Semantic Role Labeling for knowledge extraction

Davide Cavicchini
University of Trento
davide.cavicchini@studenti.unitn.it

Abstract

This project explores the development of an SRL system using transformer-based encoder models like BERT, aiming to bridge the gap between distributional semantics and logical knowledge representation. In this work, I focused on predicate detection, argument identification, and argument classification tasks. The results highlight the importance of the pre-training of the model and the architecture used on top of it, while also pointing out the shortcomings of the developed models and future directions to iron them out.

1 Introduction

In recent years, most of the advancements in Natural Language Processing have been toward generative models and distributional semantics. However, I argue these efforts can go as far since the model capacity will never enable it to know and understand all the aspects of the world. Furthermore, current generative models cannot be easily controlled to update their knowledge reliably. For this reason, the goal is to leverage all the "old school" methods for knowledge representation and reasoning to augment the capabilities of language models and make them more controllable and explainable. This project wants to be a preliminary study in this direction, trying to bridge the gap between distributional semantics and logical representations of the knowledge contained in a natural language text.

To achieve this, I focused on the already existing Semantic Role Labeling (SRL) task. SRL involves identifying the predicate-argument structures within a sentence, determining “who did what to whom”, “when”, “where”, etc. This detailed understanding of sentence structure and meaning is essential for various downstream applications, including information extraction, question-answering, and reasoning.

The objective of this project is to develop an SRL system that leverages transformer-based encoder models, such as BERT, to extract predicates and annotate semantic roles in a natural language text. This system will serve as a foundational component for constructing a knowledge graph representation of the text. This report will focus on the architecture tested for the SRL task and its performance.

2 Related Work

Older work relied on lexical and syntactic features, such as part-of-speech tags (Li et al., 2019) and syntactic trees (Roth and Lapata, 2016). However, a big problem of these approaches is the availability of such representations, which require a parser that inherently injects another layer of potential errors into the system. However, since the advent of BERT-like foundation models, the NLP community has increasingly relied on models pre-trained with a language modeling objective fine-tuned for downstream tasks. The use of BERT transformers for word-level predictions is not new, in Chen et al. (2019) they successfully use the final representation of the first token for each word produced by the BERT encoder for the slot-filling task.

Focusing on semantic role labeling, there has already been significant work on using BERT-based models, like Shi and Lin (2019). Their approach wants to make proper use of the attention mechanisms in the BERT encoder, though this requires feeding the encoder with the sentence followed by the predicate we are interested in classifying the roles for as *[[CLS] sentence [SEP] predicate [SEP]]*, thus requiring the model to be queried multiple times for each predicate contained in the text we are interested in.

3 Proposed Approach

The standard formulation of semantic role labeling decomposes into four subtasks: predicate detection, predicate sense disambiguation, argument identification, and argument classification. This work focuses on the task of predicate detection, argument identification, and classification.

3.1 Predicate Detection

The predicate detection task refers to the process of identifying and classifying the main verbs, like in (Universal-Propositions, 2019), or nouns, like in (Meyers et al., 2004), within a sentence that denotes actions, events, or states. The implemented model produces a probability for each word of being a predicate, the specifics of the architecture used will be dealt with later in section 5.

3.2 Argument identification and classification

This task requires computing the argument spans, or syntactic heads, and assigning them the correct semantic role. There exist two representations used for argument annotation: span-based and dependency-based. Here, I follow (Li et al., 2019) approach to unify these two annotation schemes.

3.2.1 Argument classification

In contrast with (Shi and Lin, 2019), I wanted to be able to perform the classification for each predicate in the sentence with a single pass in the transformer encoder. For this reason, the encoder network is fed only with the sentence we are interested in, and later the word representations, corresponding to the first token representations for each word, are combined with each predicate representation, and classified over the label set of semantic roles. More formally, given the word token representations W and predicate representations R, we want to classify the pairs $p \in W \times R$ into the power set of semantic role labels 2^L, where L is the set of all roles used in the dataset. The specific architecture used to perform this will be later discussed in section 5.

3.2.2 Argument identification

For argument identification, I implemented two simple logic to transform the predicted roles into a span-level prediction.

