Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing

Antoine Bordes
Heudiasyc, UMR 7253
UTC, Compiègne, France
antoine.bordes@utc.fr

Xavier Glorot
DIRO, Université de Montréal
Montréal, QC, Canada
glorotxa@iro.umontreal.ca

Jason Weston
Google
New York, NY, USA
jweston@google.com

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

Abstract

Open-text semantic parsers are designed to interpret any statement in natural language by inferring a corresponding *meaning representation* (MR – a formal representation of its sense). Unfortunately, large scale systems cannot be easily machine-learned due to a lack of directly supervised data. We propose a method that learns to assign MRs to a wide range of text (using a dictionary of more than 70,000 words mapped to more than 40,000 entities) thanks to a training scheme that combines learning from knowledge bases (e.g. WordNet) with learning from raw text. The model jointly learns representations of words, entities and MRs via a multi-task training process operating on these diverse sources of data. Hence, the system ends up providing methods for knowledge acquisition and word-sense disambiguation within the context of semantic parsing in a single elegant framework. Experiments on these various tasks indicate the promise of the approach.

1 Introduction

A key ambition of AI has always been to render computers able to automatically interpret text and express its meaning in a formal representation. Semantic parsing (Mooney, 2004) precisely aims at building such systems to interpret statements expressed in natural language. The purpose of the 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). This task seems too daunting to carry out manually (because of the vast quantity of knowledge engineering that would be required) so machine learning seems an appealing avenue. On the other hand, machine learning models usually require many labeled examples, which can also be costly to gather, especially when labeling properly requires the expertise of a linguist.

Hence, research in semantic parsing can be roughly divided into two tracks. The first one, which could be termed *in-domain*, aims at learning to build highly evolved and comprehensive MRs (Ge and Mooney, 2009; Zettlemoyer and Collins, 2009; Liang *et al.*, 2011). Since this requires highly annotated training data and/or MRs built specifically for one domain, such approaches typically have a restricted vocabulary (a few hundred words) and a correspondingly limited MR representation. For example, the U.S. geography dataset[1] only deals with facts like the number and relative location of cities, rivers and states, with about 800 facts. Alternatively, a second line of research, which could be termed *open-domain* (or *open-text*), works towards learning to associate a MR to any kind of natural language sentence (Shi and Mihalcea, 2004; Poon and Domingos, 2009). In this case, the supervision is weaker because it is infeasible to label large amounts of free text with MRs that capture deep semantic structure. As a result, models infer simpler MRs; this is also referred to as *shallow* semantic parsing.

This paper focuses on the open-domain category. We aim to produce 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. The process is described in Figure 1 and detailed in Section 2. For a given sentence, we infer a MR in two stages: (1) a semantic role labeling step predicts the semantic structure; (2) a disambiguation step assigns a corresponding entity to each relevant word, so as to minimize a learnt energy function. For step (1) we use an existing approach. Our key contribution is a novel inference model for performing step (2). This consists of an energy-based model that is trained with

Appearing in Proceedings of the 15^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2012, La Palma, Canary Islands. Volume 22 of JMLR: W&CP 22.

[1] See cs.utexas.edu/~ml/nldata/geoquery.html.

data from multiple sources in order to combat the lack of strong supervision. The system is large-scale with a dictionary of more than 70,000 words that can be mapped to more than 40,000 disambiguated entities.

Our energy-based model is trained to *jointly capture semantic information between words, entities and combinations of those.* This is encoded in a distributed representation in which a low dimensional embedding vector (or simply *embedding* in the following) is learnt for each symbol. Our *semantic matching energy function*, introduced in Section 3, is designed to blend such embeddings in order to assign low energy values to plausible combinations.

Resources like WordNet (Miller, 1995) and ConceptNet (Liu and Singh, 2004) encode common-sense knowledge in the form of relations between entities (e.g. $\_has\_part(\_car, \_wheel)$) but do not link this knowledge to raw text (sentences). On the other hand, text resources like Wikipedia are not grounded with entities. As we show in Section 4, our training procedure is based on multi-task learning across different data sets including the three mentioned above. In this way MRs induced from text and relations between entities are embedded (and integrated) in the same space. This allows us to learn to perform disambiguation on raw text by using large amounts of indirect supervision and little direct supervision. The model learns to use common-sense knowledge (such as WordNet relations between entities) to help disambiguate, i.e. to choose the correct WordNet sense of a word.

