DOI 10.1007/s10844-016-0414-7

Noname manuscript No.
(will be inserted by the editor)

Efficient Energy-Based Embedding Models for Link Prediction in Knowledge Graphs

Pasquale Minervini · **Claudia d'Amato** ·
Nicola Fanizzi

the date of receipt and acceptance should be inserted later

Abstract We focus on the problem of link prediction in Knowledge Graphs, with the goal of discovering new facts. To this purpose, Energy-Based Models for Knowledge Graphs that embed entities and relations in continuous vector spaces have been largely used. The main limitation in their applicability lies in the parameter learning phase, which may require a large amount of time for converging to optimal solutions. In this article, we first propose an unified view on different Energy-Based Embedding Models. Hence, for improving the model training phase, we propose the adoption of adaptive learning rates. We show that, by adopting adaptive learning rates during training, we can improve the efficiency of the parameter learning process by an order of magnitude, while leading to more accurate link prediction models in a significantly lower number of iterations. We extensively evaluate the proposed learning procedure on a variety of new models: our results show a significant improvement over state-of-the-art link prediction methods on two large Knowledge Graphs, namely WordNet and Freebase.

1 Introduction

Knowledge Graphs (KGs) are graph-structured knowledge bases, where factual knowledge is represented in the form of relationships between entities. We focus on KGs that adopt *Resource Description Framework* (RDF)[1] as their representation, since they constitute a powerful instrument for search, analytics, recommendations, and data integration. Indeed, RDF is the Web standard for expressing information about resources.

Department of Computer Science - University of Bari, Italy
E-mail: {firstname.lastname}@uniba.it

[1] `http://www.w3.org/TR/rdf11-concepts/`

A resource (hereafter also called *entity*) can be anything, including documents, people, physical objects, and abstract concepts. An RDF knowledge base (also called *RDF graph* as a KG) is a set of *RDF triples* of the form $\langle s, p, o \rangle$, where s, p and o denote the *subject*, the *predicate* (i.e. a *relation type*) and the *object* of the triple, respectively. Each triple $\langle s, p, o \rangle$ describes a statement, which can be interpreted as: *A relationship of type p holds between entities s and o.* The following example shows a set of RDF triples[2] describing the writer *William Shakespeare*[3]:

Example 1 (RDF Fragment)

⟨W. Shakespeare,	influencedBy,	G. Chaucer⟩
⟨W. Shakespeare,	religion,	Church of England⟩
⟨W. Shakespeare,	author,	Hamlet⟩
⟨Hamlet,	genre,	Tragedy⟩
⟨Hamlet,	character,	Ophelia⟩ □

Several RDF KGs are publicly available through the *Linked Open Data* (LOD) cloud, a collection of interlinked KGs such as Freebase [4], DBpedia [3] and YAGO [21]. As of April 2014, the LOD cloud is composed of 1,091 interlinked KGs, globally describing more than 8×10^6 entities, and 188×10^6 relationships holding between them[4]. However, KGs are often largely incomplete. For instance, 71% of the persons described in Freebase[5] have no known place of birth and 75% of them have no known nationality [14].

For this reason, in this work, we focus on the problem of *predicting missing links* in large KGs, so as to discover new facts about a domain of interest. In the literature, this problem is referred to as *link prediction*, or *knowledge base completion.* The aim of this work is to provide an efficient and accurate model for predicting missing RDF triples in large RDF KGs (in a *link prediction* setting), without requiring extra background knowledge.

The link prediction task is well known in *Statistical Relational Learning* (SRL) [16] which aims at modeling data from multi-relational domains, such as social networks, citation networks, protein interaction networks and knowledge graphs, and detecting missing links in such domains. Two main categories of models can be ascribed to SRL: *Probabilistic latent variable models* and *embedding models* (also frequently called *Energy-based models*). A detailed analysis of these two classes of models is reported in Sect. sec:related.

While appearing promising in terms of link prediction results, *Probabilistic latent variable models* showed limitations on scaling on large KG because of the complexity of the probabilistic inference and learning, which is intractable in general [19]. Differently from them, *embedding models* have shown interesting

[2] This description is taken from the Freebase KG [4]

[3] For readability reasons, we describe entities and relations using an intuitive way of writing down triples as text rather than using the pure RDF syntax.

[4] State of the LOD Cloud 2014: `http://lod-cloud.net/`

[5] Available at `https://developers.google.com/freebase/data`

ability to scale on large KG while maintaining comparative performance in terms of predictive accuracy [5].

We focus specifically on a class of embedding models for KGs, named as *Energy-Based Embedding Models* (EBEMs), where entities and relations are embedded in continuous vector spaces, referred to as *embedding spaces*. In such models, the probability of an RDF triple to encode a true statement is expressed in terms of *energy* of the triple: this is an unnormalized score that is inversely proportional to such a probability value, and is computed as a function of the embedding vectors of the subject, the predicate and the object of the triple. The reason why we focus on this class of models, such as *Translating Embedding* (TransE) [8] and other related ones [7,9,28], is because it has been experimentally proved that they achieve state-of-the-art predictive accuracy results on link prediction tasks, while being able to scale to large and Web-scale KGs [5, 8, 14]. However, a major limiting factor for EBEMs lies in the parameter learning algorithm, which may require a long time (even days) to converge on large KGs [11].

In order to overcome such a limitation, we propose a method for reducing the learning time in EBEMs by an order of magnitude, while leading to more accurate link prediction models. Furthermore, we employ the proposed learning method for evaluating a family of novel EBEMs with useful properties. We experimentally tested our methods on two large and commonly used KGs: namely WordNet and Freebase, following the same evaluation protocol used in [8]. Our results show a significant improvement over the state-of-the-art embedding models.

The rest of the paper is organized as follows. In Sect. sec:basics, we introduce basics on Energy-Based Models. In Sect. sec:models we propose: a) a framework for characterizing state-of-the-art EBEMs, b) a family of novel energy functions with useful properties, c) a method for improving the efficiency of the learning process in such models. In Sect. sec:related the main related works falling in the *Probabilistic latent variable models* and *Embedding Models* categories are analyzed, while in Sect. sec:evaluation we empirically evaluate the proposed learning methods and energy functions. In Sect. sec:conclusion we summarize our work and outline future research directions.

2 Basics on Energy-Based Models

Energy-Based Models [20] are a versatile and flexible framework for modeling dependencies between variables. The key component is a scalar-valued *energy function* $E(\cdot)$, which associates a scalar *energy* with a configuration of variables. The energy of a configuration of variables is inversely proportional to the probability of such a configuration. Precisely, more likely configurations correspond to lower energy values, while less likely configurations correspond to higher energy values. Two main steps can be recognized in energy-based models: the *inference* step and the *learning* step.

The *inference* step consists in finding the most likely configuration of the variables of interest, that is the one that minimizes the energy function $E(\cdot)$. Given X and Y random variables, with values in $\mathcal{X}$ and $\mathcal{Y}$, an example of the exploitation of the inference in energy-based models is given in the following.

