Transition-based Semantic Role Labeling with Pointer Networks

Daniel Fernández-González

Universidade da Coruña, CITIC, FASTPARSE Lab, LyS Group, Depto. de Ciencias de la Computación y Tecnologías de la Información, Campus de Elviña, s/n, A Coruña, 15071, Spain

ARTICLE INFO

Keywords:
Natural language processing
Computational linguistics
Semantic role labeling
Neural network
Deep learning

ABSTRACT

Semantic role labeling (SRL) focuses on recognizing the predicate-argument structure of a sentence and plays a critical role in many natural language processing tasks such as machine translation and question answering. Practically all available methods do not perform full SRL, since they rely on pre-identified predicates, and most of them follow a pipeline strategy, using specific models for undertaking one or several SRL subtasks. In addition, previous approaches have a strong dependence on syntactic information to achieve state-of-the-art performance, despite being syntactic trees equally hard to produce. These simplifications and requirements make the majority of SRL systems impractical for real-world applications. In this article, we propose the first transition-based SRL approach that is capable of completely processing an input sentence in a single left-to-right pass, with neither leveraging syntactic information nor resorting to additional modules. Thanks to our implementation based on Pointer Networks, full SRL can be accurately and efficiently done in $O(n^2)$, achieving the best performance to date on the majority of languages from the CoNLL-2009 shared task.

1. Introduction

Semantic role labeling (SRL) has been successfully applied to a wide spectrum of natural language processing (NLP) applications such as machine translation (Shi, Liu, Ren, Feng, Li, Zhou, Sun and Wang, 2016; Wang, Zhao, Ploux, Lu and Utiyama, 2016; Marcheggiani, Bastings and Titov, 2018), information extraction (Bastianelli, Castellucci, Croce and Basili, 2013), question answering (Yih, Richardson, Meek, Chang and Suh, 2016; Zheng and Kordjamshidi, 2020; Xu, Tan, Song, Wu, Zhang, Song and Yu, 2020) and text comprehension (Zhang, Wu, Li and Zhao, 2019), inter alia. This fundamental NLP task can be seen as a shallow semantic parsing that aims to extract the “who did what to whom, how, where and when” from an input text by identifying predicate-argument relations. These semantic relations are usually represented by a set of labeled dependencies, where each one connects a predicate to either the entire phrasal argument (following the span-based SRL formalism) or just the argument’s syntactic head (following the dependency-based SRL annotation). While we can find recent studies that seek to improve performance on the former (He, Lee, Levy and Zettlemoyer, 2018a; Strubell, Verga, Andor, Weiss and McCallum, 2018; Zhang, Xia, Zhou, Jiang, Fu and Zhang, 2022), this research work focuses on dependency-based SRL, which was popularized by CoNLL-2008 and CoNLL-2009 shared tasks (Surdeanu, Johansson, Meyers, Márquez and Nivre, 2008; Hajic, Ciaramita, Johansson, Kawahara, Martí, Márquez, Meyers, Nivre, Padó, Štěpánková, Straka, Surdeanu, Xue and Zhang, 2009). An example of predicate-argument relations represented as a dependency-based SRL structure is depicted in Figure 1(a).

SRL is traditionally decomposed into four simpler subtasks: predicate identification (e.g., managers in Figure 1(a)), predicate sense disambiguation (manager.01 is the sense of predicate managers in the example), argument identification (e.g., fund) and argument role labeling (fund is argument A1 for predicate managers). While the four subtasks had to be completed in the CoNLL-2008 shared task, CoNLL-2009 corpora notably simplified SRL by providing pre-identified predicates beforehand. This simplification, coupled with the fact that the vast majority of approaches are only tested on the CoNLL-2009 benchmark, resulted in the common practice of not performing full SRL and exclusively focusing on the last three subtasks.

Furthermore, we can find in the literature as most SRL systems adopt either a pipeline framework (Roth and Lapata, 2016; Marcheggiani, Frolov and Titov, 2017; He, Li, Zhao and Bai, 2018b; Cai and Lapata, 2019b) or an end-to-end strategy (He, Li and Zhao, 2019; Li, Zhao, Wang and Parnow, 2020; Conia and Navigli, 2020). While the former...
approach resorts to different specific models to separately address one or two subtasks, the latter strategy employs a single model to accomplish the SRL task. However, it is worth mentioning that the vast majority of these end-to-end approaches do not perform predicate identification and, therefore, are not considered full SRL systems.

Moreover, most previous efforts mainly focused on syntax-aware SRL methods (He et al., 2018b, 2019; Cai and Lapata, 2019b; Kasai, Friedman, Frank, Radev and Rambow, 2019; Li, Zhao, He and Cai, 2021): i.e., they leverage syntactic information (also provided by the CoNLL-2009 corpora) to produce state-of-the-art accuracies. These approaches were motivated, among other reasons, by the fact that a significant portion of arcs from syntactic dependency trees matches predicate-argument relations in SRL structures. Nevertheless, it is important to note that dependency parsing is also an equally challenging and resource-consuming NLP task, and syntactic training data can be especially scarce in low-resource languages.

The fact that practically all approaches heavily rely on information that is not always available (such as gold predicates and syntactic dependency trees) makes them impractical for real-life downstream applications that require full SRL. In addition, the use of a single model in end-to-end architectures not only mitigates the error-propagation problem in pipeline strategies, but also notably simplifies the decoding process.

To alleviate these inconveniences, Cai, He, Li and Zhao (2018) introduced the first end-to-end syntax-agnostic approach for accurately performing full SRL. They handle the four subtasks by a single graph-based model, following a two-stage decoding procedure: they first identify and classify all predicates and then search for arguments and semantic roles for each of these predicates. The latter is implemented by independently scoring all possible predicate-argument dependencies and then exhaustively searching for a high-scoring graph by combining these scores. This work was recently improved by Zhou, Xia, Li, Zhang, Hong and Zhang (2022). They proposed a syntax-agnostic method that jointly identifies and classifies predicates and arguments in a single stage. In addition, their approach leverages high-order information, scoring sets of predicate-argument dependencies and computing the high-scoring graph in cubic time. The resulting graph-based model achieves a remarkable performance in full end-to-end SRL.

Unlike graph-based methods, transition-based algorithms were barely proposed for SRL modeling. These generate a sequence of actions (transitions) to incrementally build predicate-argument dependencies (usually from left to right). This is typically done by local, greedy prediction and can efficiently process a sentence in a linear or quadratic number of actions. Although they provide higher efficiency than graph-based models and were successfully employed in other parsing tasks (Ma, Hu, Liu, Peng, Neubig and Hovy, 2018; Fernández-González and Gómez-Rodríguez, 2019, 2020; Fernández-González and Gómez-Rodríguez, 2022), only two transition-based SRL systems were presented (Choi and Palmer, 2011; Fei, Zhang, Li and Ji, 2021), neither of them performing full syntax-agnostic SRL.

In this article, we propose the first transition-based approach for full syntax-agnostic SRL.\(^1\) Our model does not rely on any kind of syntactic information, even discarding part-of-speech (PoS) tags,\(^2\) and incrementally produces

---

\(^1\)Source code available at [https://github.com/danifg/SRLPointer](https://github.com/danifg/SRLPointer).

