A Sequence-to-Sequence Model for Semantic Role Labeling

Angel Daza and Anette Frank
Leibniz ScienceCampus "Empirical Linguistics and Computational Language Modeling"
Department of Computational Linguistics
Heidelberg University
69120 Heidelberg, Germany
{daza,frank}@cl.uni-heidelberg.de

Abstract

We explore a novel approach for Semantic Role Labeling (SRL) by casting it as a sequence-to-sequence process. We employ an attention-based model enriched with a copying mechanism to ensure faithful regeneration of the input sequence, while enabling interleaved generation of argument role labels. Here, we apply this model in a monolingual setting, performing PropBank SRL on English language data. The constrained sequence generation set-up enforced with the copying mechanism allows us to analyze the performance and special properties of the model on manually labeled data and benchmarking against state-of-the-art sequence labeling models. We show that our model is able to solve the SRL argument labeling task on English data, yet further structural decoding constraints will need to be added to make the model truly competitive. Our work represents a first step towards more advanced, generative SRL labeling setups.

1 Introduction

Semantic Role Labeling (SRL) is the task of assigning semantic argument structure to constituents or phrases in a sentence, to answer the question: *Who* did *what* to *whom*, *where* and *when*? This task is normally accomplished in two steps: first, identifying the predicate and second, labeling its arguments and the roles that they play with respect to the predicate. SRL has been formalized in different frameworks, the most prominent being FrameNet (Baker et al., 1998) and PropBank (Palmer et al., 2005). In this work we focus on *argument identification and labeling* using the PropBank (PB) annotation scheme.

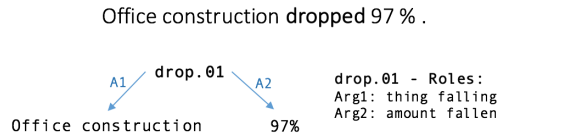

Figure 1: An input sentence (top), its PropBank predicate-argument structure (middle) and its linearized labeled sequence produced by our system.

Recent end-to-end neural models considerably improved the state-of-the-art results for SRL in English (He et al., 2017; Marcheggiani and Titov, 2017). In general, such models treat the problem as a supervised sequence labeling task, using deep LSTM architectures that assign a label to each token within the sentence.

SRL training resources for other languages are more restricted in size and thus, models suffer from sparseness problems because specific predicate-role instances occur only a handful of times in the training set. Since annotating SRL data in larger amounts is expensive, the use of a generative neural network model could be beneficial for automatically obtaining more labeled data in low-resource settings. The model that we present in this paper is a first step towards a joint label and language generation formulation for SRL, using the sequence-to-sequence architecture as a starting point.

We explore a sequence-to-sequence formulation of SRL that we apply, as a first step, in a classical monolingual setting on PropBank data, as illustrated in Figure 1. This constrained monolingual setting will allow us to analyze the suitablility of a sequence-to-sequence architecture for SRL, by benchmarking the system performance against existing sequence labeling models for SRL on well known labeled evaluation data.

Proceedings of the 3rd Workshop on Representation Learning for NLP, pages 207–216
Melbourne, Australia, July 20, 2018.

Sequence-to-sequence (seq2seq) models were pioneered by Sutskever et al. (2014), and later enhanced with an attention mechanism (Bahdanau et al., 2014; Luong et al., 2015). They have been successfully applied in many related structure prediction tasks such as syntactic parsing (Vinyals et al., 2015), parsing into Abstract Meaning Representation (Konstas et al., 2017), semantic parsing (Dong and Lapata, 2016), and cross-lingual Open Information Extraction (Zhang et al., 2017).

When applying a seq2seq model with attention in a monolingual SRL labeling setup, we need to restrict the decoder to reproduce the original input sentence, while in addition inserting PropBank labels into the target sequence in the decoding process (see Figure 1). To achieve this, we encode each input sentence into a suitable representation that will be used by the decoder to regenerate word tokens as given in the source sentence and introducing SRL labels in appropriate positions to label argument spans with semantic roles. In order to avoid lexical deviations in the output string, we add a copying mechanism (Gu et al., 2016) to the model. This technique was originally proposed to deal with rare words by copying them directly from the source when appropriate. We apply this mechanism in a novel way, with the aim of guiding the decoder to reproduce the input as closely as possible, while otherwise giving it the option of generating role labels in appropriate positions in the target sequence.