Example 2 (Energy-Based Inference) Assuming that X describes the pixels of an image, while Y describes a discrete label associated with the image (such as "car" or "tree"), let $E : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ be an energy function defined on the configurations of X and Y. The most likely label $y^* \in \mathcal{Y}$ for an image $x \in \mathcal{X}$ can be inferred by finding the label in $\mathcal{Y}$ that, given x, minimizes the energy function $E(\cdot)$:

$$y^* = \arg\min_{y \in \mathcal{Y}} E(x, y).$$

Learning in energy-based models consists in finding the most appropriate energy function within a family $\mathcal{F} = \{E_\theta \mid \theta \in \Theta\}$, indexed by parameters θ, that is in finding the function that is actually able to associate *lower energy states* with likely configurations of the variables of interests, and *higher energy states* to unlikely configurations of such variables. In practice, this corresponds to finding the energy function $E^*_\theta \in \mathcal{F}$ that minimizes a given *loss functional* $\mathcal{L}$, which measures the *quality* of the energy function on the data $\mathcal{D}$:

$$E^*_\theta = \arg\min_{E_\theta \in \mathcal{F}} \mathcal{L}(E_\theta, \mathcal{D}).$$

A normalized probability distribution can be derived from an energy-based model. Specifically, given an energy function $E : \mathcal{X} \mapsto \mathbb{R}$ defined on the possible configurations of a random variable X, it is possible to derive a corresponding probability distribution through the *Gibbs distribution*:

$$P(X = x) = \frac{1}{Z(\beta)} e^{-\beta E(x)}$$

where β is an arbitrary positive constant, and $Z(\beta) = \sum_{\tilde{x} \in \mathcal{X}} e^{-\beta E(\tilde{x})}$ is a normalizing factor [6] referred to as the *partition function.*

3 A Framework for Energy-Based Embedding Models

Energy-based models can be used for modeling the uncertainty in RDF KGs, in both statistical inference and learning tasks.

An RDF graph G can be viewed as a labeled directed multigraph, where entities are vertices, and each RDF triple is represented by a directed edge whose label is a predicate, and emanating from its source vertex to its object vertex. We denote with $\mathcal{E}_G$ the set of all entities occurring as subjects or objects in G, formally:

$$\mathcal{E}_G = \{s \mid \exists \langle s, p, o \rangle \in G\} \cup \{o \mid \exists \langle s, p, o \rangle \in G\}$$

[6] If X is a continuous random variable, then $Z(\beta) = \int_{\tilde{x} \in \mathcal{X}} e^{-\beta E(\tilde{x})}$.

and we denote with $\mathcal{R}_G$ the set of all relations appearing as predicates in G, formally:

$$\mathcal{R}_G = \{p \mid \exists \langle s, p, o \rangle \in G\}.$$

Let $\mathcal{S}_G = \mathcal{E}_G \times \mathcal{R}_G \times \mathcal{E}_G$ be the space of *possible triples* of G, with $G \subseteq \mathcal{S}_G$, and let $E_\theta : \mathcal{S}_G \to \mathbb{R}$ (with parameters θ) be an energy function that defines an energy distribution over the set of possible triples $\mathcal{S}_G$.

The inference step consists in finding the most likely configuration of the variables of interest, i.e. the one that minimizes the energy function $E(\cdot)$. For instance, assume we need to know the most likely object o^* to appear in a triple with subject s (e.g. W. SHAKESPEARE) and predicate p (e.g. NATIONALITY). It can be inferred by finding the object o that minimizes the function $E(\cdot)$, as follows:

$$o^* = \arg\min_{o \in \mathcal{E}_G} E_\theta(\langle s, p, o \rangle).$$

Similar inference tasks can be performed with respect to the subject s, the predicate p, or a subset of such variables.

Learning consists in finding an energy function $E^*_\theta \in \mathcal{F}$, within a parametric family of energy functions $\mathcal{F} = \{E_\theta \mid \theta \in \Theta\}$ indexed by parameters θ, that minimizes a given loss functional $\mathcal{L}$ defined on the RDF graph G, that is:

$$E^*_\theta = \arg\min_{E^*_\theta \in \mathcal{F}} \mathcal{L}(E_\theta, G).$$

Since the *energy* value for a triple expresses a quantity that is inversely proportional to the probability of the triple itself (see Sect. sec:basics), in a *link prediction* setting, the energy function $E^*_\theta(\cdot)$ can be exploited for assessing a ranking of the so called *unobserved* triples, that are the triples in $\mathcal{S}_G \setminus G$. As such, triples associated with lower energy values (higher probabilities) will be more likely to be considered for a completion of the graph G, differently from triples associated with the higher energy values (lower probabilities).

On this point, it is important to note that *Open World Assumption* holds in RDF, which means that when a triple is missing in G, this does not have to be interpreted as that the corresponding statement is false (like for the case of the *Closed World Assumption* typically made in database settings), but rather that its truth value is *missing/unknown*, since it cannot be observed in the KG. We will refer to all triples in G as *visible triples*, and to all triples in $\mathcal{S}_G \setminus G$ as *unobserved triples*, which might encode true statements.

Within energy-based models, we particularly focus on *Energy-Based Embedding Models* (EBEMs) which are a specific class of models where each entity $x \in \mathcal{E}_G$ is mapped to a unique low-dimensional continuous vector $\mathbf{e}_x \in \mathbb{R}^k$, that is referred to as the *embedding vector* of x, and each predicate $p \in \mathcal{R}_G$ corresponds to an operation in the embedding vector space. As already pointed out in Sect. sec:introduction, the reason for such a choice is that EBEMs, such as *Translating Embedding* (TransE) [8] and related models [7, 9, 28], achieve state-of-the-art results in link prediction tasks, while being able to scale on very large (Web-scale) Knowledge Graphs [6].

In the following sections:

Table 1: Energy-Based Embedding Models for knowledge graphs proposed in the literature, with their energy functions, shared and embedding parameters.

Model	Energy function $E(\langle s, p, o \rangle)$	Shared	Embedding
Unstructured [7]	$\lVert \mathbf{e}_s - \mathbf{e}_o \rVert_1$		$\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k$
TransE [8]	$\lVert (\mathbf{e}_s + \mathbf{e}_p) - \mathbf{e}_o \rVert_{1/2}$		$\mathbf{e}_s, \mathbf{e}_p, \mathbf{e}_o \in \mathbb{R}^k$
SE [9]	$\lVert \mathbf{R}_{p,1} \mathbf{e}_s - \mathbf{R}_{p,2} \mathbf{e}_o \rVert_1$		$\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k, \mathbf{R}_{p,\cdot} \in \mathbb{R}^{n \times k}$
RESCAL [24]	$\mathbf{e}_s^T \mathbf{R}_p \mathbf{e}_o$		$\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k, \mathbf{R}_p \in \mathbb{R}^{k \times k}$
SME lin. [7]	$(\mathbf{R}_1 \mathbf{e}_s + \mathbf{R}_2 \mathbf{e}_p)^T (\mathbf{R}_3 \mathbf{e}_o + \mathbf{R}_4 \mathbf{e}_p)$	$\mathbf{R}_\cdot \in \mathbb{R}^{n \times k}$	$\mathbf{e}_s, \mathbf{e}_p, \mathbf{e}_o \in \mathbb{R}^k$
SME bil. [7]	$[(\mathbf{R}_1 \mathbf{e}_s) \times_3 (\mathbf{R}_2 \mathbf{e}_p)]^T \, [(\mathbf{R}_3 \mathbf{e}_o) \times_3 (\mathbf{R}_4 \mathbf{e}_p)]$	$\mathbf{R}_\cdot \in \mathbb{R}^{n \times k}$	$\mathbf{e}_s, \mathbf{e}_p, \mathbf{e}_o \in \mathbb{R}^k$
NTN [28]	$\mathbf{u}_p^T \tanh\left(\mathbf{e}_s^T \mathbf{T}_p \mathbf{e}_o + \mathbf{R}_{p,1} \mathbf{e}_s + \mathbf{R}_{p,2} \mathbf{e}_o\right)$		$\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k, \mathbf{u}_p \in \mathbb{R}^n,$ $\mathbf{T}_p \in \mathbb{R}^{k \times k \times n}, \mathbf{R}_{p,\cdot} \in \mathbb{R}^{n \times k}$

(a) We present a unified general framework for formalizing EBEMs for KGs and we show that EBEMs proposed in the literature so far can be characterized with respect to their energy function. Additionally, we propose novel formulations of the energy functions with useful properties (see Sect. sec:inference).