Since no standard evaluation for open-text semantic parsing exists, we evaluated our method on different criteria to reflect its different properties. Results presented in Section 6 consider two benchmarks: word-sense disambiguation (WSD) and (WordNet) knowledge acquisition. We also demonstrate the possibility that our system can perform knowledge extraction, i.e. learn new common-sense relations that do not exist in WordNet by multi-tasking with raw text.

2 Semantic Parsing Framework

This section introduces the particular framework we are using to perform semantic parsing on free text.

2.1 WordNet-based Representations (MRs)

The MRs we consider for semantic parsing are simple logical expressions of the form $REL(A_0, \ldots, A_n)$. REL is the relation symbol, and A_0, ..., A_n are its arguments. Note that several such forms can be recursively constructed to build more complex structures. We wish to parse open-domain raw text so a large set of relation types and arguments must be considered.

We employ WordNet for defining REL and A_i arguments as proposed in (Shi and Mihalcea, 2004). WordNet encompasses comprehensive knowledge within its graph structure, whose nodes (termed *synsets*) correspond to senses, and edges define relations between those senses. Each synset is associated with a set of words sharing that sense. They are usually identified by 8-digits codes, however, for clarity reasons, we indicate in this paper a synset by the concatenation of one of its words, its part-of-speech tag (NN for nouns, VB for verbs, JJ for adjectives or RB for adverbs) and a number indicating which sense it refers to. For example, *_score_NN_1* refers to the synset representing the first sense of the noun "score", also containing the words "mark" and "grade", whereas *_score_NN_2* refers to its second meaning (i.e. a written form of a musical composition).

We denote instances of relations from WordNet using triplets (lhs, rel, rhs), where lhs depicts the left-hand side of the relation, rel its type and rhs its right-hand side. Examples are (*_score_NN_1*, *_hypernym*, *_evaluation_NN_1*) or (*_score_NN_2*, *_has_part*, *_musical_notation_NN_1*). We filtered out the synsets appearing in less than 15 triplets, as well as relation types appearing in less than 5000 triplets. We obtain a graph with the following statistics: 41,024 synsets and 18 relation types; a total of 70,116 different words belong to these synsets. For our final predicted MRs, we will represent REL and A_i arguments as tuples of WordNet synsets (hence REL can for example be any verb, and is not constrained to one of the 18 WordNet relations).

2.2 Inference Procedure

Step (1): MR structure inference Our semantic parsing consists of two stages as described in Figure 1. 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 our learning algorithm for step (2).

We use the SENNA software[2] (Collobert *et al.*, 2011) to perform part-of-speech (POS) tagging, chunking, lemmatization[3] and semantic role labeling (SRL). In the following, we term the concatenation of a lemmatized word and a POS tag (such as *_score_NN* or *_accompany_VB*) a *lemma*. Note the absence of an integer suffix, which distinguishes a lemma from a 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. It is crucial because it will be used

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

[3]Lemmatization is carried out with NLTK (`nltk.org`) and transforms a word into its canonical or base form.

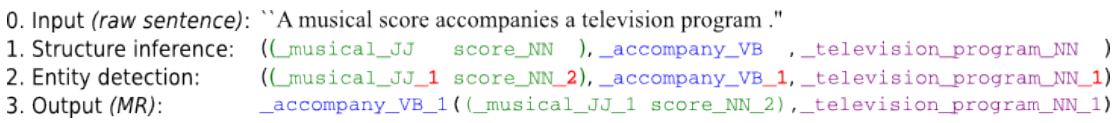

Figure 1: **Open-text semantic parsing.** To parse an input sentence (step 0), a preprocessing (lemmatization, POS, chunking, SRL) is first performed (step 1) to clean data and uncover the MR structure. Then, to each lemma is assigned a corresponding WordNet synset (step 2), hence defining a complete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

to infer the *structure* of the MR.

We only consider sentences that match the following template: (*subject*, *verb*, *direct object*). Here, each of the three elements of the template is associated with a tuple of lemmatized words (i.e. a multi-word phrase). SRL is used to structure the sentence into the (lhs = subject, rel = verb, rhs = object) template, note that the order is not necessarily subject / verb / direct object in the raw text (e.g. in passive sentences).

To summarize, this step starts from a sentence and either rejects it or outputs a triplet of lemma tuples, one 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 synsets, that is, we still have to perform disambiguation, which takes place in step (2).

Step (2): Detection of MR entities The 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 word-sense disambiguation in a particular setup, i.e., w.r.t. the logical form of the semantic parse from step (1). Depending on the lemmas, this can either be 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 our proposed semantic parsing 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 Figure 1.