Our main contributions in this work are:

(i) We propose a novel neural architecture for SRL using a seq2seq model enhanced with attention and copying mechanisms.

(ii) We evaluate this model in a monolingual setting, performing PropBank-style SRL on standard English datasets, to assess the suitability of this model type for the SRL labeling task.

(iii) We compare the performance of our model to state-of-the-art sequence labeling models, including detailed (also comparative) error analysis.

(iv) We show that the seq2seq model is suited for the task, but still lags behind sequence labeling systems that include higher-level constraints.

2 Model

We propose an extension to the Sequence-to-Sequence model of (Bahdanau et al., 2014) to perform SRL.[1] The model will learn to map an unlabeled source sequence of words $(x_1...x_{T_x})$ into a target sequence $(y_1...y_{T_y})$ consisting of word tokens and SRL label tokens (see Figure 2). The source sentence, represented as a sequence of dense word vectors, is fed to an LSTM encoder to produce a series of hidden states that represent the input. This information is used by the decoder to recursively generate tokens step-by-step, conditioned on the previous generated tokens and the source by attending the encoder's hidden states as proposed in Bahdanau et al. (2014). On top of this architecture, we add the copying mechanism (Gu et al., 2016), which helps the model to avoid lexical deviations in the output while still having the freedom of generating words and SRL labels based on the context. The attention-based generation and copying mechanism will be competing with each other so that the model learns when to copy directly from the source and when to generate the next token.

In our current setup we restrict role labeling to a single predicate per sentence. If a sentence has more than one predicate, we create a separate copy for each predicate; the same setting was applied in Zhou and Xu (2015). In each sentence copy the predicate whose roles are to be labeled is preceded by a special token <*PRED*> that marks the position of the predicate under consideration. This helps the decoder to focus on generating argument labels for that specific predicate (see Table 1.)

2.1 Vocabulary

We assume a unique vocabulary for both encoder and decoder that comprises the words occurring during training, the out-of-vocabulary token, and the special symbol used to mark the position of the predicate, thus $\mathcal{V} = \{v_1, ..., v_N\} \cup \{UNK, < PRED >\}$. In addition, we employ a set $\mathcal{L} = \{l_1, ..., l_M\}$ with all the possible labeled brackets and a set $\mathcal{X} = \{x_1..., x_{T_x}\}$, a per-instance set containing the T_x words from the current source sequence. Thus, our total vocabulary is defined for each instance as $\mathcal{V} \cup \mathcal{L} \cup \mathcal{X}$.

The label set $\mathcal{L}$ contains one common opening bracket *(#* for all argument types to indicate the beginning of an argument span, and several label-specific closing brackets, such as *P0:A1)*, which indicates in this case that the span for argument *A1* is ending (see also Table 1).

[1] In this work we restrict ourselves to argument labeling.

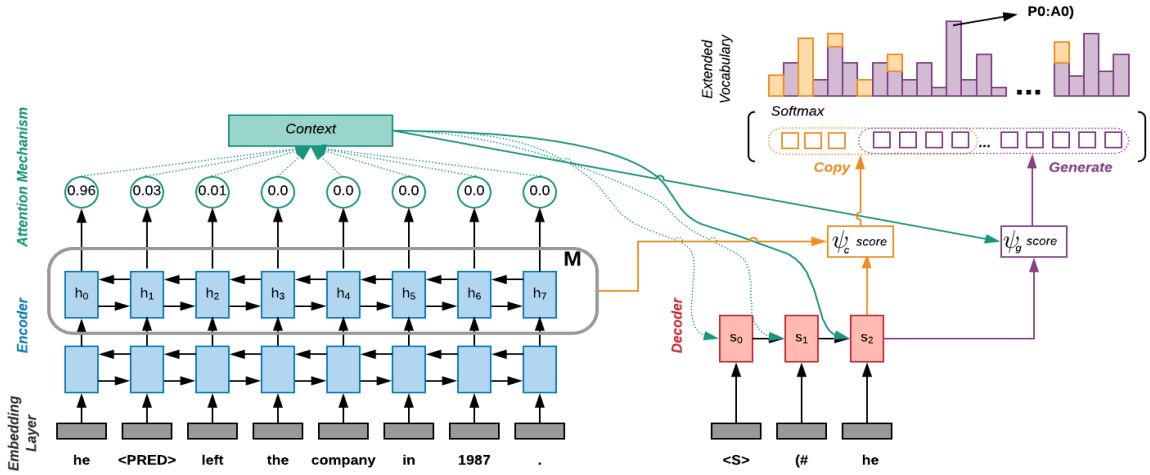

Figure 2: A sequence-to-sequence model for SRL. A score for copying and a score for generating tokens is computed at each time step and a joint softmax determines the probability of the next token over the extended vocabulary of words $\mathcal{V}$, labels $\mathcal{L}$ and current instance words $\mathcal{X}$.