(b) We then focus on the EBEM *learning* phase, by proposing a method for improving the efficiency of the parameters learning step (see Sect. sec:learning).

3.1 Energy Function Characterization and New Energy Functions

The energy function $E_\theta : \mathcal{S}_G \to \mathbb{R}$ considered in the state-of-the art EBEMs for KG can be defined by using two types of parameters:

– **Shared Parameters:** used for computing the energy of all triples in the space of the possible triples $\mathcal{S}_G$ of G.

– **Embedding Parameters:** used for computing the energy of triples containing a specific entity or relation $x \in \mathcal{E}_G \cup \mathcal{R}_G$. We denote such parameters by adding a subscript with the name of the entity or relation they are associated with. For instance, in the Translating Embeddings model, $\mathbf{e}_s$ denotes the embedding vector representing a subject s, and $\mathbf{e}_p$ denotes the translation vector representing a predicate p.

Both shared parameters and embedding parameters are learned from data. In particular, EBEMs for KGs associate each entity $x \in \mathcal{E}_G$ with a k-dimensional embedding vector $\mathbf{e}_x \in \mathbb{R}^k$, and each relation $p \in \mathcal{R}_G$ with a set of embedding parameters $\mathbf{S}_p$. Tab. tab:models summarizes the energy functions that have been used by the state-of-the-art EBEMs for link prediction in KGs, and highlights the distinction between the two different kinds of parameters reported above. Please note that, in the table, the subscript for the parameters stand for the entity/predicate to which they refer to, e.g. the subscript $_p$ is added to the parameters associated with a particular predicate p.

The energy functions can be seen as sharing a common structure: given a RDF triple $\langle s, p, o \rangle$, its energy $E(\langle s, p, o \rangle)$ is computed by the following two-step process, also depicted in Fig. fig:energystruct:

1. The embedding vectors $\mathbf{e}_s, \mathbf{e}_o \in \mathbb{R}^k$ respectively of the subject s and the object o of the triple, and the embedding parameters $\mathbf{S}_p$ associated with

the predicate p of the triple are used to obtain two new vectors $\mathbf{e}'_s, \mathbf{e}'_o \in \mathbb{R}^{k'}$ by means of two model-dependent functions $f_s(\cdot)$ and $f_o(\cdot)$:

$$\mathbf{e}'_s = f_s(\mathbf{e}_s, \mathbf{S}_p), \qquad \mathbf{e}'_o = f_o(\mathbf{e}_o, \mathbf{S}_p).$$

2. The energy of a triple $\langle s, p, o \rangle$ is computed by a model-dependent function $g(\cdot)$, with $g : \mathbb{R}^{k'} \times \mathbb{R}^{k'} \mapsto \mathbb{R}$, applied to the vectors $\mathbf{e}'_s, \mathbf{e}'_o \in \mathbb{R}^{k'}$ resulting from the previous step:

$$E(\langle s, p, o \rangle) = g(\mathbf{e}'_s, \mathbf{e}'_o) = g(f_s(\mathbf{e}_s, \mathbf{S}_p), f_o(\mathbf{e}_o, \mathbf{S}_p)). \tag{1}$$

Please note that here, the proposed unifying framework is intended for *describing* EBEMs for KG: the choice for the functions $f_s(\cdot)$, $f_o(\cdot)$ and $g(\cdot)$ is model-dependent, and different models might correspond to different choices of such functions. As an example, in the following we show how the energy function adopted by the *Translating Embeddings* model (TransE) [8], a state of the art EBEM for performing link prediction in KG, can be expressed by the use of the formalization presented above. TransE is particularly interesting: while its number of parameters grows *linearly* with the number of entities and relations in the KG, it yields state-of-the-art link prediction results on WordNet and Freebase KGs (see the empirical comparison with other link prediction methods in Sect. sec:adaptive).

$$\begin{tikzcd}
\mathbf{e}_s \arrow[d] & \mathbf{S}_p \arrow[dl] \arrow[dr] & \mathbf{e}_o \arrow[d] \\
\mathbf{e}'_s = f_s(\mathbf{e}_s, \mathbf{S}_p) \arrow[dr] & & \mathbf{e}'_o = f_o(\mathbf{e}_o, \mathbf{S}_p) \arrow[dl] \\
& E(\langle s, p, o \rangle) = g(\mathbf{e}'_s, \mathbf{e}'_o) &
\end{tikzcd}$$

Fig. 1: Structure of the energy function in Energy-Based Embedding Models for KGs: $\mathbf{e}_s$, $\mathbf{S}_p$ and $\mathbf{e}_o$ are the embedding parameters of s, p and o.

Example 3 (Energy Function in TransE) In the formulation for the energy function of TransE [8] (see also Tab.tab:models), each entity $x \in \mathcal{E}_G$ in an RDF graph G corresponds to a k-dimensional embedding vector $\mathbf{e}_x \in \mathbb{R}^k$, while each predicate $p \in \mathcal{R}_G$ corresponds to a *translation operation*, represented by a k-dimensional vector $\mathbf{e}_p \in \mathbb{R}^k$, in the embedding vector space. As from Tab. tab:models, the energy function can be formulated by using the L_1 or the L_2 distance of the (translated) subject and object embedding vectors. In the case of L_1 formulation, the *energy* of an RDF triple $\langle s, p, o \rangle$ is given by the L_1 distance of $(\mathbf{e}_s + \mathbf{e}_p)$, corresponding to $\mathbf{e}_s$ translated by $\mathbf{e}_p$, and $\mathbf{e}_o$:

$$E(\langle s, p, o \rangle) = \| (\mathbf{e}_s + \mathbf{e}_p) - \mathbf{e}_o \|_1.$$

This corresponds to the following choice of the functions $f_s(\cdot)$, $f_o(\cdot)$ and $g(\cdot)$:

$$f_s(\mathbf{e}_s, \{\mathbf{e}_p\}) = \mathbf{e}_s + \mathbf{e}_p, \qquad f_o(\mathbf{e}_o, \{\mathbf{e}_p\}) = \mathbf{e}_o, \qquad g(\mathbf{e}'_s, \mathbf{e}'_o) = \|\mathbf{e}'_s - \mathbf{e}'_o\|_1. \quad \square$$

Besides proposing a general framework for expressing an energy function to be used by EBEMs, we also investigate whether the choice of other *affine transformations* for the functions $f_s(\cdot)$ and $f_o(\cdot)$, such as *scaling*, or *composition of translation and scaling*, leads to more accurate models than those generated by TransE (using the energy function reported in Tab. tab:models), while still having a number of parameters that scales linearly in the number of entities and relations in the KG. Specifically, we investigate the choices for the following $f_s(\cdot)$ and $f_o(\cdot)$ functions:

Translation:	$f(\mathbf{e}_x, \{\mathbf{e}_p\}) = \mathbf{e}_x + \mathbf{e}_p,$
Scaling:	$f(\mathbf{e}_x, \{\mathbf{e}_p\}) = \mathbf{e}_x \odot \mathbf{e}_p,$
Scaling $\circ$ Translation:	$f(\mathbf{e}_x, \{\mathbf{e}_{p,1}, \mathbf{e}_{p,2}\}) = (\mathbf{e}_x \odot \mathbf{e}_{p,1}) + \mathbf{e}_{p,2},$

where $\circ$ denotes the composition operation between functions, and $\odot$ denotes the Hadamard product, also referred to as element-wise product. The results of such a study are reported and discussed in Sect. sec:adaptive.

