HW2: Semantic Role Labeling

Bota Duisenbay
La Sapienza University of Rome
duisenbay.1849680@studenti.uniroma1.it

1 Introduction

Semantic Role Labelling is a task in which given a sentence, it identifies various components that could play a semantic roles and classifies them based on the theme of the predicate. For example, in the sentence "Alice asked Bob to play the game", there are two predicates. For the first predicate "asked", "Alice" is the first argument (Performer/Agent), and "Bob" is the second argument (Receiver/Patient), while for the next predicate "play", "Bob" is now the first argument (Performer), and "game" is the second. Generally, it can be as filling the basic event structures such as who did what to who, when, and where.

The SRL task can be divided into 4 sub-tasks as follows: predicate identification and disambiguation, argument identification, and classification. There are several approaches to do it as learning all at once or separately. In case of this work, the focus is on the later: identification and classification of arguments, given predicates as an input. For these subtasks, pre-trained word embedding GloVe with BiLSTM and transformer based BERT model finetuning with BiLSTM are implemented and compared.

2 Dataset

Only English dataset were used for the task. The given data set is given as training and validation sets: 5501 and 1026 sentences, containing words, lemmas, POS tags, dependency heads, dependency relations, predicates(positions and senses for sense disambiguity), and roles as argument labels. A sentence may contain several predicates or no predicate. The list of arguments classes were not known beforehand, therefore, it is collected from labels of both training and validation sets and summaries in the table 1.

2.agent	3.asset	4.attribute
5.beneficiary	6.cause	7.co-agent
8.co-patient	9.co-theme	10.destination
11.experiencer	12extent	13.goal
14.instrument	15.location	16.material
17.patient	18.product	19.purpose
20.recipient	21.result	22.source
23.stimulus	24.theme	25.time
26.topic	27.value	1._

Table 1: Argument classes

2.1 Pre-processing

For argument identification and classification tasks, only words, lemmas and roles are used, while predicates are retrieved from lemmas at predicate position ids in roles. Sentences that contain more than one predicate were duplicated, so that each contains only one predicate and associates labels. For sentences with no predicate indicates, labels were set all to the label "_". This resulted in following change in sets sizes: training 5501 increased to 12641 with duplicates and validation from 1026 to 2553.

Words are converted into lowercase and cleaned by removing special characters, instead of leaving only English and numeric characters because words may contain letters from non-English alphabets.

For the BERT model, the words were tokenized by a BERT tokenizer that could result in dividing a word into sub-words. Besides, additional tokens [CLS] and [SEP] are added to the sentences. Hence, labels and predicates' positions are aligned accordingly using positional embeddings. After the prediction, a post-processing step must also be taken to readjust labels and predicates after decoding.

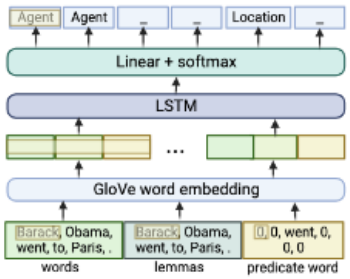

Figure 1: Model with GloVe embedding architecture

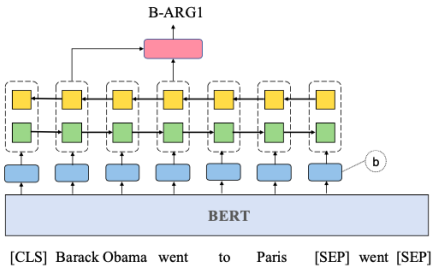

Figure 2: Simple BERT architecture by Shi and Lin (2019)

3 Model

LSTM models are powerful for sequential tasks in NLP. This includes modeling the relationships between predicates and arguments, where SRL is viewed as a sequential argument labeling. Using pretrained embeddings like GloVe to encode input words is found to improve the performance.

One of the weaknesses of the LSTM, as always stated, is the vanishing gradients over long sequences and the inability to parallelize sequential computations.

Transformer-based models overcome these problems and have shown significant improvements in various NLP tasks. The most common pre-trained models like BERT are used as encoders and fine tuned for different tasks during the training. Encoded representations are further passed to Linear/LSTM models.

Two LSTM models with different embedding strategies (GloVe and BERT) were implemented for the task.

