Towards Understanding Situated Natural Language

Antoine Bordes
LIP6, Université Paris 6,
Paris, France
antoine.bordes@lip6.fr

Nicolas Usunier
LIP6, Université Paris 6,
Paris, France
nicolas.usunier@lip6.fr

Ronan Collobert
NEC Laboratories America,
Princeton, USA
ronan@collobert.com

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

Abstract

We present a general framework and learning algorithm for the task of *concept labeling*: each word in a given sentence has to be tagged with the unique physical entity (e.g. person, object or location) or abstract concept it refers to. Our method allows both *world knowledge* and *linguistic information* to be used during learning and prediction. We show experimentally that we can *learn* to use world knowledge to resolve *ambiguities* in language, such as word senses or reference resolution, without the use of *handcrafted rules or features*.

1 Introduction

Much of the focus of the natural language processing community lies in solving syntactic or semantic tasks with the aid of sophisticated machine learning algorithms and the encoding of linguistic prior knowledge. For example, a typical way of encoding prior knowledge is to hand-code syntax-based input features or rules for a given task. However, one of the most important features of natural language is that its real-world use (as a tool for humans) is to communicate something about our physical reality or metaphysical considerations of that reality. This is strong prior knowledge that is simply ignored in most current systems.

For example, in current parsing systems there is no allowance for the ability to disambiguate a sentence given knowledge of the physical reality of the world. So, if one happened to know that Bill owned a telescope while John did not, then this should affect parsing decisions given the sentence "John saw Bill in the park with his telescope." Likewise, in terms of reference resolution one could disambiguate the sentence "He passed the exam." if one happens to know that Bill is taking an exam and John is not. Further, one can improve disambiguation of the word bank in "John went to the bank" if you happen to know whether John is out for a walk in the countryside or in the city. Many human disambiguation decisions are in fact based on whether the current sentence agrees well with one's current *world model*. Such a model is dynamic as the current *state of the world* (e.g. the existing entities and their relations) changes over time. Note that this model that we use to disambiguate can be built from our perception of the world (e.g. visual perception) and not necessarily from language at all.

Concept labeling In this paper, we propose a general framework for learning to use *world knowledge* called *concept labeling*. The "knowledge" we consider can be viewed as a database of physical entities existing in the world (e.g. people, locations or objects) as well as abstract concepts, and relations between them, e.g. the location of one entity is expressed in terms of its relation with another entity. Our task consists of labeling each word of a sentence with its corresponding *concept* from the database.

The solution to this task does not provide a full semantic interpretation of a sentence, but we believe it is a important part of that goal. Indeed, in many cases, the meaning of a sentence can only be uncovered after knowing exactly which concepts, e.g. which unique objects in the world, are involved. If one wants to interpret "He passed the exam", one has to infer not only that "He" refers to a "John", and "exam" to a school test, but also exactly which "John" and which test it was. In that sense, concept labeling is more general than the traditional tasks like word-sense disambiguation, co-reference resolution, and named-entity recognition, and can be seen as a unification of them.

Learning algorithm We then go on to propose a tractable algorithm for this task that seamlessly learns to integrate both world knowledge and linguistic con-

Appearing in Proceedings of the 13^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 9 of JMLR: W&CP 9.

tent of a sentence *without the use of any hand-crafted rules or features*. This is a challenging goal and standard algorithms do not achieve it. Our algorithm is first evaluated on human generated sentences from RoboCup commentaries (Chen & Mooney, 2008). Yet this dataset does not involve world knowledge. Hence we present experiments using a novel simulation procedure to generate natural language and concept label pairs: the simulation generates an evolving world, together with sentences describing the successive evolutions. These experiments demonstrate that our algorithm can learn to use world knowledge for word disambiguation and reference resolution when standard methods cannot.

Although clearly only a first step towards the goal of language understanding, which is AI complete, we feel our work is an original way of tackling an important and central problem. In a nut-shell, we show one can learn to resolve ambiguities using world knowledge, which is a prerequisite for further semantic analysis, e.g. for communication.

Previous work Our work concerns learning the connection between two symbolic systems: the one of natural language and the one, non-linguistic, of the concepts present in a database. Making such an association has been studied as the *symbol grounding problem* (Harnad, 1990) in the literature. More specifically, the problem of connecting natural language to another symbolic system is called *grounded language processing* (Roy & Reiter, 2005). Some of such earliest works that used world knowledge to improve linguistic processing involved hand-coded parsing and no learning at all, perhaps the most famous being situated in blocks world (Winograd et al., 1972).

