Neural Sequence Learning Models for Word Sense Disambiguation

Alessandro Raganato, Claudio Delli Bovi and **Roberto Navigli**
Department of Computer Science
Sapienza University of Rome
{raganato,dellibovi,navigli}@di.uniroma1.it

Abstract

Word Sense Disambiguation models exist in many flavors. Even though supervised ones tend to perform best in terms of accuracy, they often lose ground to more flexible knowledge-based solutions, which do not require training by a word expert for every disambiguation target. To bridge this gap we adopt a different perspective and rely on sequence learning to frame the disambiguation problem: we propose and study in depth a series of end-to-end neural architectures directly tailored to the task, from bidirectional Long Short-Term Memory to encoder-decoder models. Our extensive evaluation over standard benchmarks and in multiple languages shows that sequence learning enables more versatile all-words models that consistently lead to state-of-the-art results, even against word experts with engineered features.

1 Introduction

As one of the long-standing challenges in Natural Language Processing (NLP), Word Sense Disambiguation (Navigli, 2009, WSD) has received considerable attention over recent years. Indeed, by dealing with lexical ambiguity an effective WSD model brings numerous benefits to a variety of downstream tasks and applications, from Information Retrieval and Extraction (Zhong and Ng, 2012; Delli Bovi et al., 2015) to Machine Translation (Carpuat and Wu, 2007; Xiong and Zhang, 2014; Neale et al., 2016). Recently, WSD has also been leveraged to build continuous vector representations for word senses (Chen et al., 2014; Iacobacci et al., 2015; Flekova and Gurevych, 2016).

Inasmuch as WSD is described as the task of associating words in context with the most suitable entries in a pre-defined sense inventory, the majority of WSD approaches to date can be grouped into two main categories: supervised (or semi-supervised) and knowledge-based. Supervised models have been shown to consistently outperform knowledge-based ones in all standard benchmarks (Raganato et al., 2017), at the expense, however, of harder training and limited flexibility. First of all, obtaining reliable sense-annotated corpora is highly expensive and especially difficult when non-expert annotators are involved (de Lacalle and Agirre, 2015), and as a consequence approaches based on unlabeled data and semi-supervised learning are emerging (Taghipour and Ng, 2015b; Başkaya and Jurgens, 2016; Yuan et al., 2016; Pasini and Navigli, 2017).

Apart from the shortage of training data, a crucial limitation of current supervised approaches is that a dedicated classifier (*word expert*) needs to be trained for every target lemma, making them less flexible and hampering their use within end-to-end applications. In contrast, knowledge-based systems do not require sense-annotated data and often draw upon the structural properties of lexico-semantic resources (Agirre et al., 2014; Moro et al., 2014; Weissenborn et al., 2015). Such systems construct a model based only on the underlying resource, which is then able to handle multiple target words at the same time and disambiguate them jointly, whereas word experts are forced to treat each disambiguation target in isolation.

In this paper our focus is on supervised WSD, but we depart from previous approaches and adopt a different perspective on the task: instead of framing a separate classification problem for each given word, we aim at modeling the joint disambiguation of the target text as a whole in terms of a sequence labeling problem. From this standpoint, WSD amounts to translating a sequence of words into a sequence of potentially sense-tagged tokens.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1165–1176
Copenhagen, Denmark, September 7–11, 2017.

With this in mind, we design, analyze and compare experimentally various neural architectures of different complexities, ranging from a single bidirectional Long Short-Term Memory (Graves and Schmidhuber, 2005, LSTM) to a sequence-to-sequence approach (Sutskever et al., 2014). Each architecture reflects a particular way of modeling the disambiguation problem, but they all share some key features that set them apart from previous supervised approaches to WSD: they are trained end-to-end from sense-annotated text to sense labels, and learn a single all-words model from the training data, without fine tuning or explicit engineering of local features.

The contributions of this paper are twofold. First, we show that neural sequence learning represents a novel and effective alternative to the traditional way of modeling supervised WSD, enabling a single all-words model to compete with a pool of word experts and achieve state-of-the-art results, while also being easier to train, arguably more versatile to use within downstream applications, and directly adaptable to different languages without requiring additional sense-annotated data (as we show in Section 6.2); second, we carry out an extensive experimental evaluation where we compare various neural architectures designed for the task (and somehow left underinvestigated in previous literature), exploring different configurations and training procedures, and analyzing their strengths and weaknesses on all the standard benchmarks for all-words WSD.

2 Related Work

The literature on WSD is broad and comprehensive (Agirre and Edmonds, 2007; Navigli, 2009): new models are continuously being developed (Yuan et al., 2016; Tripodi and Pelillo, 2017; Butnaru et al., 2017) and tested over a wide variety of standard benchmarks (Edmonds and Cotton, 2001; Snyder and Palmer, 2004; Pradhan et al., 2007; Navigli et al., 2007, 2013; Moro and Navigli, 2015). Moreover, the field has been explored in depth from different angles by means of extensive empirical studies and evaluation frameworks (Pilehvar and Navigli, 2014; Iacobacci et al., 2016; McCarthy et al., 2016; Raganato et al., 2017).

As regards supervised WSD, traditional approaches are generally based on extracting local features from the words surrounding the target, and then training a classifier (Zhong and Ng, 2010; Shen et al., 2013) for each target lemma. In their latest developments, these models include more complex features based on word embeddings (Taghipour and Ng, 2015b; Rothe and Schütze, 2015; Iacobacci et al., 2016).

The recent upsurge of neural networks has also contributed to fueling WSD research: Yuan et al. (2016) rely on a powerful neural language model to obtain a latent representation for the whole sentence containing a target word w; their instance-based system then compares that representation with those of example sentences annotated with the candidate meanings of w. Similarly, Context2Vec (Melamud et al., 2016) makes use of a bidirectional LSTM architecture trained on an unlabeled corpus and learns a context vector for each sense annotation in the training data. Finally, Kågebäck and Salomonsson (2016) present a supervised classifier based on bidirectional LSTM for the lexical sample task (Kilgarriff, 2001; Mihalcea et al., 2004). All these contributions have shown that supervised neural models can achieve state-of-the-art performances without taking advantage of external resources or language-specific features. However, they all consider each target word as a separate classification problem and, to the best of our knowledge, very few attempts have been made to disambiguate a text jointly using sequence learning (Ciaramita and Altun, 2006).