2.2 Encoder

We use a two-layer bi-RNN encoder with LSTM cells (Hochreiter and Schmidhuber, 1997) that outputs a series of hidden states $h_j = \left[\overrightarrow{h_j}; \overleftarrow{h_j}\right]$ where each h_j contains information about the surrounding context of the word x_j. We refer to the complete matrix of encoder hidden states as $\mathbf{M}$, since it acts as a memory that the decoder can use to copy words directly from the source.

2.3 Attention Mechanism

We use the global dot product attention from Luong et al. (2015) to compute the context vector c_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \quad ; \quad \alpha_{ij} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T_x} \exp(e_{i,k})} \qquad (1)$$

where $e_{i,j}$ is the dot product function between decoder state s_{i-1} and each encoder hidden state h_j.

2.4 Decoder

The role of the decoder (a single-layer recurrent unidirectional LSTM) is to emit an output token y_t from a learned distribution over the vocabulary at each time step t given its state s_t, the previous output token y_{t-1}, the attention context vector c_t, and the memory $\mathbf{M}$. To get this distribution it is necessary to compute two separate modes: one for generating and one for copying.

To obtain the probability of generating y_t we use the context vector produced by the attention to learn a score ψ_g for each possible token v_i of being the next generated token. We define ψ_g as:

$$\psi_g(y_t = v_i) = W_o[s_t; c_t], \quad v_i \epsilon \mathcal{V} \cup \mathcal{L} \qquad (2)$$

where $W_o \epsilon \mathbb{R}^{N \times 2d_s}$ is a learnable parameter and s_t, c_t are the current decoder state and context vector respectively. This means that the model computes a generation score for both words and labels, based on what it is attending on at the current step.

For the probability of copying y_t we compute the score ψ_c of copying a token directly from the source as:

$$\psi_c(y_t = x_j) = \sigma(h_j^T W_c) s_t, \quad x_j \epsilon \mathcal{X} \qquad (3)$$

where $W_c \epsilon \mathbb{R}^{d_h \times d_s}$ is a learnable parameter, h_j is the encoder hidden state representing x_j, s_t is the current decoder state, and σ is a non-linear transformation; we used `tanh` for our experiments.

Using the two scoring methods, the decoder will have two competing modes: the generation mode, used to generate the most probable subsequent token based on attention; and the copying, used to choose the next token directly from the encoder memory $\mathbf{M}$, which holds both positional and content information of the source. A final mixed distribution is calculated by adding the probability of generating y_t and the probability of copying y_t:

$$p(y_t|s_t, y_{t-1}, c_t, \mathbf{M}) = p(y_t, \mathbf{g}|s_t, y_{t-1}, c_t) + p(y_t, \mathbf{c}|s_t, y_{t-1}, \mathbf{M}) \qquad (4)$$

We use a *softmax* layer to convert the two scores into a joint distribution that represents the mixed

Source-1:	The trade figures <*PRED*> **turn out** well , and all those recently unloaded bonds spurt in price .
Target-1:	*(#* The trade figures *P0:A1)* ***(# turn out P0:V)*** *(#* well *P0:A2)* , and all those recently unloaded bonds spurt in price .
Source-2:	The trade figures turn out well , and all those recently <*PRED*> **unloaded** bonds spurt in price .
Target-2:	The trade figures turn out well , and all those *(#* recently *P0:AM-TMP)* ***(# unloaded P0:V)*** *(#* bonds *P0:A1)* spurt in price .
Source-3:	The trade figures turn out well , and all those recently unloaded bonds <*PRED*> **spurt** in price .
Target-3:	The trade figures turn out well , and *(#* all those recently unloaded bonds *P0:A1)* ***(# spurt P0:V)*** *(#* in price *P0:AM-ADV)* .