\(^2\)PoS tags are considered lexical-level syntactic features and, therefore, a truly syntax-agnostic approach should not leverage that information.
Transition-based Semantic Role Labeling with Pointer Networks

For implementing our technique, we resort to Pointer Networks (Vinyals, Fortunato and Jaitly, 2015), which provide an efficient $O(n^2)$ runtime complexity in practice. We experimentally prove that our end-to-end SRL system surpasses strong baselines (including syntax-aware approaches) on the CoNLL-2009 corpora, becoming the highest-performing model in practically all languages. Our major contributions can be summarized as follows:

- We design the first syntax-agnostic approach for transition-based SRL, requiring neither syntactic dependency trees nor PoS tag information.
- Our approach is end-to-end and can be directly applied to plain text, tackling the four SRL subtasks in one shot and without requiring any external module.
- Our model is robust and achieves state-of-the-art results on the majority of languages from CoNLL-2009 benchmark with and without pre-identified predicates.
- We empirically prove that the proposed transition-based technique processes CoNLL-2009 corpora in $O(n^2)$ time, being more efficient than best-performing graph-based models ($O(n^3)$).

The remainder of this article is organized as follows: Section 2 introduces previous studies for syntax-agnostic SRL. In Section 3, we define the transition system for full SRL and detail the proposed Pointer Network architecture. In Section 4, we extensively evaluate our SRL model on CoNLL-2009 corpora, present a discussion of the experimental results, analyze the contribution of each component and study the time complexity of our approach in practice. Lastly, Section 5 includes final conclusions.

2. Related work

Syntax-based approaches (Roth and Lapata, 2016; He et al., 2018b, 2019; Cai and Lapata, 2019b; Li et al., 2021) have been the mainstream for dependency-based SRL, consistently proving that syntactic information is highly effective for achieving state-of-the-art accuracies.

Alternatively, Marcheggiani et al. (2017) proposed the first syntax-agnostic model for dependency-based SRL. Instead of leveraging syntactic features for capturing long-distance predicate-argument dependencies, they employ a BiLSTM-based encoder. Their approach exclusively addresses argument identification and labeling, directly using gold predicates from CoNLL-2009 corpora and resorting to other works (Zhao, Chen, Kazama, Uchimoto and Torisawa, 2009; Björkelund, Hafdell and Nugues, 2009; Roth and Lapata, 2016) for language-specific predicate disambiguation. This initial attempt was followed by the graph-based model by Cai et al. (2018), which is considered the first syntax-agnostic method for full end-to-end SRL. They perform the four SRL subtasks by a single model. To achieve that, they apply the widely-used biaffine attention mechanism (Dozat and Manning, 2017) for exhaustively scoring all predicate-argument relations and their semantic roles; then, during decoding, they first search for predicates and, in a second stage, each predicate is processed by identifying and labeling its arguments. This work was improved by other syntax-agnostic graph-based techniques such as (Li, He, Zhao, Zhang, Zhang, Zhou and Zhou, 2019) and (Li et al., 2020) that, instead of applying a two-stage strategy, model dependency-based SRL as a graph parsing task, where predicates and arguments are uniformly treated and jointly processed. However, while performing full SRL, these two systems follow a pipeline strategy and, therefore, are not considered end-to-end: the former follows (Roth and Lapata, 2016) for predicate disambiguation and the latter identifies predicates in advance with a separate sequence tagging model. Recently, Zhou et al. (2022) extended the work by Li et al. (2020) to full end-to-end SRL, obtaining promising results on the CoNLL-2009 English dataset without pre-identified predicates.

We can also find recent syntax-agnostic approaches that do not perform full SRL and follow a predicate-centered strategy for decoding and word representation (based on gold predicates provided by CoNLL-2009 corpora): (Chen, Lyu and Titov, 2019) and (Lyu, Cohen and Titov, 2019), which additionally implement different iterative refinement procedures, and (Conia and Navigli, 2020), which employs an additional specific encoder for contextualizing each gold predicate in the sentence.

Regarding transition-based SRL modeling, just two syntax-aware attempts were proposed. Choi and Palmer (2011) presented a pre-deep-learning system that relies on handcrafted syntactic features and is only able to identify arguments and predict their semantic roles. And, recently, Fei et al. (2021) developed an end-to-end system that resorts to TreeLSTMs (Tai, Socher and Manning, 2015) for leveraging syntactic information. In addition, this work firstly
processes the sentence from left to right in order to identify any possible predicate and, when a predicate is found, their transition system searches for arguments from near to far. Finally, while their approach can be used for full SRL, Fei et al. (2021) did not evaluate their transition-based model on CoNLL-2009 corpora without gold predicates, just testing it on the English dataset with pre-identified predicates.

On the other hand, graph-based approaches (Wang, Huang and Tu, 2019; He and Choi, 2020) are also the mainstream in other semantic parsing tasks such as Semantic Dependency Parsing (SDP) (Oepen, Kuhlmann, Miyao, Zeman, Flickinger, Hajič, Ivanova and Zhang, 2014); however, Fernández-González and Gómez-Rodríguez (2020) introduced a transition-based algorithm that yields state-of-the-art scores on that task. Inspired by the latter, we design the first transition-based model that, without any kind of syntactic features, performs full end-to-end SRL in a single forward pass.

3. Model

3.1. Graph representation

For full end-to-end SRL modeling, the four subtasks must be formulated as a single graph parsing task, where predicates and arguments are uniformly treated. To that end, the relations between predicates and arguments must be represented as labeled dependency arcs, where the predicate acts as the semantic head, the argument as the semantic dependent and the semantic role as the dependency label. In addition, end-to-end graph-based SRL models (Cai et al., 2018; He et al., 2018b; Li et al., 2019) augment the original dependency-based structure by building a single-rooted graph (not necessarily acyclic): i.e., all predicates are attached to an artificial root node (added at the begging of the sentence) and the resulting arcs are labeled with predicate sense tags. Formally, given an input sentence \( X = w_0, w_1, \ldots, w_n \) (being \( w_0 \) the artificial root node), a full end-to-end SRL system is expected to completely produce a graph \( G \) represented as a set of labeled predicate-argument relations: \( G \subseteq W \times W \times L \), where \( W \) is the set of input words \( (W = \{w_0, w_1, \ldots, w_n\}) \) and \( L \) refers to the set of semantic role plus predicate sense labels. We adopt this graph representation for implementing our transition-based model. In Figure 1(b), we present the resulting single-rooted graph obtained from the original dependency-based SRL structure in Figure 1(a).