In addition, we also evaluate the effect of enforcing the embedding vector of all entities to lie on the Euclidean unit $(k-1)$-sphere, that is $\mathbb{S}^{k-1} = \{\mathbf{x} \in \mathbb{R}^k \mid \|\mathbf{x}\|_2 = 1\}$. This is motivated by the fact that, in TransE [8] and related-models, the L_2 norm of all entity embedding vectors is enforced to be 1: hence, we take into account the effect of normalizing the results of the functions $f_s(\cdot)$ and $f_o(\cdot)$, so that the resulting projections also lie on the Euclidean unit sphere (together with all entity embedding vectors).

In the next section, we discuss the *learning* step of EBEMs, which consists in finding the most appropriate energy function to be used during the *inference* step (see Sect. sec:basics), and propose a method for improving both the efficiency of the learning process and the predictive accuracy of the learned model.

3.2 Learning the Parameters of the Energy Function

As discussed in Sect. sec:basics, *learning* in EBEMs for KGs corresponds to finding an energy function E^*_θ, within a family of functions $\mathcal{F} = \{E_\theta \mid \theta \in \Theta\}$ indexed by parameters θ, that minimizes a given *loss functional* $\mathcal{L}$ measuring the compatibility of an energy function with respect to the RDF graph G:

$$E^*_\theta = \arg\min_{E_\theta \in \mathcal{F}} \mathcal{L}(E_\theta, G). \tag{2}$$

In the following, the definition for the *loss functional* $\mathcal{L}$ is given. In agreement with the formalization presented in Sect. sec:inference, a key point for learning the (best) energy function in EBEMs consists in learning the shared and embedding parameters to be used for computing the energy function. As for [7–9], shared and embedding parameters are learned by using a *corruption*

Algorithm 1 Learning in EBEMs via *Stochastic Gradient Descent* [8]

```
Input: Learning rate $\eta$, batch size $n$
Output: Optimal model parameters $\theta^*$
1: Initialize model parameters $\theta_0$
2: for $t \in \langle 1, \ldots, \tau \rangle$ do
3:     $\mathbf{e}_x \leftarrow \mathbf{e}_x / \|\mathbf{e}_x\|, \forall x \in \mathcal{E}_G$          {Normalize all entity embeddings}
4:     $T \leftarrow \textsc{SampleBatch}(G, n)$          {Sample observed and corrupted triples}
5:     $g_t \leftarrow \nabla \sum_{(x,\tilde{x}) \in T} [\gamma + E_\theta(x) - E_\theta(\tilde{x})]_+$          {Evaluate the gradient of $\mathcal{L}$ w.r.t. $\theta$}
6:     $\Delta_t \leftarrow -\eta g_t$          {Calculate the update to model parameters $\theta$}
7:     $\theta_t \leftarrow \theta_{t-1} + \Delta_t$          {Update the model parameters $\theta$}
8: end for
9: return $\theta_\tau$
```

process $\mathcal{Q}(\tilde{x} \mid x)$ that, given a RDF triple $x \in G$, produces a *corrupted* RDF triple $\tilde{x}$, uniformly sampled from the set of corrupted triples $\mathcal{C}_x$. Formally, given an RDF triple $\langle s, p, o \rangle$ from G, the set of corrupted triples for it is given by

$$\mathcal{C}_{\langle s,p,o \rangle} = \{\langle \tilde{s}, p, o \rangle \mid \tilde{s} \in \mathcal{E}_G\} \cup \{\langle s, p, \tilde{o} \rangle \mid \tilde{o} \in \mathcal{E}_G\}$$

that is the set obtained by replacing either the subject or the object of the triple with another entity from the set of entities $\mathcal{E}_G$.

The corruption process is applied to *positive* training RDF triples in order to generate *negative* examples that are actually missing in a KG. By corrupting the subject and the object of triples in the KG, the Local Closed World Assumption (LCWA) [14] is implicitly followed. In the LCWA, the idea is to consider the knowledge about a specific property p (e.g. NATIONALITY) of a resource s (e.g. W. SHAKESPEARE) to be *locally complete* if a value for p is already specified for the resource s. For instance, knowing that the triple $\langle$W. SHAKESPEARE, NATIONALITY, ENGLISH$\rangle$ is true (because observed in the KG), allows to assume that $\langle$W. SHAKESPEARE, NATIONALITY, AMERICAN$\rangle$ - i.e. a triple obtained by corrupting the object - is very likely to be false.

Since the final goal is to learn an energy function which associates lowest energy values (highest score) with observed triples, and highest energy values (lowest score) with unobserved triples, the corruption process $\mathcal{Q}(\tilde{x} \mid x)$ is used for defining the following margin-based stochastic ranking criterion over the triples in G:

$$\mathcal{L}(E_\theta, G) = \sum_{x \in G} \sum_{\tilde{x} \sim \mathcal{Q}(\tilde{x} \mid x)} [\gamma + E_\theta(x) - E_\theta(\tilde{x})]_+ , \qquad (3)$$

where $[x]_+ = \max\{0, x\}$, $\gamma > 0$ is a hyperparameter referred to as *margin*, and the embedding vector of each entity is enforced to have an unitary norm, i.e. $\forall x \in \mathcal{E}_G : \|\mathbf{e}_x\| = 1$. Actually, the loss functional in Eq. eq:loss$_m arginenforcestheenergyofobservedtriplestobelowerthant$
$theunitarynormconstraintsintheoptimizationproblempreventthetrainingprocesstotriviallysolveitbyincreasingtheentit$
1001[8].

The minimization problem in Eq. eq:minloss can be solved by using projected Stochastic Gradient Descent (SGD) in mini-batch mode, as also proposed in [7–9] and summarized in Alg. alg:sgd. The training algorithm works as follows: given an RDF graph G, at each iteration, it samples a batch of triples from G. Similarly to [8], each batch is obtained by first randomly permuting all triples in G, then partitioning them into n_b batches of similar size, and iterating over them. A single pass over all triples in G is called an *epoch*. For each triple in the batch, the algorithm generates a *corrupted* triple by means of the corruption process $\mathcal{Q}(\tilde{x} \mid x)$: this leads to a set T of observed and corrupted pairs of triples. Hence, the observed/corrupted triple pairs in T are used to evaluate the gradient of the loss functional $\mathcal{L}$ in Eq. eq:loss$_m arginwithrespecttothecurrentmodelparameters\theta$. Finally, θ is updated in the steepest descent direction of the loss functional $\mathcal{L}$ by a fixed learning rate η. This procedure is repeated until convergence (in [8] the learning procedure was limited to 1000 epochs).

The main drawback of SGD is that it requires an initial, careful tuning of the learning rate η, that is also used across all parameters, without adapting to the characteristics of each parameter. However, if some entities are infrequent, the corresponding embedding vectors will tend to be updated less frequently during the learning process, and will require a longer time to be properly learned. For such a reason, the task of learning the model parameters in EBEMs by using SGD may require even days to terminate [11].

In order overcome such a limitation, we propose the adoption of *adaptive per-parameter learning rates* as a solution for reducing the learning time in EBEMs. The underlying idea consists in associating *smaller learning rates* to parameters updated more often (such as the embedding vectors of entities appearing more frequently) and *larger learning rates* to parameters updated less often. Specifically, while the SGD algorithm in Alg. alg:sgd uses a global, fixed learning rate η, we propose relying on methods that estimate the optimal learning rate for each parameter while still being tractable for learning large models. In particular, we consider the following criteria for selecting the optimal learning rates: the Momentum method [26], AdaGrad [15] and AdaDelta [31]. Each of these methods can be implemented in Alg. alg:sgd, by replacing the update to model parameters on line line:update as specified in the following.

