10197 http://arxiv.org/abs/1902.10197

Mihai Surdeanu. 2013. Overview of the TAC2013 Knowledge Base Population Evaluation: English Slot Filling and Temporal Slot Filling. In *Proceedings of the Sixth Text Analysis Conference*. TAC, Gaithersburg, Maryland, USA, 13 pages.

Rakshit Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-Evolve: Deep Temporal Reasoning for Dynamic Knowledge Graphs. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, International Convention Centre, Sydney, Australia, 3462–3471. https://proceedings.mlr.press/v70/trivedi17a.html

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (sep 2014), 78–85. https://doi.org/10.1145/2629489

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2020. TeRo: A Time-aware Knowledge Graph Embedding via Temporal Rotation. In *Proceedings of the 28th International Conference on Computational Linguistics*. International Committee on Computational Linguistics, Barcelona, Spain (Online), 1583–1593. https://doi.org/10.18653/v1/2020.coling-main.139

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. 2019. Temporal Knowledge Graph Embedding Model based on Additive Time Series Decomposition (ATISE). https://doi.org/10.48550/ARXIV.1911.07893

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 *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6575

A Glossary

KG Knowledge Graph
KGE Knowledge Graph Embedding

MAE Mean Average Error
MRR Mean Reciprocal Rank

QA Question Answering

TKG Temporal Knowledge Graph

B Overview of Test Sets

Table 9: Test sets, and what they contain

Test set	Contains
T_D	Dense partiton timestamps
T_P	Sparse partition timestamps
T_S	Symmetric relations
T'_S	Non-symmetric relations
T_A	Anti-symmetric relations
T'_A	Non-antisymmetric relations
T_I	Inverse relations
T'_I	Non-inverse relations

C Test Set Statistics

Table 10: Number of facts in each test set

Dataset	$\lvert\mathcal{R}\rvert$	$\lvert T_D\rvert$	$\lvert T_P\rvert$	$\lvert T'_S\rvert$	$\lvert T_S\rvert$	$\lvert T'_A\rvert$	$\lvert T_A\rvert$	$\lvert T'_I\rvert$	$\lvert T_I\rvert$
ICEWS	220	7422	6687	22902	3987	23253	3636	25581	1308
WikiData	24	4248	3924	16248	0	580	15668	16248	0
YAGO	10	2068	2000	7500	704	704	7500	8204	0

D Time Density Figures

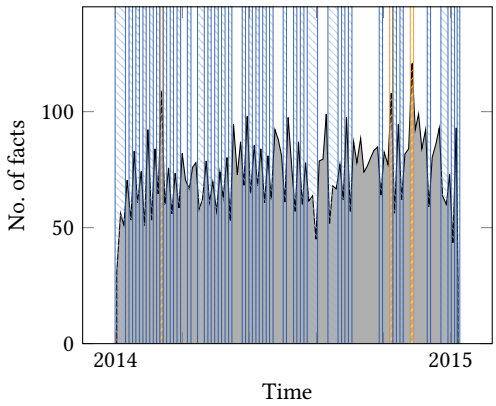

Figure 5: Dense and sparse partitions on ICEWS. Dense partition marked in yellow, sparse in blue.

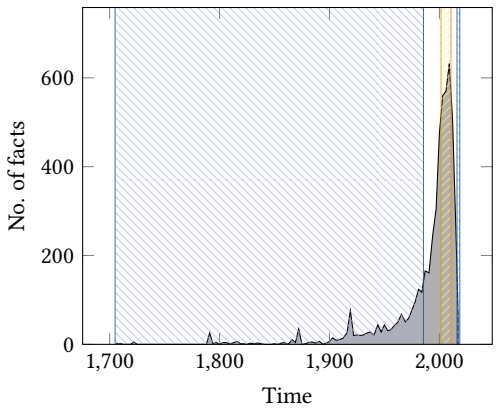

Figure 6: Dense and sparse partitions on WikiData. Dense partition marked in yellow, sparse in blue. Facts from before year 1700 are not included.

E Time Prediction Error Distribution

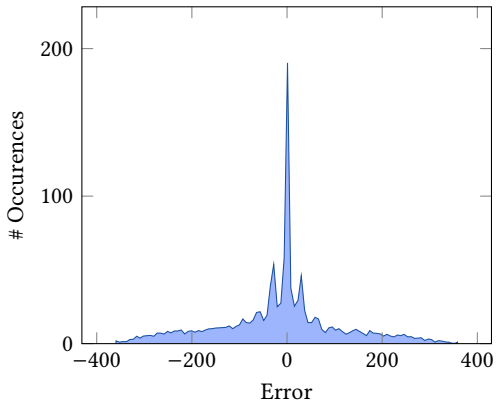

Figure 7: Distribution of predictions on timestamps for DE-TransE on ICEWS.