Sequence learning, especially using LSTM (Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005; Graves, 2013), has become a well-established standard in numerous NLP tasks (Zhou and Xu, 2015; Ma and Hovy, 2016; Wang and Chang, 2016). In particular, sequence-to-sequence models (Sutskever et al., 2014) have grown increasingly popular and are used extensively in, e.g., Machine Translation (Cho et al., 2014; Bahdanau et al., 2015), Sentence Representation (Kiros et al., 2015), Syntactic Parsing (Vinyals et al., 2015), Conversation Modeling (Vinyals and Le, 2015), Morphological Inflection (Faruqui et al., 2016) and Text Summarization (Gu et al., 2016). In line with this trend, we focus on the (so far unexplored) context of supervised WSD, and investigate state-of-the-art all-words approaches that are based on neural sequence learning and capable of disambiguating all target content words within an input text, a key feature in several knowledge-based approaches.

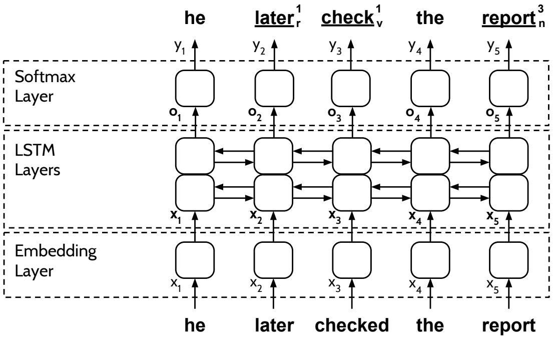

Figure 1: Bidirectional LSTM sequence labeling architecture for WSD (2 hidden layers). We use the notation of Navigli (2009) for word senses: w_p^i is the i-th sense of w with part of speech p.

3 Sequence Learning for Word Sense Disambiguation

In this section we define WSD in terms of a sequence learning problem. While in its classical formulation (Navigli, 2009) WSD is viewed as a classification problem for a given word w in context, with word senses of w being the class labels, here we consider a variable-length sequence of input symbols $\vec{x} = \langle x_1, ..., x_T \rangle$ and we aim at predicting a sequence of output symbols $\vec{y} = \langle y_1, ..., y_{T'} \rangle$.[1] Input symbols are word tokens drawn from a given vocabulary V.[2] Output symbols are either drawn from a pre-defined sense inventory S (if the corresponding input symbols are open-class content words, i.e., nouns, verbs, adjectives or adverbs), or from the same input vocabulary V (e.g., if the corresponding input symbols are function words, like prepositions or determiners). Hence, we can define a WSD model in terms of a function that maps sequences of symbols $x_i \in V$ into sequences of symbols $y_j \in O = S \cup V$.

Here all-words WSD is no longer broken down into a series of distinct and separate classification tasks (one per target word) but rather treated directly at the sequence level, with a single model handling all disambiguation decisions. In what follows, we describe three different models for accomplishing this: a traditional LSTM-based model (Section 3.1), a variant that incorporates an attention mechanism (Section 3.2), and an encoder-decoder architecture (Section 3.3).

3.1 Bidirectional LSTM Tagger

The most straightforward way of modeling WSD as formulated in Section 3 is that of considering a sequence labeling architecture that tags each symbol $x_i \in V$ in the input sequence with a label $y_j \in O$. Even though the formulation is rather general, previous contributions (Melamud et al., 2016; Kågebäck and Salomonsson, 2016) have already shown the effectiveness of recurrent neural networks for WSD. We follow the same line and employ a bidirectional LSTM architecture: in fact, important clues for disambiguating a target word could be located anywhere in the context (not necessarily before the target) and for a model to be effective it is crucial that it exploits information from the whole input sequence at every time step.

Architecture. A sketch of our bidirectional LSTM tagger is shown in Figure 1. It consists of:

- An embedding layer that converts each word $x_i \in \vec{x}$ into a real-valued d-dimensional vector $\mathbf{x}_i$ via the embedding matrix $\mathbf{W} \in \mathbb{R}^{d \times |V|}$;
- One or more stacked layers of bidirectional LSTM (Graves and Schmidhuber, 2005). The hidden state vectors $\mathbf{h}_i$ and output vectors $\mathbf{o}_i$ at the i^{th} time step are then obtained as the concatenations of the forward and backward pass vectors $\overrightarrow{\mathbf{h}}_i, \overrightarrow{\mathbf{o}}_i$ and $\overleftarrow{\mathbf{h}}_i, \overleftarrow{\mathbf{o}}_i$;
- A fully-connected layer with softmax activation that turns the output vector $\mathbf{o}_i$ at the i^{th} time step into a probability distribution over the output vocabulary O.

Training. The tagger is trained on a dataset of N labeled sequences $\{(\vec{x}_k, \vec{y}_k)\}_{k=1}^N$ directly obtained from the sentences of a sense-annotated corpus, where each $\vec{x}_k$ is a sequence of word tokens, and each $\vec{y}_k$ is a sequence containing both word tokens and sense labels. Ideally $\vec{y}_k$ is a copy of $\vec{x}_k$ where each content word is sense-tagged. This is, however, not the case in many real-world datasets, where only a subset of the content words is annotated; hence the architecture is designed to deal with both fully and partially annotated sentences. Apart from sentence splitting and tokenization, no preprocessing is required on the training data.

[1] In general $\vec{x}$ and $\vec{y}$ might have different lengths, e.g., if $\vec{x}$ contains a multi-word expression (*European Union*) which is mapped to a unique sense identifier (`European Union`^1_n).

[2] V generalizes traditional vocabularies used in WSD and includes both word lemmas and inflected forms.

3.2 Attentive Bidirectional LSTM Tagger

The bidirectional LSTM tagger of Section 3.1 exploits information from the whole input sequence $\vec{x}$, which is encoded in the hidden state $\mathbf{h}_i$. However, certain elements of $\vec{x}$ might be more discriminative than others in predicting the output label at a given time step (e.g., the syntactic subject and object when predicting the sense label of a verb).