Momentum Method The basic idea of this method is accelerating the progress along dimensions where the sign of the gradient does not change, while slowing the progress along dimensions where the sign of the gradient continues to change. This is done by keeping track of previous parameter updates with an exponential decay. The update step on line line:update of Alg. alg:sgd, in the Momentum method is given by:

$$\Delta_t \leftarrow \rho \Delta_{t-1} - \eta g_t,$$

where ρ is a hyperparameter controlling the decay of previous parameter updates.

AdaGrad The underlying idea in this method is that per parameter learning rates should grow with the inverse of gradient magnitudes: large gradients should have smaller learning rates, while small gradients should have larger learning rates, so that the progress along each dimension evens out over time. The update step on line line:update of Alg. alg:sgd, in AdaGrad is given by:

$$\Delta_t \leftarrow -\frac{\eta}{\sqrt{\sum_{j=1}^{t} g_j^2}} g_t,$$

where η is a global scaling hyperparameter. AdaGrad has been used on large scale learning tasks in a distributed environment [12].

AdaDelta This method uses an exponentially decaying average of squared gradients $E[g^2]$ and squared updates $E[\Delta^2]$, controlled by a decay term ρ, to give more importance to more recent gradients and updates. The update step on line line:update of Alg. alg:sgd, in AdaDelta is given by:

$$\Delta_t \leftarrow -\frac{\mathrm{RMS}[\Delta]_{t-1}}{\mathrm{RMS}[g]_t} g_t,$$

where $E[x]_t = \rho E[x]_{t-1} + (1 - \rho) x_t$ calculates the exponentially decaying average, $\mathrm{RMS}[x]_t = \sqrt{E[x^2]_t + \epsilon}$, and ϵ is an offset hyperparameter.

All these methods leverage each parameter's previous gradients for adaptively selecting the optimal learning rate. The additional space complexity provided by each of these methods is an additional accumulator for each parameter, containing its gradient history. We did not experience any sensible difference in runtimes in comparison with plain SGD.

4 Related Works

In this section we survey the most representative related works in the categories of *Probabilistic latent variable models* and *Embedding Models*, by jointly highlighting their main peculiarities and drawbacks.

Probabilistic Latent Variable Models Models in this class explain relations between entities by associating each entity to a set of intrinsic *latent attributes*. The term *latent* refers to the fact that the attributes are not directly observable in the data. Specifically, this class of models conditions the probability distribution of the relations between two entities on the latent attributes of such entities, and all relations are considered conditionally independent given the latent attributes. Similarly to Hidden Markov Models [19, 30], this allows the information to *propagate* through the network of interconnected latent variables.

An early model in this family is the *Stochastic Block Model* (SB) [29], which associates a *latent class* variable with each entity. In Fig. fig:latent (see left side), a simple SB for a social network is depicted. Here, each user

$u \in U$ is associated with a latent class variable Z_u which conditions both its attributes A_u, and its relations R_i with other users. The *Infinite (Hidden) Relational Model* [18,30] extends the SB by using Bayesian nonparametrics, so to automatically infer the optimal number of latent classes. The *Infinite Hidden Semantic Model* [25] further extends such model, to make use of constraints expressed in First Order Logic during the learning process, while the Mixed Membership Stochastic Block Model [1] extends the SB to allow entities to have mixed cluster-memberships. More recent works associate a set of *latent features* with each entity, instead of a single latent class. The *Nonparametric Latent Feature Relational Model* [23] is a latent feature model, which relies on Bayesian nonparametrics to automatically infer the optimal number of latent features during learning. Other approaches, such as [13], focus on the problem of inducing probabilistic logic programs. While showing interesting results in terms of predictive accuracy, a detailed analysis on the scalability issue, with particular reference to real-world large KGs is missing.

The main limitation of probabilistic latent variable models lies in the complexity of probabilistic inference and learning, which is intractable in general [19]. As a consequence, these models do not result to be fully appropriate for modeling large KGs.

Embedding Models Similarly to probabilistic latent feature models, in *Embedding Models* each entity in the KG is represented by means of a continuous k-dimensional *embedding vector* $\mathbf{e}_x \in \mathbb{R}^k$, encoding its intrinsic latent features within the KG. Nevertheless, models in this class do not necessarily rely on probabilistic inference for learning the optimal embedding vectors and this allows avoiding the issues related to the proper normalization of probability distributions, that may lead to intractable problems.

In RESCAL [24], the problem of learning the embedding vector representations of all entities and predicates is cast as a *tensor factorization* problem: by relying on a bilinear model, and by using a squared reconstruction loss, an efficient learning algorithm, based on regularized *Alternating Least Squares*,

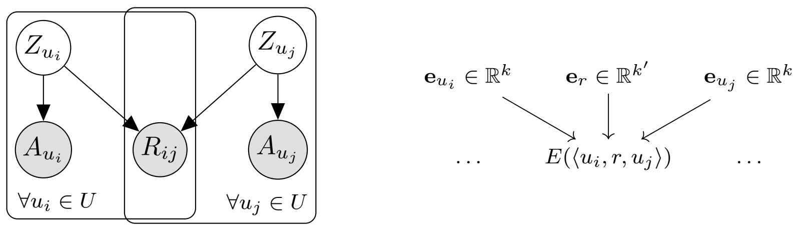
Z_{u_i}
Z_{u_j}
A_{u_i}
R_{ij}
A_{u_j}
$\forall u_i \in U$
$\forall u_j \in U$
$\mathbf{e}_{u_i} \in \mathbb{R}^k$
$\mathbf{e}_r \in \mathbb{R}^{k'}$
$\mathbf{e}_{u_j} \in \mathbb{R}^k$
$\ldots$
$E(\langle u_i, r, u_j \rangle)$
$\ldots$

Fig. 2: Left – A simple SB for a social network: each user $u \in U$ is associated with a latent class variable Z_u which conditions both its attributes A_u, and its relations with other users. Right – An example of EBEM: the k-dimensional embedding vector $\mathbf{e}_u \in \mathbb{R}^k$ of an entity u (e.g. an user in a social network) is used for computing the *energy* of all RDF triples in which u appears in.

is proposed. However, the number of parameters grows *super-linearly* with the number of predicates in the KG: for such a reason, RESCAL can hardly scale to highly-relational KGs [17]. In EBEMs (depicted on the right side of Fig. fig:latent), the *energy* of each RDF triple $\langle s, p, o \rangle$ is defined as a function of the embedding vectors $\mathbf{e}_s$ and $\mathbf{e}_o$, associated with the subject s and the object o of the triple (as already detailed in Sect. sec:models). The major limitation in EBEMs is the *learning time*, i.e. the time required for learning the parameters of the energy function. Several options have been proposed for the choice of both the *energy function* and the *loss functional* for learning the embedding vectors representation [7–9, 17, 28]. These methods have been used to achieve state-of-the-art link prediction results while scaling on large KGs.

We outperform such methods both in terms of efficiency (by reducing the learning time by an order of magnitude) and effectiveness (by obtaining a more accurate model) - as shown by the empirical evaluations provided in Sect. sec:evaluation.

5 Empirical Evaluation

In this section, we present the empirical evaluation for our proposed solution. Particularly, we aim at answering the following questions:

Q1: Can adaptive learning rates, as proposed in Sect. sec:learning, be used for improving the efficiency of parameters learning with respect to the current state-of-the-art EBEMs?
Q2: Do the energy functions proposed in Sect. sec:inference lead to more accurate link prediction models for KG completion?

In Sect. sec:adaptive, we answer **Q1** by empirically evaluating the efficiency of the proposed learning procedure and the accuracy of the learned models. In Sect. sec:functions, we answer **Q2** by evaluating the accuracy of models using the proposed energy functions in link prediction tasks.

In the following, we describe the KGs used for the evaluation, jointly with the adopted metrics.