Since the model is structured around relation 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 WordNet relation type *_has_part* playing the role of the verb.

3 Semantic Matching Energy

This section presents the main contribution of this paper: an energy function which we use to embed lemmas and WordNet entities into the same vector space (see (Lecun *et al.*, 2006) for an introduction to energy-based learning). This *semantic matching* energy function is used to predict appropriate synsets given lemmas. This is achieved by an approximate search for a set of synsets that are compatible with the observed lemmas, i.e. a set of synsets that minimize the energy.

3.1 Framework

Our model is designed with the following key concepts:

- Named symbolic entities (synsets, relation types, and lemmas) are all associated with a joint d-dimensional vector space, termed the "embedding space", following previous work in neural language models (see (Bengio, 2008)). The i^{th} entity is assigned to a vector $E_i \in \mathbb{R}^d$. These vectors are parameters of the model and are *jointly learned to perform well at the semantic parsing task.*
- 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 of the symbols to their embeddings. $\mathcal{E}$ must also be able to handle variable-size arguments, since for example there could be multiple lemmas in the subject part of the sentence.
- The energy function $\mathcal{E}$ is optimized to be lower for training examples than for other possible configurations of symbols. Hence the semantic matching energy function can distinguish plausible combinations of entities from implausible ones to choose the most likely sense for a lemma.

3.2 Parametrization

The semantic matching energy function has a parallel structure (see Figure 2): first, pairs (lhs, rel) and (rel, rhs) are combined separately and then, these semantic combinations are *matched*.

Let the input triplet be $x = ((lhs_1, lhs_2, \ldots), (rel_1, rel_2, \ldots), (rhs_1, rhs_2, \ldots))$.

(1) Each symbol i in the input tuples is mapped to its embedding $E_i \in \mathbb{R}^d$.

(2) The embeddings associated with all the symbols within the same tuple are aggregated by a pooling function π (we used the mean but other plausible

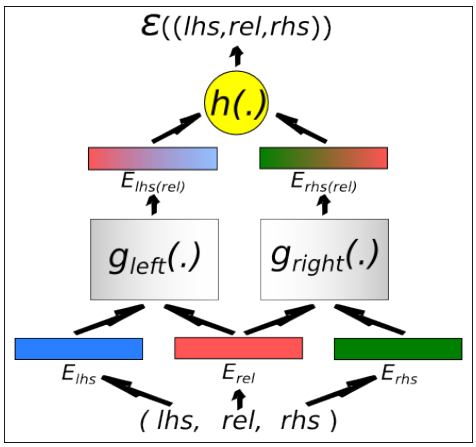

Figure 2: **Semantic matching energy function.** A triplet of tuples (lhs, rel, rhs) is first mapped to its embeddings E_{lhs}, E_{rel} and E_{rhs} (using an aggregating function for tuples involving more than one symbol). Then E_{lhs} and E_{rel} are combined using $g_{left}(.)$ to output $E_{lhs(rel)}$ (similarly $E_{rhs(rel)} = g_{right}(E_{rhs}, E_{rel})$). Finally the energy $\mathcal{E}((lhs, rel, rhs))$ is obtained by merging $E_{lhs(rel)}$ and $E_{rhs(rel)}$ with the $h(.)$ function.

candidates include the sum, the max, and combinations of several such elementwise statistics):

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

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

(3) 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.

(4) The energy is computed from the transformed embeddings of the left-hand and right-hand sides: $\mathcal{E}(x) = h(E_{lhs(rel)}, E_{rhs(rel)})$, where h is a function that can be hard-coded or parametrized. If the latter, parameters are tuned during training.

Many parametrizations are possible for the energy function (e.g., with non-linear, linear, etc. formulations for g and h) and we have explored only a few. In this paper, we only present the bilinear setting (that performed best in experiments proposed in Section 6): the g functions are bilinear layers and h is a dot-product. More precisely, we used:

$$E_{lhs(rel)} = (W_{ent,l} E_{lhs}) \otimes (W_{rel,l} E_{rel}) + b_l$$
$$E_{rhs(rel)} = (W_{ent,r} E_{rhs}) \otimes (W_{rel,r} E_{rel}) + b_r$$
$$h(E_{lhs(rel)}, E_{rhs(rel)}) = -E_{lhs(rel)} \cdot E_{rhs(rel)}$$

where $W_{ent,l}$, $W_{rel,l}$, $W_{ent,r}$ and $W_{rel,r}$ are $d \times d$ weight matrices, b_l, b_r are d bias vectors and $\otimes$ depicts the element-wise vector product.