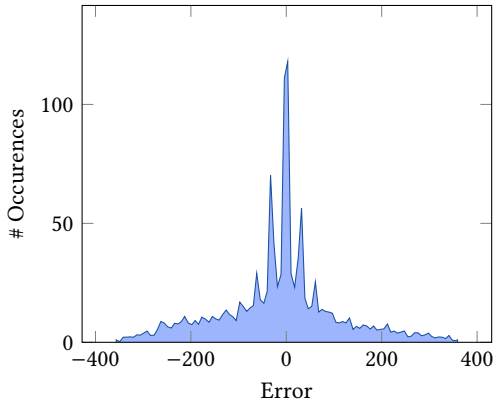

Figure 8: Distribution of predictions on timestamps for DE-DistMult on ICEWS.

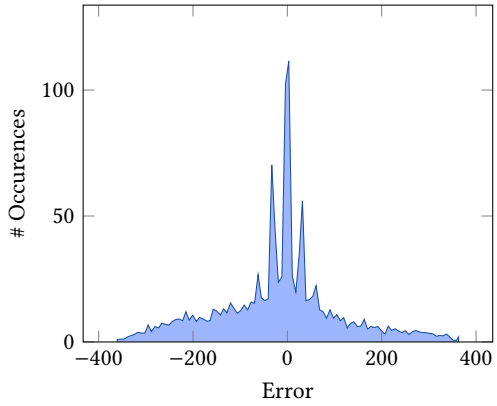

Figure 9: Distribution of predictions on timestamps for DE-SimplE on ICEWS.

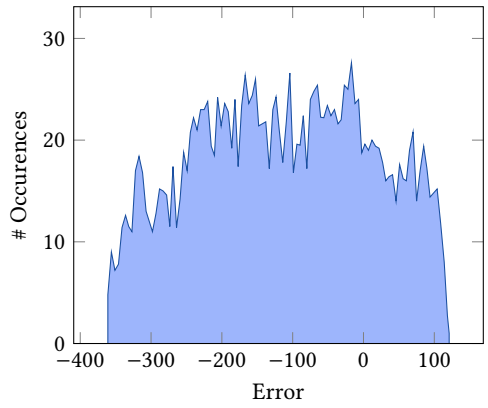

Figure 10: Distribution of predictions on timestamps for ATiSE on ICEWS.

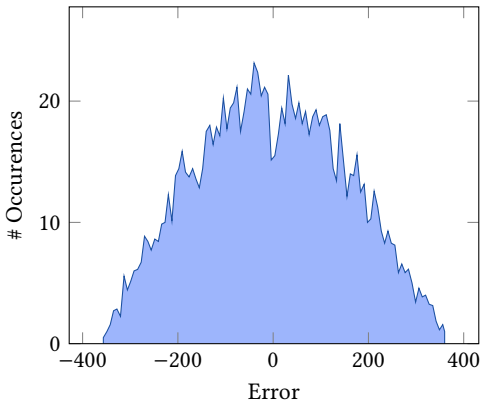

Figure 11: Distribution of predictions on timestamps for TeRo on ICEWS.

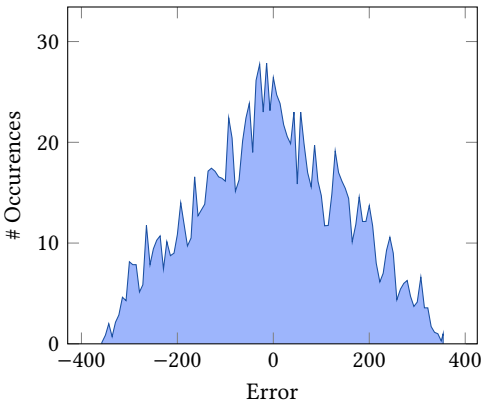

Figure 12: Distribution of predictions on timestamps for TimePlex on ICEWS.

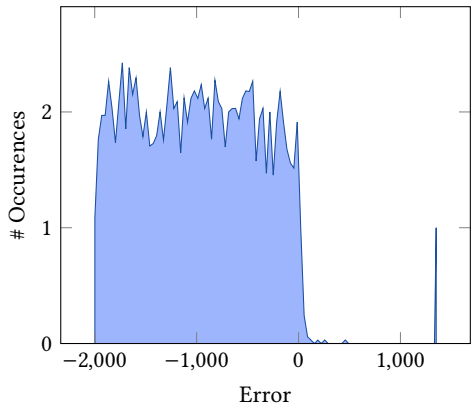

Figure 13: Distribution of predictions on timestamps for DE-TransE on WikiData.

