Mach Learn (2014) 94:233–259
DOI 10.1007/s10994-013-5363-6

A semantic matching energy function for learning with multi-relational data

Application to word-sense disambiguation

Antoine Bordes · Xavier Glorot · Jason Weston · Yoshua Bengio

Received: 16 May 2012 / Accepted: 18 April 2013 / Published online: 30 May 2013

Abstract Large-scale relational learning becomes crucial for handling the huge amounts of structured data generated daily in many application domains ranging from computational biology or information retrieval, to natural language processing. In this paper, we present a new neural network architecture designed to embed multi-relational graphs into a flexible continuous vector space in which the original data is kept and enhanced. The network is trained to encode the semantics of these graphs in order to assign high probabilities to plausible components. We empirically show that it reaches competitive performance in link prediction on standard datasets from the literature as well as on data from a real-world knowledge base (WordNet). In addition, we present how our method can be applied to perform word-sense disambiguation in a context of open-text semantic parsing, where the goal is to learn to assign a structured meaning representation to almost any sentence of free text, demonstrating that it can scale up to tens of thousands of nodes and thousands of types of relation.

Keywords Neural networks · Multi-relational data · Word-sense disambiguation

Editors: Ronan Collobert and Luke Zettlemoyer.

A. Bordes and X. Glorot contributed equally to this work.

A. Bordes (✉)
Heudiasyc UMR 7253, Université de Technlogie de Compiègne & CNRS, Compiègne, France
e-mail: antoine.bordes@utc.fr

X. Glorot · Y. Bengio
Université de Montréal, Montréal, QC, Canada

X. Glorot
e-mail: glorotxa@iro.umontreal.ca

Y. Bengio
e-mail: bengioy@iro.umontreal.ca

J. Weston
Google, New York, NY, USA
e-mail: jweston@google.com

1 Introduction

Multi-relational data, which refers to graphs whose nodes represent entities and edges correspond to relations that link these entities, plays a pivotal role in many areas such as recommender systems, the Semantic Web, or computational biology. Relations are modeled as triplets of the form (subject, relation, object), where a relation either models the relationship between two entities or between an entity and an attribute value; relations are thus of several types. Such data sources are equivalently termed multi-relational graphs. They can also be represented by 3-dimensional tensors, for which each slice represents an adjacency matrix for one relation. Multi-relational graphs are popular tools for encoding data via knowledge bases, semantic networks or any kind of database following the Resource Description Framework (RDF) format. Hence, they are widely used in the Semantic Web (Freebase, DBpedia, etc.)[1] but also for knowledge management in bioinformatics (GeneOntology, UMLS semantic network, etc.)[2] or natural language processing (WordNet),[3] to name a few. Social networks can also be represented using RDF.

In spite of their appealing ability for representing complex data, multi-relational graphs remain complicated to manipulate for several reasons. First, interactions are of multiple types and heterogeneous (various frequencies, concerning different subsets of entities, etc.). In addition, most databases have been built either collaboratively or (partly) automatically. As a consequence, data is noisy and incomplete: relations can be missing or be invalid, there can be redundancy among entities because several nodes actually refer to the same concept, etc. Finally, most multi-relational graphs are of very large dimensions in terms of numbers of entities and of relation types: Freebase contains more than 20 millions entities, DBpedia is composed of 1 billion triplets linking around 4 millions entities, GeneOntology contains more than 350k verified biological entities, etc. Conveniently represent, summarize or denoise this kind of data is now a central challenge in statistical relational learning (Getoor and Taskar 2007).

In this paper, we propose a new model to learn multi-relational semantics, that is, to encode multi-relational graphs into representations that capture the inherent complexity in the data, while seamlessly defining similarities among entities and relations and providing predictive power. Our work is based on an original energy function, which is trained to assign low energies (i.e. high probabilities) to plausible triplets of a multi-relational graph. This energy function, termed *semantic matching energy*, relies on a compact distributed representation: all elements (entity and relation type) are represented into the same relatively low (e.g. 50) dimensional embedding vector space. The embeddings are learnt by a neural network whose particular architecture and training process force them to encompass the original data structure. Unlike in previous work, in this model, relation types are modeled similarly as entities. In this way, entities can also play the role of relation type, as in natural language for instance, and this requires less parameters when the number of relation types grows. We show empirically that this model achieves competitive results on benchmark tasks of link prediction, i.e., generalizing outside of the set of given valid triplets.

We also demonstrate the flexibility and scalability of the semantic matching energy by applying it for word-sense disambiguation (WSD). The model can successfully be trained on various heterogeneous data sources (knowledge bases, free text, etc.), containing several

[1] Respect. available from freebase.com and dbpedia.org.

[2] Respect. available from geneontology.org and semanticnetwork.nlm.nih.gov.

[3] Available from wordnet.princeton.edu.

thousands of entities *and* of relation types, to jointly learn representations for words and for senses (defined as entities of a lexical knowledge base, WordNet). On two different evaluation test sets, the proposed approach outperforms both previous work for learning with multi-relational data and standard methods for unsupervised WSD.

To summarize, the main contributions of this paper are threefold:

– an original model for encoding multi-relational data, which represents relation types and entities in the same way, and is potentially able to scale up to larger numbers of relation types than previous work;
– a training algorithm based on a ranking objective, which allows to learn on large numbers of training samples and achieves competitive results on benchmarks of various dimensions;
– an adaptation of the model for word-sense disambiguation, which consists of the first successful direct application of relational embeddings of a knowledge base (WordNet) for natural language processing.

Note that this paper extends a shorter version (Bordes et al. 2012), which first introduced the model. However, the previous paper was only focused on the application to word-sense disambiguation, whereas the present paper has a wider scope and considers more problems involving multi-relational data. New elements are provided: a fresh (and cleaner) form of the bilinear formulation, new experiments comparing to the state-of-the-art in link prediction, entity ranking and WSD, a more comprehensive literature review, and more details on the model formulation and the training procedure. We also provide a link to an open-source implementation of the code and to the data used in this paper: http://goo.gl/bHWsK.

The paper is organized as follows. Section 2 presents a review of previous work on learning with multi-relational data. Section 3 introduces the semantic matching energy function and Sect. 4 its training procedure. Extensive experimental results are given in Sect. 5. Finally, the application to WSD is described in Sect. 6 and Sect. 7 concludes and sketches future work.

2 Previous work

Several methods have been explored to represent and encode multi-relational data, such as clustering approaches. Hence, Kemp et al. (2006) introduced the Infinite Relational Model, IRM, a nonparametric Bayesian model whose latent variables are used to discover meaningful partitions among entities and relations. This model provides a great interpretability of the data but suffers from a poor predictive power. Miller et al. (2009) refined this to allow entities to have a mixed cluster membership. Sutskever et al. (2009) proposed another refinement with the Bayesian Tensor Clustered Factorization model, BCTF, in which the nonparametric Bayesian framework is coupled with the learning, via collective matrix factorization, of distributed representations for the entities and relation types. Other proposals have consisted in improving the original model by adding first-order formulae with Markov Logic. Hence, MRC, for Multiple Relation Clustering (Kok and Domingos 2007), performs clustering of entities through several relation types simultaneously. Singla and Domingos (2006) presented another model based on Markov logic for the task of entity resolution (i.e. deciding whether two entities should be merged).

All these methods share the ability of providing an interpretation of the data but are slow and do not scale to very large databases due to the high cost of inference. Models based on tensor factorization can be faster and scale to larger data because of their continuous and usually convex optimization. Standard methods like CANDECOMP/PARAFAC

(CP) (Harshman and Lundy 1994) or those from (Tucker 1966) have been applied on multi-relational graphs. Franz et al. (2009) used CP for ranking data from RDF knowledge bases. Other directions have also been proposed derived from probabilistic matrix factorization for multi-dimensional data (Chu and Ghahramani 2009) or by adapting dimensionality reduction techniques such as SVD (Speer et al. 2008; Cambria et al. 2009). Recently, Nickel et al. (2011) presented RESCAL, an upgrade over previous tensor factorization methods, which achieves strong predictive accuracies on various problems. RESCAL represents entities by low dimensional vectors and relation types by low rank matrices, which are learnt using a collective learning process similar to that of CP, but with some relaxed constraints. It has achieved the best accuracies on many benchmarks of tensor factorization. RESCAL has been applied to the knowledge base YAGO (Nickel et al. 2012) and hence showed to scale well on data with large numbers of entities (few millions). However, RESCAL has never been tested on data with large numbers of relation types (YAGO has 87 such types).

Some approaches described above (e.g. BCTF, RESCAL) end up with a distributed representation of the entities and relation types obtained via factorizing or clustering the original data. A slightly different line of work consists in focusing on learning such representations, termed *embeddings*. This idea of learning embeddings has been successful in natural language processing via the framework of language models (Bengio et al. 2003) where an embedding per word is learnt in an unsupervised fashion: it has been shown that such representations can store key information about language (mostly syntactic similarities) that helps to improve performance on standard NLP tasks (Bengio 2008; Collobert et al. 2011). Bordes et al. (2010) adapted a related model to a small hand-crafted knowledge base and text for language understanding. For multi-relational data, Linear Relational Embeddings (Paccanaro 2000; Paccanaro and Hinton 2001) learn a mapping from the entities into a feature-space by imposing the constraint that relations in this feature-space are modeled by linear operations. In other words, entities are modeled by real-valued vectors and relations by matrices and parameters of both are learnt. This idea has been further improved in the Structured Embeddings (SE) framework of Bordes et al. (2011).

Our work lies in the same research area, since we also aim at learning distributed representations of multi-relational data, and we introduce several novel elements. First, unlike previous work (including BCTF, RESCAL or SE) we do not represent a relation type differently than any entity (by a matrix for instance). In our model, a relation type is represented by a vector (just like other entities) and shares the status and number of parameters of other entities. This is convenient when the number of such types is large or when they can also play the role of entities as we illustrate in Sect. 6 on a problem with more than 10k relation types. Second, we do not use a training process based on tensor reconstruction (as RESCAL) or on clustering (as BCTF), but on a predictive approach instead. The learning objective is essentially asking the model to perform link or entity prediction (i.e. filling an empty spot in a triple). This leads to an algorithm based on a ranking objective and using stochastic gradient descent and backpropagation for updating the parameters, which has a low computational complexity per epoch (independent on the number of entities and relation types). Third, we show that our model is flexible and can be successfully trained via multi-tasking on several heterogeneous sources. Finally, even if it is not presented in this paper, our approach could potentially be adapted to learn non-linear representations of multi-relational data by adding non-linear transfer functions such as *tanh* or *sigmoid* to the neural network. We empirically compare our model with IRM, BCTF, MRC, CP, RESCAL and SE on various tasks with very different properties in our experiments of Sects. 5 and 6. Even if the best performing methods from earlier papers differ from one task to another, our proposed approach is competitive for all of them. The closest counterparts are SE and RESCAL, but as we show experimentally, they are not able to handle large-scale multi-relational data as we propose, and either break or are outperformed when data dimensions grows (number of entities and/or of relation types). Note that, while preparing the final version of this work, Jenatton et al. (2012) introduced a method partly inspired by our approach, which achieves interesting performances.