This bilinear parametrization is appealing because the operation $\otimes$ allows to encode conjunctions between lhs and rel, and rhs and rel.

3.3 Training Objective

We now define the training criterion for the semantic matching energy function. Let $\mathcal{C}$ denote the dictionary which includes all entities (relation types, lemmas *and* synsets), and let $\mathcal{C}^*$ denote the set of tuples (or sequences) whose elements are taken in $\mathcal{C}$. Let $\mathcal{R} \subset \mathcal{C}$ be the subset of entities which are relation types ($\mathcal{R}^*$ is defined similarly to $\mathcal{C}^*$). We are given a training set $\mathcal{D}$ containing m triplets of the form $x = (lhs_x, rel_x, rhs_x)$, where $lhs_x \in \mathcal{C}^*$, $rel_x \in \mathcal{R}^*$, and $rhs_x \in \mathcal{C}^*$, and $\forall x \in \mathcal{C}^* \times \mathcal{R}^* \times \mathcal{C}^*$, $\mathcal{E}(x) = \mathcal{E}((lhs_x, rel_x, rhs_x))$.

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) 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, we use a ranking criterion (that is based on uniformly sampling a negative example).

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. For example, this would allow us to answer questions like "what is part of a car?" or "what does a score accompany?". 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:

$$\mathcal{E}(x) < \mathcal{E}((i, rel_x, rhs_x)), \forall i \in \mathcal{C}^* : (i, rel_x, rhs_x) \notin \mathcal{D} \quad (1)$$
$$\mathcal{E}(x) < \mathcal{E}((lhs_x, j, rhs_x)), \forall j \in \mathcal{R}^* : (lhs_x, j, rhs_x) \notin \mathcal{D} \quad (2)$$
$$\mathcal{E}(x) < \mathcal{E}((lhs_x, rel_x, k)), \forall k \in \mathcal{C}^* : (lhs_x, rel_x, k) \notin \mathcal{D} \quad (3)$$

Towards achieving this, the following stochastic criterion is minimized:

$$\sum_{x \in \mathcal{D}} \sum_{\tilde{x} \sim Q(\tilde{x}|x)} \max\left(\mathcal{E}(x) - \mathcal{E}(\tilde{x}) + 1, 0\right) \quad (4)$$

where $Q(\tilde{x}|x)$ is a corruption process that transforms a training example x into a corrupted *negative example*. In the experiments Q only changes one of the three members of the triplet, by changing only one of the lemmas, synsets or relation types in it, by sampling it uniformly from $\mathcal{C}$ or $\mathcal{R}$.

3.4 Disambiguation of Lemma Triplets

Our semantic matching energy function is used on raw text to perform step (2) of the protocol described in Section 2.2, that is to carry out the word-sense disambiguation step. A triplet of lemmas $((lhs_1^{lem}, lhs_2^{lem}, \ldots), (rel_1^{lem}, \ldots), (rhs_1^{lem}, \ldots))$ is labeled 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 lemmas and select the synset leading to the lowest energy:

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

with $\mathcal{C}(syn|lem)$ 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. 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 this crucial step. Hence, the multi-tasking training presented in the next section attempts to learn good embeddings jointly for synsets and lemmas (and good parameters for the g functions).

4 Multi-Task Training

4.1 Multiple Data Resources

In order to endow the model with as much commonsense knowledge as possible, the following heterogeneous data sources are combined.

WordNet v3.0 (WN). Described in Section 2.1, this is the main resource, defining the dictionary of entities. The 18 relation types and 40,989 synsets retained are composed to form a total of 221,017 triplets. We randomly extracted from them a validation and a test set with 5,000 triplets each.

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 both synset entities of each triplet are replaced by one of their corresponding lemmas, thus training the models with many examples that are similar up to the replacement of a lemma by a synonym. "Bridge" WN is designed to teach the model about the connection between synset and lemma embeddings, thus in its relation tuples the *lhs* or *rhs* synset is replaced by a corresponding lemma (while the other argument remains a synset). Sampling training examples from WN involves actually sampling from one of its three versions, resulting in a triplet involving synsets, lemmas or both.

ConceptNet v2.1 (CN). CN (Liu and Singh, 2004) is a common-sense knowledge base in which lemmas or groups of lemmas are linked together with rich semantic relations, for example (*_kitchen_table_NN, _used_for, _eat_VB _breakfast_NN*). It is based on *lemmas* and not synsets, and hence it does not make any distinction between different word senses. Only triplets containing lemmas from the WN dictionary are kept, to obtain a total of 11,332 training triplets.