Table 1: A single sentence with three labeled predicates is converted into three different source-target pairs. The symbol <*PRED*> in each source marks the predicate for which the model is expected to generate a correct predicate-argument structure.

likelihood of generating and copying y_t. Again following Gu et al. (2016), we define this as:

$$p(y_t, \mathbf{g}|\cdot) = \begin{cases} \frac{1}{Z} e^{\psi_g(y_t)} & y_t \epsilon \mathcal{V} \cup \mathcal{L} \\ 0 & otherwise \end{cases}$$
$$p(y_t, \mathbf{c}|\cdot) = \begin{cases} \frac{1}{Z} \sum_{j:x_j=y_t} e^{\psi_c(x_j)} & y_t \epsilon \mathcal{X} \\ 0 & otherwise \end{cases} \quad (5)$$

where Z is the normalization term shared by the two modes, $Z = \sum_{v \epsilon \mathcal{V}} e^{\psi_g(v)} + \sum_{x \epsilon \mathcal{X}} e^{\psi_c(x)}$. Since a single *softmax* is applied over the copying and generating modes, the network learns by itself when it is proper to copy a word from the source and when it needs to generate a label.

During training, the objective is to minimize the negative log-likelihood of the target token y_t for each time-step for both generate mode (given previous generated tokens) and copy mode (given source sequence X). We calculate the loss for the whole sequence as:

$$loss = -\frac{1}{T_y} \sum_{t=0}^{T_y} \log P(y_t | y_{<t}, X) \quad (6)$$

3 Experimental Setup

3.1 Datasets and Evaluation Measures

We test the performance of our system on the span-based SRL datasets CoNLL-05[2] and CoNLL-12.[3] These datasets provide the gold predicate as part of the input. Since we focus on argument identification and classification, we provide this information in the input to the system. We use the standard training, development and test splits and use the official CoNLL-05 evaluation script on both datasets. We compare our results with Collobert et al. (2011); FitzGerald et al. (2015); Zhou and Xu (2015) and He et al. (2017) who use the same datasets and evaluation script. We show results separately for the Brown and WSJ portion of the CoNLL-05 test dataset.

The CoNLL-05 Shared Task[4] evaluation script computes precision, recall and F1 measure (the harmonic mean of precision and recall) for the predicted arguments. The script expects prediction-gold pairs that have the same number of words in order to consider them comparable, and only if this is the case, it computes a score. Furthermore, an argument is only considered correct if the words spanning the argument as well as its role label match with gold (Carreras and Màrquez, 2005). This means that it is essential to predict perfect argument spans besides the correct role label.

3.2 Pre-processing

For our seq2seq model we need to provide sources and targets in a linearized manner. The sequences are sentences with zero or more predicates. Following Zhou and Xu (2015), if a sentence has n_p predicates we process the sentence n_p times, each one with its corresponding predicate-argument structure. As shown in Table 1, we linearize the target side by converting the CoNLL format into sequences of tokens that include brackets indicating the span of the argument and the argument label on the closing bracket. We inform the model about the predicate that it should focus on by adding the special token <*PRED*> to the source sequence immediately before the predicate word. This process is entirely reversible and thus we convert the system outputs back to CoNLL format and evaluate the results with the official script.

[2] http://www.lsi.upc.edu/~srlconll/home.html

[3] http://conll.cemantix.org/2012/data.html

[4] http://www.lsi.upc.edu/~srlconll/soft.html

	CoNLL-05			CoNLL-12	
	Dev	WSJ	Brown	Dev	Test
Seq2seq (attention-only)					
same length	29.19	29.98	32.24	-	-
brackets	95.25	94.93	94.24	-	-
Seq2seq (w/ Attention & Copying)					
same length	96.71	97.15	97.24	97.46	96.07
brackets	99.91	99.82	99.88	99.97	99.93

Table 2: Quality of reproducing words and SRL brackets with seq2seq: Attention-only vs. Attention & Copying, on CoNLL-05 and CoNLL-12 datasets: percentage of correctly reproduced sentence length and percentage of balanced brackets.

