Compositional Pre-Training for Semantic Parsing with BERT

Arnaud Autef, Simon Hagege

Stanford University

Overview

We study **Transformer-based** Encoder Decoder architectures on a **semantic parsing** task: **Geoquery**. We investigate the effects of a **BERT** Encoder and **data recombination** methods to augment the dataset. Since the report, our latest result with BERT achieves **0.70 strict accuracy** without copying mechanism!

Introduction

Semantic parsing: conversion of natural language utterances to logical forms

Transformer neural architecture based on Multi-head Attention and FeedForward sublayers, that we use in an Encoder Decoder framewoork as in [3]

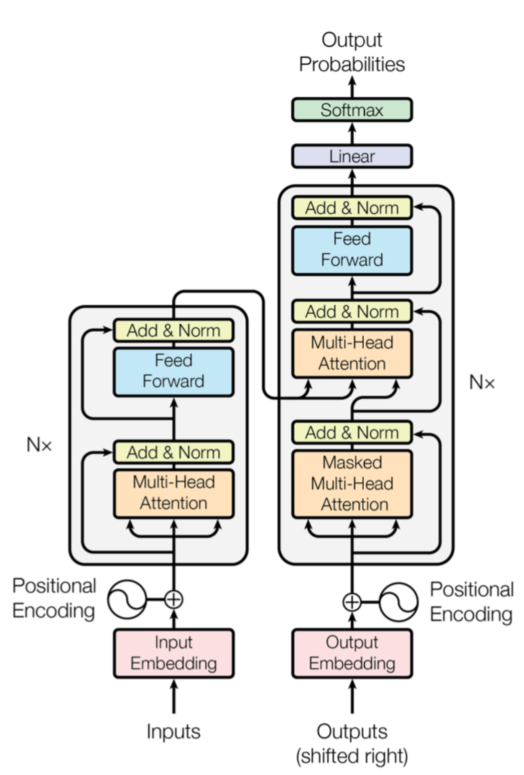

Figure: Transformer architecture - figure from [3]

BERT Transformer pre-trained to learn a language model through two tasks [1]:

- *Masked Language Model*
- *Next sequence prediction*

Data Recombination Data augmentation technique introduced in [2], generates new data using synchronous context-free grammars (SCFG):

- *Entity*: abstracting entities with their types, based on predicates in the logical form (e.g `stateid`)
- *Nesting*: abstracting both entities and whole phrases with their types
- *Concat-k*: combining k sentences into a single