3 Semantic matching energy function

This section introduces our model designed to embed multi-relational data into fully distributed representations via a custom energy function.

3.1 Notations

This work considers multi-relational databases as graph models. The data structure is defined by a set of nodes and a set of links. To each individual node of the graph corresponds an element of the database, which we term an *entity*, and each link defines a *relation* between entities. Relations are directed and there are typically several different kinds of relations. Let $\mathcal{C}$ denote the dictionary which includes all entities and relation types, and let $\mathcal{R} \subset \mathcal{C}$ be the subset of entities which are relation types. In the remainder of the paper, a relation is denoted by a triplet (*lhs*, *rel*, *rhs*), where *lhs* is the *left* entity, *rhs* the *right* one and *rel* the *type* of relation between them.

3.2 Main ideas

The main ideas behind our semantic matching energy function are the following.

– Named symbolic entities (entities *and* relation types) are associated with a d-dimensional vector space, termed the "embedding space", following previous work in neural language models (Bengio 2008). The ith entity is assigned a vector $E_i \in \mathbb{R}^d$. Note that more general mappings from an entity to its embedding are possible.[4]
– The semantic matching energy value associated with a particular triplet (*lhs*, *rel*, *rhs*) is computed by a parametrized function $\mathcal{E}$ that starts by mapping all symbols to their embeddings and then combines them in a structured fashion. Our model is termed "semantic matching" because $\mathcal{E}$ relies on a matching criterion computed between both sides of the triplet.
– The energy function $\mathcal{E}$ is optimized to be lower for training examples than for other possible configurations of symbols. Hence the semantic energy function can distinguish plausible combinations of entities from implausible ones, and can be used, for instance, to answer questions, e.g. corresponding to a triplet (*lhs*, *rel*, ?) with a missing *rhs*, by choosing among the possible entities a *rhs* with a relatively lower energy. See Lecun et al. (2006) for a review of energy-based learning.

[4]For example, if entities are not symbols but structured objects, such as images (Weston et al. 2010), which have attributes represented in a vector x, then the embedding for entity x could be obtained via a parametrized function e that maps x to $E_x = e_\theta(x)$, with θ learned by gradient-based learning and back-propagating the training criterion into θ.

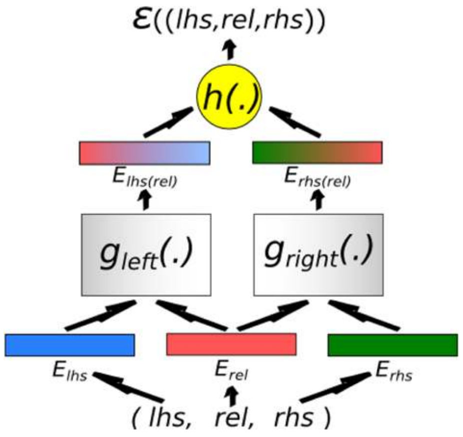


Fig. 1 Semantic matching energy function. A triple of entities (*lhs*, *rel*, *rhs*) is first mapped to its embeddings E_{lhs}, E_{rel} and E_{rhs}. Then E_{lhs} and E_{rel} are combined using $g_{left}(.)$ to output $E_{lhs(rel)}$ (similarly $E_{rhs(rel)} = g_{right}(E_{rhs}, E_{rel})$). Finally the energy $\mathcal{E}((lhs, rel, rhs))$ is obtained by matching $E_{lhs(rel)}$ and $E_{rhs(rel)}$ with the $h(.)$ function

Our approach is inspired by the framework introduced by Bordes et al. (2011) as well as by recent work of Bottou (2011). Our main motivation is to conceive a model where entities and relation types would share the same kind of representation. Embedding all symbols defining a multi-relational graph into the same space amounts to deleting the usual conceptual difference between entities (*lhs* and *rhs*) and relation types. This modeling is more natural when entities can act as *rel* as well as *lhs* or *rhs*. In Sect. 6, we apply our method to natural language data, where relation types typically correspond to verbs. Since verbs may also occur in *lhs* or *rhs*, it is reasonable to share the same representation, and this can ease learning representations of verbs appearing rarely as relation type. Our choice to encode symbols by vectors helps too by causing the overall number of tunable parameters to learn to remain low, especially when the cardinality of $\mathcal{R}$ grows.

3.3 Neural network parametrization

The energy function $\mathcal{E}$ (denoted SME) is encoded using a neural network, whose parallel architecture is based on the intuition that the relation type should first be used to extract relevant components from each argument's embedding, and put them in a space where they can then be compared (see Fig. 1). Hence, pairs (*lhs*, *rel*) and (*rel*, *rhs*) are first combined separately and then, these semantic combinations are *matched*.

(1) Each symbol of the input triplet (*lhs*, *rel*, *rhs*) is mapped to its embedding E_{lhs}, E_{rel} and $E_{rhs} \in \mathbb{R}^d$.
(2) The embeddings E_{lhs} and E_{rel} respectively associated with the *lhs* and *rel* arguments are used to construct a new relation-dependent embedding $E_{lhs(rel)}$ for the *lhs* in the context of the relation type represented by E_{rel}, and similarly for the *rhs*: $E_{lhs(rel)} = g_{left}(E_{lhs}, E_{rel})$ and $E_{rhs(rel)} = g_{right}(E_{rhs}, E_{rel})$, where g_{left} and g_{right} are parametrized functions whose parameters are tuned during training. Even if it remains low-dimensional, nothing forces the dimension of $E_{lhs(rel)}$ and $E_{rhs(rel)}$, which we denote p, to be equal to d, the one of the entity embedding space.
(3) The energy is computed by "matching" the transformed embeddings of the left-hand side and right-hand side: $\mathcal{E}((lhs, rel, rhs)) = h(E_{lhs(rel)}, E_{rhs(rel)})$, where h can be a simple operator such as a dot product or a more complex function whose parameters are learnt.

With this formulation, $\mathcal{E}$ is not able to handle variable-size arguments for *lhs* or *rhs* (like tuples of entities). However, it can be adapted to it by adding a pooling stage between steps (1) and (2) as we show in Sect. 6.1.

Different types of parametrizations can be used for the g and h functions. We chose to use a dot product for the output h function because it is simple and has shown to work well in related work (e.g. in Weston et al. 2010). For the g functions, we studied two options, one linear and the other bilinear, which lead to two versions of SME detailed below:

– *Linear form* (denoted SME(linear) in the following), in this case g functions are simply linear layers:

$$\begin{aligned} E_{lhs(rel)} &= g_{left}(E_{lhs}, E_{rel}) = W_{l1}E_{lhs}^{\intercal} + W_{l2}E_{rel}^{\intercal} + b_l^{\intercal}, \\ E_{rhs(rel)} &= g_{right}(E_{rhs}, E_{rel}) = W_{r1}E_{rhs}^{\intercal} + W_{r2}E_{rel}^{\intercal} + b_r^{\intercal} \end{aligned}$$

with $W_{l1}, W_{l2}, W_{r1}, W_{r2} \in \mathbb{R}^{p \times d}$ (weights), $b_l, b_r \in \mathbb{R}^p$ (biases) and $E^{\intercal}$ denotes the transpose of E. This leads to the following form for the energy:

$$\mathcal{E}\big((lhs, rel, rhs)\big) = -\big(W_{l1}E_{lhs}^{\intercal} + W_{l2}E_{rel}^{\intercal} + b_l^{\intercal}\big)^{\intercal}\big(W_{r1}E_{rhs}^{\intercal} + W_{r2}E_{rel}^{\intercal} + b_r^{\intercal}\big). \quad (1)$$

– *Bilinear form* (denoted SME(bilinear) in the following), in this case g functions are using 3-modes tensors as core weights:

$$\begin{aligned} E_{lhs(rel)} &= g_{left}(E_{lhs}, E_{rel}) = \big(W_l \bar{\times}_3 E_{rel}^{\intercal}\big)E_{lhs}^{\intercal} + b_l^{\intercal}, \\ E_{rhs(rel)} &= g_{right}(E_{rhs}, E_{rel}) = \big(W_r \bar{\times}_3 E_{rel}^{\intercal}\big)E_{rhs}^{\intercal} + b_r^{\intercal} \end{aligned}$$

with $W_l, W_r \in \mathbb{R}^{p \times d \times d}$ (weights) and $b_l, b_r \in \mathbb{R}^p$ (biases). $\bar{\times}_3$ denotes the n-mode vector-tensor product along the 3rd mode, which, for $U, V \in \mathbb{R}^d$ and $W \in \mathbb{R}^{p \times d \times d}$, is defined as (lower cased letters denote vector/tensor elements):

$$\forall i \in 1, \dots p, \quad \big(\big(W \bar{\times}_3 V^{\intercal}\big)U^{\intercal}\big)_i = \sum_{j=1}^{d}\sum_{k=1}^{d} w_{ijk}v_k u_j.$$

This leads to the following form for the energy:

$$\mathcal{E}\big((lhs, rel, rhs)\big) = -\big(\big(W_l \bar{\times}_3 E_{rel}^{\intercal}\big)E_{lhs}^{\intercal} + b_l^{\intercal}\big)^{\intercal}\big(\big(W_r \bar{\times}_3 E_{rel}^{\intercal}\big)E_{rhs}^{\intercal} + b_r^{\intercal}\big). \quad (2)$$

3.4 Discussion

We can notice that Eq. (1), defining the energy for SME(linear), can be re-written as (bias terms are removed for clarity):

$$\mathcal{E}\big((lhs, rel, rhs)\big) = -E_{lhs}\tilde{W}_1E_{rhs}^{\intercal} - E_{lhs}\tilde{W}_2E_{rel}^{\intercal} - E_{rel}\tilde{W}_3E_{rhs}^{\intercal} - E_{rel}\tilde{W}_4E_{rel}^{\intercal},$$

with $\tilde{W}_1 = W_{l1}^{\intercal}W_{r1}$, $\tilde{W}_2 = W_{l1}^{\intercal}W_{r2}$, $\tilde{W}_3 = W_{l2}^{\intercal}W_{r1}$ and $\tilde{W}_4 = W_{l2}^{\intercal}W_{r2} \in \mathbb{R}^{d \times d}$. Hence, the energy can be decomposed into three terms coding for pairs (*lhs*, *rhs*), (*lhs*, *rel*) and (*rel*, *rhs*), and an additional quadratic term for *rel*. This shows that SME(linear) actually represents a triplet as a combination of pairwise interactions (similar to what is captured by bigrams).

Similarly, Eq. (2), defining the energy for SME(bilinear) can be re-written as:

$$\mathcal{E}\big((lhs, rel, rhs)\big) = -E_{lhs}\tilde{W}(rel)E_{rhs}^{\intercal},$$

with $\tilde{W}(rel) = (W_l \bar{\times}_3 E_{rel}^\intercal)^\intercal (W_r \bar{\times}_3 E_{rel}^\intercal) \in \mathbb{R}^{d \times d}$. In this case, the energy is composed of a single term, which depends on all three entities, with a central role for *rel*. Hence, SME(bilinear) represents a triplet through 3-way interactions (similar to what is captured by a trigram). The choice between a linear or a bilinear form for g leads to a very different formulation overall.