Wikipedia (Wk). 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 the first paragraph of Section 2.2, we created a total of 1,484,966 triplets of lemmas. Additionally, imperfect training triplets (containing a mix of lemmas and synsets) are produced by performing the disambiguation step of Section 3.4 on one of the lemmas. This is equivalent to 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* (i.e. definitions), syntactically parsed and with content words semantically linked to WN synsets. Using the protocol of Section 2.2, 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 MR entity detection/word-sense disambiguation task. With the remaining 42,957 examples, we created unambiguous training triplets to help the performance of the disambiguation algorithm described in Section 3.4: for each lemma in each triplet, a new triplet is created by replacing the lemma by its true correspond-

ing 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 we removed 10,000 for validation.

Unambiguous Wikipedia (Wku). Finally, We built an additional training set with 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 (as detailed in next section).

4.2 Training 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 at random (composed of synsets, of lemmas or both) from one of the above sources of examples.
2. Select at random resp. constraint (1), (2) or (3).
3. Create a negative triplet $\tilde{x}$ by sampling an entity from the set of all entities $\mathcal{C}$ to replace respectively lhs_{x_i}, rel_{x_i} or rhs_{x_i}.
4. If $\mathcal{E}(x_i) > \mathcal{E}(\tilde{x}) - 1$, make a stochastic gradient step to minimize the criterion (4).
5. Enforce the constraint that each embedding vector is normalized, $||E_i|| = 1$, $\forall i$.

The constant 1 in step 4 is the *margin*. The gradient step requires a learning rate of λ. The normalization in step 5 helps remove scaling freedoms from the model.

The above algorithm was used for all the data sources except XWN and Wku. In that case, 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 true sense. Hence, to increase training efficiency and yield a more discriminant disambiguation, in step 3 with probability $\frac{1}{2}$, we either sample randomly from $\mathcal{C}$ or from the set of remaining candidate synsets corresponding to this disambiguated lemma (i.e. the set of its other meanings).

The matrix E which contains the representations of the entities is learnt via a complex *multi-task learning* procedure because a single embedding matrix is used for all relations and all data sources (each really corresponding to a different distribution of symbol tuples, i.e., a different task). As a result, the embedding of an entity contains factorized information coming from all the relations and data sources in which the entity is involved as lhs, rhs or even rel (for verbs). For each entity, the model is forced to learn how it interacts with other entities in many different ways.

5 Related Work

Our approach is original in both its energy-based model formulation and how it unifies multiple tasks and training resources for its semantic parsing goal. However, it has relations with many previous works. Shi and Mihalcea (2004) proposed a rule-based system for open-text semantic parsing using WordNet and FrameNet (Baker *et al.*, 1998) while Giuglea and Moschitti (2006) proposed a model to connect WordNet, VerbNet and PropBank (Kingsbury and Palmer, 2002) for semantic parsing using tree kernels. A method based on Markov-Logic Networks has been recently introduced for unsupervised semantic parsing that can be also used for information acquisition (Poon and Domingos, 2009, 2010). However, instead of connecting MRs to an existing ontology as the proposed method does, it constructs a new one and does not leverage pre-existing knowledge. Automatic information extraction is the topic of many models and demos (Snow *et al.*, 2006; Yates *et al.*, 2007; Wu and Weld, 2010; Suchanek *et al.*, 2008) but none of them relies on a joint embedding model. Some approaches have been directly targeting to enrich existing resources, as we do here with WordNet, (Agirre *et al.*, 2000; Cuadros and Rigau, 2008; Cimiano, 2006) but these never use learning. Finally, several previous works have targeted 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).

To our knowledge, energy-based models have not been applied to semantic parsing, but approaches have been developed for other NLP tasks. Bengio *et al.* (2003) developed a neural word embedding approach for language modeling (they only model words, not entities). Paccanaro and Hinton (2001) developed entity embedding for learning logical relations (but do not model words). Collobert and Weston (2008) developed word embedding models further and applied them to various NLP tasks. We in fact use their SRL system to solve step (1) of our system (see Figure 1) and focus on step (2), which they do not address. The architecture of our system is also related to recent work on gated models (Sutskever *et al.*, 2011) and recursive autoencoders (Bottou, 2011; Socher *et al.*, 2011).