More recent works on grounded language acquisition have focused on *learning* to match language with some other representations. Grounding text with a visual representation also in a blocks-type world was tackled in (Feldman et al., 1996). Other works use visual grounding (Siskind, 1994; Yu & Ballard, 2004; Barnard & Johnson, 2005), or a representation of the intended meaning in some formal language (Fleischman & Roy, 2005; Kate & Mooney, 2007; Wong & Mooney, 2007; Chen & Mooney, 2008; Branavan et al., 2009; Zettlemoyer & Collins, 2009; Liang et al., 2009). Example applications of such grounding include using the multimodal input to improve clustering (with respect to unimodal input) e.g. (Siskind, 1994), word-sense disambiguation (Barnard & Johnson, 2005; Fleischman & Roy, 2005) or semantic parsing (Kate & Mooney, 2007; Chen & Mooney, 2008; Branavan et al., 2009; Zettlemoyer & Collins, 2009).

Work using linguistic context, i.e. previously uttered sentences, also ranges from dialogue systems, e.g. (Allen, 1995), to co-reference resolution (Soon et al., 2001). We do not consider this type of contextual knowledge in this paper, however our framework is extensible to those settings.

2 Concept Labeling

We consider the following setup. One must learn a mapping from a natural language sentence $x \in \mathcal{X}$ to its labeling in terms of *concepts* $y \in \mathcal{Y}$, where y is an ordered set of concepts, one concept for each word in the sentence, i.e. $y = (c_1, \ldots, c_{|x|})$ where $c_i \in \mathcal{C}$, the set of concepts. These concepts belong to a current model of the world which expresses one's knowledge of it. We term it a "*universe*". The framework of concept labeling is illustrated in Figure 1.

World knowledge We define the universe as a set of concepts and their relations to other concepts: $\mathcal{U} = (\mathcal{C}, \mathcal{R}_1, \ldots, \mathcal{R}_n)$ where n is the number of types of relation and $\mathcal{R}_i \subset \mathcal{C}^2$, $\forall i = 1, \ldots, n$. Much work has been done on knowledge representation itself (see (Russell et al., 1995) for an introduction). This is not the focus of this paper and so we made this simplest possible choice. To make things concrete we now describe the template database we use in this paper.

Each concept c of the database is identified using a unique string $name(c)$. Each physical object or action (verb) of the universe has its own referent. For example, two different cartons of milk will be referred to as <*milk1*> and <*milk2*> Here, we use understandable strings as identifiers for clarity but they have no meaning for the system.

In our experiments we will consider two types of relation[1], which give our learner world knowledge about geographical properties of entities in its world, that can be expressed with the following formula:

- $location(c) = c'$ with $c, c' \in \mathcal{C}$: the location of the concept c is the concept c'.
- $containedby(c) = c'$ with $c, c' \in \mathcal{C}$: the concept c' physically holds the concept c.

Strong and weak supervision Usually in sequence labeling, learning is carried out using fully supervised data composed of input sequences explicitly aligned with label annotations. To learn the task of concept labeling, this would result in training examples composed by triples $(x, y, u) \in \mathcal{X} \times \mathcal{Y} \times \mathcal{U}$ as

[1] Of course this list is easily expanded upon. Here, we give two simple properties of physical objects.

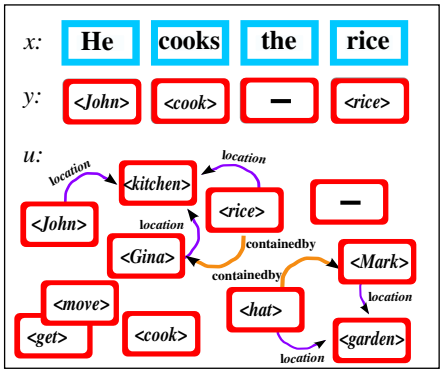

Figure 1: **Concept labeling.** The universe u contains all the concepts known to exist, as well as their relations. The goal is to predict the sequence y of the concepts that each word in the sentence x refers to, including the empty concept "-", using x and u.

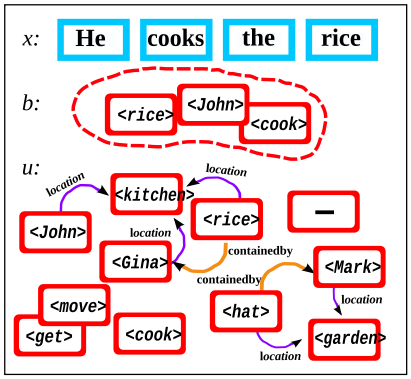

Figure 2: **Weak training triple.** In the weakly supervised setting the supervision bag b consists of the set of concepts related to the sentence x. The alignment between words and concepts is not given and must be discovered.

shown in Figure 1. In order to construct such a dataset it seems likely that humans would have to annotate data for our learner in much the same way as linguists have built labeled datasets for parsing, semantic role labeling, part-of-speech and reference resolution.