Moreover, it is worth noting that, in some languages such as Czech, the same word can serve as two or more arguments to the same predicate, resulting in two or more dependencies between the same two words. For instance, we can find two dependency arcs between predicate \( w_p \) and word \( w_a \) respectively tagged with semantic roles \( A1 \) and \( A2 \) (meaning that \( w_a \) serves as arguments \( A1 \) and \( A2 \) for predicate \( w_p \)). We handle this by keeping just one dependency between words \( w_p \) and \( w_a \), and by assigning the concatenation of both semantic roles \( (A1 \# A2) \) as dependency label. Additionally, in some datasets, a predicate can be also an argument of itself (i.e., a dependency where the head and the dependent are the same word). An example of this can be seen in Figure 1, where the word managers acts as the predicate and argument \( A0 \). To encode that information in our final representation, we concatenate the semantic role label of this dependency \( (A0 \text{ in the example}) \) with the predicate sense \( (01 \text{ for predicate managers}) \) and use the resulting label \( (01\#A0) \) for tagging the arc that will attach that predicate to the artificial root. In both pre-processing strategies, the original structure is easily recovered before evaluation.

3.2. Transition system

Inspired by (Fernández-González and Gómez-Rodríguez, 2020), we design a transition system for generating a graph \( G \) for the input sentence by applying a sequence of actions \( A = a_1, \ldots, a_t \). These actions (or transitions) will be sequentially predicted by a neural model. In this section, we formally define the main components of the proposed transition system: state configurations and actions.

While other transition-based SRL systems require more complex state configurations with several stacks and additional data structures for temporarily storing partially-processed words (Choi and Palmer, 2011; Fei et al., 2021), we just need to implement two pointers for building any dependency graph. More in detail, the proposed transition system has state configurations of the form \( c = \langle i, j, \Sigma \rangle \), where \( i \) points at the word \( w_i \) currently being processed, \( j \) indicates the position of the last identified predicate \( w_j \) for \( w_i \) and \( \Sigma \) contains the set of already-created edges. Given a sentence \( X = w_0, w_1, \ldots, w_n \) (with \( w_0 \) as artificial root node), the process starts at the initial state configuration \( c_{\text{initial}} = \langle 1, -1, \emptyset \rangle \), where \( i \) is pointing at the first input word \( w_1 \), no predicate position has been saved yet at \( j \) and \( \Sigma \) is empty. Then, after applying a sequence of actions \( A \), the transition system reaches a final configuration of the form

\[^3\text{As common practice, the lemma is removed from the predicate sense tag (e.g., manager in manager:01), since that information is typically provided in datasets from CoNLL-2009.} \]
Transition-based Semantic Role Labeling with Pointer Networks

| transition | state configuration | focus word, | last predicate, | added arc |
|------------|---------------------|-------------|----------------|-----------|
| ARC-4      | ⟨1, 4, Σ ∪ {4 → 1}⟩ | But₁        | say₄          | say₄ → But₁ |
| Shift      | ⟨2, 1, Σ⟩           | fund₂       | managers₂     | managers₂ → fund₂ |
| ARC-3      | ⟨2, 3, Σ ∪ {3 → 2}⟩ | fund₂       | managers₃     | managers₃ → fund₂ |
| Shift      | ⟨3, 1, Σ⟩           | managers₃   | Root₀         | Root₀ → managers₁ |
| ARC-0      | ⟨3, 0, Σ ∪ {0 → 3}⟩ | managers₃   | Root₀         | Root₀ → say₄ |
| ARC-4      | ⟨3, 4, Σ ∪ {4 → 3}⟩ | say₄        | say₄          | say₄ → managers₁ |
| Shift      | ⟨4, 1, Σ⟩           | say₄        | Root₀         | Root₀ → say₄ |
| ARC-0      | ⟨4, 0, Σ ∪ {0 → 4}⟩ | say₄        | Root₀         | Root₀ → say₄ |
| Shift      | ⟨5, 1, Σ⟩           | they₅       |                |            |
| Shift      | ⟨6, 1, Σ⟩           | 're₆        | say₄          | say₄ → 're₆ |
| ARC-4      | ⟨6, 4, Σ ∪ {4 → 6}⟩ | 're₆        |                |            |
| Shift      | ⟨7, 1, Σ⟩           | ready₇      |                |            |
| Shift      | ⟨8, 1, Σ⟩           | 's          |                |            |
| Shift      | ⟨9, 1, Σ⟩           |             |                |            |

Table 1
Transition sequence and resulting state configurations for incrementally generating arcs of the dependency graph in Figure 1(b).

c_{terminal} = ⟨n + 1, −1, Σ⟩, where all the words have been shifted and Σ contains the edges of the graph G for the input sentence X.

Unlike the works by Choi and Palmer (2011) and Fei et al. (2021) that define six different actions to produce SRL structures, we just require two transitions:

- **ARC-p** that attaches the current focus word wᵢ to the head word at position p, building a semantic dependency arc from the identified predicate wₚ to argument wᵢ. By applying this action, the transition system moves from state configurations ⟨i, j, Σ⟩ to ⟨i, p, Σ ∪ {wᵢ → wₚ}⟩. This transition can only be applied if the resulting edge has not been created yet (i.e., wᵢ → wₚ ∉ Σ) and the predicate wₚ is in a higher position than the last identified predicate for wᵢ in position j (i.e., j < p). The latter condition is necessary since the head assignment to the current focus word must follow the left-to-right order used for training.¹ Finally, the resulting dependency arc is labeled by a jointly-trained classifier, as described in the following section.

- **SHIFT** that moves i one position to the right, pointing at the word wᵢ₊₁. And, since we will start searching for predicates for that unprocessed word, j is initialized to −1. Therefore, we move from state configurations ⟨i, j, Σ⟩ to ⟨i + 1, −1, Σ⟩ by using this action.

The resulting transition system processes an input sentence from left to right by applying a sequence of SHIFT-ARC actions that attaches some words to one or several predicates and leaves others unattached, incrementally building a dependency graph. Please see in Table 1 how this transition-based algorithm generates the graph in Figure 1(b).

While the described transition system was originally designed for performing full end-to-end SRL, it can be easily adapted to leverage gold predicate information provided by the CoNLL-2009 corpora (where no predicate identification is required). For that purpose, a third condition must be added to the ARC-p transition: it can be applied only if the word wₚ is a gold predicate. In addition, each word wᵢ that is a gold predicate is directly attached to the artificial root w₀. Please note that, while these modifications allow a fairer comparison on the CoNLL-2009 benchmark, our approach does not follow a predicate-centered strategy as the vast majority of SRL systems evaluated on that shared task. These typically base their training and decoding procedures on the existence of pre-identified predicates and, as a consequence, they simply focus on individually processing each given predicate by searching for its arguments over the whole input.