	CoNLL-05				CoNLL-12	
	dev	test	WSJ	Brown	dev	test
Collobert	72.29	74.15	-	-	-	-
FitzGerald	78.3	-	79.4	71.2	79.2	79.6
Zhou & Xu	79.55	81.27	82.84	69.41	81.07	81.27
He	81.6	81.6	83.1	72.1	81.5	81.7
Ours (min)	76.05	76.7	78.13	66.28	73.4	73.61
Ours (max)	77.29	77.87	79.23	68.39	75.05	75.43

Table 3: F1 measure for argument role labeling of our seq2seq model w/ Attention & Copying on CoNLL-05 and CoNLL-12 dev and test sets, compared to Collobert w/o parser, FitzGerald single model, Zhou & Xu, and He single model .

3.3 Training

Since we process as many copies of sentences as it has predicates, the final amount of sequences is approximately 94K for CoNLL-05 and 185K for CoNLL-12 training sets. We keep linearized sequences up to 100 tokens long and lowercase all tokens. Given this limit, we omit 30 (CoNLL-05) and 900 (CoNLL-12) sequences from training. We initialize the model with pre-trained 100-dimensional GloVe embeddings (Pennington et al., 2014) and update them during training.[5] All the tokens that are not covered by GloVe or that appear less frequently than a given threshold[6] in the training dataset are mapped to the UNK embedding. We keep track of this mapping to be able to post-process the sequence and recover the rare tokens. Our vocabulary size is set to $|\mathcal{V}| \approx 20K$ words for CoNLL-05 and $|\mathcal{V}| \approx 18K$ words for CoNLL-12.

We use Adam optimizer (Kingma and Ba, 2014), a learning rate $l_r = 0.001$ and gradient clipping at 5.0. Both encoder and decoder have hidden layer of 512 LSTMs. We use dropout (Srivastava et al., 2014) of 0.4 and train for 4 epochs with batch size of 6.

4 Evaluation and Results

Initially, we trained a model using attention only, and it learned to generate balanced brackets (every opening bracket has a corresponding closing bracket within the sequence) without further constraints. Yet, due to its generative nature, many target sequences diverged from the source in both length and token sequences. This was expected, because the system has to learn to generate not only the labels at the correct time-step but also to re-generate the complete sentence accurately. This is a disadvantage compared to the sequence labeling models where the words are already given.

By adding copying mechanism the model successfully regenerates the source sentence in the majority (up to 99%) of cases, as shown in Table 2. Such behavior also enables us to measure the performance of the model as an argument role classifier against the gold standard. Thus, we can benchmark its labeling performance against previous architectures built to solve the SRL task.

Table 3 displays the overall labeling performance of our copying-enhanced seq2seq model in comparison to previous neural sequence labeling architectures. For sequences that do not fully reproduce the input, we cannot compute appropriate scores against the gold standard. We compute two alternative scores for these cases: *oracle-min*, by setting the score for these sentences to 0.0 F1, and *oracle-max*, by setting their results to the scores we would obtain with perfect (= gold) labels. With these scores, we can better estimate the loss we are experiencing by non-perfectly reproduced sequences (see Table 2.)

As seen in Table 3, our model achieves an F1 score of 76.05 on the CoNLL-05 development set, and 73.4 on CoNLL-12 (min-oracle), and 77.29 and 75.05 (max-oracle), respectively. While these scores are still low compared to the latest neural SRL architectures, they are above the relatively simple model of Collobert et al. (2011). Note also

[5] We also experimented with word2vec word embeddings (Mikolov et al., 2013) but found GloVe6B (trained on Wikipedia2014+Gigaword5) embeddings to perform better. Available at https://nlp.stanford.edu/projects/glove/

[6] We used a threshold of 10 for CoNLL-05 and 15 for CoNLL-12.

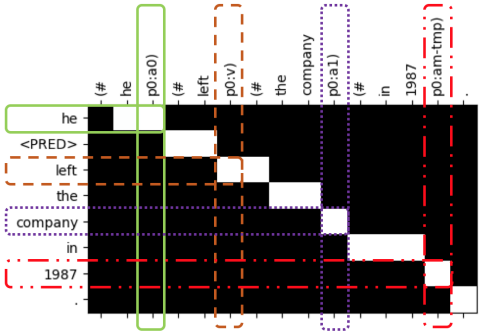

Figure 3: Example of the alignments learned by the attention mechanism.