We model this hunch by introducing an attention mechanism, already proven to be effective in other NLP tasks (Bahdanau et al., 2015; Vinyals et al., 2015), into the sequence labeling architecture of Section 3.1. The resulting *attentive* bidirectional LSTM tagger augments the original architecture with an attention layer, where a context vector $\mathbf{c}$ is computed from all the hidden states $\mathbf{h}_1, ..., \mathbf{h}_T$ of the bidirectional LSTM. The attentive tagger first reads the entire input sequence $\vec{x}$ to construct $\mathbf{c}$, and then exploits $\mathbf{c}$ to predict the output label y_j at each time step, by concatenating it with the output vector $\mathbf{o}_j$ of the bidirectional LSTM (Figure 2).

We follow previous work (Vinyals et al., 2015; Zhou et al., 2016) and compute $\mathbf{c}$ as the weighted sum of the hidden state vectors $\mathbf{h}_1, ..., \mathbf{h}_T$. Formally, let $H \in \mathbb{R}^{n \times T}$ be the matrix of hidden state vectors $[\,\mathbf{h}_1, ..., \mathbf{h}_T\,]$, where n is the hidden state dimension and T is the input sequence length (cf. Section 3). $\mathbf{c}$ is obtained as follows:

$$\begin{aligned} \mathbf{u} &= \omega^T \tanh(H) \\ \mathbf{a} &= softmax(\mathbf{u}) \\ \mathbf{c} &= H\mathbf{a}^T \end{aligned} \tag{1}$$

where $\omega \in \mathbb{R}^n$ is a parameter vector, and $\mathbf{a} \in \mathbb{R}^T$ is the vector of normalized attention weights.

3.3 Sequence-to-Sequence Model

The attentive tagger of Section 3.2 performs a two-pass procedure by first reading the input sequence $\vec{x}$ to construct the context vector $\mathbf{c}$, and then predicting an output label y_j for each element in $\vec{x}$. In this respect, the attentive architecture can effectively be viewed as an encoder for $\vec{x}$. A further generalization of this model would then be a complete encoder-decoder architecture (Sutskever et al., 2014) where WSD is treated as a sequence-to-sequence mapping (*sequence-to-sequence WSD*), i.e., as the "translation" of word sequences into sequences of potentially sense-tagged tokens.

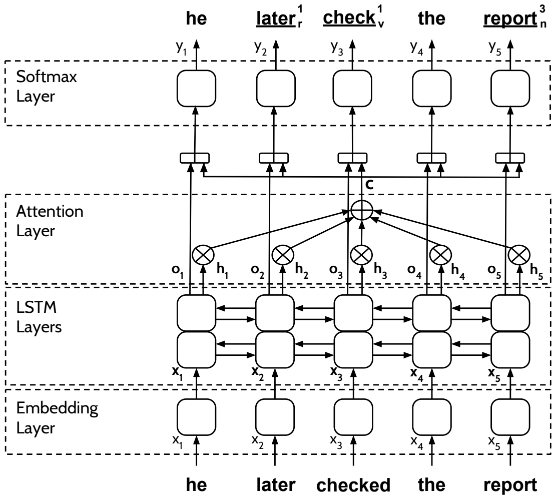

Figure 2: Attentive bidirectional LSTM sequence labeling architecture for WSD (2 hidden layers).

In the sequence-to-sequence framework, a variable-length sequence of input symbols $\vec{x}$ is represented as a sequence of vectors $\vec{\mathbf{x}} = \langle \mathbf{x}_1, ..., \mathbf{x}_T \rangle$ by converting each symbol $x_i \in \vec{x}$ into a real-valued vector $\mathbf{x}_i$ via an embedding layer, and then fed to an encoder, which generates a fixed-dimensional vector representation of the sequence. Traditionally, the encoder function is a Recurrent Neural Network (RNN) such that:

$$\begin{aligned} \mathbf{h}_t &= f(\mathbf{h}_{t-1}, \mathbf{x}_t) \\ \mathbf{c} &= q(\{\mathbf{h}_1, ..., \mathbf{h}_T\}) \end{aligned} \tag{2}$$

where $\mathbf{h}_t \in \mathbb{R}^n$ is the n-dimensional hidden state vector at time t, $\mathbf{c} \in \mathbb{R}^n$ is a vector generated from the whole sequence of input states, and f and q are non-linear functions.[3] A decoder is then trained to predict the next output symbol y_t given the encoded input vector $\mathbf{c}$ and all the previously predicted output symbols $\langle y_1, ..., y_{t-1} \rangle$. More formally, the decoder defines a probability over the output sequence $\vec{y} = \langle y_1, ..., y_{T'} \rangle$ by decomposing the joint probability into ordered conditionals:

$$p(\vec{y} \,|\, \vec{x}) = \prod_{t=1}^{T'} p(y_t \,|\, \mathbf{c}, \langle y_1, ..., y_{t-1} \rangle) \tag{3}$$

Typically a decoder RNN defines the hidden state at time t as $\mathbf{s}_t = g(\mathbf{s}_{t-1}, \{\mathbf{c}, y_{t-1}\})$ and then feeds $\mathbf{s}_t$ to a softmax layer in order to obtain a conditional probability over output symbols.

[3] For instance, Sutskever et al. (2014) used an LSTM as f, and $q(\{\mathbf{h}_1, ..., \mathbf{h}_T\}) = \mathbf{h}_T$.