The trigram formulation can model ternary interactions but requires more parameters. SME(bilinear) has $\mathcal{O}(n_e d + n_r d + p d^2)$ parameters while SME(linear) has only $\mathcal{O}(n_e d + n_r d + p d)$, with n_e the number of entities (never being a relation type), n_r the number of relation types, d the low-level embedding dimension and p the dimension of the higher-level relation-dependent representation (with both p and d much smaller than n_e and n_r). Still, both formulations can scale up to large numbers of relation types ($n_r \gg 1$) without requiring too many parameters, contrary to previous methods such as RESCAL and SE, which entail $\mathcal{O}(n_e d + n_r d^2)$ parameters. This property comes from our original choice of modeling *rel* in the same way as *lhs* and *rhs*, using vectors. Interestingly, we can remark that, in Eq. (2), $W_l \bar{\times}_3 E_{rel}^\intercal$ and $W_r \bar{\times}_3 E_{rel}^\intercal$ act as a pair of matrices coding for *rel*: this can be seen as a distributed or factorized version of what RESCAL or SE proposed.

4 Training

This section details the training procedure for the semantic matching energy function, SME.

4.1 Training criterion

We are given a training set $\mathcal{D}$ containing m triplets $x = (x_{lhs}, x_{rel}, x_{rhs})$, where $x_{lhs} \in \mathcal{C}$, $x_{rel} \in \mathcal{R}$, and $x_{rhs} \in \mathcal{C}$. We recall that the energy of a triplet is denoted $\mathcal{E}(x) = \mathcal{E}(x_{lhs}, x_{rel}, x_{rhs})$. Ideally, we would like to perform maximum likelihood over $P(x) \propto e^{-\mathcal{E}(x)}$ but this is intractable. The approach we follow here has already been used successfully in ranking settings (Collobert et al. 2011; Weston et al. 2010) and corresponds to performing two approximations. First, like in pseudo-likelihood we only consider one input at a time given the others, e.g. *lhs* given *rel* and *rhs*, which makes normalization tractable. Second, instead of sampling a negative example from the model posterior,[5] we use a ranking criterion (based on uniformly sampling a negative example, in a way that is reminiscent of Noise Contrastive Estimation (Gutmann and Hyvärinen 2010)).

Intuitively, if one of the elements of a given triplet were missing, then we would like the model to be able to predict the correct entity. The objective of training is to learn the semantic energy function $\mathcal{E}$ such that it can successfully rank the training samples x below all other possible triplets:

$$\mathcal{E}(x) < \mathcal{E}\big((i, x_{rel}, x_{rhs})\big) \quad \forall i \in \mathcal{C} : (i, x_{rel}, x_{rhs}) \notin \mathcal{D}, \tag{3}$$

$$\mathcal{E}(x) < \mathcal{E}\big((x_{lhs}, j, x_{rhs})\big) \quad \forall j \in \mathcal{R} : (x_{lhs}, k, x_{rhs}) \notin \mathcal{D}, \tag{4}$$

$$\mathcal{E}(x) < \mathcal{E}\big((x_{lhs}, x_{rel}, k)\big) \quad \forall k \in \mathcal{C} : (x_{lhs}, x_{rel}, j) \notin \mathcal{D}. \tag{5}$$

[5] In an energy-based model such as the Boltzmann machine, the gradient of the negative log-likelihood is equal to the gradient of the energy of a positive example (observed and valid) minus the expected value of the gradient of a negative example (sampled from the model). In the case of pseudo-likelihood training one would consider conditional likelihoods $P(x_i | x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_d)$, and only the x_i part of the positive example needs to be resampled for constructing the negative example, using this same posterior.

Towards achieving this, the following stochastic criterion is minimized:

$$\sum_{x \in \mathcal{D}} \sum_{\tilde{x} \sim Q(\tilde{x}|x)} \max\big(\mathcal{E}(x) - \mathcal{E}(\tilde{x}) + 1, 0\big) \tag{6}$$

where $Q(\tilde{x}|x)$ is a corruption process that transforms a training example x into a corrupted *negative example*. Note that $\max(\mathcal{E}(x) - \mathcal{E}(\tilde{x}) + 1, 0)$ is similar in shape to the negative log-likelihood $-\log \frac{e^{-\mathcal{E}(x)}}{e^{-\mathcal{E}(x)} + e^{-\mathcal{E}(\tilde{x})}} = -\log \operatorname{sigmoid}(\mathcal{E}(\tilde{x}) - \mathcal{E}(x))$, which corresponds to the probability of sampling x given that only x and $\tilde{x}$ are considered. In the experiments, Q only changes one of the three members of the triplet (as in a pseudo-likelihood setup), replacing it by an entity uniformly sampled either from $\mathcal{R}$ if the replaced entity is a relation type, or from $\mathcal{C}/\mathcal{R}$ otherwise. We do not actually check if the negative example is in $\mathcal{D}$. Note that this is not necessary because if we have the symmetry $Q((\tilde{a}, b, c)|(a, b, c)) = Q((a, b, c)|(\tilde{a}, b, c))$ etc. for all elements of the triplet, and it is true here, then the expected contribution to the total expected gradient due to cases where $\tilde{x} \in \mathcal{D}$ is 0. This is because if we consider only the pairs $x, \tilde{x} \in \mathcal{D}$, the average over $\mathcal{D}$ of the gradients $\frac{\partial \mathcal{E}(x)}{\partial \theta}$ equals the average over $\mathcal{D}$ of the gradients $\frac{\partial \mathcal{E}(\tilde{x})}{\partial \theta}$, by our symmetry assumption.

4.2 Ranking algorithm

To train the parameters of the energy function $\mathcal{E}$ we loop over all of the training data resources and use stochastic gradient descent (Robbins and Monro 1951). That is, we iterate the following steps:

1. Select a positive training triplet $x_i = (lhs_i, rel_i, rhs_i)$ at random from $\mathcal{D}$.
2. Select at random resp. constraint (3), (4) or (5).
3. Create a negative triplet $\tilde{x}$ by sampling one entity either from $\mathcal{R}$ to replace rel_i or from $\mathcal{C}/\mathcal{R}$ to replace lhs_i or rhs_i.
4. If $\mathcal{E}(x_i) > \mathcal{E}(\tilde{x}) - 1$, make a stochastic gradient step to minimize (6).
5. Enforce the constraint that each embedding is normalized, $\|E_i\| = 1, \forall i$.

The gradient step requires a learning rate λ. The constant 1 in step 4 is the *margin* as is commonly used in many margin-based models such as SVMs (Boser et al. 1992). The normalization in step 5 helps remove scaling freedoms from the model, makes the impact of the margin actually effective and regularizes the optimization objective, preventing weights to collapse or diverge.

Each update of the model parameters is carried out by backpropagation. Its computational complexity is dominated by the cost of the n-mode vector-tensor product for SME(bilinear): $\mathcal{O}(pd^2)$, and by the cost of the matrix product for SME(linear): $\mathcal{O}(pd)$. Therefore, to perform one epoch over $\mathcal{D}$, the bilinear form requires in the order of $\mathcal{O}(mpd^2)$ operations and the linear form in the order of $\mathcal{O}(mpd)$. *Note that this is independent of the number of entities or relation types.*

Matrix E contains the representations of the entities and is learnt via a *multi-task learning* procedure (Caruana 1995; Collobert and Weston 2008) because the same embedding matrix is used for all relation types (each corresponding to a different distribution of entities, i.e., a different task). As a result, the embedding of an entity contains factorized information coming from all the relations in which the entity is involved as *lhs*, *rhs* or even *rel*. For each entity, the model is forced to learn how an entity interacts with other entities in many different ways. One can think of the elements $e_{i,j}$ of each embedding vector E_i as *learned attributes* for entity i or relation type i. Different tasks may demand different attributes, so that entities that have a common behavior[6] in some way will get the same values for some of their attributes. If the same attributes can be useful for several tasks, then statistical power is gained through parameter sharing, and transfer of information between tasks can happen, making the data of some task informative for generalizing properly on another task.

4.3 Implementation details

All the code for the experiments has been implemented in Python and using the Theano library (Bergstra et al. 2010). Training is carried out using mini-batches (we create 200 mini-batches for each dataset, independent of its size). All hyperparameter values are set using a validation set. The dimension of the embeddings (d) and the dimension of the output space of g functions (p) are selected among $\{10, 25, 100\}$. There is a different learning rate for the embedding matrix E and for the parameters of g. It is chosen among $\{0.03, 0.01, 0.003, 0.001, 0.0003\}$ for E and among $\{3., 1., 0.3, 0.1, 0.03\}$ for g. Training stops using early stopping on the validation set error (or after a maximum of 2,000 epochs).

5 Empirical evaluation

This section proposes an experimental comparison of SME with current state-of-the-art methods for learning representations of multi-relational data.

5.1 Datasets

In order to evaluate against existing methods, we performed experiments on benchmarks from the literature. Kinships and UMLS are fully observed, i.e. for each relation type and each potential pair of entities it has been observed whether the given triplet is valid or not. They are also sparse, i.e. only a small fraction of triplets are valid. We also illustrate the properties of our model on Nations and WordNet , which are partially observed: we only observe some valid triplets in the case of WordNet and some valid or invalid triplets for Nations. The rest is unknown, that is, missing triplets can be valid or not. And of course, in that case only a tiny fraction of potential triplets are observed. We describe all datasets below, with some statistics displayed in Table 1.

Table 1 Statistics of datasets used in our experiments. The top two are fully observed, very sparse i.e. only a small minority of relations are valid and hence are used in a cross-validation scheme. Only a fraction of relations are observed in Nations and WordNet

Dataset	Nb. of relation types	Nb. of entities	Nb. of observed relations	% valid relations in obs. ones
UMLS	49	135	893,025	0.76
Kinships	26	104	281,216	3.84
Nations	56	14	11,191	22.9
WordNet	18	40,943	151,442	100

[6]E.g., appear in semantically similar contexts, i.e., in instances containing the same entities or ones with close-by values of their embedding.

Table 2 Relation types of WordNet used in our experiments

WordNet
_hypernym, *_hyponym*, *_instance_hyponym*, *_instance_hypernym*, *_related_form*, *_has_part*, *_part_of*, *_member_has_part*, *_member_part_of*, *_also_see*, *_attribute*, *_synset_domain_region*, *_synset_domain_usage*, *_synset_domain_topic*, *_verb_group*, *_member_of_domain_region*, *_member_of_domain_usage*, *_member_of_domain_topic*

UMLS This dataset contains data from the Unified Medical Language System semantic work gathered by McCray (2003). This consists in a graph with 135 entities and 49 relation types. The entities are high-level concepts like 'Disease or Syndrome', 'Diagnostic Procedure', or 'Mammal'. The relations represent verbs depicting causal influence between concepts like 'affect' or 'cause'.

