Learning Structured Embeddings of Knowledge Bases

Antoine Bordes[*]
Heudiasyc
CNRS – UTC
Compiègne, France
bordesa@iro.umontreal.ca

Jason Weston
Google
111 8th Avenue
New York. NY, USA
jaseweston@gmail.com

Ronan Collobert
IDIAP
Rue Marconi 19
Martigny, Switzerland
ronan@collobert.com

Yoshua Bengio
Département IRO
Université de Montréal
Montréal. QC, Canada
bengioy@iro.umontreal.ca

1 Introduction

A fundamental challenge for AI has always been to be able to gather, organize and make intelligent use of the colossal amounts of information generated daily (Davis *et al.*, 1993). Recent developments in this area aim at building large web-based Knowledge Bases (KBs), special kinds of relational databases especially designed for knowledge management, collection, and retrieval. Thanks to long-term funding efforts or collaborative processes, promising progress has been accomplished and several KBs, which encompass a huge amount of data regarding general and specific knowledge, are now readily available on-line. Examples are OpenCyc, Freebase, WordNet, DBpedia, etc.

These KBs have been conceived for differing purposes, such as approaching human-like reasoning, producing an intuitively usable dictionary and thesaurus or proposing a global on-line information resource for *semantic web* applications. However, their highly-structured and organized data could also be useful in many other AI areas such as Natural Language Processing (NLP), for word-sense disambiguation or natural language understanding, in computer vision for scene classification or image semantic annotation, or in collaborative filtering. Even if WordNet is widely used in NLP (e.g. in (Snow *et al.*, 2007)), this remains a small contribution compared to what could be achieved with such gigantic knowledge quantities. This could be explained by the fact that it is usually hard to take advantage of KBs data in other systems. Indeed, their underlying symbolic frameworks, whilst being very efficient for their original purposes, are not flexible enough to be fruitfully exported, especially to statistical learning approaches.

This work studies an original way of leveraging the structured data encompassed by KBs into statistical learning systems. Our work is based on a model that learns to represent elements of any KB into a relatively low (e.g. 50) dimensional *embedding vector space*. The embeddings are established by a neural network whose particular architecture allows to integrate the original data structure within the learnt representations. This new framework is appealing for several reasons: it is flexible (simple to adapt to many KBs), compact (only a low-dimension vector per entity and a low-dimension matrix per relation type to store) and also exhibits generalization ability (the ability to infer new relations from existing ones). Moreover, such representations potentially allow integration of KBs within systems of the recent machine learning trend of Deep Learning, e.g., those applied to NLP (Collobert and Weston, 2008).

2 Structured Embeddings

This work considers Knowledge Bases as graph models. This means that the data structure of KBs is not necessarily hierarchical, and is just 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. In the following, a relation is denoted by a triplet (e^l, r, e^r), where e^l is the *left* entity, e^r the *right* one and r the *type* of relation between them.

Our structural embedding of KBs is based on 2 main ideas. First, entities can be modeled in a d-dimensional vector space, termed the "embedding space". The i^{th} entity is assigned a vector $E_i \in \mathbb{R}^d$. Second, within that embedding space, for any given relation type, there is a specific similarity measure that captures that relation between entities. For example, in WordNet, the *_part_of* relation would use one measure of similarity, whereas *_similar_to* would use another. Note that these similarities are not generally symmetric. We model this by assigning for the k^{th} given relation a pair $R_k = (R_k^{lhs}, R_k^{rhs})$, where R_j^{lhs} and R_j^{rhs} are both $d \times d$ matrices. The similarity function for a given entity is thus defined as $S_k(E_i, E_j) = ||R_k^{lhs} E_i - R_k^{rhs} E_j||_p$, using the p-norm (we chose $p = 1$ for the simplicity of the associated gradient learning). In other words, we transform the entity embedding vectors E_i and E_j by the corresponding left and right hand relation matrices for the relation R_k and then similarity is measured according to the 1-norm distance

[*]Work done while Antoine Bordes was visiting Université de Montréal.

Table 1: **Ranking**. Predicted ranks on WordNet (55,166 candidates) and Freebase (81,061 candidates).