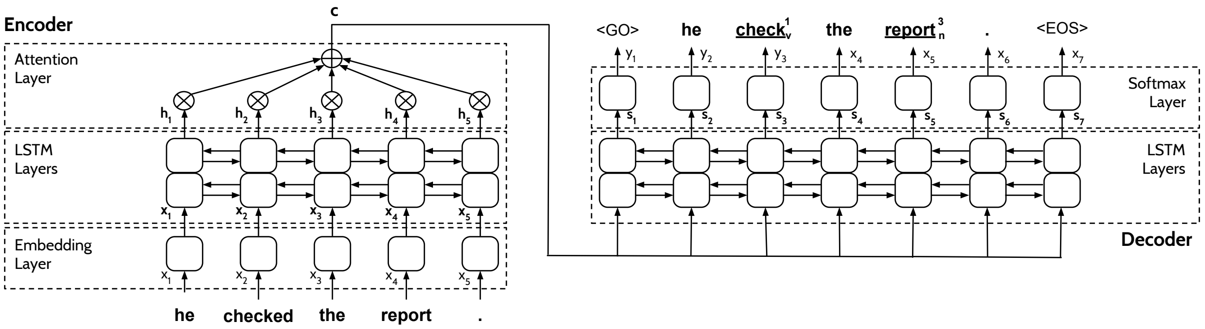
Encoder
Attention Layer
LSTM Layers
Embedding Layer
c
h_1 h_2 h_3 h_4 h_5
x_1 x_2 x_3 x_4 x_5
he checked the report .
<GO> he check^1_v the report^3_n . <EOS>
y_1 y_2 y_3 x_4 x_5 x_6 x_7
s_1 s_2 s_3 s_4 s_5 s_6 s_7
Softmax Layer
LSTM Layers
Decoder

Figure 3: Encoder-decoder architecture for sequence-to-sequence WSD, with 2 bidirectional LSTM layers and an attention layer.

In the context of WSD framed as a sequence learning problem, a sequence-to-sequence model takes as input a training set of labeled sequences (cf. Section 3.1) and learns to replicate an input sequence $\vec{x}$ while replacing each content word with its most suitable word sense from S. In other words, sequence-to-sequence WSD can be viewed as the combination of two sub-tasks:

- A *memorization* task, where the model learns to replicate the input sequence token by token at decoding time;
- The actual *disambiguation* task where the model learns to replace content words across the input sequence with their most suitable senses from the sense inventory S.

In the latter stage, multi-word expressions (such as nominal entity mentions or phrasal verbs) are replaced by their sense identifiers, hence yielding an output sequence that might have a different length than $\vec{x}$.

Architecture. The encoder-decoder architecture generalizes over both the models in Sections 3.1 and 3.2. In particular, we include one or more bidirectional LSTM layers at the core of both the encoder and the decoder modules. The encoder utilizes an embedding layer (cf. Section 3.1) to convert input symbols into embedded representations, feeds it to the bidirectional LSTM layer, and then constructs the context vector $\mathbf{c}$, either by simply letting $\mathbf{c} = \mathbf{h}_T$ (i.e., the hidden state of the bidirectional LSTM layer after reading the whole input sequence), or by computing the weighted sum described in Section 3.2 (if an attention mechanism is employed). In either case, the context vector $\mathbf{c}$ is passed over to the decoder, which generates the output symbols sequentially based on $\mathbf{c}$ and the current hidden state $\mathbf{s}_t$, using one or more bidirectional LSTM layers as in the encoder module. Instead of feeding $\mathbf{c}$ to the decoder only at the first time step (Sutskever et al., 2014; Vinyals and Le, 2015), we condition each output symbol y_t on $\mathbf{c}$, allowing the decoder to peek into the input at every step, as in Cho et al. (2014). Finally, a fully-connected layer with softmax activation converts the current output vector of the last LSTM layer into a probability distribution over the output vocabulary O. The complete encoder-decoder architecture (including the attention mechanism) is shown in Figure 3.

4 Multitask Learning with Multiple Auxiliary Losses

Several recent contributions (Søgaard and Goldberg, 2016; Bjerva et al., 2016; Plank et al., 2016; Luong et al., 2016) have shown the effectiveness of *multitask learning* (Caruana, 1997, MTL) in a sequence learning scenario. In MTL the idea is that of improving generalization performance by leveraging training signals contained in related tasks, in order to exploit their commonalities and differences. MTL is typically carried out by training a single architecture using multiple loss functions and a shared representation, with the underlying intention of improving a main task by incorporating joint learning of one or more related auxiliary tasks. From a practical point of view, MTL works by including one task-specific output layer per additional task, usually at the outermost level of the architecture, while keeping the remaining hidden layers common across all tasks.

In line with previous approaches, and guided by the intuition that WSD is strongly linked to other NLP tasks at various levels, we also design and study experimentally a multitask augmentation of the models described in Section 3. In particular, we consider two auxiliary tasks:

- **Part-of-speech (POS) tagging**, a standard auxiliary task extensively studied in previous work (Søgaard and Goldberg, 2016; Plank et al., 2016). Predicting the part-of-speech tag for a given token can also be informative for word senses, and help in dealing with cross-POS lexical ambiguities (e.g., *book a flight* vs. *reading a good book*);
- **Coarse-grained semantic labels (LEX)** based on the WordNet (Miller et al., 1990) lexicographer files,[4] i.e., 45 coarse-grained semantic categories manually associated with all the synsets in WordNet on the basis of both syntactic and logical groupings (e.g., *noun.location*, or *verb.motion*). These very coarse semantic labels, recently employed in a multitask setting by Alonso and Plank (2017), group together related senses and help the model to generalize, especially over senses less covered at training time.

We follow previous work (Plank et al., 2016; Alonso and Plank, 2017) and define an auxiliary loss function for each additional task. The overall loss is then computed by summing the main loss (i.e., the one associated with word sense labels) and all the auxiliary losses taken into account.

As regards the architecture, we consider both the models described in Sections 3.2 and 3.3 and modify them by adding two softmax layers in addition to the one in the original architecture. Figure 4 illustrates this for the attentive tagger of Section 3.2, considering both POS and LEX as auxiliary tasks. At the j^{th} time step the model predicts a sense label y_j together with a part-of-speech tag POS_j and a coarse semantic label LEX_j.[5]

5 Experimental Setup

In this section we detail the setup of our experimental evaluation. We first describe the training corpus and all the standard benchmarks for all-words WSD; we then report technical details on the architecture and on the training process for all the models described throughout Section 3 and their multitask augmentations (Section 4).

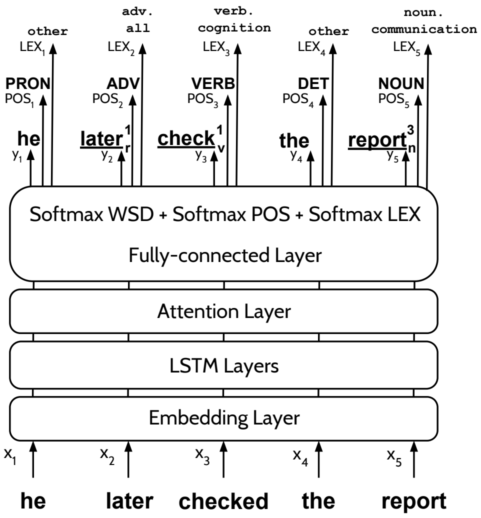

Figure 4: Multitask augmentation (with both POS and LEX as auxiliary tasks) for the attentive bidirectional LSTM tagger of Section 3.2.