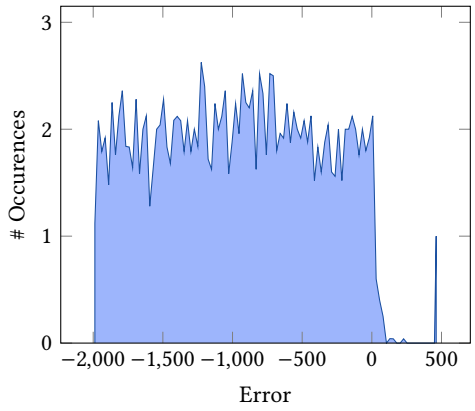

Figure 14: Distribution of predictions on timestamps for DE-DistMult on WikiData.

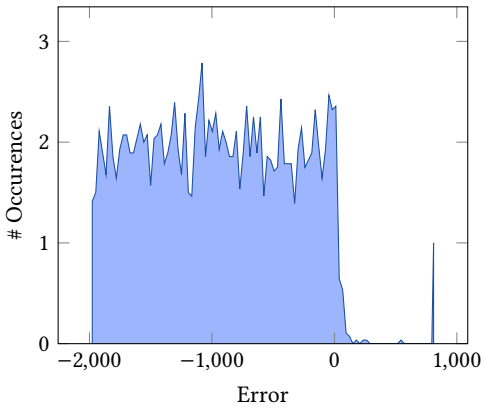

Figure 15: Distribution of predictions on timestamps for DE-SimplE on WikiData.

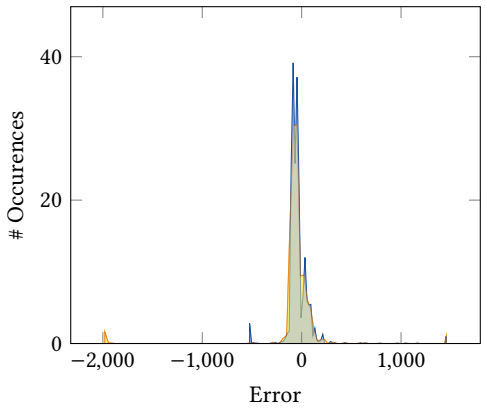

Figure 16: Distribution of predictions on timestamps for TeRo on WikiData. Blue indicates the best timestamp in the interval prediction, yellow indicates the worst.

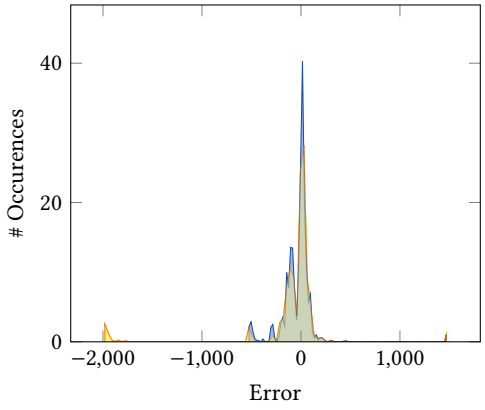

Figure 17: Distribution of predictions on timestamps for ATiSE on WikiData. Blue indicates the best timestamp in the interval prediction, yellow indicates the worst.

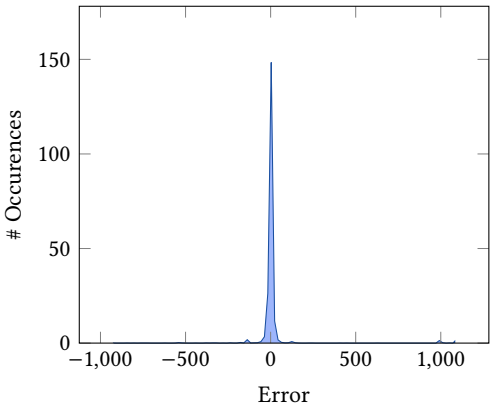

Figure 18: Distribution of predictions on timestamps for TimePlex on WikiData.