Knowledge Graphs As KGs, WordNet [22] and Freebase (FB15k) [4] have been adopted:

- WordNet is a lexical ontology for the English language. It is composed of over 151×10^3 triples, describing 40943 entities and their relations by means of 18 predicate names.
- Freebase (FB15k) is a large collaborative knowledge base that is composed of over 592×10^3 triples, describing 14951 entities and their relations by means of 1345 predicate names.

As for the experiments, for comparison purpose, we use the very same training, validation and test sets adopted in [8]. Specifically, as regards WordNet, given the whole KG, 5000 triples were used for validation and 5000 were used for testing. As regards FB15k, 50000 triples were used for validation while 59071 were used for testing (the interested reader may refer to [8] for more informations about the creation of such datasets).

Evaluation Metrics As for [8], the following metrics have been used:

– *averaged rank* (denoted as Mean Rank)
– *proportion of ranks not larger than* 10 (denoted as Hits@10).

They have been computed as follows. For each test triple $\langle s, p, o \rangle$, the object o is replaced by each entity $\tilde{o} \in \mathcal{E}_G$ in G thus generating a *corrupted* triple $\langle s, p, \tilde{o} \rangle$. The energy values of corrupted triples are computed by the model, and successively sorted in ascending order. The rank of the correct triple is finally stored. Similarly, this procedure is repeated by corrupting the subject s of each test triple $\langle s, p, o \rangle$. Aggregated over all test triples, this procedure leads to the two metrics: the *averaged rank* (denoted as Mean Rank) that measures the average position of the true test triple in the ranking, and the *proportion of ranks not larger than* 10 (denoted as Hits@10) that measures the number of times the true test triple is ranked among the most likely 10 triples. This setting is referred to as the Raw setting.

Please note that, if a generated corrupted triple already exists in the KG, ranking it before the original triple $\langle s, p, o \rangle$ is not wrong. For such a reason, an alternative setting, referred to as Filtered setting (abbreviated with Filt.) is also considered. In this setting, corrupted triples that exist in either training, validation or test set are removed, before computing the rank of each triple.

In both Raw and Filtered settings, it would be desirable to have lower Mean Rank and higher Hits@10.

5.1 Evaluation of Adaptive Learning Rates

In order to reply to question **Q1**, that is, for assessing whether Momentum, AdaGrad and AdaDelta are more efficient than SGD in minimizing the loss functional in Eq. eq:loss$_m argin, weempiricallyevaluatedthesemethodsonthetaskoflearningtheparametersinTransEonV$
1001[8]$thatis$:

$k = 20$, $\gamma = 2$, $d = L_1$ for WordNet
$k = 50$, $\gamma = 1$, $d = L_1$ for FB15k.

Following the empirical comparison of optimization methods in [27], we compared SGD, Momentum, AdaGrad and AdaDelta using an extensive grid of hyperparameters. Specifically, given $\mathcal{G}_\eta = \{10^{-6}, 10^{-5}, \ldots, 10^{1}\}$, $\mathcal{G}_\rho = \{1 - 10^{-4}, 1 - 10^{-3}, \ldots, 1 - 10^{-1}, 0.5\}$ and $\mathcal{G}_\epsilon = \{10^{-6}, 10^{-3}\}$, the grids of hyperparameters for each of the optimization methods were defined as follows:

– **SGD** and **AdaGrad:** rate $\eta \in \mathcal{G}_\eta$.
– **Momentum:** rate $\eta \in \mathcal{G}_\eta$, decay rate $\rho \in \mathcal{G}_\rho$.
– **AdaDelta:** decay rate $\rho \in \mathcal{G}_\rho$, offset $\epsilon \in \mathcal{G}_\epsilon$.

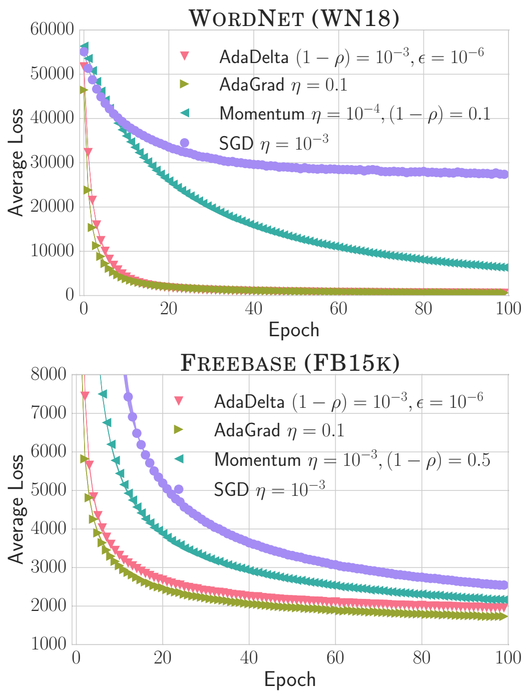

Fig. 3: Average loss across 10 TransE parameters learning tasks on WordNet (top) and Freebase FB15k (bottom) knowledge graphs, using the optimal settings in [8]. For each of the optimization methods, the hyperparameters settings that after 100 epochs achieve the lowest average loss are reported.

For each possible combination of optimization method and hyperparameter values, we performed an evaluation consisting in 10 learning tasks, each time using a different random seed for initializing the model parameters in TransE. The same 10 random seeds were used for each of the evaluation tasks.

Fig. fig:loss shows the behavior of the loss function for each of the optimization methods, for the best hyperparameter settings after 100 epochs, over the training set. It is immediate to see that, for both WordNet and FB15k, AdaGrad (with $\eta = 0.1$) and AdaDelta (with $(1-\rho) = 10^{-3}$ and $\epsilon = 10^{6}$) provide sensibly lower values of the loss functional $\mathcal{L}$ than SGD and Momentum, even after a low number of iterations (< 10 epochs), and that AdaGrad and AdaDelta, in their optimal hyperparameter settings, provide very similar loss values. Since AdaGrad has only one hyperparameter η and a lower complexity (it only requires one per parameter accumulator and a rescaling operation at each iteration) than AdaDelta, we select AdaGrad (with $\eta = 0.1$) as the optimization method of choice. Specifically, as a successive step, we needed to assess whether AdaGrad (with $\eta = 0.1$) leads to *more accurate models*, i.e. with lower Mean Rank and higher Hits@10, than SGD. For the purpose,

Table 2: **Link Prediction Results:** Test performance of several state-of-the-art Link Prediction methods on the WordNet and Freebase (FB15k) KGs. Results show the Mean Rank (the lower, the better) and Hits@10 (the higher, the better) for both the Raw and the Filtered settings [8].

Knowledge Graph	WordNet				Freebase (FB15k)			
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
Unstructured [7]	315	304	35.3	38.2	1074	979	4.5	6.3
RESCAL [24]	1180	1163	37.2	52.8	828	683	28.4	44.1
SE [9]	1011	985	68.5	80.5	273	162	28.8	39.8
SME linear [7]	545	533	65.1	74.1	274	154	30.7	40.8
SME bilinear [7]	526	509	54.7	61.3	284	158	31.3	41.3
LFM [17]	469	456	71.4	81.6	283	164	26.0	33.1
TransE [8]	263	251	75.4	89.2	243	125	34.9	47.1
TransE (AdaGrad)	**169**	**158**	**80.5**	**93.5**	**189**	**73**	**44.0**	**60.1**

we trained TransE by using AdaGrad (with $\eta = 0.1$) for 100 epochs on a link prediction task on WordNet and FB15k, under the same evaluation setting used in [8]. Hyperparameters were selected according to the performance on the validation set using the same grid of hyperparameters adopted in [8]. Specifically, we chose the margin $\gamma \in \{1, 2, 10\}$, the embedding vector dimension $k \in \{20, 50\}$, and the dissimilarity $d \in \{L_1, L_2\}$. Tab. tab:TransEa shows the results obtained by TransE trained using AdaGrad (with $\eta = 0.1$) for 100 epochs, in comparison with state-of-the-art results as reported in [8].