¹ Please note that, while applying a different order in head attachments (e.g., inside-out (Ma et al., 2018)) could lead to slight improvements in accuracy, we decided to keep it simple and follow a left-to-right strategy.
3.3. Pointer Network

We use a Pointer Network (Vinyals et al., 2015) for implementing the proposed transition-based algorithm. We manage to properly represent state configurations in this neural model and use that information for generating transition sequences necessary for processing input sentences. We detail below the different components of our neural architecture:

**Word representation** Each input token $w_i$ is represented by the concatenation of word ($e_i^{\text{word}}$), lemma ($e_i^{\text{lemma}}$) and character-level ($e_i^{\text{char}}$) embeddings. The latter is obtained by encoding characters inside $w_i$ with convolutional neural networks (CNNs) (Ma and Hovy, 2016) and, unlike most SRL systems, we do not include PoS tag embeddings in order to develop a truly syntax-agnostic approach. In addition, we also evaluate our model with deep contextualized word embeddings ($e_i^{\text{BERT}}$) from pre-trained language model BERT (Devlin, Chang, Lee and Toutanova, 2019). In particular, we use mean pooling (i.e., the average value of all subword embeddings) to extract word-level representations from weights of the second-to-last layer. Lastly, we follow previous works (Cai et al., 2018; He et al., 2018b; Li et al., 2020) and leverage a predicate indicator embedding ($e_i^{\text{indicator}}$) when our model is tested following the CoNLL-2009 setting and, therefore, pre-identified predicates are available. When all these embeddings are exploited, the word representation $e_i$ is obtained as follows:

$$e_i = e_i^{\text{word}} \oplus e_i^{\text{lemma}} \oplus e_i^{\text{char}} \oplus e_i^{\text{BERT}} \oplus e_i^{\text{indicator}}$$

**Encoder** As a common practice, we feed a three-layer bidirectional LSTM (BiLSTM) (Hochreiter and Schmidhuber, 1997) to generate a context-aware vector representation $h_i$ for each word vector $e_i$:

$$h_i = \text{BiLSTM}(e_i) = f_i \oplus b_i$$

where $f_i$ and $b_i$ are respectively the forward and backward hidden states of the last LSTM layer at the $i$th position. Additionally, a randomly-initialized vector $h_0$ is used for denoting the artificial root node. As a result, the sequence of word representations $E = e_1, \ldots, e_n$ is encoded into a sequence of encoder vectors $H = h_0, h_1, \ldots, h_n$.

**Decoder** We employ a unidirectional one-layer LSTM plus an attention mechanism for decoding. Firstly, at each time step $t$, state configurations $c_t = (i, j, \Sigma)$ are encoded by feeding the LSTM with the combination\(^5\) of the respective encoder representations $h_i$ and $h_j$ of the current focus word ($w_i$) and its last assigned predicate ($w_j$) if available. This will generate the state configuration representation $s_t$:\(^7\)

$$r_t = h_i + h_j$$

$$s_t = \text{LSTM}(r_t)$$

Please note that, while introducing high-order information (such as the co-parent representation $h_j$ of a future predicate of $w_j$) significantly penalizes graph-based models’ runtime complexity, transition-based algorithms can straightforwardly leverage this information without harming their efficiency.

Once the current state configuration $c_t$ is properly represented as $s_t$, an attention mechanism is used for selecting the action $a_t$ to be applied at time step $t$. In particular, this mechanism employs the biasline scoring function (Dozat and Manning, 2017) to compute the score between each input word $w_k$ (represented by the encoder vector $h_k$ with $k \in [0, n]$) and the state configuration representation $s_t$; and then normalizes the resulting vector $v'$ of length $n$ to output the attention vector $a_t$, which is a softmax distribution with dictionary size equal to the length of the input:

$$v'_k = \text{score}(s_t, h_k) = f_1(s_t)^T W f_2(h_k) + U^T f_1(s_t) + V^T f_2(h_k) + b$$

$$a_t = \text{softmax}(v')$$

where $W$ is the weight matrix of the bi-linear term, $U$ and $V$ are the weight tensors of the linear terms, $b$ is the bias vector and $f_1(\cdot)$ and $f_2(\cdot)$ are two one-layer perceptrons with ELU activation to obtain lower-dimensionality and avoid overfitting. From the attention vector $a_t$, we select the highest-scoring position $p_t$ from the input and, being $c_t = (i, j, \Sigma)$, use that information to choose the current action $a_t$ between the two available transitions as follows:

\(^5\)This embedding simply marks whether $w_i$ is a predicate or not based on the information provided by CoNLL-2009 corpora.

\(^6\)Instead of concatenating both vectors, we compute the element-wise summation to avoid increasing the dimension of the resulting vector $r_t$.

\(^7\)Please note that $\Sigma$ in $c_t$ is not used for generating the state configuration representation $s_t$. $\Sigma$ was exclusively designed to collect already-built edges and its main purpose is to prevent the transition system from creating dependency arcs already added to $\Sigma$. 

• if \( p_t = i \), then a \( \text{SHIFT} \) action is applied, moving the focus word pointer to the next word.

• On the contrary, if \( p_t \neq i \), then the \( \text{ARC} \) transition parameterized with \( p_t \) will be considered, building a dependency arc between the word \( w_i \) and its predicate \( w_{p_t} \). In case that conditions required by this action are not satisfied, then the next highest-scoring position in \( \alpha_t \) will be used for choosing again between the \( \text{SHIFT} \) and \( \text{ARC} \) transitions.

In Figure 2, we include a sketch of the proposed Pointer Network architecture and decoding steps for partially building the graph structure in Figure 1(b).

Finally, when an \( \text{ARC}-p_t \) transition creates an edge between the current focus word \( w_i \) and the predicate \( w_{p_t} \), we apply a classifier for labeling it. This labeler is implemented by the biaffine scoring function previously described. Concretely, for each available semantic role or predicate sense label \( l \in L \), we compute the score of assigning \( l \) to the predicted arc between the argument \( w_i \) (encoded as \( s_t \)) and the predicate \( w_{p_t} \) (represented by \( h_{p_t} \)) as follows:

\[
u_t^l = \text{score}(s_t, h_{p_t}, l) = g_1(s_t)^T W_l g_2(h_{p_t}) + U_l^T g_1(s_t) + V_l^T g_2(h_{p_t}) + b_l
\]

where \( W_l \) is a weight matrix, \( U_l \) and \( V_l \) are weight tensors and \( b_l \) is the bias vector exclusively used for each label \( l \), and \( g_1(\cdot) \) and \( g_2(\cdot) \) are two one-layer perceptrons with ELU activation. Then, we select the highest-scoring label in the vector \( \beta_t \) for tagging the predicted arc.

**Training objectives** The loss of our model comes from the transition system and the labeler. On the one hand, the transition system is trained by minimizing the total log loss of choosing the correct sequence of \( \text{SHIFT-ARC} \) transitions \( A \) to output the gold dependency graph \( \tilde{G} \) for the input sentence \( X \) (i.e., predicting the correct sequence of indices \( p_t \), with each decision at time step \( t \) being conditioned by previous ones \( p_{<t} \)):

\[
\mathcal{L}_{\text{trans}}(\theta) = - \sum_{t=1}^{T} \log P_g(p_t|p_{<t}, X)
\]

On the other hand, the labeler is trained to minimize the total log loss of assigning the correct label \( l \), given a dependency arc from predicate \( w_{p_t} \) to dependent word \( w_i \):

\[
\mathcal{L}_{\text{label}}(\theta) = - \sum_{t=1}^{T} \log P_g(l|w_{p_t}, w_i)
\]
Table 2
Data statistics for CoNLL-2009 training sets. We report the number of sentences, annotated sentences (with at least one predicate), tokens, predicates and arguments, as well as the average sentence length.

