Frame-Semantic Parsing with BERT Extracted Word and Span Embeddings

Romy Zilkha, Jonathan Schwartz, Jonathan Bofman, Gabriel Clinger

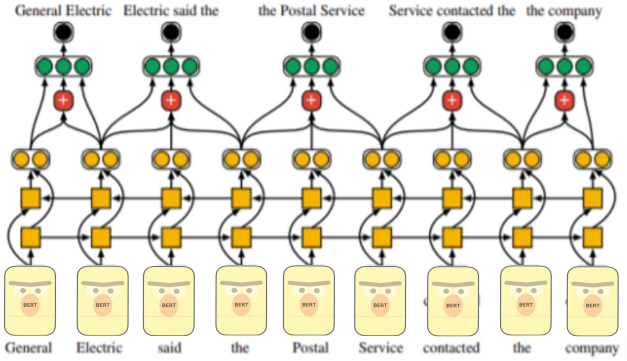
COMS 4995, Columbia University
Professor Bauer

Introduction

Inspired by the Frame-Semantic Parsing paper (open-SESAME) that uses a Syntactic Scaffolding approach for Frame-Semantic Parsing. We build upon the existing scaffolding approach by incorporating attention mechanism into two distinct embedding phases (words and spans). We decided to use the huggingface implementation of BERT in order to integrate the attention mechanism into the existing argument identification model.

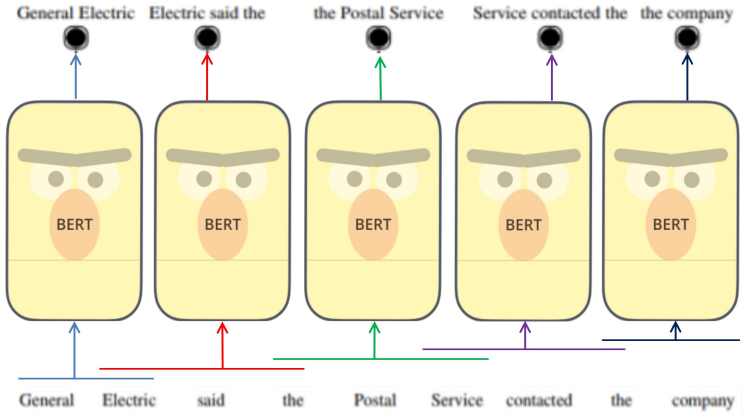
Method I – Word Embeddings

In this method we decided to use the original span embeddings creation phase (using bidirectional-RNNs for generating the embedded span representation). Our modification to this process is changing the word (token) embeddings, which are extracted from the BERT model. Using BERT, we were able to create new word representations which are created with respect to the most relevant words (to that token) in the sentence. We used two vector dimensionality reduction techniques in order to reduce our embedding vectors from [1,762] to [1,64] due to computational restrictions (the large dimension of the BERT token embeddings could not be handled by the rest of the Frame Semantic model).
The first technique that we used is PCA (Principal Component Analysis) and the second is Autoencoders where we built a new model and trained it to reduce our embedding vectors.



Method II – Span Embeddings

In this method we ignored the span embeddings creation phase (using the bi-RNN's) and instead created our own span representations using BERT. We extracted the entire sentence (or in our case, span) representation from BERT, reduced the dimensionality, and fed that embedding to the scoring function (next part of the pipeline). Using this method, we eliminated the need for word embeddings completely since we are extracting the entire span representation. Similar to the first method, we used both PCA and Autoencoders for the dimensionality reduction. However, due to computational restrictions caused by the very large number of possible spans, we were not able to finish training the model.



Results

In this section, we will focus only on the first method (token embeddings). Comparing the two dimensionality reduction techniques - PCA and Autoencoders, we were able to see better results for the Autoencoders technique in precision and very similar results between the techniques in recall and f1. However, the original argid model provided better results compared to both PCA and Autoencoders.

	Precision			Recall			F1		
	wpr	upr	lpr	wre	ure	lre	wf1	uf1	lf1
PCA	0.49358	0.51077	0.48688	0.26166	0.19663	0.19891	0.34201	0.28395	0.28244
Autoenconers	0.53057	0.58062	0.56802	0.12986	0.18472	0.19391	0.20866	0.28027	0.28912
Original	0.63739	0.67262	0.58487	0.45778	0.54611	0.49486	0.53286	0.60280	0.53611

Conclusions

Looking at the results of the token embeddings method, we believe that main factor for the lower results compared to the original argid originates from the loss of information due to the dimensionality reduction (using both PCA and Autoencoders). However, we were very happy to see that the results were not too far away from the baseline (original open-sesame argid). Looking ahead, we believe that we should examine both training the autoencoder model for the dimensionality reduction on the objective of the entire argument identification model and secondly, optimize the overall code in order to be able to use GPUs for faster computations. Regarding the span embeddings (Method II), we believe that if we could solve the computational restrictions, we would be able to show improved results.

References

1. Frame-Semantic Parsing With Softmax-Margin Segmental RNNs and a Syntactic Scaffold paper - https://arxiv.org/pdf/1706.09528.pdf
2. open-SESAME - https://github.com/swabhs/open-sesame
3. Extracting BERT word and span representation - https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/#3-extracting-embeddings
4. Huggingface BERT model - https://huggingface.co/transformers/model_doc/bert.html
5. Autoencoders - https://towardsdatascience.com/dimensionality-reduction-pca-versus-autoencoders-338fcaf3297d