		WordNet		Freebase
		rank e^l	rank e^r	rank e^r
COUNTS	*Train*	662.7	804.1	541.8
	Test	6202.3	5894.2	804.9
EMB	*Train*	16.2	23.3	–
	Test	3414.7	3380.8	–
EMB$_{MT}$	*Train*	13.6	20.9	2.9
	Test	97.3	223.0	317.2
EMB$_{MT}$+KDE	*Train*	**11.8**	**19.9**	**1.6**
	Test	**87.8**	**192.5**	**314.5**

Table 2: **Generalization** . Lists of e^r predicted using EMB$_{MT}$+KDE after training on WordNet. We removed from the lists all elements from the training set: the predictions below are generalized by the system.

e^l	_everest_1	_brain_1
r	_part_of	_has_part
e^r	*_north_vietnam_1*	*_subthalamic_nucleus_1*
	_hindu_kush_1	*_cladode_1*
	_karakoram_1	*_subthalamus_1*
	_federal_2	*_fluid_ounce_1*
	_burma_1	*_sympathetic_nervous_system_1*

in the transformed embedding space. The entities embeddings E and the relations operators (R^{lhs}, R^{rhs}) are learnt with a neural network, which can be seen as a generalization of a siamese network (Bromley *et al.*, 1993). Note that E is learnt via a *multi-tasking* process because a single embedding matrix is used for all relations.

Unfortunately, this approach has the weakness that it somewhat dilutes some crucial information given by the KB data: training triples are *true* facts, for which we have a high degree of certainty. So, after training the structured embeddings, we propose to estimate the probability density at any point of the defined embedding space using Kernel Density Estimation (KDE). Because KDE bases its estimation on the training points, this guarantees that they get a high probability density, which is exactly what we are looking for.

3 Empirical Evaluation

Ranking The quality of our representations can be assessed using the following ranking task: for triplets (e^l,r,e^r), we remove e^l (or e^r), and record at which position it would be ranked by the system, among all possible entities. This setting somewhat corresponds to question answering. We compare our method, denoted EMB$_{MT}$+KDE, with 3 counterparts which rank triplets with different procedures: EMB$_{MT}$ which uses the same embeddings but performs ranking without KDE, EMB which also ranks without KDE but using embeddings learnt without multi-tasking (a different E per relation type) and COUNTS which does not perform any learning but only counts the number of times pairs (e^l, r) and (r,e^r), for all e^l, r and e^r, appear in the training set. We use data from 2 KBs: WordNet which contains 55,166 entities, 11 relation types, 164,467 training triples and 4,000 testing triples, and Freebase which contains 81,061 entities, 13 relation types, 356,517 training triples and 4,000 testing triples.

Results of Table 1 show that EMB$_{MT}$ and EMB$_{MT}$+KDE perform best. As expected KDE helps on training examples. The rank is almost perfect on Freebase and the values are misleading on WordNet. Indeed, the correct answer is not ranked on top all the time because some other training triplets happen to be as correct as the considered test example:if training examples are removed from the lists, the ranks on WordNet become 1.1 and 1.3. Hence, KDE achieves its goal since EMB$_{MT}$+KDE replicates (almost) perfectly the training KB.

Generalization COUNTS can record some information about training examples but can not generalize. To some extent, EMB exhibits the same behavior since it can almost replicate the training set, but is bad on test triplets. Since both EMB$_{MT}$ and EMB$_{MT}$+KDE perform much better on test examples, we deduce that generalization can only be possible via multi-tasking. This allows to encode information coming from different relations in the embeddings of entities, which can then be exploited by relation operators. This is a kind of analogy process which seems to be more efficient on WordNet than on Freebase because the same entities appear in more different types of relations. Table 2 illustrates this a bit more by displaying top ranked entities for 2 WordNet relations. Since we removed any training examples from these lists, these are analogies performed by the system. The chosen entities are not always exactly correct but do make sense most of the time.

References

Bromley, J., Guyon, I., LeCun, Y., Sackinger, E., and Shah, R. (1993). Signature verification using a siamese time delay neural network. In *NIPS 6*, volume 6.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *ICML '08*.

Davis, R., Shrobe, H., and Szolovits, P. (1993). What is a knowledge representation? *AI Magazine*, **14**(1), 17–33.

Snow, R., Prakash, S., Jurafsky, D., and Ng, A. Y. (2007). Learning to merge word senses. In *EMNLP'07*.

Topic: learning algorithms **Preference: oral**