Finally, Bordes *et al.* (2011) describe an entity embedding model for knowledge acquisition: given relation

Table 1: **WordNet Knowledge Acquisition (cols. 2-3) and Word Sense Disambiguation (cols. 4-5).** MFS uses the Most Frequent Sense. **All+MFS** is our best system, combining all sources of information.

Model	WordNet rank	WordNet p@10	F1 XWN	F1 Senseval3
All+MFS	–	–	**72.35**%	**70.19**%
All	139.30	3.47%	67.52%	51.44%
WN+CN+Wk	95.9	4.60%	34.80%	34.13%
WN	72.1	5.88%	29.55%	28.36%
MFS	–	–	67.17	67.79%
Gamble (Decadt *et al.*, 2004)	–	–	–	66.41%
SE (Bordes *et al.*, 2011)	53.2	7.45%	–	–
SE (no KDE) (Bordes *et al.*, 2011)	87.6	4.91%	–	–
Random	20512	0.01%	26.71%	29.55%

training triplets, they also measure how well the model generalizes to new relations. Their model embeds each relation type into a pair of matrices instead of using a bilinear parametrization. This gives relation types a different status (they cannot appear as *lhs* or *rhs*) and necessitates many more parameters. Training on raw text with thousands of verbs would be impossible, hence their model is not viable for semantic parsing.

6 Experiments

This section starts by an evaluation of our model and then proposes an illustration of its properties.

6.1 Benchmarks

To assess the performance w.r.t. choices made with the multi-task joint training and the diverse data sources, we evaluated models trained with several combinations of data sources on two benchmark tasks: WordNet knowledge encoding and WSD. **WN** denotes our model trained only on WordNet, "Ambiguated" WordNet and "Bridge" WordNet, **WN+CN+Wk** is also trained on CN and Wk datasets, and **All** is trained on all sources. Results are summarized in Table 1.

Knowledge Acquisition The ability to generalize from given knowledge (training relations) to new relations is measured with the following procedure. For each test WordNet triplet, either the left or right entity is removed and replaced by each of the 41,024 synsets of the dictionary in turn. Energies of those triplets are computed by the model and sorted by ascending order and the rank of the correct synset is stored. We then measure the mean predicted rank (the average of those ranks), **WordNet rank**, and the precision@10 (p@10) (the proportion of ranks that are within 1 and 10, divided by 10), **WordNet p@10**. Hyperparameters (λ and d) are selected with a validation set.

Columns 2 and 3 of Table 1 present the comparative results, together with performance of (Bordes *et al.*, 2011) (**SE**). Our model trained on WordNet alone (**WN**) is a bit worse than **SE** (line 8). However, Bordes *et al.* (2011) stack a Kernel Density Estimator (KDE) on top of their structured embeddings to improve prediction. Compared with **SE** without KDE (line 9), our model is clearly competitive. Multi-tasking with other data (**WN+CN+Wk** and **All**) still allows to encode WordNet knowledge well, even if it is slightly worse than with WordNet alone (**WN**).

By multi-tasking with raw text, the numbers of relation types grows from 18 to several thousands. Our model learns similarities, which are more complex with so many relations: by adding text relations, the problem of extracting knowledge from WordNet becomes harder. This degrading effect is a current limitation of the multi-tasking process, even if performance is still very good if one keeps in mind that ranks of Table 1 are over 41,024 entities. Besides, this offers the ability to combine multiple training sources, which is crucial for WSD and eventually for semantic parsing.

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.[4] For the latter, we processed the original data using the protocol of Section 2.2 and kept only triplets (*subject*, *verb*, *direct object*) for which all lemmas were belonging to our vocabulary defined by WordNet. We obtained a total of 208 words to disambiguate (out of $\approx$ 2000 originally). The performance of the most frequent sense (**MFS**) based on WordNet frequencies is also evaluated. Finally, we also report the results of **Gamble** (Decadt *et al.*, 2004), winner of Senseval-3, on our subset of its data.[5]

F1 scores are presented in Table 1 (cols. 4-5). The difference between **WN** and **WN+CN+Wk** indicates that, even with no direct supervision, the model can

[4] `www.senseval.org/senseval3`.

[5] 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 than during the challenge and is outperformed by **MFS**.

Table 2: **Embeddings.** Closest neighbors for some lemmas/synsets in the embedding space (euclidean distance).