From Tab. tab:TransEa can be noted that, despite of the sensibly lower number of training epochs (100, compared to 1000 used for training TransE with SGD, as reported by [8]), TransE trained using AdaGrad provides more accurate link prediction models (i.e. lower Mean Rank and higher Hits@10 values) than every other model in the comparison. A possible explanation for this phenomenon is the following. AdaGrad uses each parameter's previous gradients for rescaling its learning rate: for such a reason, entities and predicates occurring less (resp. more) frequently will be associated with an higher (resp. lower) learning rate. As a result, the learning process for each parameter evens out over time, and all embedding parameters are learned at the same pace.

The results showed in this section largely prove that our solution is able to give a positive answer to **Q1**. Specifically, besides of experimentally proving that the adaptive learning rates proposed in Sect. 3.2 are able to improve the efficiency of parameters learning with respect to the current state-of-the-art EBEMs, we have also proved that the final learned model is able to outperform current state-of-the-art models in terms of Mean Rank and Hits@10.

5.2 Evaluation of the Proposed Energy Functions

In this section, we evaluate the energy functions proposed in Sect. sec:inference in the definition of an EBEM, with the final goal of replying to question **Q2**, that is to assess whether the proposed energy functions lead to more accurate link prediction models for KGs completion than models at the state-of-the-art.

As from (eq:energy), the energy function of an EBEM can be rewritten as:

$$E(\langle s, p, o \rangle) = g(f_s(\mathbf{e}_s, \mathbf{S}_p), f_o(\mathbf{e}_o, \mathbf{S}_p))$$

where $\mathbf{e}_s$ and $\mathbf{e}_o$ denote the embedding vectors of the subject s and the object o of the triple, and $\mathbf{S}_p$ denotes the set of embedding parameters associated with the predicate p. In Sect. sec:inference we proposed alternative choices for functions $f_s(\cdot)$ and $f_o(\cdot)$, that allow defining models whose number of parameters grows *linearly* with the number of entities and relations in the KG. Specifically, we proposed using *translation*, *scaling*, composition, and projection on the Euclidean unit sphere $n(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$. For each of the considered choices, we trained the corresponding EBEM on WordNet and FB15k. Hyperparameters were selected on the basis of the model performance on the validation set: we selected the embedding vector dimension $k \in \{20, 50, 100\}$, the margin $\gamma \in \{2, 5, 10\}$, and the $g(\cdot)$ function $g(\mathbf{x}, \mathbf{y}) \in \{\|\mathbf{x} - \mathbf{y}\|_1, \|\mathbf{x} - \mathbf{y}\|_2, -\mathbf{x}^T\mathbf{y}\}$, corresponding to the L_1 and L_2 distances, and the negative dot product. Following the results from Sect. sec:adaptive, model parameters were learned using AdaGrad (with $\eta = 0.1$) for 100 training epochs.

Tab. tab:results shows the test results obtained with different choices of $f_s(\cdot)$ and $f_o(\cdot)$ function. Note that we used the notation $\mathbf{e}_{p,1}$ and $\mathbf{e}_{p,2}$ for referring to two distinct predicate embedding vectors, one used in the formulation of $f_s(\cdot)$ and the other in $f_o(\cdot)$, for avoiding name clashing. For the purposes of comparison, Tab. tab:results also shows the results obtained, on the same link prediction tasks, by TransE (as reported in [8]) that is the best performing model in the literature.

From the table, it is interesting to note that, especially for highly multi-relational KGs such as Freebase (FB15k), *simpler models* for $f_s(\cdot)$ and $f_o(\cdot)$ provide better results than their more complex variants. A possible explanation is that many predicates in FB15k only occur in a limited number of triples (only 736 predicates out of 1345 occur in more than 20 triples) and in cases like this more expressive models are less likely to generalize correctly than simpler models. Given $f_o(\mathbf{e}_o, \{\mathbf{e}_p\}) = \mathbf{e}_o$, the best performing models, in terms of Hits@10, are:

- $f_s(\mathbf{e}_s, \{\mathbf{e}_p\}) = \mathbf{e}_s + \mathbf{e}_p$, representing the predicate-dependent *translation* of the subject's embedding vector
- $f_s(\mathbf{e}_s, \{\mathbf{e}_p\}) = \mathbf{e}_s \odot \mathbf{e}_p$, representing the predicate-dependent *scaling*.

This indicates that, despite the very different geometric interpretations, relying on simpler models improves link prediction results, especially in highly-relational KGs. This is probably due to the fact that, especially for the case of real-world KGs (which, by nature, tend to be very sparse), *simpler models*

Table 3: **Link Prediction Results:** Test performances of several EBEMs (on different choices of the $f_s(\cdot)$ and $f_o(\cdot)$ functions) in comparison with TransE [8] on WordNet and Freebase (FB15k). Results show the Mean Rank (the lower, the better) and Hits@10 (the higher, the better) in the Raw and Filtered settings.

Knowledge Graph	WordNet				Freebase (FB15k)			
Metric	Mean Rank		Hits@10 (%)		Mean Rank		Hits@10 (%)	
	Raw	Filt.	Raw	Filt.	Raw	Filt.	Raw	Filt.
TransE [8]	263	251	75.4	89.2	243	125	34.9	47.1
$f_s = \mathbf{e}_s + \mathbf{e}_p$, $f_o = \mathbf{e}_o$	**161**	**150**	**80.5**	**93.5**	**189**	**65**	**47.9**	**67.6**
$f_s = \mathbf{e}_s \odot \mathbf{e}_p$, $f_o = \mathbf{e}_o$	**229**	**215**	**81.4**	**93.5**	**207**	**81**	**46.5**	**65.3**
$f_s = (\mathbf{e}_s \odot \mathbf{e}_{p,1}) + \mathbf{e}_{p,2}$, $f_o = \mathbf{e}_o$	168	155	81.3	93.2	214	88	41.8	57.3
$f_s = \mathbf{e}_s + \mathbf{e}_{p,1}$, $f_o = \mathbf{e}_o + \mathbf{e}_{p,2}$	171	159	79.6	92.6	196	78	44.9	62.4
$f_s = \mathbf{e}_s \odot \mathbf{e}_{p,1}$, $f_o = \mathbf{e}_o \odot \mathbf{e}_{p,2}$	337	325	**83.0**	**95.2**	202	75	44.9	62.9
$f_s = (\mathbf{e}_s \odot \mathbf{e}_{p,1}) + \mathbf{e}_{p,2}$, $f_o = \mathbf{e}_o \odot \mathbf{e}_{p,3}$	279	266	82.4	94.3	210	88	42.3	59.1
$f_s = (\mathbf{e}_s \odot \mathbf{e}_{p,1}) + \mathbf{e}_{p,2}$, $f_o = (\mathbf{e}_o \odot \mathbf{e}_{p,3}) + \mathbf{e}_{p,4}$	320	308	81.6	93.6	211	87	40.0	54.9
$f_s = n(\mathbf{e}_s + \mathbf{e}_p)$, $f_o = \mathbf{e}_o$	211	200	75.7	88.7	237	115	39.5	55.4
$f_s = n(\mathbf{e}_s \odot \mathbf{e}_p)$, $f_o = \mathbf{e}_o$	226	213	77.6	89.2	262	132	42.0	59.9
$f_s = n((\mathbf{e}_s \odot \mathbf{e}_{p,1}) + \mathbf{e}_{p,2})$, $f_o = \mathbf{e}_o$	160	148	77.7	88.7	239	103	42.8	59.1
$f_s = n(\mathbf{e}_s + \mathbf{e}_{p,1})$, $f_o = n(\mathbf{e}_o + \mathbf{e}_{p,2})$	262	251	79.3	91.6	206	86	**47.5**	**66.5**
$f_s = n(\mathbf{e}_s \odot \mathbf{e}_{p,1})$, $f_o = n(\mathbf{e}_o \odot \mathbf{e}_{p,2})$	761	750	73.4	83.5	249	120	42.0	61.0
$f_s = n(\mathbf{e}_s \odot \mathbf{e}_{p,1} + \mathbf{e}_{p,2})$, $f_o = n(\mathbf{e}_o \odot \mathbf{e}_{p,3} + \mathbf{e}_{p,4})$	624	613	74.7	83.6	238	114	42.7	60.4