| Language  | Sentences | Annotated | Avg. length | Tokens | Predicates | Arguments |
|-----------|-----------|-----------|-------------|--------|------------|-----------|
| Catalan (CA) | 13,200     | 12,873    | 30.2        | 390,302| 37,431     | 84,367    |
| Chinese (ZH)  | 22,277     | 21,071    | 28.5        | 609,060| 102,813    | 231,869   |
| Czech (CZ)    | 38,727     | 38,578    | 16.9        | 652,544| 414,237    | 365,255   |
| English (EN)  | 39,279     | 37,847    | 25.0        | 958,167| 179,014    | 393,699   |
| German (DE)   | 36,020     | 14,282    | 22.2        | 648,677| 17,400     | 34,276    |
| Spanish (ES)  | 14,329     | 13,835    | 30.7        | 427,442| 43,824     | 99,054    |

Finally, we jointly train the transition system and the labeler by optimizing the sum of their losses:

\[ \mathcal{L}(\theta) = \mathcal{L}_{\text{tran}}(\theta) + \mathcal{L}_{\text{label}}(\theta) \]

4. Experiments

4.1. Data
We conduct experiments on datasets from CoNLL-2008 and CoNLL-2009 shared tasks (Surdeanu et al., 2008; Hajič et al., 2009). The former was an English-only benchmark with in-domain (from the Wall Street Journal corpus (Marcus, Santorini and Marcinkiewicz, 1993)) and out-of-domain (from the Brown Corpus (Francis and Kucera, 1982)) test sets. This was extended by the CoNLL-2009 shared task to a multilingual benchmark by adding 6 more languages (Catalan, Chinese, Czech, German, Japanese and Spanish) with 6 in-domain and 2 out-of-domain test sets. Unlike CoNLL-2008 English dataset, CoNLL-2009 corpora notably simplified the SRL task by providing gold predicates beforehand. In order to properly test our model under a real-world usage scenario and also compare it to most previous works, we evaluate our approach on CoNLL-2009 datasets following two different settings: w/ pre-identified predicates and w/o pre-identified predicates (requiring the latter full SRL as in the CoNLL-2008 shared task).

In Table 2, we summarize the training data statistics for each language. From this information, we can see as, while having a notable amount of sentences in the training set, German is considered a low-resource language due to the low proportion of annotated sentences and predicate instances in comparison to other languages (Conia and Navigli, 2020). For the same reasons, Catalan and Spanish are also classified as low-resource languages.

Finally, we use the CoNLL-2009 official scoring script for performance evaluation. This measures the labeled precision, recall and F₁ score for semantic dependencies.

4.2. Setup
In our experiments, word and lemma embeddings are initialized with 300-dimensional GloVe vectors (Pennington, Socher and Manning, 2014) for English; structured-skipgram embeddings (Ling, Dyer, Black and Trancoso, 2015) for Chinese (dimension 80), German (dimension 64) and Spanish (dimension 64); and 64-dimensional Polyglot embeddings for Catalan and Czech. 100-dimensional character-level embeddings are randomly initialized and, for CNNs, we use 100 filters with a window size of 3 and max-pooling. For BERT-based embeddings, we extract respectively 768-dimensional and 1024-dimensional vectors from specific BERT_{base} and BERT_{large} models (Devlin et al., 2019): bert-large-cased for English, bert-base-multilingual-cased for Catalan, bert-base-chinese for Chinese, deepset/gbert-large (Chan, Schweter and Möller, 2020) for German, bert-base-bg-cs-pl-ru-cased (Arkhipov, Trofimova, Kuratov and Sorokin, 2019) for Czech and bert-base-spanish-wwm-cased (Cañete, Chaperon, Fuentes, Ho, Kang and Pérez, 2020) for Spanish. Following a greener and less resource-consuming strategy, BERT-based embeddings are not fine-tuned during training. Finally, we randomly initialize a 16-dimensional predicate indicator embedding under the w/ pre-identified predicate setup.

Most hyperparameters were taken from (Fernández-González and Gómez-Rodríguez, 2020) and we directly apply them to all datasets and languages without further optimization. For training, we employ Adam optimizer (Kingma

---

8Please note that the CoNLL-2009 shared task also augmented CoNLL-2008 English dataset with pre-identified predicates.
9We do not evaluate our approach on the Japanese dataset since it is no longer available due to licensing problems.
10https://ufal.mff.cuni.cz/conll2009-st/scorer.html
### Table 3

| System | w/o pre-identified predicates | w/ pre-identified predicates |
|--------|--------------------------------|-----------------------------|
| **(Syntax-aware)** | | |
| He et al. (2018b) | n 83.9 82.7 83.3 | n 84.2 87.5 85.9 |
| Zhou, Li and Zhao (2020) + Joint | n 87.4 89.0 88.2 | n 84.2 89.0 85.9 |
| Zhou et al. (2020) + Joint + BERT fine-tuned | n 87.4 89.0 88.2 | n 87.4 89.0 88.2 |
| Munir, Zhao and Li (2021) + ELMO | n 85.8 84.4 85.1 | n 85.8 84.4 85.1 |
| Li et al. (2021) + ELMO | n 86.2 86.0 86.1 | n 86.2 86.0 86.1 |
| **(Syntax-agnostic)** | | |
| Cai et al. (2018) | y 84.7 85.2 85.0 | y 85.9 88.0 86.9 |
| Li et al. (2019)† | n 86.0 85.6 85.8 | y 86.7 86.2 86.5 |
| Li et al. (2020)† | n 86.0 85.6 85.8 | y 86.7 86.2 86.5 |
| Zhou et al. (2022)† | y 86.7 86.2 86.5 | y 86.7 86.2 86.5 |
| This work † + ELMO | n 84.5 86.1 85.3 | y 87.6 90.2 88.9 |
| Li et al. (2020)† + BERT fine-tuned | n 88.6 88.6 88.6 | y 87.6 90.2 88.9 |
| Zhou et al. (2022)† + BERT fine-tuned | y 87.6 90.2 88.9 | y 87.6 90.2 88.9 |
| This work † + BERT | y 87.2 89.8 88.5 | y 87.2 89.8 88.5 |

Note: * † indicates truly syntax-agnostic models that do not leverage PoS tag information.