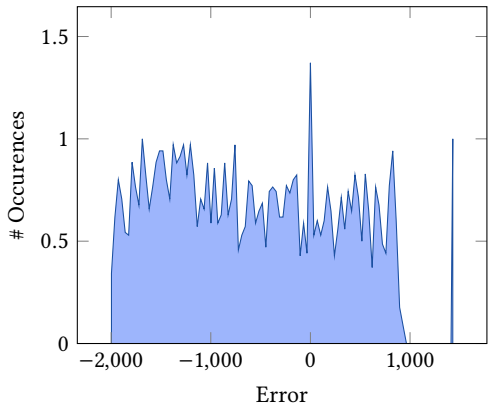

Figure 19: Distribution of predictions on timestamps for DE-TransE on YAGO.

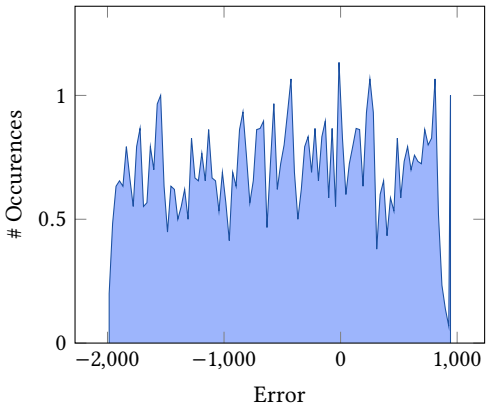

Figure 20: Distribution of predictions on timestamps for DE-DistMult on YAGO.

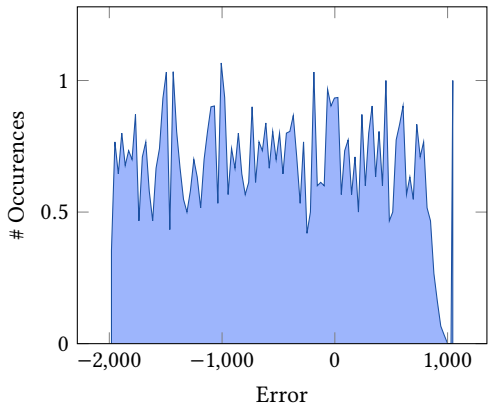

Figure 21: Distribution of predictions on timestamps for DE-SimplE on YAGO.

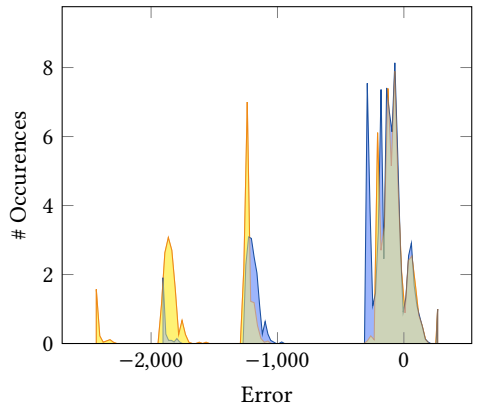

Figure 22: Distribution of predictions on timestamps for TeRo on YAGO. Blue indicates the best timestamp in the interval prediction, yellow indicates the worst.

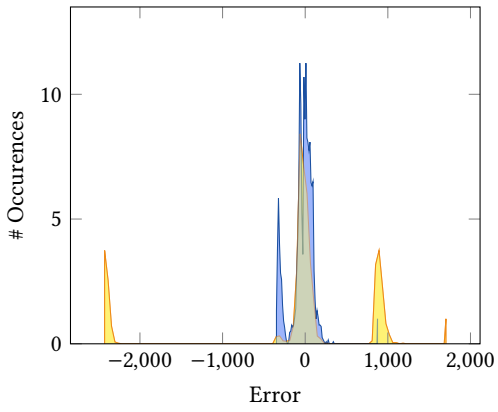

Figure 23: Distribution of predictions on timestamps for ATiSE on YAGO. Blue indicates the best timestamp in the interval prediction, yellow indicates the worst.

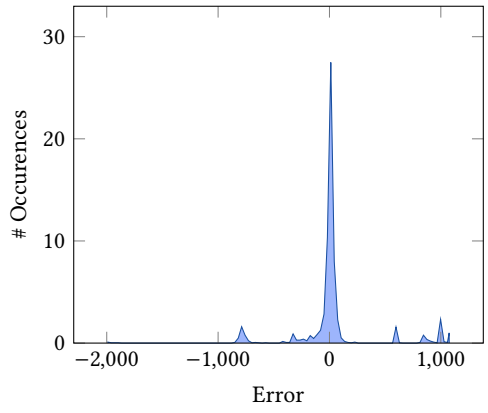

Figure 24: Distribution of predictions on timestamps for TimePlex on YAGO.