Approach

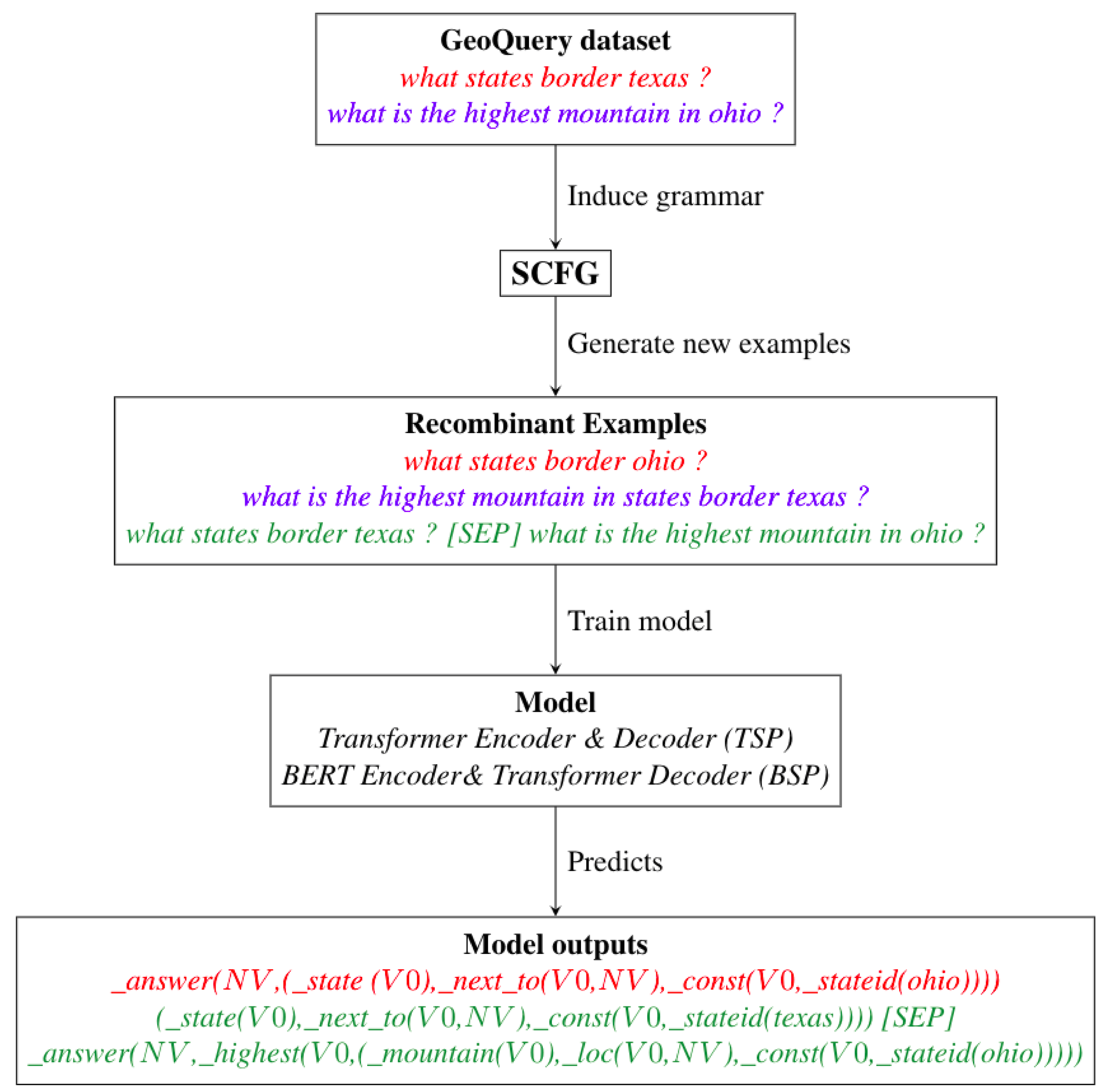

Figure: Overview of our model

TSP Transformer semantic parser: Encoder-Decoder with N stacked transformer layers

BSP BERT semantic parser: TSP with BERT as the Encoder

Evaluation metrics

Strict evaluation Exact match between output queries strings $(\hat{y}_i)_{1\leq i\leq n}$ and the target strings $(y_i)_{1\leq i\leq n}$

$$\text{Strict} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(y_i = \hat{y}_i)$$

Jaccard evaluation Match between the sets of characters of a model output string and the corresponding target query string

$$\frac{1}{n}\sum_{i=1}^{n} \text{Jac}(y_i, \hat{y}_i)$$

where

$$\text{Jac}(y_i, \hat{y}_i) = \frac{\#(Y_i \cap \hat{Y}_i)}{\#(Y_i \cup \hat{Y}_i)}$$

$$Y_i = set(y_i) \quad \hat{Y}_i = set(\hat{y}_i)$$

Knowledge-based evaluation (KB) Interpreting the outputs of our model $\hat{y}_i$ as SQL queries and compute the share of model outputs that both

- Correspond to a valid query to the database
- Yield to an identical output to the target query y_i

Large models

Data Recombination methods

Recombination	Strict	KB	Jaccard	$\text{Jac}_{\text{strict}}$
No recombination	0.136	0.169	0.891	0.225
Entity	**0.189**	**0.259**	**0.902**	**0.282**
Nesting	**0.189**	0.241	0.900	**0.282**
Concat-2	0.157	0.192	0.892	0.204