Evaluation Benchmarks. We evaluated our models on the English all-words WSD task, considering both the fine-grained and coarse-grained benchmarks (Section 6.1). As regards fine-grained WSD, we relied on the evaluation framework of Raganato et al. (2017), which includes five standardized test sets from the Senseval/SemEval series: Senseval-2 (Edmonds and Cotton, 2001, **SE2**), Senseval-3 (Snyder and Palmer, 2004, **SE3**), SemEval-2007 (Pradhan et al., 2007, **SE07**), SemEval-2013 (Navigli et al., 2013, **SE13**) and SemEval-2015 (Moro and Navigli, 2015, **SE15**). Due to the lack of a reasonably large development set for our setup, we considered the smallest among these test sets, i.e., **SE07**, as development set and excluded it from the evaluation of Section 6.1. As for coarse-grained WSD, we used the SemEval-2007 task 7 test set (Navigli et al., 2007), which is not included in the standardized framework, and mapped the original sense inventory from WordNet 2.1 to WordNet 3.0.[6] Finally, we carried out an experiment on multilingual WSD using the Italian, German, French and Spanish data of **SE13**. For these benchmarks we relied on BabelNet (Navigli and Ponzetto, 2012)[7] as unified sense inventory.

[4] `https://wordnet.princeton.edu/man/lexnames.5WN.html`

[5] We use a dummy LEX label (`other`) for punctuation and function words.

[6] We utilized the original sense-key mappings available at `http://wordnetcode.princeton.edu/3.0` for nouns and verbs, and the automatic mappings by Daudé et al. (2003) for the remaining parts of speech (not available in the original mappings).

[7] `http://babelnet.org`

	Dev	Test Datasets				Concatenation of All Test Datasets				
	SE07	SE2	SE3	SE13	SE15	Nouns	Verbs	Adj.	Adv.	All
BLSTM	61.8	71.4	68.8	65.6	69.2	70.2	56.3	75.2	**84.4**	68.9
BLSTM + att.	62.4	71.4	**70.2**	66.4	70.8	71.0	**58.4**	75.2	83.5	69.7
BLSTM + att. + LEX	63.7	**72.0**	69.4	66.4	**72.4**	**71.6**	57.1	**75.6**	83.2	**69.9**
BLSTM + att. + LEX + POS	**64.8**	**72.0**	69.1	**66.9**	71.5	71.5	57.5	75.0	83.8	**69.9**
Seq2Seq	60.9	68.5	67.9	65.3	67.0	68.7	54.5	74.0	81.2	67.3
Seq2Seq + att.	62.9	69.9	69.6	65.6	67.7	69.5	57.2	74.5	81.8	68.4
Seq2Seq + att. + LEX	64.6	70.6	67.8	66.5	68.7	70.4	55.7	73.3	82.9	68.5
Seq2Seq + att. + LEX + POS	63.1	70.1	68.5	66.5	69.2	70.1	55.2	75.1	84.4	68.6
IMS	61.3	70.9	69.3	65.3	69.5	70.5	55.8	75.6	82.9	68.9
IMS+emb	**62.6**	**72.2**	**70.4**	65.9	71.5	**71.9**	56.6	**75.9**	**84.7**	**70.1**
Context2Vec	61.3	71.8	69.1	65.6	**71.9**	71.2	**57.4**	75.2	82.7	69.6
Lesk$_{ext}$+emb	$\star$56.7	63.0	63.7	66.2	64.6	70.0	51.1	51.7	80.6	64.2
UKB$_{gloss}$ w2w	42.9	63.5	55.4	$\star$62.9	63.3	64.9	41.4	69.5	69.7	61.1
Babelfy	51.6	$\star$67.0	63.5	**66.4**	70.3	68.9	50.7	73.2	79.8	66.4
MFS	54.5	65.6	$\star$66.0	63.8	$\star$67.1	67.7	49.8	73.1	80.5	65.5

Table 1: F-scores (%) for English all-words fine-grained WSD on the test sets in the framework of Raganato et al. (2017) (including the development set **SE07**). The first system with a statistically significant difference from our best models is marked with $\star$ (unpaired t-test, $p < 0.05$).

At testing time, given a target word w, our models used the probability distribution over O, computed by the softmax layer at the corresponding time step, to rank the candidate senses of w; we then simply selected the top ranking candidate as output of the model.

Architecture Details. To set a level playing field with comparison systems on English all-words WSD, we followed Raganato et al. (2017) and, for all our models, we used a layer of word embeddings pre-trained[8] on the English ukWaC corpus (Baroni et al., 2009) as initialization, and kept them fixed during the training process. For all architectures we then employed 2 layers of bidirectional LSTM with 2048 hidden units (1024 units per direction).

As regards multilingual all-words WSD (Section 6.2), we experimented, instead, with two different configurations of the embedding layer: the pre-trained bilingual embeddings by Mrkšić et al. (2017) for all the language pairs of interest (EN-IT, EN-FR, EN-DE, and EN-ES), and the pre-trained multilingual 512-dimensional embeddings for 12 languages by Ammar et al. (2016).

Training. We used SemCor 3.0 (Miller et al., 1993) as training corpus for all our experiments. Widely known and utilized in the WSD literature, SemCor is one of the largest corpora annotated manually with word senses from the sense inventory of WordNet (Miller et al., 1990) for all open-class parts of speech. We used the standardized version of SemCor as provided in the evaluation framework[9] which also includes coarse-grained POS tags from the universal tagset. All models were trained for a fixed number of epochs $E = 40$ using Adadelta (Zeiler, 2012) with learning rate 1.0 and batch size 32. After each epoch we evaluated our models on the development set, and then compared the best iterations (E^*) on the development set with the reported state of the art in each benchmark.