A more difficult, but more realistic, learning problem would be for our model to learn from *weakly* supervised data of just observing language given the evolving world-state context. Hence, one instead considers weakly labeled training triples (x, b, u): for a given input sentence x, the supervision b is now a "bag" of concepts (there is no information about ordering/alignment of elements of b to x). This is similar to the setting of (Kate & Mooney, 2007) except we learn to use world knowledge. An illustrating example of a training triple (x, b, u) is given on Figure 2.

To conduct experiments with such a situated learner an immediately realisable setting is within a computer game environment, e.g. multiplayer Internet games. For such games, the knowledge base of concepts is already given by the underlying game model defined by the game designers (e.g. the locations, actors, etc. as well as concepts such as an object being located or worn). Then all that remains is to convert this into our *universe* formalism. Real-world settings are also possible but require technologies for building world knowledge beyond the scope of this work.

Why is this task challenging? The main difficulty of concept labeling arises with *ambiguous words* that can be mislabeled. A concept labeling algorithm must be able to use the available information to solve the ambiguities. In our work, we consider the following kinds of ambiguities, which of course, can be mixed within a sentence:

- **Location-based** ambiguities that can be resolved by the locations of the concepts. Examples: "John picked *it* up" or "*He* dropped his coat in the hall". Information about the location of John, co-located objects and so on can improve the disambiguation.
- **Containedby-based** ambiguities that can be resolved through knowledge of *containedby* relations as in "the *milk* in the closet" or "the *one* in the closet" where there are several cartons of milk (e.g. one in the fridge and one in the closet).
- **Category-based:** A concept is identified in a sentence by an ambiguous term and the disambiguation can be resolved by using semantic categorization. Example: "*He* cooks the rice in the kitchen" where two persons, a male and a female, are in the kitchen.

The first two kinds of ambiguities require the algorithm to be able to *learn* rules based on its available universe knowledge. The last kind can be solved using linguistic information such as word gender or category. However, the necessary rules or linguistic information are *not* given as input features and again the algorithm has to *learn* to infer them. This is one of the main goals of our work. Figure 3 describes the necessary steps for an algorithm to perform such disambiguation. Even for a simple sentence the procedure is rather complex and somehow requires "reasoning".

3 Learning Algorithm

Inference A straight-forward approach to learn a function that maps from sentence x to concept se-

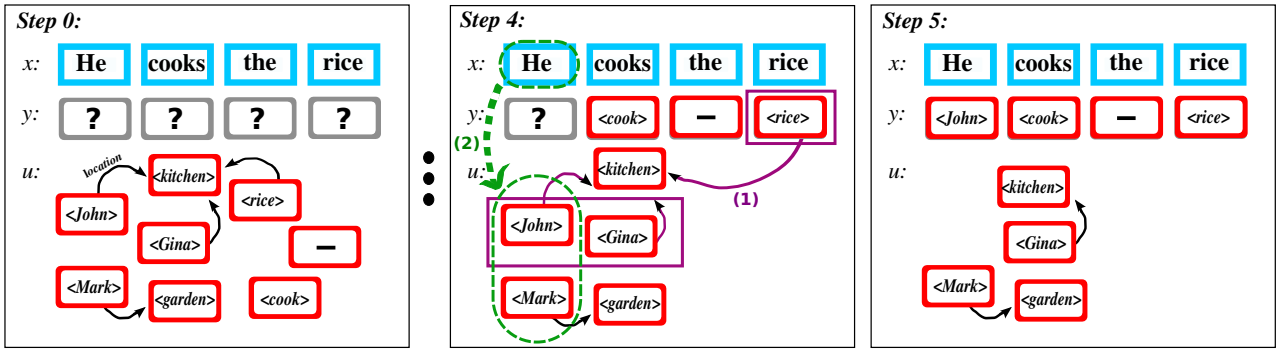

Figure 3: **Inference scheme.** Step 0 defines the task: recover the concepts y given a sentence x and the current state of the *universe* u. For simplicity only relevant concepts and location relations are depicted in this figure. First, non-ambiguous words are labeled in steps 1-3 (not shown). In step 4, to tag the ambiguous pronoun "he", the system has to combine two pieces of information: (1) <*rice*> and the unknown concept might share the same location, <*kitchen*>, and, (2) "he" only refers to a subset of concepts in u (the males).

quence y given u is to consider a model of the form:

$$y = f(x, u) = \text{argmax}_{y'} \quad g(x, y', u), \tag{1}$$

where $g(\cdot)$ returns a scalar that should be a large value when the output concepts y' are consistent with both the sentence x and the current state of the universe u. To find such a function, one can choose a family of functions $g(\cdot)$ and pick the member which minimizes the error: $L = \sum_{\{(x,b,u)\}} \ell(b, f(x, u))$ where the loss function ℓ is 1 if b and the set containing the elements of $f(x, u)$ differ, and 0 otherwise. However, one practical issue of this choice of algorithm is that the exhaustive search over all possible concepts in Equation (1) could be rather slow.