Nations This dataset groups 14 countries (Brazil, China, Egypt, etc.) with 56 binary relation types representing interactions among them like 'economic_aid', 'treaties' or 'rel_diplomacy', and 111 features describing each country. See Rummel (1999) for details.

Kinships Australian tribes are renowned among anthropologists for the complex relational structure of their kinship systems. This dataset, created by Denham (1973), focuses on the Alyawarra, a tribe from Central Australia. 104 tribe members were asked to provide kinship terms for each other. This results in a graph of 104 entities and 26 relation types, each of them depicting a different kinship term, such as Adiadya or Umbaidya. See Denham (1973) or Kemp et al. (2006) for more details.

WordNet This knowledge base is designed to produce intuitively usable dictionary and thesaurus, and supports automatic text analysis. It encompasses comprehensive knowledge within its graph structure, whose entities (termed *synsets*) correspond to senses, and relation types define lexical relations between those senses. We considered all the entities that were connected with the relation types given in Table 2, although we did remove some entities for which we have too little information: we filtered out the synsets appearing in less that 15 triplets. We obtain a graph with 40,943 synsets and 18 relations types. Examples of triplets are (*_score_NN_1*, *_hypernym*, *_evaluation_NN_1*) or (*_score_NN_2*, *_has_part*, *_musical_notation_NN_1*). As WordNet is composed of words with different meanings, we describe its entities by the concatenation of the word, its part-of-speech tag ('NN' for noun, 'VB' for verb, 'JJ' for adjective and 'RB' for adverb) and a digit indicating which sense it refers to i.e. *_score_NN_1* is the entity encoding the first meaning of the noun "score". This version of WordNet is different from that used in Bordes et al. (2011) because the original data has been preprocessed differently: this version contains less entities but more relation types.

5.2 Link prediction

The link prediction task consists in predicting whether two entities should be connected by a given relation type. This is useful for completing missing values of a graph, forecasting the behavior of a network, etc. but also to assess the quality of a representation. We evaluate our model on UMLS, Nations and Kinships, following the setting introduced in Kemp et al. (2006): data tensors are split in ten folds of valid configurations using (*lhs*, *rel*, *rhs*) triplets as statistical units, and experiments are performed by cross-validation. The standard evaluation

Table 3 Link prediction. Comparisons of area under the precision-recall curve (AUC) computed in a 10-fold cross-validation setting between two versions of SME (this paper) and previously published algorithms (SE, RESCAL, CP, BCTF, MRC, IRM) on UMLS, Nations and Kinships. Emb. is an unstructured version of SME. Best performing methods, with a significant difference with the rest, are indicated in bold

Method	UMLS	Nations	Kinships
SME(linear)	**0.979** ± 0.003	0.777 ± 0.025	0.149 ± 0.003
SME(bilinear)	**0.985** ± 0.003	**0.865** ± 0.015	0.894 ± 0.011
Emb.	0.035 ± 0.002	0.345 ± 0.025	0.038 ± 0.001
SE	**0.983** ± 0.004	**0.869** ± 0.016	0.913 ± 0.006
RESCAL	**0.98**	0.84	**0.95**
CP	0.95	0.83	**0.94**
BCTF	**0.98**	n/a	0.90
MRC	**0.98**	0.75	0.85
IRM	0.70	0.75	0.66

metric is *area under the precision-recall curve* (AUC). For our own models, we used one of the nine training folds for validation. Table 3 presents results of SME along with those of RESCAL, BCTF, MRC, IRM and CP, which have been extracted from Kemp et al. (2006), Kok and Domingos (2007), Sutskever et al. (2009), Nickel et al. (2011) and that of SE, which we computed ourselves. Table 3 also displays performance of an *unstructured version* of SME, termed Emb.: in this case, the score of a triplet (*lhs*, *rel*, *rhs*) is simply determined by the dot product between *lhs* and *rhs* embeddings, without any influence of the relation type.

The linear formulation of SME is outperformed by SME(bilinear) on all three tasks. The largest differences for Nations and Kinships indicate that, for these problems, a joint interaction between both *lhs*, *rel* and *rhs* is crucial to represent the data well: relations cannot be simply decomposed as a sum of bigrams. This is particularly true for the complex kinship systems of the Alyawarra. On the contrary, interactions within the UMLS network can be represented by simply considering the various (entity, entity) and (entity, relation type) bigrams. Compared to other methods, SME(bilinear) performs similarly to SE, RESCAL, BCTF and MRC on UMLS and similarly to SE on Nations. It is worth noting than, on Nations, SE and SME(bilinear) perform better by a vast margin. On Kinships, it is outperformed by RESCAL and CP: on this dataset with complex ternary interactions, the training process of these tensor factorization methods, based on reconstruction, seems to be beneficial compared to our predictive approach. Simply representing a relation by a vector might also be detrimental w.r.t. using matrices, but SME reaches similar performance as SE (using matrices and a predictive training process). Still, compared to MRC, which is not using a matrix-based encoding, SME(bilinear) remains highly competitive. As expected, Emb. performs poorly, outlining the crucial influence of *rel* for correctly modeling such data.

To summarize, on these 3 benchmarks with moderate sizes, our method is either state-of-the-art (represented mainly by SE and RESCAL) or very close to it, and can be considered as the best performing method on average. However, SME is primarily designed for large-scale conditions and we show in the following that it outperforms RESCAL when the number of entities increases (in Sect. 5.3) and SE when the number of relation types does (in Sect. 6.4.1).

5.3 Entity ranking

Performing an evaluation based on link prediction for WordNet is problematic because only positive triplets are observed. Hence, in this case, there is no negative triplet but only unknown ones for which it is impossible to state whether they are valid or not. For this setting,

Table 4 Entity ranking on WordNet. Comparisons between two versions of SME (this paper) and SE, RESCAL and Emb., an unstructured version of SME. Mean/median predicted rank and precision@10 (p@10, in %) are computed on the test set. Best performances are indicated in bold, worst in italic

Method	Rank (median / mean)	p@10
SME(linear)	5 / 559	6.51
SME(bilinear)	8 / 526	5.47
Emb.	*26* / **317**	*3.51*
SE	**3** / *1011*	**6.85**
RESCAL	12 / 893	4.76

for which we only have access to positive training and test examples, AUC is not a satisfying metric anymore. Hence, we evaluate our model on this data using the ranking setting proposed in Bordes et al. (2011) and described below, which allows an analysis on positive samples only.

We measure the mean and median predicted ranks and the prediction@10, computed with the following procedure. For each test triplet, the left entity is removed and replaced by each of the entities of the dictionary in turn. Energies of those corrupted triplets are computed by the model and sorted by ascending order and the rank of the correct synset is stored. This whole procedure is also repeated when removing the right-hand argument instead. We report the mean and median of those predicted ranks and the precision@10 (or p@10), which is the proportion of ranks within 1 and 10, divided by 10. WordNet data was split in training, validation and test sets with 141,442 observed triplets for training, 5,000 for validation and 5,000 for testing.

Table 4 presents comparative results on the test set, together with the performance of Emb., SE and RESCAL, which we all computed. No method is able to perform best for all metrics. SE obtains a low median rank and the best p@10, but has the worst mean rank. This indicates an instability: SE works very well for most examples but can be terrible in some cases. In the original paper introducing SE, Bordes et al. (2011) proposed to stack a Kernel Density Estimator (KDE) on top of the structured embeddings to improve stability. However, throughout this paper, when we refer to SE, we mean *without* KDE, because this makes a fairer comparison. We could also stack KDE on top of SME but this involves a very expensive extra-cost, that forbids any large-scale ambition. Emb. performs quite well and reaches the best mean rank indicating that, on WordNet, the influence of the relation type is not crucial to get a fair rough estimate of the likely *lhs* given *rhs*, and vice-versa. Still, Emb. does not solve the task of handling multi-relational data, since it simply ignores the relation types. This seems to be fine on this task, but it was terrible on those of the previous section.

SME is the only method able to perform well for all metrics (while never reaching on top). In particular, SME(linear) is very close to SE in median rank and p@10 while being much better in mean rank: it does not seem to suffer from instability issues. It is hard on WordNet to be accurate on average (low mean rank) and still have a large proportion of examples very well ranked (low median rank) and SME appears to be the best for this compromise. RESCAL performs consistently worse than SME. We tried very hard to make the code provided by the authors work as well as possible: to behave properly, the model requires large latent dimensions d but this slows it down a lot. Results of Table 4 have been obtained with $d = 2000$ and a training time of almost 2 days (compared to around 4 h for SME).

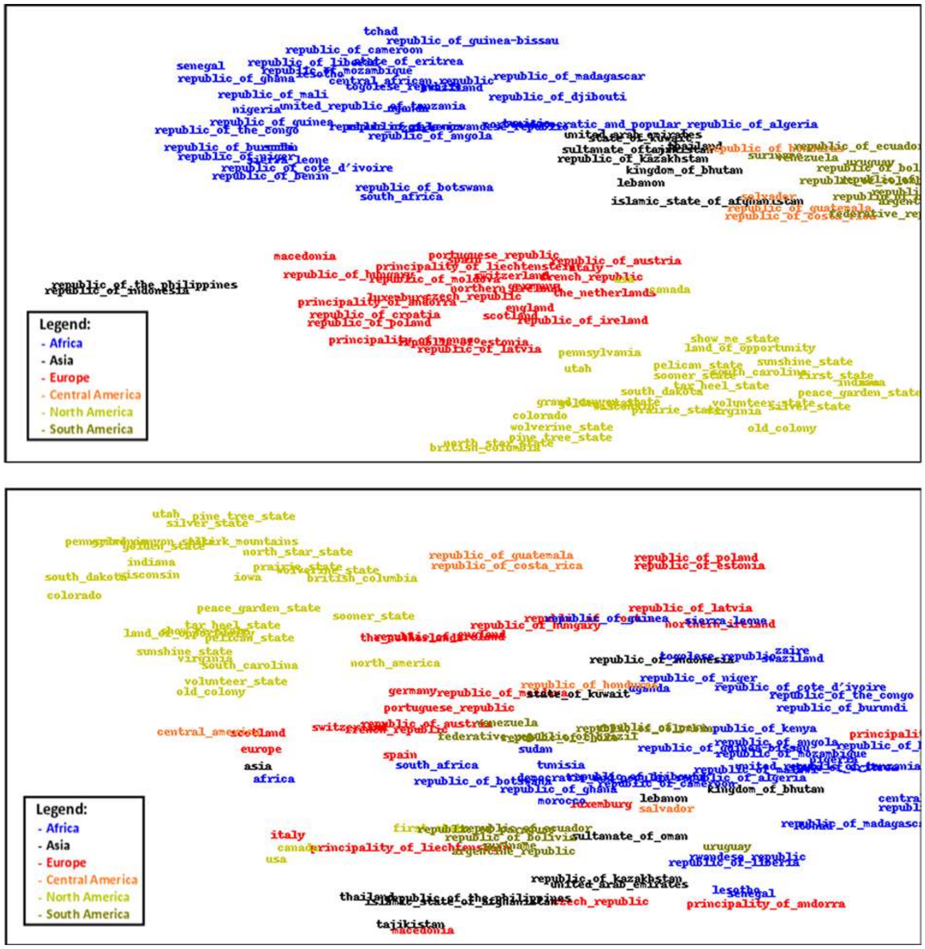


Fig. 2 Entity embeddings. Plots of representations (matrix E), learnt by SME(linear) (*top*) and SME(bilinear) (*bottom*), for 115 countries selected from WordNet and projected in 2D by t-SNE. SME(linear) encoded geographical similarities within the embeddings

5.4 Entity embeddings

The matrix E factorizes information from all relations in which the entity appears. We propose here to illustrate the kind of semantics captured by the representations.