6 Experimental Results

Throughout this section we identify the models based on the LSTM tagger (Sections 3.1-3.2) by the label **BLSTM**, and the sequence-to-sequence models (Section 3.3) by the label **Seq2Seq**.

6.1 English All-words WSD

Table 1 shows the performance of our models on the standardized benchmarks for all-words fine-grained WSD. We report the F1-score on each in-

[8] We followed Iacobacci et al. (2016) and used the Word2Vec (Mikolov et al., 2013) skip-gram model with 400 dimensions, 10 negative samples and a window size of 10.

[9] http://lcl.uniroma1.it/wsdeval

SemEval-2007 task 7			
BLSTM + att. + LEX	83.0	IMS	81.9
BLSTM + att. + LEX + POS	**83.1**	Chen et al. (2014)	82.6
Seq2Seq + att. + LEX	82.3	Yuan et al. (2016)	**82.8**
Seq2Seq + att. + LEX + POS	81.6	UKB w2w	80.1

Table 2: F-scores (%) for coarse-grained WSD.

	SemEval-2013 task 12			
	IT	FR	DE	ES
BLSTM (bilingual)	61.6	55.2	**69.2**	65.0
BLSTM (multilingual)	62.0	55.5	**69.2**	66.4
UMCC-DLSI	**65.8**	**60.5**	62.1	**71.0**
DAEBAK!	61.3	53.8	59.1	60.0
MFS	57.5	45.3	67.4	64.5

Table 3: F-scores (%) for multilingual WSD.

dividual test set, as well as the F1-score obtained on the concatenation of all four test sets, divided by part-of-speech tag.

We compared against the best supervised and knowledge-based systems evaluated on the same framework. As supervised systems, we considered **Context2Vec** (Melamud et al., 2016) and It Makes Sense (Zhong and Ng, 2010, **IMS**), both the original implementation and the best configuration reported by Iacobacci et al. (2016, **IMS+emb**), which also integrates word embeddings using exponential decay.[10] All these supervised systems were trained on the standardized version of SemCor. As knowledge-based systems we considered the embeddings-enhanced version of Lesk by Basile et al. (2014, **Lesk**$_{ext}$**+emb**), UKB (Agirre et al., 2014) (**UKB**$_{gloss}$ **w2w**) , and **Babelfy** (Moro et al., 2014). All these systems relied on the Most Frequent Sense (**MFS**) baseline as back-off strategy.[11] Overall, both **BLSTM** and **Seq2Seq** achieved results that are either state-of-the-art or statistically equivalent (unpaired t-test, $p < 0.05$) to the best supervised system in each benchmark, performing on par with word experts tuned over explicitly engineered features (Iacobacci et al., 2016). Interestingly enough, **BLSTM** models tended consistently to outperform their **Seq2Seq** counterparts, suggesting that an encoder-decoder architecture, despite being more powerful, might be suboptimal for WSD. Furthermore, introducing LEX (cf. Section 4) as auxiliary task was generally helpful; on the other hand, POS did not seem to help, corroborating previous findings (Alonso and Plank, 2017; Bingel and Søgaard, 2017).

The overall performance by part of speech was consistent with the above analysis, showing that our models outperformed all knowledge-based systems, while obtaining results that are superior or equivalent to the best supervised models. It is worth noting that RNN-based architectures outperformed classical supervised approaches (Zhong and Ng, 2010; Iacobacci et al., 2016) when dealing with verbs, which are shown to be highly ambiguous (Raganato et al., 2017).

The performance on coarse-grained WSD followed the same trend (Table 2). Both **BLSTM** and **Seq2Seq** outperformed UKB (Agirre et al., 2014) and IMS trained on SemCor (Taghipour and Ng, 2015a), as well as recent supervised approaches based on distributional semantics and neural architectures (Chen et al., 2014; Yuan et al., 2016).

6.2 Multilingual All-words WSD

All the neural architectures described in this paper can be readily adapted to work with different languages without adding sense-annotated data in the target language. In fact, as long as the first layer (cf. Figures 1-3) is equipped with *bilingual* or *multilingual* embeddings where word vectors in the training and target language are defined in the same space, the training process can be left unchanged, even if based only on English data. The underlying assumption is that words that are translations of each other (e.g., *house* in English and *casa* in Italian) are mapped to word embeddings that are as close as possible in the vector space.

In order to assess this, we considered one of our best models (**BLSTM+att.+LEX**) and replaced the monolingual embeddings with bilingual and multilingual embeddings (as specified in Section 5), leaving the rest of the architecture unchanged. We then trained these architectures on the same English training data, and ran the resulting models on the multilingual benchmarks of SemEval-2013 for Italian, French, German and Spanish. While doing this, we exploited BabelNet's inter-resource mappings to convert WordNet sense labels (used at training time) into BabelNet synsets compliant with the sense inventory of the task.

F-score figures (Table 3) show that bilingual and multilingual models, despite being trained only on English data, consistently outperformed the MFS

[10] We are not including Yuan et al. (2016), as their models are not available and not replicable on the standardized test sets, being based on proprietary data.

[11] Since each system always outputs an answer, F-score equals both precision and recall, and statistical significance can be expressed with respect to any of these measures.

baseline and achieved results that are competitive with the best participating systems in the task. We also note that the overall F-score performance did not change substantially (and slightly improved) when moving from bilingual to multilingual models, despite the increase in the number of target languages treated simultaneously.

6.3 Discussion and Error Analysis

All the neural models evaluated in Section 6.1 utilized the MFS back-off strategy for instances unseen at training time, which amounted to 9.4% overall for fine-grained WSD and 10.5% for coarse-grained WSD. Back-off strategy aside, 85% of the times the top candidate sense for a target instance lay within the 10 most probable entries in the probability distribution over O computed by the softmax layer.[12] In fact, our sequence models learned, on the one hand, to associate a target word with its candidate senses (something word experts are not required to learn, as they only deal with a single word type at a time); on the other, they tended to generate softmax distributions reflecting the semantics of the surronding context. For example, in the sentence:

(a) The two *justices* have been attending federalist society events for years,

our model correctly disambiguated *justices* with the WordNet sense $\texttt{justice}_n^3$ (public official) rather than $\texttt{justice}_n^1$ (the quality of being just), and the corresponding softmax distribution was heavily biased towards words and senses related to persons or groups (*commissioners*, *defendants*, *jury*, *cabinet*, *directors*). On the other hand, in the sentence:

(b) Xavi Hernandez, the player of Barcelona, has 106 *matches*,

the same model disambiguated *matches* with the wrong WordNet sense $\texttt{match}_n^1$ (tool for starting a fire). This suggests that the signal carried by discriminative words like *player* vanishes rather quickly. In order to enforce global coherence further, recent contributions have proposed more sophisticated models where recurrent architectures are combined with Conditional Random Fields (Huang et al., 2015; Ma and Hovy, 2016). Finally, a number of errors were connected to shorter sentences with limited context for disambiguation: in fact, we noted that the average precision of our model, without MFS back-off, increased by 6.2% (from 74.6% to 80.8%) on sentences with more than 20 word tokens.