Concatenation A simple concatenation strategy where all the predicted spans for each role are concatenated.

Top Assumes the spans to only be contiguous and only predicts the top-scoring span identified as the one with the highest mean probability of its components. This assumption does not always hold in the used datasets. However, given their relatively rare occurrence, it allows for slightly higher F1 scores and makes the model behave better for long sequences with many predicates.

4 Dataset

The used datasets are: the Universal Prepositions Dataset (Universal-Propositions, 2019), and the NomBnak Dataset ((Meyers et al., 2004)).

4.1 Universal Prepositions Dataset

The Universal Prepositions Dataset consists of sentences annotated with semantic roles, focusing on verbal prepositions. The English version of the dataset was obtained from the merge of the sources: UD_English-EWT, propbank-release, and LDC2012T13. Offering a broad range of linguistic contexts, making it suitable for training models that need to generalize across different linguistic domains.

The dataset uses the convention of labeling the syntactic heads of the arguments, which, as mentioned, are then processed to use the representations proposed in (Li et al., 2019).

Unfortunately, in the test and dev set the dataset contains predicate senses which are not included in the training set. For this reason, this project does not tackle the problem of predicate sense disambiguation but would require minimal modification to include it.

4.2 NomBank Dataset

The NomBank Dataset annotates noun predicates with their corresponding arguments, providing a different perspective to the previous dataset. Unfortunately, this makes it incompatible with the previous dataset to be used to train the same model. However, it might be useful for the final goal of creating a KG representation of text, enabling the decomposition of the entities in the semantic arguments.

The dataset was preprocessed using a modified version of the scripts in the (Cognitive-Computation-Group) repository. Obtaining an analogous representation to the one used for the UP dataset.

5 Model

As mentioned, the model uses a pre-trained BERT-like model as a backbone encoder and extends it with custom layers for the SRL tasks I set out to tackle. The model architecture includes several

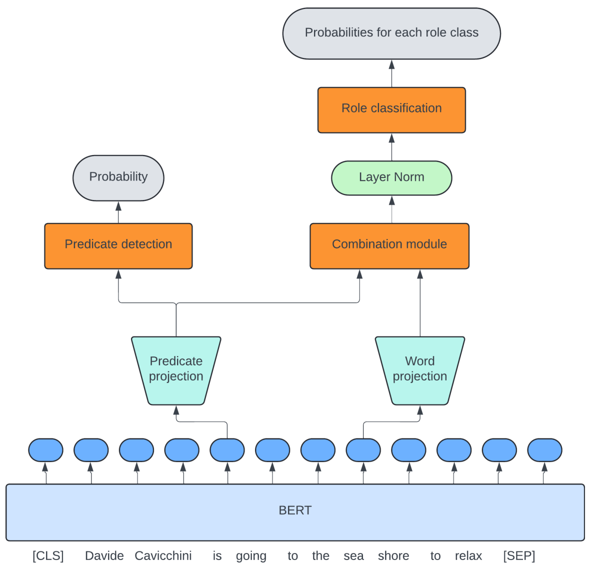

Figure 1: Full architecture for the SRL model

key components, each designed to address specific aspects of the SRL task and problems noticed during the testing of the configurations. The complete model architecture is illustrated in image 1.

5.1 Transformer Encoder

The transformer encoder serves as the core of the model. It provides powerful contextual representations, crucial for accurately identifying and classifying semantic roles and prepositions within sentences.

5.2 Projection Layers

Initial experiments with the model revealed its tendency to over-fit the training data, signaled by the increase of the dev loss, diverging from the F1 score which was increasing. This probably meant that the model was becoming increasingly confident with its predictions even when wrong.

To alleviate this problem, I decided to use two separate FC layers to shrink the dimensions of the word representations and predicate representations before the classification of the semantic roles. This allowed me to cut, up to a factor of 8, the parameters on top of the encoder model. Later, the predicate projection layer output was also used as input to the predicate detection allowing it to take gradients from both tasks.