Algorithm 1 Order-free inference

Require: (x, u)
1: $\hat{y}_j^0 = \perp$, $j = 1, \ldots, |x|$, where $\perp$ means *unlabeled*.
2: **while** $t < |x|$ **do**
3: $\quad t = t + 1$.
4: $\quad S_t = \bigcup_{j:\ \hat{y}_j^{t-1} = \perp} \{y' \,|y'_j \in \mathcal{C} \text{ and } \forall i \neq j, y'_i = \hat{y}_i^{t-1}\}$.
5: $\quad \hat{y}^t = \text{argmax}_{y' \in S_t} \quad g(x, y', u)$.
6: **end while**
Return: $\hat{y}^{|x|}$.

The intuition in Figure 3 rather suggests us to use a greedy "order-free" inference process (Shen et al., 2007): we label the word we are most confident in (possibly the least ambiguous, which can be in any position in the sentence) and then use the known features of that concept to help label the remaining ones.

This is detailed in Algorithm 1. At each step, the set of output candidates S_t does not depend on the position in the sentence, one can label any thus far unlabeled word with a concept (line 4). The confidence-based choice is carried out with the function $g(\cdot)$ (line 5), which is learnt to optimize the loss of interest L. Note that Algorithm 1 is quite efficient as it requires only $|\mathcal{C}| \times |x|^2$ computations of $g(\cdot)$, whereas solving (1) requires $|\mathcal{C}|^{|x|}$ (and $|\mathcal{C}| \gg |x|$).

Family of functions Following Bengio et al. (2003) and Collobert & Weston (2008), we chose a neural-network architecture for $g(\cdot)$ which is capable of learning to encode some of the linguistic information required to perform disambiguation (such as, "he" only refers to males, in the example of Figure 3). The actual form of the function $g(\cdot)$ is:

$$g(x, y, u) = \sum_{i=1}^{|x|} g_i(x, y_{-i}, u)^\top h(y_i, u) \tag{2}$$

where $g_i(\cdot) \in \mathbb{R}^N$ is a "sliding window" representation of width w centered on the i^{th} position in the sentence, y_{-i} is the same as y except that the i^{th} position $(y_{-i})_i = \perp$, and $h(\cdot) \in \mathbb{R}^N$ is a linear mapping into the same space as $g(\cdot)$. We constrain $||h(\perp, u)|| = 0$ so that as yet unlabeled outputs do not play a role. A less mathematical explanation of this model is as follows: $g_i(\cdot)$ takes a window of the input sentence *and* previously labeled concepts centered around the i^{th} word and embeds them into an N dimensional space. $h(y_i, u)$ embeds the i^{th} concept into the same space, where both mappings are *learnt*. The magnitude of their dot product in this space indicates how confident the model is that the i^{th} word, given its context, should be labeled with concept y_i. This representation is also useful from a computational point of view because $g_i(\cdot)$ and $h(\cdot)$ can be cached and reused in Algorithm 1, making inference fast.

The functions $g_i(\cdot)$ and $h(\cdot)$ are simple two-layer lin-

ear neural networks in a similar spirit to (Collobert & Weston, 2008). The first layer of both are so-called "Lookup Tables". We represent each word m in the dictionary with a unique vector $D(m) \in \mathbb{R}^d$ and every unique concept name $name(c)$ also with a unique vector $C(name(c)) \in \mathbb{R}^d$, where we *learn these mappings*. No hand-crafted syntactic features are used.