tend to generalize better and are less prone to over-fitting than more complex models. This characteristic can be very advantageous in real-world scenarios: relying on simpler models such as TransE (where the number of parameters scales *linearly* with the number of entities and predicates) can sensibly improve the training time, making learning from large and Web-scale KGs feasible.

We can conclude that, constraining the expressiveness of the models while using adaptive learning rates, yields a significant improvement over state-of-the-art methods discussed in [8].

Source code and datasets for reproducing the experiments presented in this paper are available on-line[7].

[7] `https://github.com/pminervini/ebemkg/`

6 Conclusions and Future Works

We focused on Energy-Based Embedding Models, a novel class of link prediction models for knowledge graph completion where each entity in the graph is represented by a continuous *embedding* vector. Models in this class, like the *Translating Embedding* model [8], have been used to achieve performance that is comparable with the main state-of-the-art methods while scaling on very large knowledge graphs.

In this work, we proposed: (i) a general framework for describing state-of-the-art Energy-Based Embedding Models, (ii) a family of novel energy functions, with useful properties, (iii) a method for improving the efficiency of the learning process by an order of magnitude, while leading to more accurate link prediction models. We empirically evaluated the adoption of the proposed adaptive learning rates in the context of Energy-Based Embedding Models by showing that they provide more accurate link prediction models while reducing the learning time by an order of magnitude in comparison with state-of-the-art learning algorithms. We also empirically evaluated the newly proposed energy functions (with a number of parameters) that scales *linearly* with the number of entities and relations in the knowledge graph. Our results showed a significant improvement over state-of-the-art link prediction methods on the very same considered large KGs, which are WordNet and Freebase.

For the future we plan to investigate on the formalization of Energy-Based Embedding Models that are able to take into account the available background knowledge. Other research directions include dynamically controlling the complexity of learned models, and further optimizing the learning process.

References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9, 1981–2014 (2008)
2. Bartlett, P.L., et al. (eds.): Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States (2012)
3. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the web of data. J. Web Sem. 7(3), 154–165 (2009)
4. Bollacker, K.D., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Wang, J.T. (ed.) Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008. pp. 1247–1250. ACM (2008)
5. Antoine Bordes, A., Gabrilovich, E.: Constructing and mining Web-scale Knowledge Graphs. WWW 2015 Tutorial. In: Gangemi, A., Leonardi, S., Panconesi, A. (ed.) Proceedings of the 24th International Conference on World Wide Web Companion, WWW 2015 - Companion Volume. ACM (2015)
6. Bordes, A., Gabrilovich, E.: Constructing and mining Web-scale Knowledge Graphs: KDD 2014 tutorial. In the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, page 1967, 2014.
7. Bordes, A., Glorot, X., Weston, J., Bengio, Y.: A semantic matching energy function for learning with multi-relational data - application to word-sense disambiguation. Machine Learning 94(2), 233–259 (2014)

8. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Burges et al. [10], pp. 2787–2795
9. Bordes, A., Weston, J., Collobert, R., Bengio, Y.: Learning structured embeddings of knowledge bases. In: Burgard, W., et al. (eds.) Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011. AAAI Press (2011)
10. Burges, C.J.C., et al. (eds.): Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States (2013)
11. Chang, K., Yih, W., Yang, B., Meek, C.: Typed tensor decomposition of knowledge bases for relation extraction. In: Moschitti, A., et al. (eds.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. pp. 1568–1579. ACL (2014)
12. Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Le, Q.V., Mao, M.Z., Ranzato, M., Senior, A.W., Tucker, P.A., Yang, K., Ng, A.Y.: Large scale distributed deep networks. In: Bartlett et al. [2], pp. 1232–1240
13. De Raedt, L., Dries, A., Thon, I., Van den Broeck, G., Verbeke, M. Inducing probabilistic relational rules from probabilistic examples, In: Qiang Yang, Q., Wooldridge, M. (eds.) Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI), 2015. 1835–1843 AAAI Press. (2015)
14. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., Zhang, W.: Knowledge vault: a Web-scale approach to probabilistic knowledge fusion. In: Macskassy, S.A., et al. (eds.) The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014. pp. 601–610. ACM (2014)
15. Duchi, J.C., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research 12, 2121–2159 (2011)
16. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. The MIT Press (2007)
17. Jenatton, R., Roux, N.L., Bordes, A., Obozinski, G.: A latent factor model for highly multi-relational data. In: Bartlett et al. [2], pp. 3176–3184
18. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proceedings, The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, July 16-20, 2006, Boston, Massachusetts, USA. pp. 381–388. AAAI Press (2006)
19. Koller, D., Friedman, N.: Probabilistic Graphical Models: principles and techniques. MIT Press (2009)
20. LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., Huang, F.: A tutorial on energy-based learning. In: Bakir, G., et al. (eds.) Predicting Structured Data. MIT Press (2006)
21. Mahdisoltani, F., Biega, J., Suchanek, F.M.: YAGO3: A knowledge base from multilingual Wikipedias. In: CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Online Proceedings (2015)
22. Miller, G.A.: Wordnet: A lexical database for English. Commun. ACM 38(11), 39–41 (1995)
23. Miller, K.T., Griffiths, T.L., Jordan, M.I.: Nonparametric latent feature models for link prediction. In: Bengio, Y., et al. (eds.) Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada. pp. 1276–1284. Curran Associates, Inc. (2009)
24. Nickel, M., Tresp, V., Kriegel, H.: A three-way model for collective learning on multi-relational data. In: Getoor, L., et al. (eds.) Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011. pp. 809–816. Omnipress (2011)
25. Rettinger, A., Nickles, M., Tresp, V.: Statistical relational learning with formal ontologies. In: Buntine, W.L., et al. (eds.) Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2009, Bled, Slovenia, September 7-11, 2009, Proceedings, Part II. LNCS, vol. 5782, pp. 286–301. Springer (2009)

26. Rumelhart, D.E., Hinton, G.E., Wilson, R.J.: Learning representations by back-propagating errors. Nature 323, 533–536 (1986)
27. Schaul, T., Antonoglou, I., Silver, D.: Unit tests for stochastic optimization. In: International Conference on Learning Representations. Banff, Canada (2014)
28. Socher, R., Chen, D., Manning, C.D., Ng, A.Y.: Reasoning with neural tensor networks for knowledge base completion. In: Burges et al. [10], pp. 926–934
29. Wang, Y.J., Wong, G.Y.: Stochastic blockmodels for directed graphs. Journal of the American Statistical Association 82(397), pp. 8–19 (1987)
30. Xu, Z., Tresp, V., Yu, K., Kriegel, H.: Infinite hidden relational models. In: UAI '06, Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence, Cambridge, MA, USA, July 13-16, 2006. AUAI Press (2006)
31. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR abs/1212.5701 (2012)