However, from later experiments, the problem was resolved by using a different LR schedule and loss function. Nonetheless, these layers are useful to increase the representation capabilities of the model while managing the parameter count, since the obtained final results are comparable while being less prone to over-fitting and being over-confident as we will see in section 7.

5.3 Predicate Detection

The predicate detection classifier is responsible for identifying the predicates contained in the input sentence. It directly maps the word representations to the probability of each being a predicate.

5.4 Combination Strategy

As said in Section 3.2, the model uses pairs of predicate representation and word representation (w, r) to assign a semantic role to the word for that predicate. This means that the model has to somehow combine these representations to be later used by a classification head. Following this step, the obtained vector is normalized using a Layer Norm, useful to reduce the risk of saturating the activation functions. The implemented solutions are four: concatenation, mean, gating, and soft-attention.

Concatenation The representations are just concatenated to obtain the combined one.

$$F = r \mathbin{+\!\!+} w$$

Mean I use the mean of the two representations.

$$F = \frac{w + r}{2}$$

Gating Taking inspiration from the LSTM gates design, the model computes a transformed representation from the concatenation of the word and predicate. It then calculates a gating score for each dimension, used as weights for combining the transformed and word representation.

$$F = \sigma(r \mathbin{+\!\!+} w) \circ t(r \mathbin{+\!\!+} w) + (1 - \sigma(r \mathbin{+\!\!+} w)) \circ w$$

Soft-attention It only computes a relevance score from the concatenated representations used to sum up the weighted predicate and word representations.

$$F = \sigma(r \mathbin{+\!\!+} w) \circ r + (1 - \sigma(r \mathbin{+\!\!+} w)) \circ w$$

5.5 Role Classification

The final stage of the model involves classifying the identified pairs into the semantic role power set. This component uses the combined representation F to assign accurate role label probabilities using a FC head mapping, completing the SRL process.

From my experiments, this head needs at least one hidden layer to inject some non-linearity. Otherwise, it is unable to learn an effective classification function.

Since the datasets contain overlapping roles, the problem is formulated as a multi-labeling classification. Requiring the output of the model to be the probabilities for each semantic role an argument can take. The final prediction is a subset of semantic roles for each word in a sentence.

6 Training Procedure

The inference and training loop of the model is designed to allow training using a multi-objective loss function.

6.1 Loss Function

For the loss in a batch, I combine the loss from the predicate identification task and argument classification:

predicate identification The logits computed by the model for each word in each sentence in a batch are unrolled into a 2D tensor, on which I apply a BCE loss, making sure that the positive and negative examples have the same weight in the computation.

argument classification I compute the BCE loss for each role adjusting the positive weight of the example to have the same importance as the negative ones. Then I mean along the same role and finally sum up the results.

Final Loss The final loss is obtained by combining the two, rescaled by the F1 score for the two tasks. This allows the model to limit overfitting, especially for the predicate identification task, being the easiest of the two.

$$Loss = \frac{Loss_{rel}}{F1_{rel}} + \frac{Loss_{role}}{F1_{role}}$$

6.2 Optimization

The models were trained for 10 epochs, using a batch size of 32 for the large models and of 64 for the smaller ones. Each sentence in a batch accounts for all the prepositions contained in it, allowing for a single pass in the transformer encoder for that particular token sequence during training. The optimizer used is AdamW with an initial LR for the encoder of $1e-5$ and $1e-3$ for the rest of the model, with a cosine annealing learning scheduler having $T\_max = epochs$ and $eta\_min = 1e-5$.

6.3 Evaluation Metrics

To assess the performance of the SRL models, I used standard evaluation metrics, including precision, recall, and F1-score. These metrics provide a comprehensive view of the model's ability to correctly identify and classify semantic roles.

7 Results

For my analysis, I will break down this section following the main components of my model which more heavily impact the results.