To represent a concept *and* its relations we do something slightly more complicated. A particular concept c (e.g. an object in a particular location, or being held by a particular person) is expressed as the concatenation of the three unique concept name vectors:

$$\bar{C}(c) = (C(name(c)), C(name(location(c))), \quad (3)$$
$$C(name(containedby(c)))).$$

In this way, the learning algorithm can take these *dynamic* relations from the *universe* into account, if they are relevant for the labeling task. Hence, the first layer of the network $g_i(\cdot)$ outputs[2]:

$$g_i^1(x, y_{-i}, u) = \Big(D(x_{i-\frac{w-1}{2}}), \ldots, D(x_{i+\frac{w-1}{2}}),$$
$$\bar{C}((y_{-i})_{i-\frac{w-1}{2}}), \ldots, \bar{C}((y_{-i})_{i+\frac{w-1}{2}})\Big) \ .$$

The second layer is a linear layer that maps from this $4wd$ dimensional vector to the N dimensional output, i.e. overall we have the function (with parameters $W_g \in \Re^{4wd \times N}$ and $b_g \in \Re^N$) :

$$g_i(x, y_{-i}, u) = W_g \ g_i^1(x, y_{-i}, u) + b_g.$$

Likewise, $h(y_i, u)$ has a first layer which outputs $\bar{C}(y_i)$, followed by a linear layer mapping from this $3d$ dimensional vector to N, i.e. (with parameters $W_h \in \Re^{3d \times N}$ and $b_h \in \Re^N$):

$$h(y_i, u) = W_h \ \bar{C}(y_i) + b_h.$$

Overall, we chose a linear architecture that avoids strongly engineered features, assumes little prior knowledge about the mapping task in hand, but is powerful enough to capture many kinds of relations between words and concepts. We justify our algorithmic choices by comparing with alternative choices in the experimental section 4.

It is important that algorithms for this task both capture the complexity of the task, and remain scalable. Our algorithm should scale to a large number of concepts, indeed it was designed with that in mind: inference scales linearly with the number of concepts. In the model described so far, however, an increase in the number of relations increase the number of parameters to learn in the network which might not be scalable. This can be remedied by mapping a large set of relations to a low dimensional subspace first using one more layer in the network. However, in this paper for simplicity we do not describe this extension further.

LaSO-type training We train our system online by employing a variation on the LaSO (Learning As Search Optimization) training process (Daumé III & Marcu, 2005). LaSO's central idea is to mix training and inference in a single *learning for search* strategy. During training, we thus perform inference on each given triple (x, b, u) using Algorithm 1, with the following updates in order to learn the model parameters.

Strong supervision We define the predicted labeling $\hat{y}^t$ at inference step t (see Algorithm 1, step 5) as y-good, compared to the true labeling y, if either $\hat{y}_i^t = y_i$ or $\hat{y}_i^t = \perp$ for all i. As soon as $\hat{y}^t$ is no longer y-good we make an "early update" to the model, similar to (Collins & Roark, 2004). The update is a stochastic gradient step so that each possible y-good state one can choose from $\hat{y}^{t-1}$ is ranked higher than the current incorrect state, i.e. we would like to satisfy the ranking constraints:

$$\forall i : \hat{y}_i^{t-1} = \perp \ , \ g(x, \hat{y}_{+(i,y_i)}^{t-1}, u) > g(x, \hat{y}^t, u),$$

where $\hat{y}_{(i,+y_i)}^{t-1}$ denotes a vector which is the same as $\hat{y}^{t-1}$ except its i^{th} element is set to y_i. Note that if all such constraints are satisfied then all training examples must be correctly classified.

Weak supervision For the weakly supervised case we are given only a supervision bag b. In that case, at each round we define the predicted labeling $\hat{y}^t$ as y-good if either $\hat{y}_i^t \in b$ or $\hat{y}_i^t = \perp$ for all i (all the words are either unlabeled or labeled with an element of bag).[3] As soon as $\hat{y}^t$ is no longer y-good the model is updated to satisfy the ranking constraints:

$$\forall i : \hat{y}_i^{t-1} = \perp \ , \ \forall c \in b, \quad g(x, \hat{y}_{+(i,c)}^{t-1}, u) > g(x, \hat{y}^t, u)$$

where $\hat{y}_{+(i,c)}^{t-1}$ is a vector which is the same as $\hat{y}^{t-1}$ except its i^{th} element is set to the concept c. Intuitively, if a label prediction for the word x_j in position j does not belong to the bag b then we require any prediction that *does* belong to it to be ranked above this incorrect prediction. (If all such constraints are satisfied, the correct bags are predicted.)

After a concept c from b has been picked, we remove it from the bag. Otherwise the same element could

[2] Padding must be used when indices are out of bounds.

[3] For weak supervision, to simplify learning we chose to initialize the weights $C(\cdot)$ and $D(\cdot)$ of our network using CCA (Li & Shawe-Taylor (2006)) treating both x and y as bags. For unambiguous words this essentially initializes their representation similar to their corresponding concept.

be picked again and again, resulting in an output sequence that does not violate the constraints. Yet, as a bag can contain duplicates, this does not forbid associating the same concept to several words.