We selected 115 entities from WordNet corresponding to countries from all over the world and to U.S. states. We selected this subset because we know that there exist an underlying structure among them. Then, we projected the corresponding embeddings learnt by the linear and bilinear versions of SME and created 2D plots using t-SNE (van der Maaten and Hinton 2008). They are given in Fig. 2: a different color is used for each continent; suffixes depicting POS tag and sense indices have been removed for clarity.

The representations learnt by the linear model seem to nicely reflect the geographical semantics, hence encoding the "part-of" information contained in WordNet: nice clusters are formed for each continent. To assess more objectively the quality of this plot, Fig. 3 proposes the one obtained for the same entities with the Lesk similarity measure of the

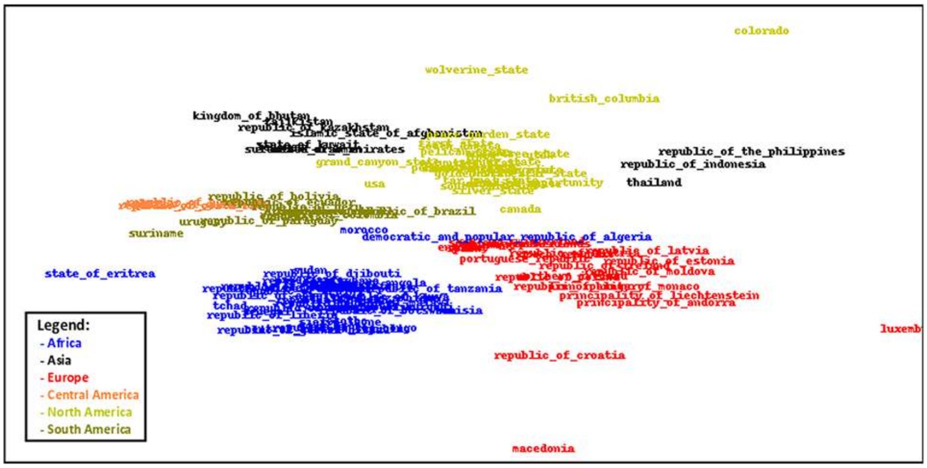


Fig. 3 Pairwise similarities. A pairwise similarity matrix of 115 countries from WordNet is computed with the Lesk measure of Wordnet::Similarity and projected in 2D by t-SNE

WordNet::Similarity package (Banerjee and Pedersen 2002).[7] We tried several measures and chose Lesk because it gave the best result. Comparing both plots tends to indicate that embeddings learnt by SME(linear) could be used to form a very decent similarity measure on WordNet. But, the comparison is not fair because the Lesk measure does not only rely on the WordNet graph but is also improved using *glosses* (i.e. definitions). Performing the same experiment with WordNet::Similarity measures based only on the graph gives much worse results. SME seems able to nicely capture the multi-relational semantics of the WordNet graph, without any other source of information.

The picture changes with the representations learnt by the bilinear models: the plot (bottom of Fig. 2) is much harder to interpret and suggests that the interactions occurring in the SME(bilinear) neural network are more complex, with a more invasive role for the relation type. This intuition is confirmed by the plots of Fig. 4. They still display t-SNE projections of representations of the same models for the same entities but not taken at the same level in the network. In this case, we projected the representations obtained by the embeddings when combined with the embedding of the relation type *_part_of* by the g_{left} function. In other words, these are plots of $E_{lhs(rel)}$. The top plot corresponds to the linear model and resemble to the one of Fig. 2: as expected, the linear g_{left} does not have a dramatic effect on the embedding landscape. The bottom plot, depicting SME(bilinear), is much more interesting because it shows that what was messy at the root level is much more organized: clusters are now formed for continents with the one corresponding to U.S. states further apart from the countries. Embeddings of SME(bilinear) are more interpretable *given a relation type*. The bilinear g functions drastically modify the distances within the embedding space depending on the relation type, as we expect that it should.

This last remark indicates that, by encoding data with SME(bilinear), one can expect that similarities between two entities existing given one relation (i.e. short distances between their transformed embeddings) would not automatically translate into a similarity between them for any relation type. This kind of behavior seems much harder to reproduce

[7]Freely available from wn-similarity.sourceforge.net.

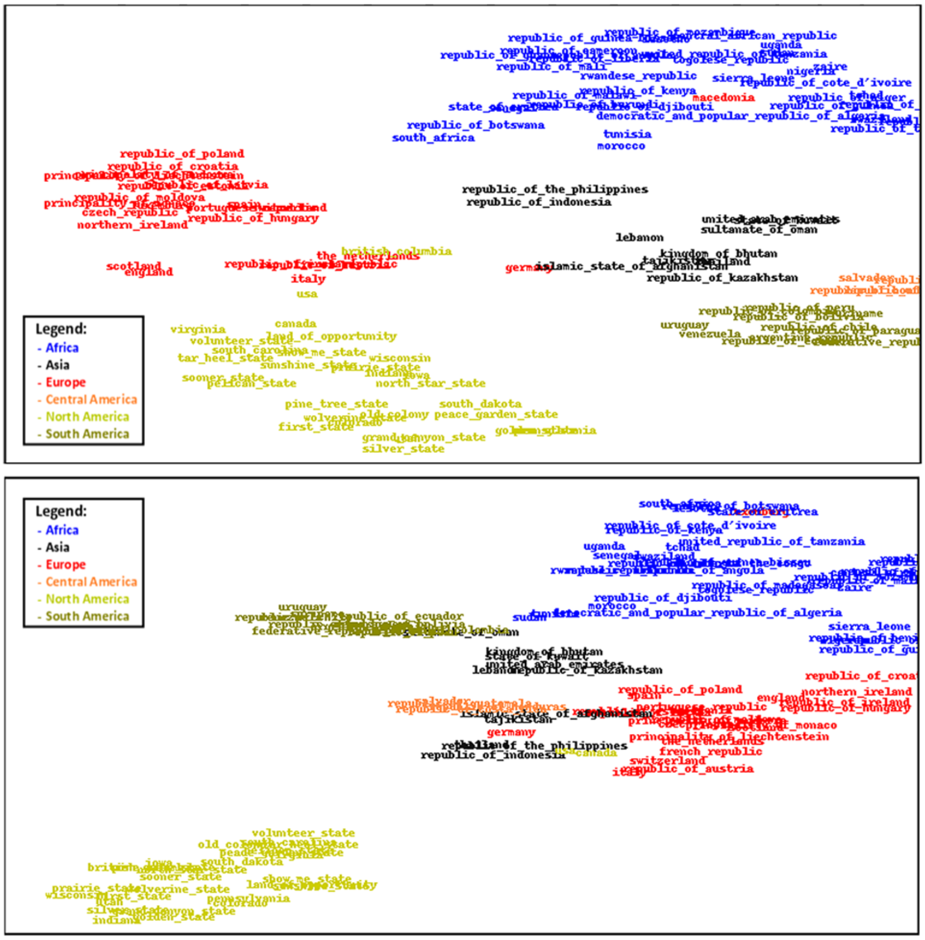
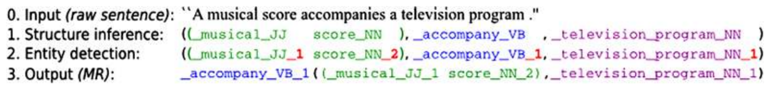


Fig. 4 Entity-relation embeddings. Plots of $E_{lhs(rel)}$ representations, learnt by SME(linear) (*top*) and SME(bilinear) (*bottom*), with 115 countries selected from WordNet as *lhs* and *_part_of* as *rel*, projected in 2D by t-SNE. SME(linear) and SME(bilinear) encoded geographical similarities at this stage

for SME(linear), where a similarity between entities seems to exist independent of the relation. This could explain the bad performance of this model on Kinships, where correct associations highly depend on the relation types. Still, drastic relations like antonymy remain unlikely to be well encoded by SME(bilinear), this might require to add non-linearities to the model or to use larger sparse representations (to code for orthogonality among entities).

6 Application for word-sense disambiguation

We have introduced a new neural network architecture for learning multi-relational semantics. Its stochastic learning process and its distributed representation of entities *and* relations allow it to scale to large graphs in terms of nodes and link types. In this section, we illustrate



```
0. Input (raw sentence):  ``A musical score accompanies a television program ."
1. Structure inference:   ((_musical_JJ   score_NN  ),_accompany_VB  ,_television_program_NN  )
2. Entity detection:      ((_musical_JJ_1 score_NN_2),_accompany_VB_1,_television_program_NN_1)
3. Output (MR):           _accompany_VB_1((_musical_JJ_1 score_NN_2),_television_program_NN_1)
```

Fig. 5 Open-text semantic parsing on simple sentences. To parse an input sentence (0.), a preprocessing (lemmatization, POS, chunking, SRL) is first performed (1.) to clean data and uncover the MR structure. Then, to each lemma is assigned a corresponding WordNet synset (2.), hence defining a complete meaning representation (3.)

these appealing properties by applying our model for learning to carry out all-words word-sense disambiguation on knowledge extracted from free text with semantic role labeling (SRL), with a view to performing open-text semantic parsing.

Semantic parsing (Mooney 2004) aims at building systems able to read text and express its meaning in a formal representation i.e. able to interpret statements expressed in natural language. The purpose of a semantic parser is to analyze the structure of sentence meaning and, formally, this consists of mapping a natural language sentence into a logical *meaning representation* (MR). Open-text semantic parsing consists of learning to associate a MR to any kind of natural language.

SME could make an interesting piece of a SRL system, especially for producing MRs of the following form: *relation*(*subject*, *object*), i.e. relations with subject and object arguments, where each component of the resulting triplet refers to a disambiguated entity, via the following two-stages process: (1) SRL step predicts the semantic structure, and (2) a disambiguation step assigns a corresponding entity to each relevant word, so as to minimize an energy given to the whole input. Even if such a SRL system using SME would not be able to cover the whole range of sentences and solve the highly complex problem of open-text semantic parsing, it could make a useful tool. Its process is illustrated in Fig. 5 and detailed in the next section. Our focus is on the application of SME for Step (2), which is an all-words WSD step on extracted knowledge.