7.1 Encoder

Model	Role		Predicate	
	F1	Loss	F1	Loss
DISTILBERT	61.7	1.592	85.5	0.550
BERT	61.9	1.520	86.8	**0.547**
XML ROB	62.0	1.372	87.0	0.561
XML ROB L CONLL	**69.4**	**1.349**	**91.8**	0.550

Table 1: F1 and loss results on test set with different backbones for UP dataset

I evaluated various backbone pre-trained encoder models, including DISTILBERT[1] from (Sanh et al., 2019), BERT[2] from (Devlin et al., 2019), XML ROBERTA[3] from (Conneau et al., 2019), and XML ROBERTA LARGE finetuned on the CONLL03 dataset[4].

Looking at Table 1, XML ROBERTA finetuned on CONLL03-English outperforms all the other models. Its performance is probably related to its bigger size. However, it is also helped by the named entity recognition task in the CONLL03 dataset (Tjong Kim Sang and De Meulder, 2003) the model was fine-tuned with since it is closely related to the semantic representations we want to achieve.

7.2 Projection & Combination Layers

The results highlight that reducing the final layers parameter count through projection retains strong performance, and hits that with enough data, it might also be used to enhance them while managing overfitting.

In terms of the combination strategy, I will only showcase the gating and soft-attention mechanisms

[1]distilbert/distilbert-base-uncased

[2]google-bert/bert-base-uncased

[3]FacebookAI/xlm-roberta-base

[4]FacebookAI/xlm-roberta-large-finetuned-conll03-english

as they performed better than the other two strategies during my testing.

7.2.1 UP datasest

Model	Role		Predicate	
	F1	Loss	F1	Loss
Gating				
no projection	71.8	1.408	90.7	**0.549**
750	69.9	1.413	91.8	0.551
200	68.1	1.354	91.8	0.555
100	67.8	1.369	91.3	0.554
Soft attention				
no projection	**73.0**	1.488	**92.1**	0.551
750	69.7	1.363	91.9	0.552
200	68.8	**1.349**	90.1	0.550
100	69.4	**1.349**	91.8	0.550

Table 2: Projection layer effect on UP dataset. Bold is overall best, underline is best for that combination strategy

For the UP dataset (Table 2), the Soft Attention method without projection achieves the best F1 score both in role classification and predicate identification. However, a projection size of 100 still maintains competitive F1 scores while also having lower loss values, indicating that parameter reduction is possible without a major drop in performance. Similarly, the Gating method shows solid performance with projections which substantially lower the number of parameters other than the encoder.

7.2.2 NomBank dataset

Model	Role		Predicate	
	F1	Loss	F1	Loss
Gating				
no projection	**78.7**	1.054	86.6	0.500
750	76.3	1.046	86.4	0.498
200	72.6	0.977	85.0	0.502
100	73.1	**0.962**	85.2	0.501
Soft attention				
no projection	76.4	1.116	86.5	**0.496**
750	77.6	1.089	**87.0**	0.497
200	75.0	0.992	85.6	0.499
100	73.6	0.980	85.5	0.502

Table 3: Projection layer effect on NomBank dataset. Bold is overall best, underline is best for that combination strategy

For the NomBank dataset (Table 3), the larger dataset size enables better performance with higher parameter counts, favoring the gating mechanism to the soft attention one. The Gating method without projection achieves the highest F1 score both for predicate identification and role classification. On the other hand, the Soft Attention method reaches the best Predicate and role F1 score with a projection size of 750, showing that projection can enhance performance given sufficient data.

7.3 Argument span identification strategy

Span strategy	role F1 UP	role F1 NOM
Concatenation	72.9	78.2
Top	**73.0**	**78.7**

Table 4: Aggregation strategy for argument span identification effects for UP and NOM datasets, using the best-performing model for each dataset

The results in Table 4 indicate that the 'Top' selection strategy delivers the best Role F1 scores for both UP and NOM datasets. I hypothesize that the 'top' strategy reduces the errors from confusing arguments between roles, which has a greater effect on the overall performance than the cases where the semantic roles are split into multiple spans.

The 'Concatenation' strategy still performs well, though probably thanks to the contained length of the sentences used for evaluating the model. Empirically, the 'Top' strategy works much better for long sequences with many predicates, as hinted in section 3.2, which is helpful to process a long text in a single pass.