Why does this work? Consider again the example "He cooks the rice" in Figure 3. We cannot resolve the first word in the sentence "He" with the true concept label <*John*> until we know that "rice" corresponds to the concept <*rice*> which we know is located in the kitchen, as is John, thereby making him the most likely referent. This is why we choose to label words with concepts in an order independent of the position in the sentence in Algorithm 1, simply label from left to right could not work. The algorithm has to learn which word to label first, and presumably, it starts by labeling the least ambiguous ones. This is what we have observed experimentally. Once <*rice*> has been identified, its features including its location will influence the function $g(x, y, u)$ and the word "He" is more easily disambiguated. Simultaneously, our method must learn the N dimensional representations $g_i(\cdot)$ and $h(\cdot)$ such that "He" matches with <*John*> rather than <*Gina*>, i.e. equation (2) is a larger value. This should happen because during training <*John*> and "He" often co-occur and allow to conclude the disambiguation.

Note that our system can learn the general principle that two things that are in the same place are more likely to be referred to in the same sentence, and does not have to re-learn that for all possible places and things. In general, it can resolve many kinds of ambiguities, both from syntax, semantics, or a combination, such as, for example all the cases given in Section 2. Furthermore, the LaSO-type training for weak supervision can implicitly learn the alignment between words and concepts, even though it is never given.

4 Experiments

4.1 RoboCup commentaries

Before using any world knowledge, we wanted to assess that our algorithm was able to be trained under weak supervision on natural language. Hence we tested it on the RoboCup commentary dataset[4]. This data contains human commentaries on football simulations over four games labeled with semantic descriptions of actions: passes, offside, penalties, ... along with the players involved. For example, a typical sentence one has to provide the semantic parse of is "Pink5 passed the ball to Pink4 who kicked it offside". However, each training sentence given is weakly supervised with all the actions that took place around the same time of the commentary. We treated each semantic description as a "bag" of concepts for weak supervision.

Following Chen & Mooney (2008), we trained on one match and tested on the other three, averaging over all four possible splits and we report the "matching" score: each input commentary is associated with several actions, we measure how often the system retrieve the correct one. We tackled this in two successive steps: (1) a bag of labels is predicted using Algorithm 1, (2) we choose to match to the bag from the set of actions that has the highest cosine similarity with the prediction. Our system achieves an F1 score of 0.669. Previously reported methods from (Chen & Mooney, 2008) Krisper (0.645 F1) and Wasper-Gen (0.65 F1) achieve similar results (random matching yields 0.465 F1). Results on this benchmark clearly indicate that our method can handle both weak supervision and natural language sentences.

4.2 Simulated world (with world knowledge)

Simulation RoboCup sentences do not involve any lexical ambiguity. To evaluate the ability of our method to use world knowledge to perform disambiguation, we thus created a simulation.

To conduct experiments on an environment with a reasonably large size we built the following artificial universe designed to simulate a house interior. It contains 58 concepts: 15 verbs (<*move*>, <*get*>, <*give*>,...) along with 10 actors (<*John*>, <*dog*>,...), 15 small objects (<*water*>, <*chocolate*>, <*doll*>,...), 6 rooms (<*kitchen*>,...) and 12 pieces of furniture (<*couch*>, ...). We define the set of describing words for each concept to contain at least two terms: an ambiguous one (using a pronoun) and a unique one. 75 words are used for generating sentences $x \in \mathcal{X}$. The simulation generates actions in the world along with sentences that describe them using a simple grammar. For example a simulation step could produce the results:

1. Pick the event <*move*>(<*Gina*>, <*hall*>).
2. Generate the training sample (x, b, u) = ("she goes from the bedroom to the hall", {<*hall*>,<*Gina*>,<*bedroom*>,<*move*>}, u).
3. Update u with *location*(<*Gina*>) = <*hall*>.

Some examples of generated sentences are given in Table 1. For our experiments we record 50,000 triples (x, y, u) for training and 20,000 for testing. Around 55% of sentences contain lexical ambiguities.[5]

[4] see (Chen & Mooney, 2008) or `http://www.cs.utexas.edu/~ml/clamp/sportscasting/#data` for details.

[5] The dataset can be downloaded from `http://webia.lip6.fr/~bordes/mywiki/doku.php?id=worlddata` .

x:	he sits on the chair
y:	<*Mark*> <*sit*> - - <*chair*>
x:	the brother gives the toy to her
y:	- <*Mark*> <*give*> - <*toy*> - <*Gina*>
x:	the father gets some yoghurt from the sideboard
y:	- <*John*> <*get*> - <*yoghurt*> - - <*sideboard*>
x:	she goes from the bedroom to the kitchen
y:	<*Gina*> <*move*> - - <*bedroom*> - - <*kitchen*>

Table 1: **Simulated world examples.** Our task is to label sentences x given world knowledge u (not shown).