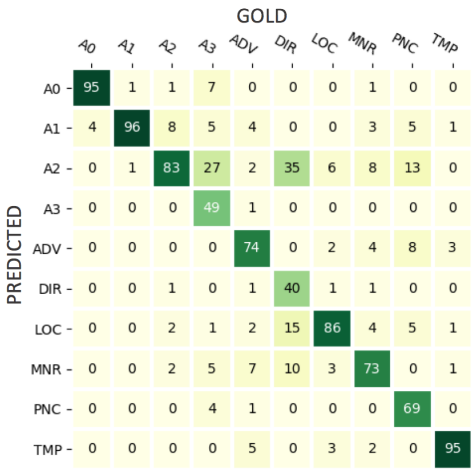

Figure 4: Confusion matrix showing percentage of predicted labels compared to the gold labels on the CoNLL-05 development set.

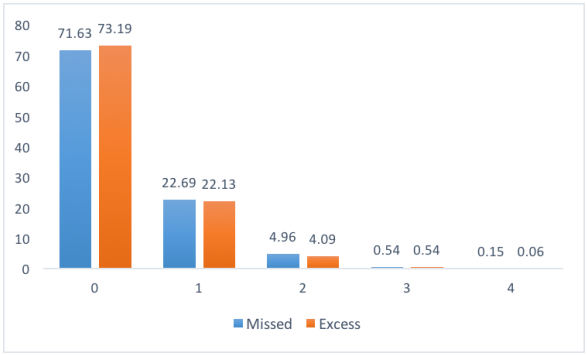

Figure 5: Percentage of sentences with 0,1,2 or more missing (blue) or excess (orange) arguments (seq2seq w/Copying, CoNLL-05, dev set).

that in contrast to the stronger models of FitzGerald et al. (2015); Zhou and Xu (2015) and He et al. (2017), our architecture is very lean and does not (yet) employ structured prediction (e.g. Conditional Random Field), to impose structural constraints on the label assignment. While this is certainly an extension we are going to explore in future work, here we will conduct deeper investigation to learn more about the kind of errors that our unconstrained seq2seq model makes. We report the analysis on CoNLL-05 development set.

4.1 Analysis

Argument Spans The model needs to generate labeled brackets at the appropriate time-step, in other words, the prediction of correct spans for arguments. To verify how well it is doing this, we measure how much overlap exists between the generated spans and the gold ones. This is equivalent to computing unlabeled argument assignment. We found that 77.5% of the spans match the gold spans completely, 21.2% of spans are partially overlapping with gold spans, and only 1.2% of the spans do not overlap at all with gold.

Argument Labels Recall from Section 2 that our model is labeling the sentences as in a translation task. It learns to use information from relevant words in the source sequence, aligning the labels to the argument words via learned attention weights as it is shown in Figure 3. This allows us to see where the model is looking when generating the labeled bracket. The confusion matrix in Figure 4 shows predicted vs. gold labels for all correctly assigned argument spans (i.e., the spans that match the gold boundaries). We observe that the model does very well for A0 and A1 gold roles, and that it causes only few misclassifications for A2. However, it frequently predicts core argument roles A0–A3 for non-argument roles, and also tends to mix predictions among non-core arguments. Since A0 and A1 roles are most frequent in the data, this indicates that the seq2seq model would benefit from more training data, particularly for less frequent roles, to better differentiate roles, and this is more prominent for the ones that are marked with prepositions.

Role co-occurrence and role set constraints Despite the absence of more refined decoding constraints, our model learns to avoid generating duplicated argument labels in most of the sequences. We find duplicated argument labels in less than 1% of the sequences. Figure 5 shows that the majority (about 70%) of sentences do not involve any missing or excess arguments; about 24/20% of sentences experience a single missing/excess role, and only 5/4% of the sentences experience a higher amount of missed/excess roles. Overall,

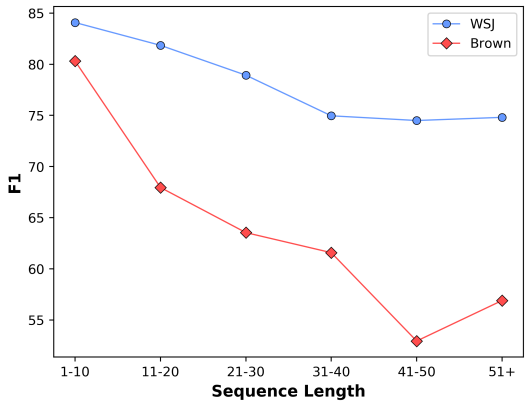

Figure 6: Performance of the model based on the number of tokens that the sequence has.