7.4 Confidence threshold

Model	role F1	role precision	role recall
DISTILBERT	61.7	59.6	82.3
BERT	61.9	59.7	83.7
XML ROB	62.0	58.6	86.0
XML ROB L CONLL	**69.4**	**68.7**	**86.24**

Table 5: Precision and Recall score for some models

All the results reported so far were obtained using a confidence threshold for the probability of 0.5. However, looking at the Precision and Recall scores in table 5, we can see how these two measures diverge. This is probably the combination of the dataset structure, containing many more negative examples than positive ones for each role, and the loss I use. This made the resulting models very

Model	Confidence	F1	precision	recall
BERT	0.5	61.9	59.7	83.7
	0.75	67.3	71.0	78.5
XML ROB	0.5	62.0	58.6	86.0
	0.75	68.4	70.7	80.7
XML ROB L CONLL	0.5	69.4	68.7	**86.2**
	0.75	**72.5**	**76.6**	82.3

Table 6: Confidence effect on evaluation metrics

Model	F1	ΔF1	precision	recall
Gating				
no projection	74.4	2.6	78.3	83.8
750	73.3	3.4	76.9	**83.9**
200	71.9	3.8	74.9	83.4
100	72.3	**4.5**	75.6	83.1
Soft attention				
no projection	**74.7**	1.7	**80.0**	82.7
750	73.5	3.8	77.3	83.6
200	72.1	3.3	76.1	83.0
100	72.5	3.1	76.7	82.3

Table 7: Final Role classification performance for UP dataset

good at discerning when a word has a role associated with it, whereas they fall behind when the word should not have any role.

To address this, we can higher the confidence threshold needed to assign a role to a word, obtaining a boost in the F1 scores across the board as we can see in Table 6.

Putting into practice this new knowledge, we can see from Tables 7 and 8 how the models using projections gain a lot more performance from this simple modification. Some models even have room for further improvement. This is likely due to their better behavior, which does not saturate the activation functions, allowing for a more meaningful confidence score.

8 Limitations & Future Work

The presented model shows good results in semantic role labeling. However, it has several limitations and areas for future improvements, allowing for better performance and applicability.

For long text sequences, the top selection strategy for argument span identification is needed. However, this strategy may result in the loss of information from arguments composed of multiple spans. This problem probably stems from the dataset it is been trained on, composed of short, self-contained sentences. To address this, future work could explore data augmentation techniques

Model	F1	ΔF1	precision	recall
Gating				
no projection	80.9	2.2	**81.1**	90.3
750	79.5	3.2	79.1	90.7
200	77.3	**4.7**	76.3	**91.1**
100	77.4	4.3	76.9	90.3
Soft attention				
no projection	79.5	3.1	80.0	90.4
750	**81.1**	3.5	**81.1**	90.3
200	78.7	3.7	78.7	90.3
100	77.0	3.4	76.3	90.2

Table 8: Final Role classification performance for NOM dataset

that involve, for example, combining multiple training sentences into a longer one.

In addition, integrating entity resolution into the SRL framework could significantly improve the understanding of the text by resolving ambiguities related to entity references. For future work, I would also work on using a more logic-based role label set, which can abstract from the predicate sense and is more suitable for constructing meaningful knowledge representations to reason on.

In conclusion, while the current model obtains a good result for semantic role labeling, future work focusing on these areas could improve the model's performance and utility for the ultimate goal of creating a formalized representation of the knowledge in a natural language text.

References

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *ArXiv*, abs/1902.10909.

Cognitive-Computation-Group. Bert-based nominal srl. `https://github.com/CogComp/SRL-English`. This repository contains all files created to perform the BERT-based nominal SRL, both using the Nombank dataset and the Ontonotes dataset.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2019. Dependency or span, end-to-end uniform semantic role labeling. *Preprint*, arXiv:1901.05280.

A. Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The nombank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, Massachusetts, USA. Association for Computational Linguistics.

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.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Peng Shi and Jimmy J. Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *ArXiv*, abs/1904.05255.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Universal-Propositions. 2019. Universal proposition banks: Release 1.0. `https://github.com/UniversalPropositions/UP-1.0.git`. Built upon release 1.4 of the Universal Dependency Treebanks.