Method	Supervision	Features	Train Err	Test Err
SVM_{struct}	strong	$x + u$ (*loc*, *contain*)	18.68%	23.57%
NN_{LR}	strong	$x + u$ (*loc*, *contain*)	5.42%	5.75%
NN_{OF}	strong	x	32.50%	35.87%
NN_{OF}	strong	$x + u$ (*contain*)	15.15%	17.04%
NN_{OF}	strong	$x + u$ (*loc*)	5.07%	5.22%
NN_{OF}	strong	$x + u$ (*loc*, *contain*)	**0.0%**	**0.11%**
NN_{OF}	weak	$x + u$ (*loc*, *contain*)	**0.64%**	**0.72%**

Table 2: **Simulation results.** We compare our order-free neural network (NN_{OF}) with world knowledge trained using strong (line 6) and weak (line 7) supervision to several variants (no world knowledge - line 3; partial world knowledge – lines 4-5; SVMs – line 1; and left-right inference – line 2). NN_{OF} using full world knowledge performs best.

Algorithms We compare several models. Firstly, we evaluate our "order-free" neural network based algorithm presented in Section 3 (NN_{OF} using $x + u$) trained with two kinds of supervision: the *strong* setting for which word-concepts alignments are given, and our realistic weak setting using a bag. We compare to a model with no access to world knowledge (NN_{OF} using x) or only partial access via *location* or *containedby* knowledge only (rather than both).

Finally, we compare to two more strongly supervised methods: a greedy left-to-right labeling NN, NN_{LR} (in order to assess the necessity of order-free) and a structured output SVM, SVM_{struct} (Tsochantaridis et al., 2005). For the SVM, the features from the world model are used as additional input features and Viterbi is used to decode the outputs. Only a linear model was used due to the infeasibility of training non-linear ones (and all the NNs are linear). In all experiments we used word and concept dimension $d = 20$, $g(\cdot)$ and $h(\cdot)$ have dimension $N = 200$, a sliding window width of $w = 13$ (i.e., 6 words on either side of a central word), and we chose the learning rate that minimized the training error as described in Section 3.

Results The results are given in Table 2. The error rates express the proportion of predicted sequences with *at least one* incorrect tag. Our model (NN_{OF}) learns to use world knowledge to disambiguate on this task: it obtains a test error close to 0% (line 6) with this knowledge, and around 35% error without (line 3). It is worth noting that training under weak supervision (last line) does not degrade accuracy: the same model using the less realistic strong setting (line 6) is only slightly better.

Confirming our intuitions about the inference (as in Figure 3), the comparison with other algorithms highlights the following points: (i) order-free labeling of concepts is important compared to more restricted labeling schemes such as left-right labeling (NN_{LR}); (ii) the architecture of our NN which embeds concepts is able to capture some useful linguistic information and thus helps generalization; this should be compared to SVM_{struct} which does not perform as well. Note that a nonlinear SVM *or* a linear SVM with hand-crafted features are likely to perform better, but the former is too slow and the latter is what we are trying to avoid as such methods will not generalize to harder tasks.

We believe SVM_{struct} fails for several reasons. First, it uses a Viterbi decoding that can only employ first-order Markov interactions (for tractability). Thus, simple reasoning involving previously labeled examples located "too far" before or after the current word is impossible. Second, its fixed feature representation encodes world knowledge in a binary encoding and e.g. one cannot learn that concepts sharing the same location are likely to appear within a sentence other than by memorizing all the possibilities. Our method can discover that and adapt its feature representation accordingly while SVM_{struct} cannot.

We constructed our simulation such that all ambiguities could be resolved with world knowledge, which is why we can obtain almost 0%: this is a good sanity check showing that our method is working well. We believe it is a prerequisite that we do well here if we hope to do well on harder tasks. The simulation we built uses rules to generate actions and utterances, but our learning algorithm uses no such hand-built rules but instead successfully *learns* them. We believe this could make our algorithm generalize to harder tasks.

One may still be concerned that the environment is

simple and that we *know a priori* that the model we are learning is sufficiently expressive to capture all the relevant information in the world. In the real world one might not have enough information to achieve such high recognition rates. We therefore considered settings where aspects of the world could not be captured directly in the model that is learned: NN_{OF} using $x + u$ (*contain*) employs a world model with only a subset of the relational information (it does not have access to the *loc* relations). Similarly, we tried NN_{OF} using $x + u$ (*loc*) as well. The results in Table 2 show that our model still learns to perform well (i.e. better than no world knowledge at all) in the presence of hidden/unavailable world knowledge. Finally, if the amount of training data is reduced we can still perform well. With 5000 training examples for NN_{OF} ($x + u$ (*loc*, *contain*)) with the same parameters we obtain 3.1% test error.