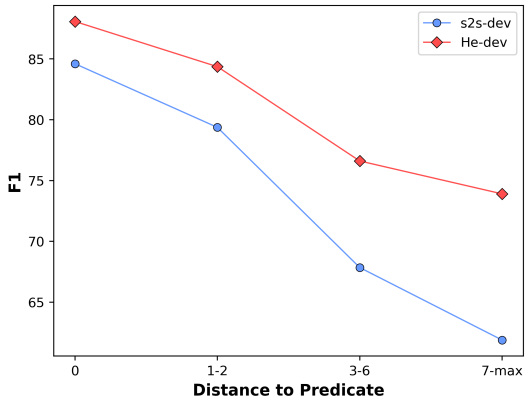

Figure 7: F1 score of arguments in buckets of increasing distances from their predicate, with distance normalized by sentence length (CoNLL-05, dev). We compare our model with He et al. (2017).

missed vs. excess arguments are balanced.

Sequence Length Another characteristic of the seq2seq model is that it encodes within a single sequence both words and labeled brackets. This increases the length of the sequences that need to be processed, and it is a well known problem that sequence length affects performance of recurrent neural models, even with the use of attention.

To measure the labeling performance difficulty with increasing sequence length, we partitioned the system outputs in six different bins containing groups of sentences of similar length (see Figure 6). As expected, the F1 score degrades proportionally to the length of the sequence, especially in sentences with more than 30 tokens.

Distance to predicate He et al. (2017) show

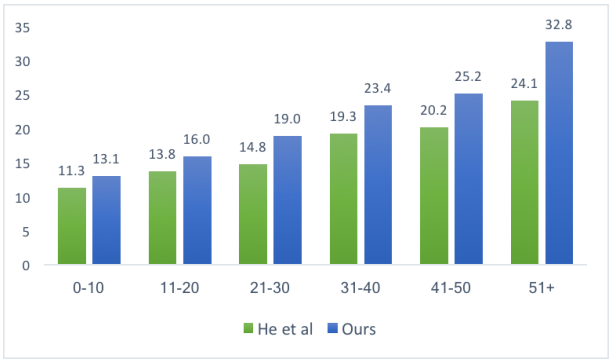

Figure 8: Error ratio of arguments in different regions of the sequences (CoNLL-05, dev).

that the surface distance between the argument and the predicate is also proportional to the amount of labeling errors. In our model, the distance between argument words and the predicate is even longer because of labeled brackets embedded in the sequence. Figure 7 displays the F1 score for different token distances between predicate and the respective argument. We see that the seq2seq model follows the same trend as the sequence labeling model, despite the fact that our model has access to the hidden states from the encoded input sentence; however, the real distance between predicate and argument in the decoder is also bigger.

Distance from sentence beginning. With each token that the model generates in decoding, the distance to the end position of the encoded sentence representation grows. While intuitively we would expect the model performance to degrade with larger distance to the input, it is also true that the model could be more prone to making mistakes at the beginning of the sequence, when the decoder has not yet generated enough context. To investigate this, we traced the ratio of errors that occur in several ranges of the sequence. We can see in Figure 8 that the first intuition was correct, the distance to the encoded representation is proportional to the mistakes that the model makes. We compare the error ratio to He et al. (2017) and show that the seq2seq system follows a similar trend but degrades faster with sequence length.

5 Related Work

Semantic Role Labeling. Traditional approaches to SRL relied on carefully designed features and expensive techniques to achieve global consistency such as Integer Linear Programming (Punyakanok et al., 2008) or dynamic programming

(Täckström et al., 2015). First neural SRL attempts tried to mix syntactic features with neural network representations. For example, FitzGerald et al. (2015) created argument and role representations using a feed-forward NN, and used a graphical model to enforce global constraints. Roth and Lapata (2016), on the other hand, proposed a neural classifier using dependency path embeddings to assign semantic labels to syntactic arguments.