7 Conclusion

In this paper we adopted a new perspective on supervised WSD, so far typically viewed as a classification problem at the word level, and framed it using neural sequence learning. To this aim we defined, analyzed and compared experimentally different end-to-end models of varying complexities, including augmentations based on an attention mechanism and multitask learning.

Unlike previous supervised approaches, where a dedicated model needs to be trained for every content word and each disambiguation target is treated in isolation, sequence learning approaches learn a single model in one pass from the training data, and then disambiguate jointly all target words within an input text. The resulting models consistently achieved state-of-the-art (or statistically equivalent) figures in all benchmarks for all-words WSD, both fine-grained and coarse-grained, effectively demonstrating that we can overcome the so far undisputed and long-standing word-expert assumption of supervised WSD, while retaining the accuracy of supervised word experts.

Furthermore, these models are sufficiently flexible to allow them, for the first time in WSD, to be readily adapted to languages different from the one used at training time, and still achieve competitive results (as shown in Section 6.2). This crucial feature could potentially pave the way for cross-lingual supervised WSD, and overcome the shortage of sense-annotated data in multiple languages that, to date, has prevented the development of supervised models for languages other than English.

As future work, we plan to extend our evaluation to larger sense-annotated corpora (Raganato et al., 2016) as well as to different sense inventories and different languages. We also plan to exploit the flexibility of our models by integrating them into downstream applications, such as Machine Translation and Information Extraction.

Acknowledgments

The authors gratefully acknowledge the support of the ERC Consolidator Grant MOUSSE No. 726487.

[12] We refer here to the same model considered in Section 6.2 (i.e., **BLSTM+att.+LEX**).

References

Eneko Agirre and Philip Edmonds. 2007. *Word sense disambiguation: Algorithms and applications*, volume 33. Springer Science & Business Media.

Eneko Agirre, Oier Lopez de Lacalle, and Aitor Soroa. 2014. Random Walks for Knowledge-Based Word Sense Disambiguation. *Comput. Linguist.*, 40(1):57–84.

Héctor Martínez Alonso and Barbara Plank. 2017. When is Multitask Learning Effective? Semantic Sequence Prediction under Varying Data Conditions. In *Proc. of ACL*, pages 44–53.

Waleed Ammar, George Mulcaire, Yulia Tsvetkov, Guillaume Lample, Chris Dyer, and Noah A. Smith. 2016. Massively Multilingual Word Embeddings. *CoRR*, abs/1602.01925.

Osman Başkaya and David Jurgens. 2016. Semi-supervised Learning with Induced Word Senses for State of the Art Word Sense Disambiguation. *J. Artif. Int. Res.*, 55(1):1025–1058.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *ICLR Workshop*.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2014. An Enhanced Lesk Word Sense Disambiguation Algorithm through a Distributional Semantic Model. In *Proc. of COLING*, pages 1591–1600.

Joachim Bingel and Anders Søgaard. 2017. Identifying Beneficial Task Relations for Multi-task Learning in Deep Neural Networks. In *Proc. of EACL*, pages 164–169.

Johannes Bjerva, Barbara Plank, and Johan Bos. 2016. Semantic Tagging with Deep Residual Networks. In *Proc. of COLING*, pages 3531–3541.

Andrei Butnaru, Radu Tudor Ionescu, and Florentina Hristea. 2017. ShotgunWSD: an Unsupervised Algorithm for Global Word Sense Disambiguation Inspired by DNA Sequencing. In *Proc. of EACL*, pages 916–926.

Marine Carpuat and Dekai Wu. 2007. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Proc. of EMNLP*, pages 61–72.

Rich Caruana. 1997. Multitask learning. *Machine Learning*, 28(1):41–75.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A Unified Model for Word Sense Representation and Disambiguation. In *Proc. of EMNLP*, pages 1025–1035.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proc. of EMNLP*, pages 1724–1734.

Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage Sense Disambiguation and Information Extraction with a Supersense Sequence Tagger. In *Proc. of EMNLP*, pages 594–602.

Jordi Daudé, Lluís Padró, and German Rigau. 2003. Validation and Tuning of WordNet Mapping Techniques. In *Proc. of RANLP*, pages 117–123.

Oier Lopez de Lacalle and Eneko Agirre. 2015. A Methodology for Word Sense Disambiguation at 90% based on large-scale Crowd Sourcing. In *Proc. of SEM*, pages 61–70.

Claudio Delli Bovi, Luca Telesca, and Roberto Navigli. 2015. Large-Scale Information Extraction from Textual Definitions through Deep Syntactic and Semantic Analysis. *TACL*, 3:529–543.

Philip Edmonds and Scott Cotton. 2001. Senseval-2: Overview. In *Proc. of SENSEVAL*, pages 1–5.

Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. 2016. Morphological Inflection Generation Using Character Sequence to Sequence Learning. In *Proc. of NAACL-HLT*, pages 634–643.

Lucie Flekova and Iryna Gurevych. 2016. Supersense embeddings: A unified model for supersense interpretation, prediction, and utilization. In *Proc. of ACL*, pages 2029–2041.

Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR*, abs/1308.0850.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. In *Proc. of ACL*, pages 1631–1640.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *CoRR*, abs/1508.01991.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. SensEmbed: Learning Sense Embeddings for Word and Relational Similarity. In *Proc. of ACL*, pages 95–105.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2016. Embeddings for Word Sense Disambiguation: An Evaluation Study. In *Proc. of ACL*, pages 897–907.