_mark_NN	*_mark_NN_1*	*_mark_NN_2*
_indication_NN	*_score_NN_1*	*_marking_NN_1*
_print_NN_3	*_number_NN_2*	*_symbolizing_NN_1*
_print_NN	*_gradation_NN*	*_naming_NN_1*
_roll_NN	*_evaluation_NN_1*	*_marking_NN*
_pointer_NN	*_tier_NN_1*	*_punctuation_NN_3*
_take_VB	*_canary_NN*	*_different_JJ_1*
_bring_VB	*_sea_mew_NN_1*	*_eccentric_NN*
_put_VB	*_yellowbird_NN_2*	*_dissimilar_JJ*
_ask_VB	*_canary_bird_NN_1*	*_same_JJ_2*
_hold_VB	*_larus_marinus_NN_1*	*_similarity_NN_1*
_provide_VB	*_mew_NN*	*_common_JJ_1*

extract meaningful information from text to disambiguate some words (**WN+CN+Wk** is significantly above **Random** and **WN**). But this is not enough, the labeled examples from XWN and Wku are crucial (+30%) and yields performance better than **MFS** (a strong baseline in WSD) on the XWN test set.

However, performance can be greatly improved by combining the **All** sources model 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 achieves the best performance of all the methods tried.

6.2 Representations

Entity Embeddings Table 2 presents the closest neighbors in the embedding space defined by the model **All** for a few entities. As expected, such neighbors are composed of a mix of lemmas and synsets. The first row depicts variation around the word "mark". Neighbors corresponding to the lemma (column 1) consist in other generic lemmas whereas those for two different synsets (columns 2 and 3) are mainly synsets with clearly different meanings. The second row shows that for common lemmas (column 1), neighbors are also generic lemmas, but precise ones (column 2) are close to synsets defining a sharp meaning. The list returned for *_different_JJ_1* indicates that learnt embeddings do not encode antonymy.

WordNet Enrichment WordNet and ConceptNet use a limited number of relation types (less than 20 of them, e.g. *_has_part* and *_hypernym*), and so they do not consider most verbs as relations. Thanks to our multi-task training and unified representation for MRs and WordNet/ConceptNet relations, our model is potentially able to generalize to such relations that do not exist in WordNet.

Table 3: **Predicted relations** reported by our system (filtering out lemmas) and by TextRunner.

	Model (**All**)	TextRunner
lhs	*_army_NN_1*	army
rel	*_attack_VB_1*	attacked
top ranked *rhs*	*_troop_NN_4*	Israel
	_armed_service_NN_1	the village
	_ship_NN_1	another army
	_territory_NN_1	the city
	_military_unit_NN_1	the fort
top ranked *lhs*	*_business_firm_NN_1*	People
	_person_NN_1	Players
	_family_NN_1	one
	_payoff_NN_3	Students
	_card_game_NN_1	business
rel	*_earn_VB_1*	earn
rhs	*_money_NN_1*	money

As illustration, predicted lists of synsets for relation types that do not exist in the two knowledge bases are given in Table 3. We also compare with lists returned by TextRunner (Yates *et al.*, 2007) (an information extraction tool having extracted information from 100M webpages, to be compared with our 50k Wikipedia articles). Lists from both systems seem to reflect common-sense. However, contrary to our system, TextRunner does not disambiguate different senses of a lemma, and thus it cannot connect its knowledge to an existing resource to enrich it.

7 Conclusion

This paper presented a large-scale system for semantic parsing mapping raw text to disambiguated MRs. The key contributions are: (i) an energy-based model that scores triplets of relations between ambiguous lemmas and unambiguous entities (synsets); and (ii) multi-tasking the learning of such a model over several resources so that we can effectively learn to build disambiguated meaning representations from raw text with relatively limited supervision.

The final system can potentially capture the deep semantics of sentences in its energy function by generalizing the knowledge of multiple resources and linking it to raw text. We obtained positive experimental results on several tasks that appear to support this assertion. Future work should explore the capabilities of such systems further including other semantic tasks, and more evolved grammars, e.g. with FrameNet (Baker *et al.*, 1998; Coppola and Moschitti, 2010).

Acknowledgments

The authors would like to acknowledge L. Bottou and R. Collobert for inspiring discussions. This work was supported by DARPA DL Program, NSERC, mprime, RQCHP, SHARCNET and Pascal2. All code was implemented using Theano (Bergstra *et al.*, 2010).

References

Agirre, E., Ansa, O., Hovy, E., and Martinez, D. (2000). Enriching very large ontologies using the WWW. In *ECAI'00 Ontology Learning Worskshop*.

Baker, C., Fillmore, C., and Lowe, J. (1998). The berkeley FrameNet project. In *ACL '98*, pages 86–90.

Bengio, Y. (2008). Neural net language models. *Scholarpedia*, **3**(1), 3881.

Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. (2003). A neural probabilistic language model. *JMLR*, **3**, 1137–1155.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *SciPy'10*. Oral.

Bordes, A., Weston, J., Collobert, R., and Bengio, Y. (2011). Learning structured embeddings of knowledge bases. In *AAAI'11*, San Francisco, USA.

Bottou, L. (2011). From machine learning to machine reasoning. Technical report, arXiv.1102.1808.

Cimiano, P. (2006). *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer-Verlag.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proc. of the 25th Inter. Conf. on Mach. Learn.*

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *JMLR*, **12**, 2493–2537.

Coppola, B. and Moschitti, A. (2010). A general purpose FrameNet-based shallow semantic parser. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC'10)*.

Cuadros, M. and Rigau, G. (2008). Knownet: using topic signatures acquired from the web for building automatically highly dense knowledge bases. In *COLING'08*.

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

Ge, R. and Mooney, R. J. (2009). Learning a Compositional Semantic Parser using an Existing Syntactic Parser. In *Proc. of the 47th An. Meeting of the ACL*.

Giuglea, A. and Moschitti, A. (2006). Shallow semantic parsing based on FrameNet, VerbNet and PropBank. In *Proceeding of the 17th European Conference on Artificial Intelligence (ECAI'06)*, pages 563–567.

Harabagiu, S. and Moldovan, D. (2002). Knowledge processing on extended WordNet. In C. Fellbaum, editor, *WordNet: An Electronic Lexical Database and Some of its Applications*, pages 379–405. MIT Press.

Havasi, C., Speer, R., and Pustejovsky, J. (2010). Coarse Word-Sense Disambiguation using common sense. In *AAAI Fall Symposium Series*.

Kingsbury, P. and Palmer, M. (2002). From Treebank to PropBank. In *Proc. of the 3rd International Conference on Language Resources and Evaluation*.

Lecun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, f. (2006). A tutorial on Energy-Based learning. In G. Bakir, T. Hofman, B. schölkopf, A. Smola, and B. Taskar, editors, *Predicting Structured Data*. MIT Press.

Liang, P., Jordan, M. I., and Klein, D. (2011). Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*.

Liu, H. and Singh, P. (2004). Focusing on conceptnet's natural language knowledge representation. In *Proc. of the 8th Intl Conf. on Knowledge-Based Intelligent Information and Engineering Syst.*

Martinez, D., de Lacalle, O., and Agirre, E. (2008). On the use of automatically acquired examples for all-nouns word sense disambiguation. *JAIR*, **33**, 79–107.

Miller, G. (1995). WordNet: a Lexical Database for English. *Communications of the ACM*, **38**(11), 39–41.

Mooney, R. (2004). Learning Semantic Parsers: An Important But Under-Studied Problem. In *Proc. of the 19th AAAI Conf. on Artif. Intel.*

Paccanaro, A. and Hinton, G. (2001). Learning distributed representations of concepts using linear relational embedding. *IEEE TKDE*, **13**, 232–244.

Poon, H. and Domingos, P. (2009). Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore.

Poon, H. and Domingos, P. (2010). Unsupervised ontology induction from text. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 296–305, Uppsala, Sweden.

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

Shi, L. and Mihalcea, R. (2004). Open text semantic parsing using FrameNet and WordNet. In *HLT-NAACL 2004: Demo*, pages 19–22.

Snow, R., Jurafsky, D., and Ng, A. (2006). Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 44th annual meeting of the Association for Computational Linguistics*, pages 801–808.

Socher, R., Lin, C., Ng, A. Y., and Manning, C. (2011). Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *ICML'11*.

Suchanek, F., Kasneci, G., and Weikum, G. (2008). Yago: A large ontology from Wikipedia and WordNet. *Web Semant.*, **6**, 203–217.

Sutskever, I., Martens, J., and Hinton, G. (2011). Generating text with recurrent neural networks. In *ICML'11*, pages 1017–1024.

Wu, F. and Weld, D. (2010). Open information extraction using Wikipedia. In *ACL'10*, pages 118–127, Uppsala, Sweden.

Yates, A., Banko, M., Broadhead, M., Cafarella, M., Etzioni, O., and Soderland, S. (2007). TextRunner: Open information extraction on the Web. In *NAACL-HLT '07*, pages 25–26.

Zettlemoyer, L. and Collins, M. (2009). Learning Context-Dependent Mappings from Sentences to Logical Form. In *Proceedings of the 47th Annual Meeting of the ACL*.