Collobert et al. (2011) proposed the first SRL neural model that did not depend on hand-crafted features and treated the task as an IOB sequence labeling problem. Later, Zhou and Xu (2015) proposed a deep bi-directional LSTM model with a CRF layer on top. This model takes only the original text as input and assigns a label to each individual word in the sentence. He et al. (2017) also treat SRL as a IOB tagging problem, and use again a deep bi-LSTM incorporating highway connections, recurrent dropout and hard decoding constraints together with an ensemble of experts. This represents the best performing system on two span-based benchmark datasets so far (namely, CoNLL-05 and CoNLL-12). Marcheggiani et al. (2017) show that it is possible to construct a very accurate dependency-based SRL system without using any kind of explicit syntactic information. In subsequent work, Marcheggiani and Titov (2017) combine their LSTM model with a graph convolutional network to encode syntactic information at word level, which improves their LSTM classifier results on the dependency-based benchmark dataset (CoNLL-09).

Sequence-to-sequence models. Seq2seq models were first discovered as powerful models for Neural Machine Translation (Sutskever et al., 2014; Cho et al., 2014) but soon proved to be useful for any kind of problem that could be represented as a mapping between source and target sequences. Vinyals et al. (2015) demonstrate that constituent parsing can be formulated as a seq2seq problem by linearizing the parse tree. They obtain close to state-of-the-art results by using a large automatically parsed dataset. Dong and Lapata (2016) built a model for a related problem, semantic parsing, by mapping sentences to logical form. Seq2seq models have also been widely used for language generation (e.g. Karpathy and Li (2015); Chisholm et al. (2017)) given their ability to produce linguistic variation in the output sequences.

More closely related to SRL is the AMR parsing and generation system proposed by Konstas et al. (2017). This work successfully constructs a two-way mapping: generation of text given AMR representations as well as AMR parsing of natural language sentences. Finally, Zhang et al. (2017) went one step further by proposing a cross-lingual end-to-end system that learns to encode natural language (i.e. Chinese source sentences) and to decode them into sentences on the target side containing open semantic relations in English, using a parallel corpus for training.

6 Conclusions

In this paper we explore the properties of a Sequence-to-Sequence model for identifying and labeling PropBank roles. This is motivated by the fact that using a seq2seq model gives more flexibility for further tasks such as constrained generation and cross-lingual label projection. Another advantage is that our model is a very lean architecture compared to the deep Bi-LSTM of the recent SRL models.

To our knowledge, this is the first attempt to perform SRL using a seq2seq approach. Specific challenges emerged by formulating the problem in this way, such as: (i) the decoding of labels and words within a single sequence; (ii) generating balanced labeled brackets at the correct position; (iii) avoiding repetition of tokens, and especially, (iv) generating labeled sequences that perfectly match the source sentence in order to make the labeled sequence absolutely comparable.

Despite these difficulties, we could show that a sequence-to-sequence model with attention and copying achieves quite respectable labeling performance with a lean architecture and without yet considering structural constraints. For future work we consider extensions towards joint semantic role labeling and constrained generation, to produce new variations of existing labeled data.

Acknowledgements

We thank the reviewers for their insightful comments. We are also grateful to Éva Mújdricza-Maydt for assistance with the PropBank labeling scheme and CoNLL datasets. This research is funded by the Leibniz ScienceCampus Empirical Linguistics & Computational Language Modeling, supported by Leibniz Association grant no. SAS2015-IDS-LWC and by the Ministry of Science, Research, and Art of Baden-Wurttemberg.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate.

Collin F. Baker, Charles J. Fillmore, John B. Lowe, Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics*, volume 1, page 86, Morristown, NJ, USA. Association for Computational Linguistics.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, CONLL '05, pages 152–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Andrew Chisholm, Will Radford, and Ben Hachey. 2017. Learning to generate one-sentence biographies from wikidata. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 633–642. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Lon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43. Association for Computational Linguistics.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP '15)*.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Andrej Karpathy and Fei-Fei Li. 2015. Deep visual-semantic alignments for generating image descriptions. pages 3128–3137. IEEE Computer Society.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. pages 146–157.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420, Vancouver, Canada. Association for Computational Linguistics.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1506–1515, Copenhagen, Denmark. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Martha Palmer, Paul Kingsbury, and Daniel Gildea. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Comput. Linguist.*, 34(2):257–287.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. pages 3104–3112.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41.

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

Sheng Zhang, Kevin Duh, and Benjamin Van Durme. 2017. Mt/ie: Cross-lingual open information extraction with neural sequence-to-sequence models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 64–70. Association for Computational Linguistics.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1127–1137, Beijing, China. Association for Computational Linguistics.