- Size $d_{model} = 512$, $N = 6$ transformer sublayers, Adam optimizer with fixed learning rate
- Two-steps approach: training - fine tuning with / without data recombination examples
- Entity best single-shot recombination method

BSP - TSP comparison

Model	Strict	KB	Jaccard	$\text{Jac}_{\text{strict}}$
TSP fixed	0.189	0.259	0.902	0.282
TSP adaptive	0.086	0.144	0.859	0.118
BSP adaptive	**0.293**	**0.425**	**0.944**	**0.457**
Relative Impr.	*55.4%*	*64.1%*	*4.7%*	*62.1%*
Absolute Impr.	*0.104*	*0.166*	*0.042*	*0.175*

- Tests with adaptive (increasing then decreasing) learning rate
- Recombination method used: *Entity*
- Poor performances, better results with BSP

Models retained

Model	Strict	KB	Jaccard	$\text{Jac}_{\text{strict}}$
Shallow BSP	**0.704**	0.*	**0.978**	**0.793**
Shallow TSP	0.657	**0.630***	0.977	0.768
Baseline TSP	0.471	-	0.950	0.579
Relative improv.	*49.5%*	-	*2.95%*	*37.0%*
Absolute improv.	*0.233*	-	*0.028*	*0.214*

- Size $d_{model} = 128$, $N = 2$ Transformer sublayers, Adam optimizer with adaptive learning rate
- Combination of *Entity*, *Nesting* and *Concat-k*
- Only the last layer of BERT is fine-tuned, other pre-trained layers are frozen

Multi-Head Attention

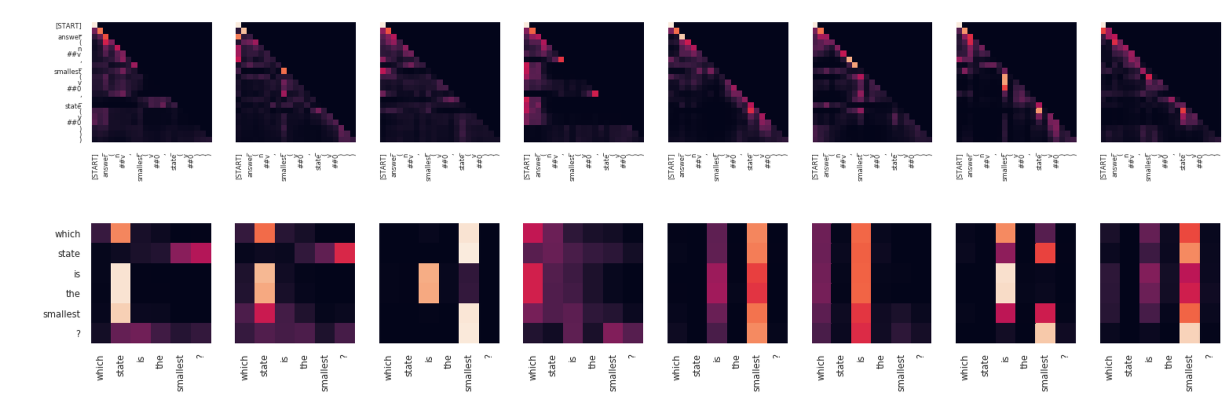

Figure: Visualization of self-attention activation on Encoder and Decoder first layer

Conclusion

- Strong performances of Transformers, improvements with BERT encoder, even with shallow architectures
- Best results obtained with a **shallow architecture**, due to the limited number of training examples

Next steps to improve the results:

- Push **data recombination further**.
- Implement a **copying mechanism** for the Decoder, as in [2] with RNNs
- Models architecture engineering: Decoder dimensions, freezing fewer BSP layers

References

[1] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018).

[2] Robin Jia and Percy Liang. “Data Recombination for Neural Semantic Parsing”. In: *Association for Computational Linguistics (ACL)*. 2016.

[3] Ashish Vaswani et al. “Attention is All you Need”. In: *Annual Conference on Neural Information Processing Systems 2017*.

Stanford University