**Table 3**

Precision (P), recall (R) and F\(1\) scores obtained by full SRL systems on the CoNLL-2008/CoNLL-2009 English in-domain (Wall Street Journal, WSJ) and out-of-domain (Brown) test sets w/o pre-identified predicates. The first block gathers methods enhanced with syntactic information and, the second block, those that are syntax-agnostic. We also indicate in column “end-to-end” whether approaches follow an end-to-end (y) or a pipeline (n) strategy, using the latter one or more extra models to accomplish full SRL. +ELMO and +BERT stand for augmentations with deep contextualized word-level embeddings from pre-trained language models ELMO (Peters, Neumann, Iyyer, Gardner, Clark, Lee and Zettlemoyer, 2018) and BERT, respectively; and +Joint means that the system learns SRL jointly with other tasks. Please note that we keep BERT-based embeddings frozen in order to avoid increasing the computational cost, and we denote with “fine-tuned” those systems that do undertake an expensive fine-tuning during training in order to adapt them to SRL. Finally, we mark with † those truly syntax-agnostic models that do not leverage PoS tag information.

and Ba, 2014) with initial learning rate of \(\eta_0 = 0.001\), \(\beta_1 = 0.9\) and \(\beta_2 = 0.9\). We also use a fixed decay rate of 0.75 and a gradient clipping of 5.0 in order to mitigate the gradient exploding effect (Pascanu, Mikolov and Bengio, 2013). Moreover, we use LSTMs with 512-dimensional hidden states for both encoder and decoder, applying recurrent dropout (Gal and Ghahramani, 2016) with a drop rate of 0.33 between hidden states and layers. We also apply a 0.33 dropout to all embeddings. In addition, all models are trained up to 600 epochs with batch size 32, and beam-search decoding with beam size 5 is utilized in all experiments. Finally, we choose the checkpoint with the highest labeled F\(1\) score on the development set for posterior in-domain and out-of-domain evaluations.

### 4.3. Results and discussion

#### English results

In Table 3, we compare our model against previous full SRL systems on English in-domain and out-of-domain tests sets under the w/o pre-identified predicates setting. Our single-model approach achieves the best performance on both test sets among syntax-agnostic SRL systems without deep contextualized word embeddings, even outperforming all syntax-based models on the WSJ test set. When pre-trained language models come into play, our system obtains competitive accuracies (improving over again all syntax-aware approaches on the in-domain test set), but it is slightly surpassed by those graph-based models (Li et al., 2020; Zhou et al., 2022) that adapt BERT-based embeddings to SRL by fine-tuning them during training.

Table 4 presents the results on English in-domain and out-of-domain tests sets w/ pre-identified predicates. It can be seen as predicate-centered SRL systems outnumber those developed for full SRL (which are reported in Table 3). While our approach was originally designed for dealing with the lack of gold predicates and just minor adaptations were undertaken for leveraging pre-identified predicate information, our system behaves similarly to the full SRL setting: it is the best-performing syntax-agnostic approach without contextualized word representations (also surpassing syntax-aware systems on the WSJ test set), but the highest scores are reported by graph-based models that fine-tune BERT-based embeddings (Li et al., 2020) or, while keeping them frozen, leverage syntactic information (Li et al., 2021).
**Table 4**

| System | WSJ end-to-end P | R | F₁ | Brown P | R | F₁ |
|--------|------------------|---|----|--------|---|----|
| Lei, Zhang, Márquez, Moschitti and Barzilay (2015) | n | - | 86.6 | n | - | 75.6 |
| FitzGerald, Täckström, Ganchev and Das (2015) + Ens | n | - | 86.7 | n | - | 75.2 |
| Roth and Lapata (2016) + Ens | n | 90.3 | 85.7 | 87.9 | 79.7 | 73.6 | 76.5 |
| Marcheggiani and Titov (2017) + Ens | n | 90.5 | 87.7 | 89.1 | 80.8 | 77.1 | 78.9 |
| He et al. (2018b) | y | 89.7 | 89.3 | 89.5 | 81.9 | 76.9 | 79.3 |
| Cai and Lapata (2019b) + Joint | n | 90.5 | 88.6 | 89.6 | 80.5 | 78.2 | 79.4 |
| Kasai et al. (2019) | n | 89.0 | 88.2 | 88.6 | 78.0 | 77.2 | 77.6 |
| He et al. (2019) | n | 90.0 | 90.0 | 90.0 | - | - | - |
| Zhou et al. (2020) + Joint | n | 88.7 | 89.8 | 89.3 | - | - | - |
| Li et al. (2021) | n | - | 89.2 | - | - | 80.1 |
| Li, He, Cai, Zhang, Zhao, Liu, Li and Si (2018) + ELMO | n | 90.3 | 89.3 | 89.8 | 80.6 | 79.0 | 79.8 |
| Cai and Lapata (2019b) + Joint + ELMO | n | 90.9 | 89.1 | 90.0 | 80.8 | 78.6 | 79.7 |
| Cai and Lapata (2019a) + ELMO | n | 91.1 | 90.4 | 90.7 | 82.1 | 81.3 | 81.6 |
| Cai and Lapata (2019a) + ELMO + Semi | n | 91.7 | 90.8 | 91.2 | 83.2 | 81.9 | 82.5 |
| Kasai et al. (2019) + ELMO | n | 90.3 | 90.0 | 90.2 | 81.0 | 80.5 | 80.8 |
| He et al. (2019) + BERT | n | 90.4 | 91.3 | 90.9 | - | - | - |
| Zhou et al. (2020) + Joint + BERT<sub>fine-tuned</sub> | n | 91.2 | 91.2 | 91.2 | 85.7 | 86.1 | 85.9 |
| Munir et al. (2021) + ELMO | n | 91.2 | 90.6 | 90.9 | 83.1 | 82.6 | 82.8 |
| Li et al. (2021) + ELMO | n | 90.5 | 91.7 | 91.1 | 83.3 | 80.9 | 82.1 |
| Li et al. (2021) + BERT | n | - | - | 91.8 | - | - | 83.2 |

**Multilingual results** Table 5 summarizes the performance on the remaining CoNLL-2009 languages (including out-of-domain test sets if available) under both w/ pre-identified predicates and w/o pre-identified predicates settings. Compared with previous methods, our approach yields strong performance consistent across languages, regardless of the availability of gold predicate information.

Without contextualized word representations, our end-to-end SRL system improves over the best syntax-agnostic methods on all languages in the in-domain setting (with and without pre-identified predicates), bringing notable improvements on both high-resource (e.g., Czech) and low-resource (e.g., German) datasets. Our proposal also outperforms syntax-based models in all datasets except German, meaning that leveraging syntactic information (Cai and Lapata, 2019a) is still crucial for obtaining state-of-the-art results on that language.

When our model is augmented with frozen BERT-based embeddings, we achieve the highest score to date on 3 out of 5 languages in the in-domain setting (with and without given predicates), being only outperformed by (Li et al.,...
## Transition-based Semantic Role Labeling with Pointer Networks