6.1 Methodology

This section details how SME could be inserted into a simple open-text semantic parsing system. In this framework, MRs are simple logical expressions $REL(A_0, \dots, A_n)$, where REL is the relation symbol, and $A_0, \dots, A_n$ are its arguments. Note that several such forms can be recursively constructed to build more complex structures. The goal is to parse open-domain raw text so a large set of relation types and arguments should be considered. Hence, WordNet is used for defining REL and A_i arguments as proposed in Shi and Mihalcea (2004), using the version introduced in Sect. 5.1. This results in a dictionary of 70,116 words that can be mapped to 40,943 possible entities. The simplified semantic parsing process consists of two stages.

Step (1): MR structure inference The first stage consists in preprocessing the text and inferring the structure of the MR. For this stage we use standard approaches, the major novelty of our work lies in applying SME for step (2).

We use the SENNA software[8] (Collobert et al. 2011) to perform part-of-speech (POS) tagging, chunking, lemmatization[9] and SRL. In the following, we call a *lemma* the concatenation of a lemmatized word and a POS tag (such as *_score_NN* or *_accompany_VB*). Note

[8]Freely available from ml.nec-labs.com/senna/.

[9]Lemmatization is not carried out with SENNA but with the NLTK toolkit (nltk.org) and transforms a word into its canonical or base form.

the absence of an integer suffix, which distinguishes a lemma from a WordNet synset: a lemma is allowed to be semantically ambiguous. The SRL step consists in assigning a semantic role label to each grammatical argument associated with a verb for each proposition.

As a simplification, only sentences that match the template (*subject*, *verb*, *direct object*) are considered here. Each of the three elements of the template is associated with a tuple of lemmatized words (i.e. with a multi-word phrase) and SRL is used to structure the sentence into the ($lhs =$ subject, $rel =$ verb, $rhs =$ object) template. The order is not necessarily subject / verb / direct object in the raw text (e.g. in passive sentences). Clearly, the subject-verb-object composition causes the resulting MRs to have a straightforward structure (with a single relation), but this pattern is common and a good choice to test our ideas at scale. Learning to infer more elaborate grammatical patterns and MR structures is left as future work: we chose here to focus on handling the large scale of the set of entities.

To summarize, this step starts from a sentence and either rejects it or outputs a triplet of lemma tuples, one tuple for the subject, one for the relation or verb, and one for the direct object. To complete our semantic parse (or MR), lemmas must be converted into WordNet synsets, that is, we still have to perform disambiguation, which takes place in step (2).

Step (2): Detection of MR entities This second step aims at identifying each semantic entity expressed in a sentence. Given a relation triplet (lhs^{lem}, rel^{lem}, rhs^{lem}) where each element of the triplet is associated with a tuple of lemmas, a corresponding triplet (lhs^{syn}, rel^{syn}, rhs^{syn}) is produced, where the lemmas are replaced by synsets. This step is a form of all-words WSD in a particular setup, i.e., w.r.t. the logical form of the semantic parse from Step (1). This can be either straightforward (some lemmas such as *_television_program_NN* or *_world_war_ii_NN* correspond to a single synset) or very challenging (*_run_VB* can be mapped to 33 different synsets and *_run_NN* to 10). Hence, in this proposed framework, MRs correspond to triplets of synsets (lhs^{syn}, rel^{syn}, rhs^{syn}), which can be reorganized to the form $rel^{syn}(lhs^{syn}, rhs^{syn})$, as shown in Fig. 5.

Since the model is structured around triplets, MRs and WordNet relations are cast into the same scheme. For example, the WordNet relation (*_ score_NN_2*, *_has_part*, *_musical_notation_NN_1*) fits the same pattern as our MRs, with the relation type *_has_part* playing the role of the verb, and the same entities being present in WordNet relations and MRs. The semantic matching energy function is trained to assign energies to triplets of lemmas and synsets. It is important to notice that such triplets involve a large number of relation types because any verb (under a lemma or a synset form) can act like it. Hence, our data, detailed in Sect. 6.2.1, contains more than 16,000 relation types.

The architecture introduced in Sect. 3.3 cannot be applied directly. Indeed, here $\mathcal{E}$ must be able to handle variable-size arguments, since for example there could be multiple lemmas in the subject part of the sentence. Hence, we add a pooling stage between steps (1) and (2) (of Sect. 3.3). The embeddings associated with all the symbols (synsets or lemmas) within the same tuple are aggregated by a pooling function π (we used the mean but other plausible candidates include the sum, the max, and combinations of several such element-wise statistics, such as in Hamel et al. (2011)). This re-defines E_{lhs}, E_{rel} and E_{rhs} as follows:

$$E_{lhs} = \pi(E_{lhs_1}, E_{lhs_2}, \ldots),$$
$$E_{rel} = \pi(E_{rel_1}, E_{rel_2}, \ldots),$$
$$E_{rhs} = \pi(E_{rhs_1}, E_{rhs_2}, \ldots),$$

where lhs_j denotes the j-th individual element of the left-hand side tuple, etc.

Table 5 Multiple data sources used for learning representations of 37,141 lemmas and 40,943 synsets. "Labeled" indicates when triplets consist of text lemmas for which the corresponding synsets are known. The total number of observed training triplets is 3,328,703 and the total number of relation types appearing in those triplets is around 10,000

Dataset	Train. size	Test size	Labeled	Symbols
WordNet	146,442	5,000	No	synsets
ConceptNet	11,332	0	No	lemmas
Wikipedia	2,146,131	10,000	No	lemmas
Extended WordNet	42,957	5,000	Yes	lemmas + synsets
Unambig. Wikipedia	981,841	0	Yes	lemmas + synsets

We use this slightly modified semantic matching energy function to solve the WSD step. We label a triplet of lemmas $((lhs_1^{lem}, lhs_2^{lem}, \ldots), (rel_1^{lem}, \ldots), (rhs_1^{lem}, \ldots))$ with synsets in a greedy fashion, one lemma at a time. For labeling lhs_2^{lem} for instance, we fix all the remaining elements of the triplet to their lemma and select the synset leading to the lowest energy:

$$lhs_2^{syn} = \operatorname{argmin}_{S \in \mathcal{C}(syn|lem)} \mathcal{E}\big(\big(lhs_1^{lem}, S, \ldots\big), \big(rel_1^{lem}, \ldots\big), \big(rhs_1^{lem}, \ldots\big)\big)$$

with $\mathcal{C}(syn|len)$ the set of allowed synsets to which lhs_2^{lem} can be mapped. We repeat that for all lemmas. We always use lemmas as context, and never the already assigned synsets. Future work should investigate more advanced inference schemes, which would probably be iterative and would gradually refine the estimated set of synsets taking into account their mutual agreement. Nevertheless, this is an efficient process as it only requires the computation of a small number of energies, equal to the number of senses for a lemma, for each position of a sentence. However, it requires good representations (i.e. good embedding vectors E_i) for synsets and lemmas because they are used jointly to perform disambiguation.

6.2 Multi-task training

This section describes how we adapted the training scheme presented in Sect. 4 for learning embeddings for synsets and lemmas using various data sources.

6.2.1 Multiple data resources

In order to endow the model with as much common-sense knowledge as possible, the following heterogeneous data sources are combined. Their statistics are summarized in Table 5. On overall, this large-scale complex data groups 78,084 entities (37,141 lemmas and 40,943 synsets) and 9,560 relation types (4,077 encoded by verb synsets, 5,448 by verb lemmas, 18 for WordNet and 17 for ConceptNet) into 3,328,703 observed training triplets (and 20,000 for testing). Note that the actual number of triplets used for training our system is much larger than that, because we generate negative triplets (by perturbing observed ones) during the learning phase (see Sect. 6.2.2).

WordNet (WN) Already described in Sect. 5, this is the main resource, defining the dictionary of 40,943 entities. WordNet contains only relations between synsets. However, the disambiguation process needs embeddings for synsets and for lemmas. Following Havasi et al. (2010), we created two other versions of this dataset to leverage WN in order to also learn lemma embeddings: "Ambiguated" WN and "Bridge" WN. In "Ambiguated" WN synset entities of each triplet are replaced by one of their corresponding lemmas. "Bridge" WN is designed to teach the model about the connection between synset and lemma embeddings, thus in its relations the *lhs* or *rhs* synset is replaced by a corresponding lemma. Sampling training examples from WN involves actually sampling from one of its three versions, resulting in a triplet involving synsets, lemmas or both.

ConceptNet This common-sense knowledge base (Liu and Singh 2004) groups lemmas or groups of lemmas, which are linked together with rich semantic relations such as (*_kitchen_table_NN*, *_used_for*, *_eat_VB _ breakfast_NN*). It is based on *lemmas* and not synsets, and it does not make distinctions between different senses of a word. Only triplets containing lemmas from the WN dictionary are kept, to finally obtain a total of 11,332 training triplets.

Wikipedia This resource is simply raw text meant to provide knowledge to the model in an unsupervised fashion. In this work 50,000 Wikipedia articles were considered, although many more could be used. Using the protocol of Step (1) of Sect. 6.1, we created a total of 2,146,131 triplets of lemmas. Imperfect training triplets (containing a mix of lemmas and synsets) are produced by performing the disambiguation step on one of the lemmas. This is equivalent to MAP (Maximum A Posteriori) training, i.e., we replace an unobserved latent variable by its mode according to a posterior distribution (i.e. to the minimum of the energy function, given the observed variables). We have used the 50,000 articles to generate more than 3M examples.

EXtended WordNet (XWN) XWN (Harabagiu and Moldovan 2002) is built from WordNet *glosses*, syntactically parsed and with content words semantically linked to WN synsets. Using the protocol of Step (1) of Sect. 6.1, we processed these sentences and collected 47,957 lemma triplets for which the synset MRs were known. We removed 5,000 of these examples to use them as an evaluation set for the word-sense disambiguation task. With the remaining 42,957 examples, we created unambiguous training triplets to help the performance of the disambiguation algorithm: for each lemma in each triplet, a new triplet is created by replacing the lemma by its true corresponding synset and by keeping the other members of the triplet in lemma form (to serve as examples of lemma-based context). This led to a total of 786,105 training triplets, from which 10k were removed for validation.

Unambiguous Wikipedia (Wku) We added to this training set some triplets extracted from the Wikipedia corpus which were modified with the following trick: if one of its lemmas corresponds unambiguously to a synset, and if this synset maps to other ambiguous lemmas, we create a new triplet by replacing the unambiguous lemma by an ambiguous one. Hence, we know the true synset in that ambiguous context. This allowed to create 981,841 additional triplets with supervision.

6.2.2 Training procedure

The training algorithm described in Sect. 4 was used for all the data sources except XWN and Wku. In those two cases, positive triplets are composed of lemmas (as context) and of a disambiguated lemma replaced by its synset. Unlike for Wikipedia, this is labeled data, so we are certain that this synset is the valid sense. Hence, to increase training efficiency and yield a more discriminant disambiguation, in step 3 of the ranking algorithm with probability $\frac{1}{2}$ we either sample randomly from the set of all entities or we sample randomly from the set of remaining candidate synsets corresponding to this disambiguated lemma (i.e. the set of its other meanings).