Mikael Kågebäck and Hans Salomonsson. 2016. Word Sense Disambiguation using a Bidirectional LSTM. In *Proceedings of CogALex*, pages 51–56.

Adam Kilgarriff. 2001. English Lexical Sample Task Description. In *Proc. of SENSEVAL*, pages 17–20.

Ryan Kiros, Yukun Zhu, Ruslan R. Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Proc. of NIPS*, pages 3294–3302.

Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. 2016. Multi-task sequence to sequence learning. In *ICLR Workshop*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proc. of ACL*, pages 1064–1074.

Diana McCarthy, Marianna Apidianaki, and Katrin Erk. 2016. Word Sense Clustering and Clusterability. *Comput. Linguist.*, 42(2):245–275.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In *Proc. of CoNLL*, pages 51–61.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. The Senseval-3 English Lexical Sample Task. In *Proc. of SENSEVAL*, pages 25–28.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR Workshop*.

George A. Miller, R.T. Beckwith, Christiane D. Fellbaum, D. Gross, and K. Miller. 1990. WordNet: an online lexical database. *International Journal of Lexicography*, 3(4):235–244.

George A. Miller, Claudia Leacock, Randee Tengi, and Ross T. Bunker. 1993. A semantic concordance. In *Proc. of HLT*, pages 303–308.

Andrea Moro and Roberto Navigli. 2015. SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking. In *Proc. of SemEval*, pages 288–297.

Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *TACL*, 2:231–244.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Roi Reichart, Ira Leviant, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic Specialisation of Distributional Word Vector Spaces using Monolingual and Cross-Lingual Constraints. *TACL*, 5.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):10.

Roberto Navigli, David Jurgens, and Daniele Vannella. 2013. SemEval-2013 Task 12: Multilingual Word Sense Disambiguation. In *Proc. of SemEval*, volume 2, pages 222–231.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. 2007. SemEval-2007 Task 07: Coarse-grained English All-words Task. In *Proc. of SemEval*, pages 30–35.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The Automatic Construction, Evaluation and Application of a Wide-Coverage Multilingual Semantic Network. *Artificial Intelligence*, 193:217–250.

Steven Neale, Lus Gomes, Eneko Agirre, Oier Lopez de Lacalle, and Antnio Branco. 2016. Word Sense-Aware Machine Translation: Including Senses as Contextual Features for Improved Translation Models. In *Proc. of LREC*, pages 2777–2783.

Tommaso Pasini and Roberto Navigli. 2017. Train-O-Matic: Large-Scale Supervised Word Sense Disambiguation in Multiple Languages without Manual Training Data. In *Proc. of EMNLP*.

Mohammad Taher Pilehvar and Roberto Navigli. 2014. A Large-scale Pseudoword-based Evaluation Framework for State-of-the-art Word Sense Disambiguation. *Comput. Linguist.*, 40(4):837–881.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In *Proc. of ACL*, pages 412–418.

Sameer S. Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 Task 17: English Lexical Sample, SRL and All Words. In *Proc. of SemEval-2007*, pages 87–92.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2016. Automatic Construction and Evaluation of a Large Semantically Enriched Wikipedia. In *Proc. of IJCAI*, pages 2894–2900.

Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. In *Proc. of EACL*, pages 99–110.

Sascha Rothe and Hinrich Schütze. 2015. AutoExtend: Extending Word Embeddings to Embeddings for Synsets and Lexemes. In *Proc. of ACL*, pages 1793–1803.

Hui Shen, Razvan Bunescu, and Rada Mihalcea. 2013. Coarse to Fine Grained Sense Disambiguation in Wikipedia. In *Proc. of SEM*, pages 22–31.

Benjamin Snyder and Martha Palmer. 2004. The english all-words task. In *Proc. of Senseval-3*, pages 41–43.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proc. of ACL*, pages 231–235.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proc. of NIPS*, pages 3104–3112.

Kaveh Taghipour and Hwee Tou Ng. 2015a. One Million Sense-Tagged Instances for Word Sense Disambiguation and Induction. In *Proc. of CoNLL*, pages 338–344.

Kaveh Taghipour and Hwee Tou Ng. 2015b. Semi-Supervised Word Sense Disambiguation Using Word Embeddings in General and Specific Domains. In *Proc. of NAACL-HLT*, pages 314–323.

Rocco Tripodi and Marcello Pelillo. 2017. A Game-Theoretic Approach to Word Sense Disambiguation. *Comput. Linguist.*, 43(1):31–70.

Oriol Vinyals, Ł ukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Proc. of NIPS*, pages 2773–2781.

Oriol Vinyals and Quoc V. Le. 2015. A Neural Conversational Model. In *Proc. of ICML, JMLR: W&CP*, volume 37.

Wenhui Wang and Baobao Chang. 2016. Graph-based Dependency Parsing with Bidirectional LSTM. In *Proc. of ACL*, pages 2306–2315.

Dirk Weissenborn, Leonhard Hennig, Feiyu Xu, and Hans Uszkoreit. 2015. Multi-Objective Optimization for the Joint Disambiguation of Nouns and Named Entities. In *Proc. of ACL*, pages 596–605.

Deyi Xiong and Min Zhang. 2014. A Sense-Based Translation Model for Statistical Machine Translation. In *Proc. of ACL*, pages 1459–1469.

Dayu Yuan, Ryan Doherty, Julian Richardson, Colin Evans, and Eric Altendorf. 2016. Semi-supervised Word Sense Disambiguation with Neural Models. In *Proc. of COLING*, pages 1374–1385.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Zhi Zhong and Hwee Tou Ng. 2010. It Makes Sense: A Wide-Coverage Word Sense Disambiguation System for Free Text. In *Proc. of ACL: System Demonstrations*, pages 78–83.

Zhi Zhong and Hwee Tou Ng. 2012. Word Sense Disambiguation Improves Information Retrieval. In *Proc. of ACL*, pages 273–282.

Jie Zhou and Wei Xu. 2015. End-to-End Learning of Semantic Role Labeling using Recurrent Neural Networks. In *Proc. of ACL*, pages 1127–1137.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification. In *Proc. of ACL*, pages 207–212.