**Table 5**

| System (w/ pre-identified predicates) | end-to-end | CA\(_{id}\) | CZ\(_{id}\) | CZ\(_{ood}\) | DE\(_{id}\) | DE\(_{ood}\) | ES\(_{id}\) | ZH\(_{id}\) |
|--------------------------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| CoNLL-2009 ST best                  |            | 80.3        | 86.5        | 85.4        | 79.7        | 65.9        | 80.5        | 78.6        |
| Zhao et al. (2009)                  | n          | 80.3        | 85.2        | 82.7        | 76.0        | 67.8        | 80.5        | 77.7        |
| Roth and Lapata (2016) + Ens        | n          | -           | -           | -           | 80.1        | -           | 80.2        | 79.4        |
| Cai and Lapata (2019b) + Joint      | n          | -           | -           | -           | 82.7        | -           | 81.8        | 83.6        |
| Cai and Lapata (2019a)              | n          | -           | -           | -           | 83.3        | -           | 82.1        | 84.6        |
| Cai and Lapata (2019a) + Semi       | n          | -           | -           | -           | 83.8        | -           | 82.9        | 85.0        |
| He et al. (2019)                    | y          | 84.9        | 88.8        | -           | 78.5        | -           | 83.9        | 84.8        |
| He et al. (2019) + BERT             | y          | 86.0        | 89.7        | -           | 81.1        | -           | 85.2        | 86.9        |
| Li et al. (2021)                    | y          | 85.5        | 90.5        | -           | 76.6        | -           | 84.3        | 86.1        |

| System (w/o pre-identified predicates) | end-to-end | CA\(_{id}\) | CZ\(_{id}\) | CZ\(_{ood}\) | DE\(_{id}\) | DE\(_{ood}\) | ES\(_{id}\) | ZH\(_{id}\) |
|--------------------------------------|------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Marcheggiani et al. (2017)           | n          | -           | 86.0        | 87.2        | -           | -           | 80.3        | 81.2        |
| Mulcaire, Swayamdipta and Smith (2018)† | y          | 79.5        | 85.1        | -           | 70.0        | -           | 77.3        | 81.9        |
| Chen et al. (2019)                   | y          | 81.7        | 88.1        | -           | 76.4        | -           | 81.3        | 81.7        |
| Lyu et al. (2019)                    | n          | 80.9        | 87.6        | 86.0        | 75.9        | 65.7        | 80.5        | 83.3        |
| Li et al. (2020)†                    | y          | 84.7        | 90.6        | 90.8        | 76.4        | 70.1        | 84.2        | 86.0        |
| This work†                           | y          | 85.9        | 93.3        | 92.5        | 79.5        | 71.0        | 85.2        | 86.7        |
| Conia and Navigli (2020)† + BERT     | y          | 86.2        | 90.1        | 90.6        | 86.5        | 73.1        | 85.3        | 87.3        |
| Li et al. (2020)† + BERT fine-tuned  | y          | 86.8        | 91.6        | 91.7        | 85.5        | 71.9        | 86.9        | 88.5        |
| This work† + BERT                    | y          | 86.2        | 93.3        | 92.2        | 86.7        | 75.1        | 85.7        | 89.1        |

Results on the out-of-domain data (with and without gold predicates), our model (with and without BERT-based embeddings) outperforms existing SRL systems (including syntax-aware approaches) by a wide margin on Czech and German test sets (being especially challenging the latter, since it contains numerous infrequent predicates specifically included for the CoNLL-2009 shared task). Lastly, we observe in Czech as adding deep contextualized word embeddings has no effect without given predicates and is harmful with pre-identified predicates, probably meaning that a task-specific fine-tuning would be helpful in that case.

Finally, it is worth mentioning that, to the best of our knowledge, our proposal is the first end-to-end system that provides scores for full multilingual SRL (without given predicates) on CoNLL-2009 datasets, since (Li et al., 2020) (the only graph-based model included in that setting) adopts a pipeline strategy. Moreover, our model is the best-performing approach among truly syntax-agnostic SRL systems, which do not exploit PoS tag embeddings. Lastly, we do not fine-tune hyperparameters for individual languages, suggesting that the presented approach is robust and can be directly applied to other languages.

### 4.4. Ablation study

The previous section has already shown as BERT-based embeddings substantially boost our model accuracy; however, we do not know the performance impact of, for instance, beam-search decoding or high-order features provided by the last assigned predicate. Thus, we conduct an ablation study of our neural architecture in order to

---

D. Fernández-González: Preprint submitted to Elsevier
better understand the contribution of each component in the final accuracy. In particular, we successively remove from
the full model: the beam-search decoding, co-parent features (i.e., state configuration representations $s_i$ are generated
without the addition of $h_j$ to $h_i$), lemma embeddings and character embeddings. In Table 6, we can observe that the
removal of every component leads to an overall performance degradation; however, character embedding ablation has
the largest impact on the performance of our model, resulting in significant drops in $F_1$ (−1.68), $F^{pred}_{1}$ (−1.73) and
$F^{arg}_{1}$ scores (−1.68). Lemma embeddings also play an important role in our neural architecture, notably increasing our
model performance. In addition, we can also see as the lack of co-parent features improves the accuracy on predicate
identification and disambiguation ($F^{pred}_{1}$), but penalizes the performance on argument identification and labeling ($F^{arg}_{1}$).
This means that co-parent features are especially beneficial for argument processing subtasks, which are more complex
and have a larger impact on the overall $F_1$ score (since, except in Czech, there are significantly more arguments
than predicates in CoNLL-2009 corpora). Finally, the beam-search decoding with beam size 5 has a minor impact
in comparison to the addition of lemma and character embeddings. We think that a further beam-size exploration
might probably increase its contribution to the final accuracy.

### 4.5. Time complexity

The full time complexity of best-performing graph-based models (Li et al., 2020; Zhou et al., 2022) is $O(n^3)$ due
to the leverage of higher-order information. We will prove that our approach is more efficient, being $O(n^2)$ its overall
expected worst-case running time for the range of data tested in our experiments.

Being $n$ the sentence length, a general directed graph can have at most $\Theta(n^2)$ edges, requiring our transition system
$O(n^2)$ actions to build it in the worst case (i.e., $n$ SHIFT transitions for processing all words and $n$ ARC actions per word
for assigning all its heads). Nevertheless, predicate-argument graphs from CoNLL-2009 datasets can be produced with
$O(n)$ transitions. In order to prove that, we need to determine the complexity of the proposed transition system in practice.
This can be done by examining, for each sentence, how the predicted transition sequence length varies as a
function of sentence length (Kübler, McDonald and Nivre, 2009). Figure 3 graphically shows the relation between the
number of predicted transitions and the number of words for every sentence from CoNLL-2009 development splits. We
can clearly observe a linear relationship across all languages, which means that the number of ARC transitions required
per word is notably low and behaves like a constant in the represented linear function. This behavior is supported by the
fact that, due to the significant amount of unattached words, there are substantially less predicate-argument edges
than words in graphs from CoNLL-2009 data, being the average ratio of edges per word in a sentence less than 1 in
practically all training sets: 0.32 in Catalan, 0.53 in Chinese, 0.08 in German, 0.56 in English and 0.34 in Spanish. The
exception is observed in graphs from the Czech dataset, where we have more than one edge per word on average (1.15).
From this information, we can state that every sentence from CoNLL-2009 corpora (except Czech) can be processed
with $2n$ transitions at most (i.e., $n$ SHIFT actions plus $n$ ARC transitions); and we will require $3n$ transitions in the worst
case for generating graphs from Czech (i.e., $n$ SHIFT transitions plus $2n$ ARC actions). In both cases, the resulting
number of transitions is linear.