During training, we sequentially alternate between all sources, performing an update of the model parameters with one mini-batch of examples each time. Sizes of mini-batches differ between sources because we decided to split each source into 50 mini-batches. We always loop over sources in the same order. Training is stopped after 8,000 epochs on all sources (or 7 computation days). There is one learning rate for the g functions and another for the embeddings: their values are set using a grid search, choosing among $\{3., 1., 0.3, 0.1, 0.03, 0.01\}$ and $\{0.03, 0.01, 0.003, 0.001, 0.0003, 0.0001\}$ respectively. The model selection criterion is the mean rank from the entity ranking task on the WordNet validation set. Dimensions of embeddings and of the g output space are equal for these experiments and set to 50 (i.e. $d = p = 50$).

6.3 Related work

The application of SME to WSD is related to work on vector-based models of word meaning (Lund and Burgess 1996; Landauer and Dumais 1997) and neural language models (Bengio 2008; Collobert et al. 2011), in the sense that we learn a vector representation for each lemma to disambiguate. More precisely, it is connected to models aiming at composing such vector embeddings for obtaining phrase or context representations (Mitchell and Lapata 2008), via tensor products (Smolensky 1990) or matrix operations (Paccanaro and Hinton 2001; Socher et al. 2012). However, besides that many of these methods would not scale to the problems introduced here, there exist a major difference with our work: we aim at learning jointly representations for words (lemmas) and senses (synsets), considering structures within language (via SRL triplets) and within a knowledge base (via WordNet triplets) together. To the best of our knowledge, this is the first attempt of mixing relational embeddings of a knowledge base and word embeddings. Note that the original architecture of SME could be adapted for recursively learning phrase representations as proposed by Socher et al. (2012) but this is beyond the scope of this paper.

Our approach is also connected to previous work targeting to improve WSD by using extra-knowledge by either automatically acquiring examples (Martinez et al. 2008) or by connecting different knowledge bases (Havasi et al. 2010), but uses a totally different method.

Finally, our ambition towards open-text semantic parsing is related to previous work by Shi and Mihalcea (2004), who proposed a rule-based system for open-text semantic parsing using WordNet and FrameNet (Baker et al. 1998) and by Giuglea and Moschitti (2006), who proposed a model to connect WordNet, VerbNet and PropBank (Kingsbury and Palmer 2002) for semantic parsing using tree kernels. Poon and Domingos (2009, 2010) introduced a method based on Markov-Logic Networks for unsupervised semantic parsing that can be also used for information acquisition. However, instead of connecting MRs to an existing ontology as done here, it constructs a new ontology and does not leverage pre-existing knowledge.

6.4 Experiments

To assess the performance w.r.t. the multi-task training and the diverse data sources, we evaluated models trained with several combinations of data sources. WN denotes SME models trained on WordNet, "Ambiguated" WordNet and "Bridge" WordNet, while ALL denotes models are trained on all sources.

Table 6 Relation type ranking on Wikipedia. Comparisons between two versions of SME (this paper) and SE. Mean/median predicted rank and precision@10 (p@10, in %) are computed on the test set. Best performances are indicated in bold

Method	Rank (median / mean)	p@10
SME(linear)	**92 / 267**	**1.951**
SME(bilinear)	97 / 286	1.862
SE	118 / 283	0.882

6.4.1 Entity ranking

We first evaluated SME(linear), SME(bilinear), SE and RESCAL in ranking on Wikipedia (Wk) because this dataset offers a test-bed with many types of relation (5,448). The goal of the task was to rank the correct *rel* given a pair of (*lhs*, *rhs*), because this is much easier than ranking *lhs* or *rhs* that contain multiple lemmas. All methods has only been trained on the Wk training set of 2,146,131 observed lemma triplets and evaluated on the corresponding test set of 10k triplets. Hence, we measure the mean and median predicted ranks and the prediction@10, computed with the following procedure. For each test triplet, the relation type is removed and replaced by each of the relation types of the dictionary in turn. Energies of those corrupted triplets are computed by the model and sorted by ascending order and the rank of the correct type is stored. Table 6 reports the average and median of those predicted ranks and the precision@10 (or p@10). Results clearly indicate the advantage of using SME on data with large numbers of relation types, compared to SE. There is no result for RESCAL because we have not been able to run it on Wk. This confirms that SME is the method of choice when dealing with large-scale multi-relational data.

6.4.2 Word-sense disambiguation

Performance on WSD is assessed on two test sets: the XWN test set and a subset of English All-words WSD task of SensEval-3.[10] For the latter, we processed the original data using the protocol of Step (1) of Sect. 6.1 and obtained a total of 208 words to disambiguate (out of ≈2000 originally). We compare with results obtained by Emb., which uses the same embeddings as SME, but without the structure of its energy function. The performance of SE and of the most frequent sense for a given lemma (MFS) as well as of the standard Lesk algorithm are also evaluated. MFS frequencies have been obtained from WordNet, which provides such information (in `cntlist` files). We attempted to adapt RESCAL to this task, but the code took too long to converge with the whole set or even a reduced set of relation types. We implemented a version of Lesk by following the work of Banerjee and Pedersen (2002), which performs WSD based on the WordNet::Similarity package. Hence, a triplet of lemmas is labeled with the triplet of synsets which exhibit the highest cumulated Lesk similarity measure (according to WordNet::Similarity): this cumulated measure is computed as the sum of Lesk similarities of all pairs of synsets composing the triplet. Finally, we report the F1-score of Gamble (Decadt et al. 2004), winner of Senseval-3, on our subset of its data.[11]

[10]More details at www.senseval.org/senseval3.

[11]A side effect of our preprocessing of SensEval-3 data is that our subset contains mostly frequent words. This is easier for MFS than for Gamble because Gamble is efficient on rare terms. Hence, Gamble performs worse

Table 7 Word-sense disambiguation results. F1-scores (in %) of different versions of SME (linear/bilinear, with different data sources, combined with the most frequent sense (MFS) information or not) are compared on XWN and a subset of SensEval-3, with previous work SE; Emb., the unstructured version of SME; Lesk, a standard WSD method; Gamble, the algorithm that won SensEval-3 on the full test set; and Random, which chooses uniformly among allowed synsets. Best performing methods, with a significant difference, are indicated in bold

Method		XWN	SensEval3
SME(linear)	ALL + MFS	**72.3**	**65.9**
	ALL	66.0	44.7
	WN	31.6	29.3
SME(bilinear)	ALL + MFS	**72.1**	**68.3**
	ALL	67.1	49.5
	WN	29.6	28.4
SE	ALL + MFS	68.9	**68.3**
	ALL	51.9	47.1
Emb.	ALL + MFS	68.7	**69.2**
	ALL	39.2	40.4
Lesk		70.2	50.5
Gamble		n/a	**66.4**
MFS		67.2	**67.8**
Random		26.7	29.6

F1 scores are presented in Table 7. The difference between models learnt on ALL and on WN indicates that the information from Wikipedia, XWN and Wku is crucial (+35 %) and yields performance equivalent to that of MFS(a strong baseline in WSD) on the XWN test set. Performance of the model trained on WN alone are roughly equivalent to that of Random. This confirms that knowledge from WordNet and free text are difficult to combine. Still, it is interesting to see that SME is able to train on these various sources and to somewhat capture information from them all. Emb., without the structure taking the relation type into account, performs very poorly. SE is affected by the large number of relation types: on XWN, it remains 15 % below SME. It can not learn proper matrix representations (d^2 parameters) for all verbs involved as relation types, especially for the rare ones. SME does not undergo this problem.

Performance can be greatly improved by combining models trained on the ALL sources and the MFS score. To do so, we converted the frequency information into an energy by taking minus the log frequency and used it as an extra energy term. The total energy function is used for disambiguation. This yields the results denoted by ALL + MFS which achieve the best results of all the methods tried. On the XWN test set, the difference in performance between SME(linear) and SME(bilinear) is not statistically significant but their gap with Lesk is (according to a χ^2 test at the 0.05 level). For SensEval-3, differences between F1 scores of SME(linear), SME(bilinear) (with ALL + MFS), Gamble and MFS are not statistically significant (χ^2 test—0.05 level). Emb. and SE can also be upgraded using MFS information. On SensEval-3, this makes them to equal the best performance. However, on XWN (a better indicator), they remain significantly outperformed by SME. Our method seems to be the best for encoding supportive information to that of MFS: combined with SME, MFS's F1-score increases by 5 %.

than during the challenge and seems to be outperformed by MFS. However, performance of both systems are statistically equivalent.

Table 8 Predicted triplets reported by SME(bilinear) trained on all data sources (ALL), by ReVerb and using the Lesk measure from Wordnet::Similarity

	SME(bilinear)	ReVerb	Lesk
lhs	_army_NN_1	army	_army_NN_1
rel	_attack_VB_1	attacked	_attack_VB_1
top ranked *rhs*	_troop_NN_4	the city	_army_unit_NN_1
	_armed_service_NN_1	the village	_army_corps_NN_1
	_ship_NN_1	Israel	_invade_VB_1
	_territory_NN_1	Poland	_military_unit_NN_1
	_military_unit_NN_1	force	_armed_service_NN_1
top ranked *lhs*	_business_firm_NN_1	People	_monetary_system_NN_1
	_person_NN_1	Players	_money_supply_NN_1
	_family_NN_1	Interest	_currency_NN_1
	_payoff_NN_3	Work	_monetary_standard_NN_1
	_card_game_NN_1	Students	_monetary_resource_NN_1
rel	_earn_VB_1	earn	_earn_VB_1
rhs	_money_NN_1	money	_money_NN_1

6.4.3 *WordNet enrichment*

WordNet uses a limited number of relation types (18 in our version), and does not consider most verbs as relations. Thanks to our multi-task training and unified representation for MRs and WordNet relations, our model is potentially able to generalize to such relations that do not exist in WordNet originally.

As illustration, predicted lists of synsets for relation types that do not exist in WordNet are given in Table 8. We also compare with lists returned by ReVerb (Fader et al. 2011) (an information extraction tool having extracted information from millions of web pages,[12] to be compared with our 50k Wikipedia articles + knowledge bases). Lists from both systems seem to reflect common sense. However, contrary to our system, ReVerb does not disambiguate different senses of a lemma, and thus it cannot connect its knowledge to an existing resource to enrich it. We also provide the lists predicted by the Lesk measure of Wordnet::Similarity, where the score of a WordNet synset is simply defined as the sum of its similarity measures with both input synsets. The predictions are semantically related but they do not reflect the triplet structure because semantic roles of *lhs*, *rel* or *rhs* do not mean anything for Lesk.

7 Conclusion

This paper presented SME, a new energy-based model for learning multi-relational semantics. This method encodes multi-relational graphs or tensors into a flexible continuous vector space in which the original data is stored and enhanced. We empirically showed that SME: (i) is highly competitive with the state-of-the-art methods for modeling multi-relational data,

[12] See the online ReVerb demo at http://openie.cs.washington.edu/.