F Comparisons of scores between prediction targets on the original splits of all datasets and all models

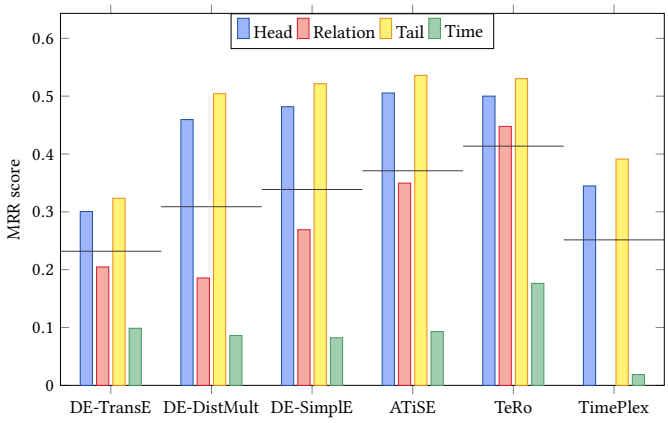

Figure 25: ICEWS, split original

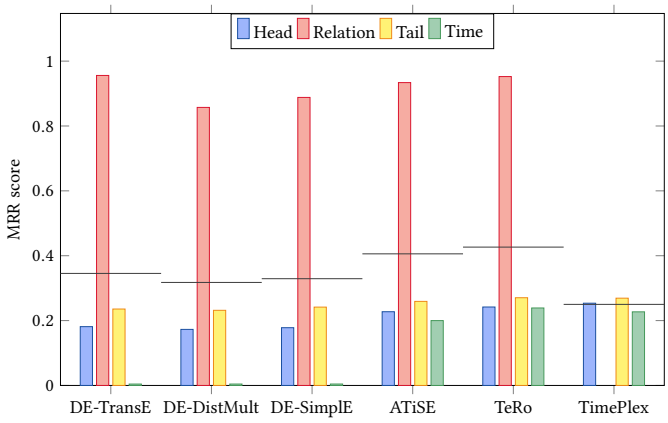

Figure 26: WikiData, split original

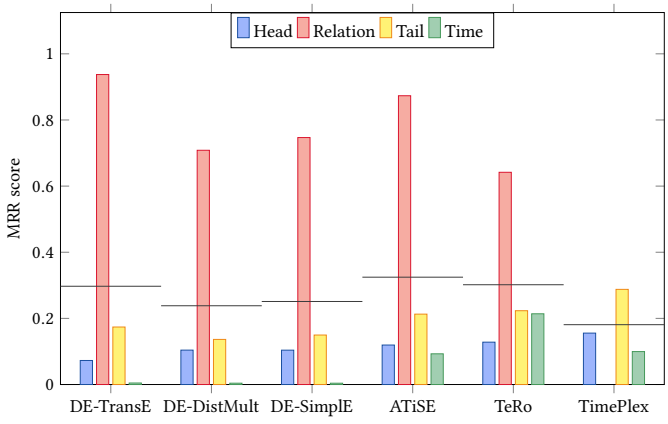

Figure 27: YAGO, split original

G Comparisons with original papers

Table 11: MRR results of the methods on only head and tail predictions on the given datasets, along with models scores from the original papers.

	ICEWS		WikiData		YAGO	
Method	Our MRR	Paper	Our MRR	Paper	Our MRR	Paper
DE-TransE	0.31	0.33	0.21	–	0.12	–
DE-DistMult	0.48	0.50	0.20	–	0.12	–
DE-SimplE	0.50	0.53	0.21	–	0.13	–
TeRo	0.52	0.56	0.26	0.30	0.18	0.19
ATiSE	0.52	0.55	0.24	0.28	0.17	0.17
TimePlex	0.37	0.60	0.26	0.33	0.22	0.24

Table 12: MRR results of the methods on all prediction targets on the given datasets along with the score from the original papers, that only does head and tail predicion.

	ICEWS		WikiData		YAGO	
Method	All MRR	Paper	All MRR	Paper	All MRR	Paper
DE-TransE	0.23	0.33	0.35	–	0.30	–
DE-DistMult	0.31	0.50	0.32	–	0.24	–
DE-SimplE	0.34	0.53	0.33	–	0.25	–
TeRo	0.41	0.56	0.43	0.30	0.30	0.19
ATiSE	0.37	0.55	0.41	0.28	0.32	0.17
TimePlex	0.25	0.60	0.25	0.33	0.18	0.24