5 Conclusion and Future Work

We have described a *general* framework for language grounding based on the task of *concept labeling*. The learning algorithm we propose is *scalable* and *flexible*: it learns with only weakly supervised raw data, and no prior knowledge of how concepts in the world are expressed in natural language. We have tested our framework within a simulation, showing that it is possible to *learn* (rather than engineer) to resolve ambiguities using world knowledge. We also showed we can learn with real *human annotated* data (RoboCup commentaries). Although clearly only a first step towards the goal of language understanding we feel our work is an original way of tackling an important and central problem. The most direct application of our work is within computer games, but other communication tasks could also apply with an increased effort.

In terms of algorithms, we feel two important research directions concern (*i*) addressing the scalability of learning with increased numbers of relations and database concepts and (*ii*) improving the algorithms for dealing with weak supervision for larger, noisier and more realistic datasets.

Acknowledgments

Part of this work was supported by ANR-09-EMER-001 and by the Network of Excellence PASCAL2. Antoine Bordes was also supported by the French DGA.

References

Allen, J. *Natural language understanding*. Benjamin-Cummings Publishing Co., 1995.

Barnard, K. and Johnson, M. Word Sense Disambiguation with Pictures. *Artificial Intelligence*, 167(1-2), 2005.

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

Branavan, S.R.K., Chen, H., Zettlemoyer, L., and Barzilay, R. Reinforcement learning for mapping instructions to actions. In *ACL '09*, 2009.

Chen, D.L. and Mooney, R.J. Learning to Sportscast: A Test of Grounded Language Acquisition. In *ICML '08*, 2008.

Collins, M. and Roark, B. Incremental parsing with the perceptron algorithm. In *ACL '04*, 2004.

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

Daumé III, H. and Marcu, D. Learning as Search Optimization: Approximate Large Margin Methods for Structured Prediction. In *ICML '05*, 2005.

Feldman, J., Lakoff, G., Bailey, D., Narayanan, S., Regier, T., and Stolcke, A. L 0-The first five years of an automated language acquisition project. *Artificial Intelligence Review*, 10(1):103–129, 1996.

Fleischman, M. and Roy, D. Intentional Context in Situated Language Learning. In *9th Conference on Computational Natural Language Learning*, 2005.

Harnad, S. The Symbol Grounding Problem. *Physica D*, 42(1-3):335–346, 1990.

Kate, R.J. and Mooney, R.J. Learning Language Semantics from Ambiguous Supervision. In *AAAI '07*, 2007.

Li, Y. and Shawe-Taylor, J. Using KCCA for Japanese–English cross-language information retrieval and document classification. *Journal of intelligent information systems*, 27(2):117–133, 2006.

Liang, P., Jordan, M. I., and Klein, D. Learning semantic correspondences with less supervision. In *ACL '09*, 2009.

Roy, D. and Reiter, E. Connecting Language to the World. *Artificial Intelligence*, 167(1-2):1–12, 2005.

Russell, S.J., Norvig, P., Canny, J.F., Malik, J., and Edwards, D.D. *Artificial intelligence: a modern approach*. Prentice Hall Englewood Cliffs, NJ, 1995.

Shen, L., Satta, G., and Joshi, A. Guided Learning for Bidirectional Sequence Classification. In *ACL '07*, 2007.

Siskind, J.M. Grounding Language in Perception. *Artificial Intelligence Review*, 8(5):371–391, 1994.

Soon, W.M., Ng, H.T., and Lim, D.C.Y. A Machine Learning Approach to Coreference Resolution of Noun Phrases. *Computational Linguistics*, 27(4):521–544, 2001.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. Large Margin Methods for Structured and Interdependent Output Variables. *JMLR*, 6:1453–1484, 2005.

Winograd, T., Barbour, M.G., and Stocking, C.R. *Understanding natural language*. Academic Press, 1972.

Wong, Y.W. and Mooney, R. Learning Synchronous Grammars for Semantic Parsing with Lambda Calculus. In *ACL '07*, 2007.

Yu, C. and Ballard, D.H. On the Integration of Grounding Language and Learning Objects. In *AAAI '04*, 2004.

Zettlemoyer, L. and Collins, M. Learning context-dependent mappings from sentences to logical form. In *ACL '09*, 2009.