(ii) outperforms them in large-scale conditions, (iii) can be successfully trained on graphs with tens of thousands of entities and thousands of kinds of relation (more than 100k entities, 10k relation types and 3.5M training triplets). This is the only method able to be efficient on all the different datasets considered in this paper.

In addition, we presented how SME could be applied to perform WSD using a dictionary of more than 70,000 words based on WordNet. Our system, trained on WordNet and free text (and other sources), can capture the deep semantics of sentences in its energy function, which, combined with most frequent sense information, leads to improvement in disambiguation over standard methods. Future work could explore the capabilities of such systems further including more general sentence structures, other semantic tasks, and more evolved grammars, e.g. with FrameNet (Baker et al. 1998; Coppola and Moschitti 2010).

An interesting extension of the model presented here extends its applicability to domains where the objects of interest are not all symbolic (i.e., from a finite set). In that case, one cannot associate a free parameter (its embedding vector) to each possible object. An example of such objects are image patches, which are generally described by a "raw" feature vector. We could learn a mapping from this raw feature space to the embedding space, where the symbolic objects are mapped (in a similar fashion as WSABIE Weston et al. 2010). Whereas for discrete object, one can view the object's embedding as the product of the embedding matrix by a one-hot vector (with a 1 at the position associated with the object symbol), for continuous objects, in the linear mapping case, the "embedding matrix" maps the raw features (richer than one-hot) to the embedding vector. In this way, relations could involve both discrete and continuous objects. Such extensions are possible because of the flexibility and scalability, that models based on embeddings like SME or WSABIE offer for dealing with multimodal inputs.

Acknowledgements The authors would like to acknowledge Léon Bottou, Ronan Collobert, Nicolas Usunier, Nicolas Le Roux, Rodolphe Jenatton and Guillaume Obozinski for inspiring discussions. This work was supported by the French ANR (EVEREST-12-JS02-005-01), the Pascal2 European NoE, the DARPA Deep Learning Program, NSERC, CIFAR, the Canada Research Chairs, and Compute Canada.

References

Baker, C., Fillmore, C., & Lowe, J. (1998). The Berkeley FrameNet project. In *ACL '98* (pp. 86–90).

Banerjee, S., & Pedersen, T. (2002). An adapted lesk algorithm for word sense disambiguation using wordnet. In *Proceedings of the third international conference on computational linguistics and intelligent text processing, CICLing '02* (pp. 136–145). Berlin: Springer.

Bengio, Y. (2008). Neural net language models. *Scholarpedia*, *3*(1), 3881. http://www.scholarpedia.org/article/Neural_net_language_models.

Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, *3*, 1137–1155.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., & Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*. http://www.iro.umontreal.ca/~lisa/pointeurs/theano_scipy2010.pdf, oral Presentation.

Bordes, A., Usunier, N., Collobert, R., & Weston, J. (2010). Towards understanding situated natural language. In *Proceedings of the 13th international conference on artificial intelligence and statistics* (Vol. 9, pp. 65–72).

Bordes, A., Weston, J., Collobert, R., & Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *Proceedings of the 25th conference on artificial intelligence (AAAI-11)*, San Francisco, USA.

Bordes, A., Glorot, X., Weston, J., & Bengio, Y. (2012). Joint learning of words and meaning representations for open-text semantic parsing. *Journal of Machine Learning Research*, *22*, 127–135.

Boser, B., Guyon, I., & Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on computational learning theory* (pp. 144–152).

Bottou, L. (2011). *From machine learning to machine reasoning*. Tech. rep. arXiv:1102.1808.

Cambria, E., Hussain, A., Havasi, C., & Eckl, C. (2009). Affectivespace: blending common sense and affective knowledge to perform emotive reasoning. In *WOMSA at CAEPIA* (pp. 32–41).

Caruana, R. (1995). Learning many related tasks at the same time with backpropagation. In G. Tesauro, D. Touretzky, & T. Leen (Eds.), *Advances in neural information processing systems: Vol. 7. NIPS'94* (pp. 657–664). Cambridge: MIT Press.

Chu, W., & Ghahramani, Z. (2009). Probabilistic models for incomplete multi-dimensional arrays. *Journal of Machine Learning Research*, *5*, 89–96.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proceedings of the 25th international conference on machine learning*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, *12*, 2493–2537.

Coppola, B., & Moschitti, A. (2010). A general purpose FrameNet-based shallow semantic parser. In *Proceedings of the international conference on language resources and evaluation, LREC'10*.

Decadt, B., Hoste, V., Daeleamns, W., & van den Bosh, A. (2004). Gamble, genetic algorithm optimization of memory-based WSD. In *Proceeding of ACL/SIGLEX Senseval-3*.

Denham, W. (1973). *The detection of patterns in alyawarra nonverbal behavior*. PhD thesis, University of Washington.

Fader, A., Soderland, S., & Etzioni, O. (2011). Identifying relations for open information extraction. In *Proceedings of the conference on empirical methods in natural language processing, association for computational linguistics, EMNLP '11* (pp. 1535–1545).

Franz, T., Schultz, A., Sizov, S., & Staab, S. (2009). Triplerank: ranking semantic web data by tensor decomposition. In *Proceedings of the 8th international semantic web conference, ISWC '09* (pp. 213–228).

Getoor, L., & Taskar, B. (2007). *Introduction to statistical relational learning (Adaptive computation and machine learning)*. Cambridge: MIT Press.

Giuglea, A., & Moschitti, A. (2006). Shallow semantic parsing based on FrameNet, VerbNet and PropBank. In *Proceeding of the 17th European conference on artificial intelligence (ECAI'06)* (pp. 563–567).

Gutmann, M., & Hyvärinen, A. (2010). Noise-contrastive estimation: a new estimation principle for unnormalized statistical models. In *Proceedings of the international conference on artificial intelligence and statistics (AISTATS2010)*.

Hamel, P., Lemieux, S., Bengio, Y., & Eck, D. (2011). Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *Proceedings of the 12th international conference on music information retrieval (ISMIR11)*.

Harabagiu, S., & Moldovan, D. (2002). Knowledge processing on extended WordNet. In C. Fellbaum (Ed.), *WordNet: an electronic lexical database and some of its applications* (pp. 379–405). Cambridge: MIT Press.

Harshman, R. A., & Lundy, M. E. (1994). Parafac: parallel factor analysis. *Computational Statistics & Data Analysis*, *18*(1), 39–72.

Havasi, C., Speer, R., & Pustejovsky, J. (2010). Coarse Word-sense disambiguation using common sense. In *AAAI Fall symposium series*.

Jenatton, R., Le Roux, N., Bordes, A., Obozinski, G., et al. (2012). A latent factor model for highly multi-relational data. In *Neural information processing systems, NIPS 2012*.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., & Ueda, N. (2006). Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st national conference on artificial intelligence, AAAI'06* (Vol. 1, pp. 381–388). Menlo Park: AAAI Press.

Kingsbury, P., & Palmer, M. (2002). From treebank to PropBank. In *Proceedings of the 3rd international conference on language resources and evaluation*.

Kok, S., & Domingos, P. (2007). Statistical predicate invention. In *Proceedings of the 24th international conference on machine learning, ICML '07* (pp. 433–440). New York: ACM.

Landauer, T., & Dumais, S. (1997). A solution to plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, *104*(2), 211.

Lecun, Y., Chopra, S., Hadsell, R., Marc'aurelio, R., & Huang, F. (2006). A tutorial on energy-based learning. In G. Bakir, T. Hofman, B. Schölkopf, A. Smola, & B. Taskar (Eds.), *Predicting structured data*. Cambridge: MIT Press.

Liu, H., & Singh, P. (2004). Focusing on conceptnet's natural language knowledge representation. In *Proceedings of the 8th international conference on knowledge-based intelligent information and engineering systems*.

Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods*, *28*(2), 203–208.

Martinez, D., de Lacalle, O., & Agirre, E. (2008). On the use of automatically acquired examples for all-nouns word sense disambiguation. *The Journal of Artificial Intelligence Research*, *33*, 79–107.

McCray, A. T. (2003). An upper level ontology for the biomedical domain. *Comparative and Functional Genomics*, *4*, 80–88.

Miller, K., Griffiths, T., & Jordan, M. (2009). Nonparametric latent feature models for link prediction. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, & A. Culotta (Eds.), *Advances in neural information processing systems* (Vol. 22, pp. 1276–1284).

Mitchell, J., & Lapata, M. (2008). Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT* (pp. 236–244).

Mooney, R. (2004). Learning semantic parsers: an important but under-studied problem. In *Proceedings of the 19th AAAI conference on artificial intelligence*

Nickel, M., Tresp, V., & Kriegel, H. P. (2011). A three-way model for collective learning on multi-relational data. In L. Getoor & T. Scheffer (Eds.), *Proceedings of the 28th international conference on machine learning (ICML-11)* (pp. 809–816). New York: ACM.

Nickel, M., Tresp, V., & Kriegel, H. P. (2012). Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on world wide web, WWW '12* (pp. 271–280).

Paccanaro, A. (2000). Learning distributed representations of concepts from relational data. *IEEE Transactions on Knowledge and Data Engineering*, *13*, 200.

Paccanaro, A., & Hinton, G. (2001). Learning distributed representations of concepts using linear relational embedding. *IEEE Transactions on Knowledge and Data Engineering*, *13*, 232–244.

Poon, H., & Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, Singapore (pp. 1–10).

Poon, H., & Domingos, P. (2010). Unsupervised ontology induction from text. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, Uppsala, Sweden (pp. 296–305).

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, *22*, 400–407.

Rummel, R. J. (1999). Dimensionality of nations project: attributes of nations and behavior of nation dyads. In *ICPSR data file* (pp. 1950–1965).

Shi, L., & Mihalcea, R. (2004). Open text semantic parsing using FrameNet and WordNet. In *HLT-NAACL 2004: demonstration papers*, Boston, MA, USA (pp. 19–22).

Singla, P., & Domingos, P. (2006). Entity resolution with Markov logic. In *Proceedings of the sixth international conference on data mining* (pp. 572–582). Los Alamitos: IEEE Computer Society.

Smolensky, P. (1990). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, *46*(1), 159–216.

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 conference on empirical methods in natural language processing (EMNLP)*.

Speer, R., Havasi, C., & Lieberman, H. (2008). Analogyspace: reducing the dimensionality of common sense knowledge. In *Proceedings of the 23rd national conference on artificial intelligence, AAAI'08* (Vol. 1, pp. 548–553). Menlo Park: AAAI Press.

Sutskever, I., Salakhutdinov, R., & Tenenbaum, J. (2009). Modelling relational data using Bayesian clustered tensor factorization. In *Advances in neural information processing systems* (Vol. 22).

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, *31*, 279–311.

van der Maaten, L., & Hinton, G. (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, *9*, 2579–2605.

Weston, J., Bengio, S., & Usunier, N. (2010). Large scale image annotation: learning to rank with joint word-image embeddings. *Machine Learning*, *81*, 21–35.