Finally, the time complexity of our approach not only comes from the transition system: for predicting each
transition, the attention vector $a_t$ must be computed over the whole input sentence in $O(n)$ time. Consequently, the
overall time complexity of the proposed SRL system on CoNLL-2009 corpora is $O(n^2)$.

\begin{table}[]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
System & $P$ & $R$ & $F_1$ & $F^{pred}_{1}$ & $F^{arg}_{1}$ \\
\hline
Full model & 85.36 & 85.89 & 85.63 & 90.34 & 83.46 \\
- beam search & 84.69 & 86.22 & 85.44 & 90.20 & 82.26 \\
- co-parent features & 84.47 & 86.13 & 85.29 & 90.45 & 82.91 \\
- lemma embeddings & 82.65 & 86.23 & 84.40 & 89.77 & 80.91 \\
- character embeddings & 80.94 & 84.58 & 82.72 & 88.04 & 80.28 \\
\hline
\end{tabular}
\caption{Overall precision, recall and $F_1$ scores, as well as specific $F_1$ scores measured only on predicate
identification+disambiguation ($F^{pred}_{1}$) and argument identification+labeling ($F^{arg}_{1}$) subtasks on the CoNLL-2009 English development split under the w/o pre-identified predicates setup.}
\end{table}
Transition-based Semantic Role Labeling with Pointer Networks

![Graphs for different languages showing the number of transitions predicted by the model relative to the sentence length for CoNLL-2009 development sets.]

Figure 3: Number of transitions predicted by our model relative to the sentence length for CoNLL-2009 development sets.

5. Conclusions

In this article, we propose the first syntax-agnostic transition-based approach for full end-to-end SRL. This exclusively relies on raw text as input, neither requiring syntactic trees nor resorting to external models to accomplish any of the SRL subtasks. In addition, we prove that our technique is more efficient in practice ($O(n^2)$) than best-performing graph-based models ($O(n^3)$). Thanks to these advantageous features, it can be easily applied in real-world applications and low-resource languages, where, for instance, syntactic information is scarce.

We not only extensively evaluate our model on in-domain and out-of-domain CoNLL-2009 corpora under the full SRL setting, but additionally adapt our approach to handle gold predicate information in order to perform a fair comparison against the vast majority of previous methods, which do not address predicate identification. While our single-model proposal obtains competitive accuracies on the CoNLL-2009 English data, it excels in the remaining five languages, achieving a strong performance across in-domain and out-of-domain test sets as well as high-resource and low-resource languages.

Lastly, although our transition-based SRL system is robust and accurate, it is outperformed by graph-based models that either fine-tune deep contextualized word embeddings or use additional syntactic information. Therefore, while we think that leveraging syntactic trees makes SRL systems less cost-effective and more dependent on high-resource languages, our model can benefit from syntax to further improve its performance. And, additionally, we could also perform a task-specific fine-tuning of BERT-based embeddings to obtain substantial accuracy gains in low-resource languages.

Acknowledgments

We acknowledge the European Research Council (ERC), which has funded this research under the European Union’s Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150), ERDF/MICINN-AEI (SCANNER-UDC, PID2020-113230RB-C21), Xunta de Galicia (ED431C 2020/11), and Centro de Investigación de Galicia “CITIC”, funded by Xunta de Galicia and the European Union (ERDF - Galicia 2014-2020 Program), by grant ED431G 2019/01.
CRediT authorship contribution statement

Daniel Fernández-González: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing - original draft, writing - review & editing, visualization.

References

Arkhipov, M., Trofimova, M., Kuratov, Y., Sorokin, A., 2019. Tuning multilingual transformers for language-specific named entity recognition, in: Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing, Association for Computational Linguistics, Florence, Italy. pp. 89–93. URL: https://aclanthology.org/W19-3712, doi:10.18653/v1/W19-3712.

Bastianelli, E., Castellucci, G., Croce, D., Basili, R., 2013. Textual inference and meaning representation in human robot interaction, in: Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora, Trento, Italy. pp. 65–69. URL: https://aclanthology.org/W13-3820.

Björkland, A., Hafell, L., Nogue, P., 2009. Multilingual semantic role labeling, in: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task, Association for Computational Linguistics, Boulder, Colorado. pp. 43–48. URL: https://aclanthology.org/W09-1206.

Cai, R., Lapata, M., 2019a. Semi-supervised semantic role labeling with cross-view training, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China. pp. 1018–1027. URL: https://aclanthology.org/D19-1094, doi:10.18653/v1/D19-1094.

Cai, R., Lapata, M., 2019b. Syntax-aware semantic role labeling without parsing. Transactions of the Association for Computational Linguistics 7, 343–356. URL: https://aclanthology.org/Q19-1022, doi:10.1162/tacl_a_00272.

Cai, J., He, S., Li, Z., Zhao, H., 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware?, in: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA. pp. 2753–2765. URL: https://aclanthology.org/C18-1233.

Cai, R., Gómez-Rodríguez, C., 2020. Transition-based semantic dependency parsing with pointer networks, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Online. pp. 710–716. URL: https://aclanthology.org/N19-1423, doi:10.18653/v1/N19-1423.

Conia, S., Navigli, R., 2020. Bridging the gap in multilingual semantic role labeling: a language-agnostic approach, in: Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online). pp. 6788–6796. URL: https://aclanthology.org/2020.coling-main.598, doi:10.18653/v1/2020.coling-main.598.

Dozat, T., Manning, C.D., 2017. Deep biaffine attention for neural dependency parsing, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France. doi:10.18653/v1/D17-1396.

Devlin, J., Chang, M.W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota. pp. 4171–4186. URL: https://aclanthology.org/N19-1423, doi:10.18653/v1/N19-1423.

Fernández-González, D., Gómez-Rodríguez, C., 2020. Multitask pointer network for multi-representational parsing. Knowledge-Based Systems 236, 107760. URL: https://www.sciencedirect.com/science/article/pii/S0950705121009849, doi:10.1016/j.knosys.2021.107760.

Fernández-González, D., Gómez-Rodríguez, C., 2022. Multitask pointer network for multi-representational parsing. Knowledge-Based Systems 236, 107760. URL: https://www.sciencedirect.com/science/article/pii/S0950705121009849, doi:10.1016/j.knosys.2021.107760.

FitzGerald, N., Täckström, O., Ganchev, K., Das, D., 2015. Semantic role labeling with neural network factors, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal. pp. 960–970. URL: https://aclanthology.org/D15-1112, doi:10.18653/v1/D15-1112.
Zheng, C., Kordjamshidi, P., 2020. SRLGRN: Semantic role labeling graph reasoning network, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Online. pp. 8881–8891. URL: https://aclanthology.org/2020.emnlp-main.714, doi:10.18653/v1/2020.emnlp-main.714.

Zhou, J., Li, Z., Zhao, H., 2020. Parsing all: Syntax and semantics, dependencies and spans, in: Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online. pp. 4438–4449. URL: https://aclanthology.org/2020.findings-emnlp.398, doi:10.18653/v1/2020.findings-emnlp.398.

Zhou, S., Xia, Q., Li, Z., Zhang, Y., Hong, Y., Zhang, M., 2022. Fast and accurate end-to-end span-based semantic role labeling as word-based graph parsing, in: Proceedings of the 29th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Gyeongju, Republic of Korea. pp. 4160–4171. URL: https://aclanthology.org/2022.coling-1.365.