3.1 GloVe model

For this model pre-trained GloVe word embedding was used to encode all the inputs separately: words, lemmas and predicate words. All three inputs were concatenated and passed to LSTM layers, followed by dropout, linear, and softmax layers. As a loss function, negative logarithmic likelihood loss was used. The architecture of the model is shown in Figure 1

3.2 Simple BERT

The architecture of the BERT-based model used for this task consists of a finetunable BERT transformer, BiLSTM and linear layers as depicted in the figure 2. Pre-trained models used are BERT Devlin et al. (2018) and ALBERT Lan et al. (2019).

As suggested by Shi and Lin (2019) we encode the sentence together with predicate in predicate-aware manner as follows. Sentence words and a predicate word are tokenized together by BERT tokenizer and enclosed as [CLS] sentence [SEP] predicate [SEP]. Then this is a input to the BERT encoder. From the last hidden layer of its output, we retrieve only a contextualized representation of the sentence (without added predicate encoding) [CLS] sentence [SEP], and concatenate to it the embedding layer of binary predicate indicator of a fixed size. The resulting vectors are fed into the BiLSTM layer. Obtained BiLSTM hidden states of each word concatenated with hidden states of the predicate word, and this goes as an input to the classifier, which is 2 linear layers with softmax layer in between.

4 Experiments and results

4.1 GloVe model

I experimented with different models starting with LSTM and gradually increasing the complexity by stacking another layer and using BiLSTM. The major F1 improvements are caused by an increase in the hidden size: 512 and 256 for LSTM and BiLSTM respectively. The parameters used for the training are that the batch size is 32, and the learning rate is 1e-3 with Adam optimizer and 5 epochs. Glove6B with embedding size 100.

4.2 BERT based model

The parameters used for training BERT based model are summarized in the table 2. Two BERT-based pretrained models were used: ”bert-base-cased” and ”bert-base-uncased” for BERT and ’albert-base-v1’ for ALBERT. ALBERT is a version of BERT with reduced number of parameters

batch size	32
epochs	10
transformer learning rate	5e-5
transformer weight decay	1e-4
learning rate	1e-4
LSTM hidden size	768
MLP hidden size	300

Table 2: Bert based model parameters

	identification	classification
GloVe based		
LSTM	0.6820	0.6334
2 LSTM	0.7022	0.6414
biLSTM	0.8240	0.7851
2 biLSTM	0.8335	0.7870
BERT based		
bert-base-uncased	0.8001	0.6538
bert-base-cased	0.8463	0.7543
albert-base-v1	0.8558	0.7642

Table 3: F1 scores of model variations

for a fast training and less memory consumption, that on top of that over-preforms BERT in various datasets and tasks.

5 Results

The results of these experiments are summarized in the table 3. For the model with GloVe word embedding, using two biLSTM over LSTM and a single BiLSTM has shown the highest results for both identification and classification. As expected, models with BERT encoders have shown a considerable increase in F1 scores. Comparing Albert and Bert base, the Albert model has shown slightly higher performance than Bert.

The confusion matrix for the GloVe-based (2BiLSTM) 3 and BERT-based (albert) models is shown to identify the most confused argument labels. For both plots, the main misclassified labels account for an identification problem ("_"), and bias for "agent" label that is prevalent in the dataset.

6 Conclusion

For argument identification and classification subtasks of SRL, LSTM model with different pretrained word embedding models were tested. LSTM has been proven to be powerful to use for SRL task. Combining it with transformer-based

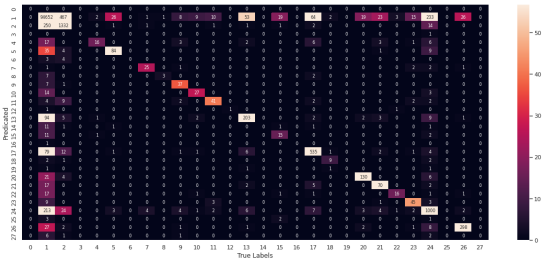

Figure 3: Confusion matrix for GloVe-biLSTM

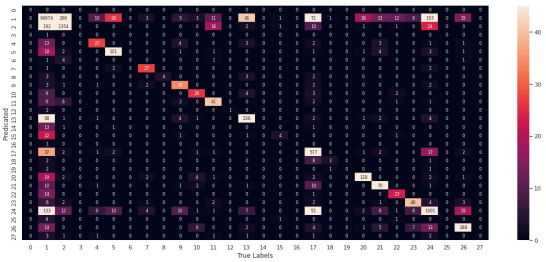

Figure 4: Confusion matrix for ALBERT

word representation leads to even better performance. This happens because the predicate word representation was encoded into the representation of other words in a sentence. For further study, different architectures and predicate encoding techniques can be tested.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Peng Shi and Jimmy Lin. 2019. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.