Extracting Syntactic Trees from NMT Encoder Self-Attentions

David Mareček, Rudolf Rosa

Institute of Formal and Applied Linguistics, Charles University, Prague

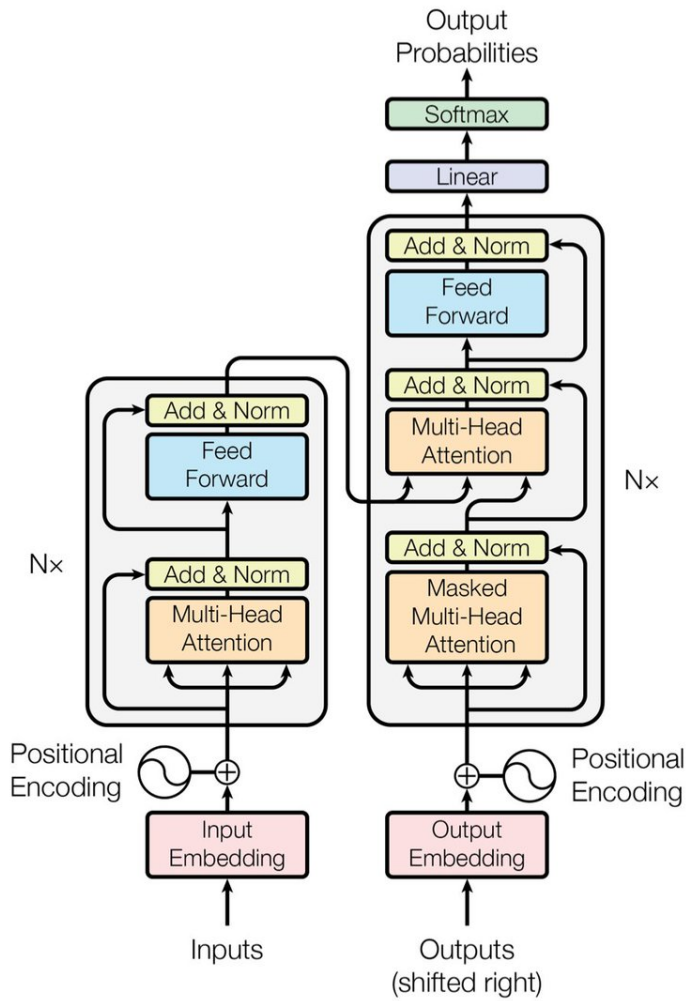
BlackboxNLP Workshop, EMNLP, Brussels

November 1st, 2018

Current state of the art in Machine Translation

Neural Machine Translation

- e.g. the Transformer architecture (Vaswani et al. 2017)
- translation quality comarable with human translators (WMT18 news-translation shared taks)



Is there any latent syntax learned inside?

Current end-to-end NMT systems are very hard to interpret.

- plenty of high-dimensional vectors
- the self-attention mechanism mixes everything together

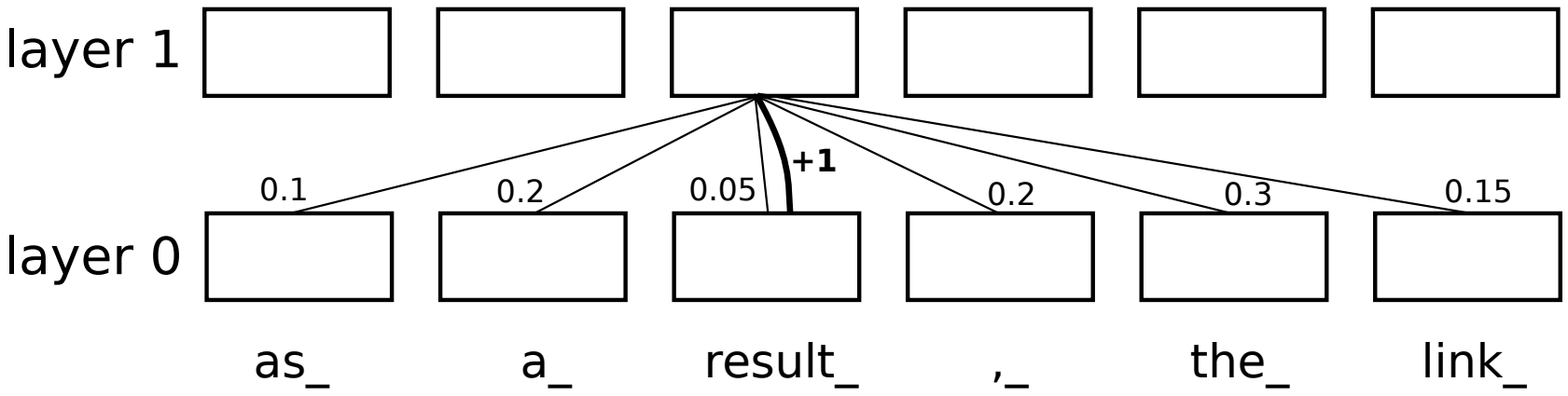
Is there any latent syntax learned inside?

This work:

- Use state-of-the art NMT system (Transformer).
- Extract and analyze its encoder self-attentions.
- Based on the self-attentions, induce parse trees and compare them to manually annotated treebanks.

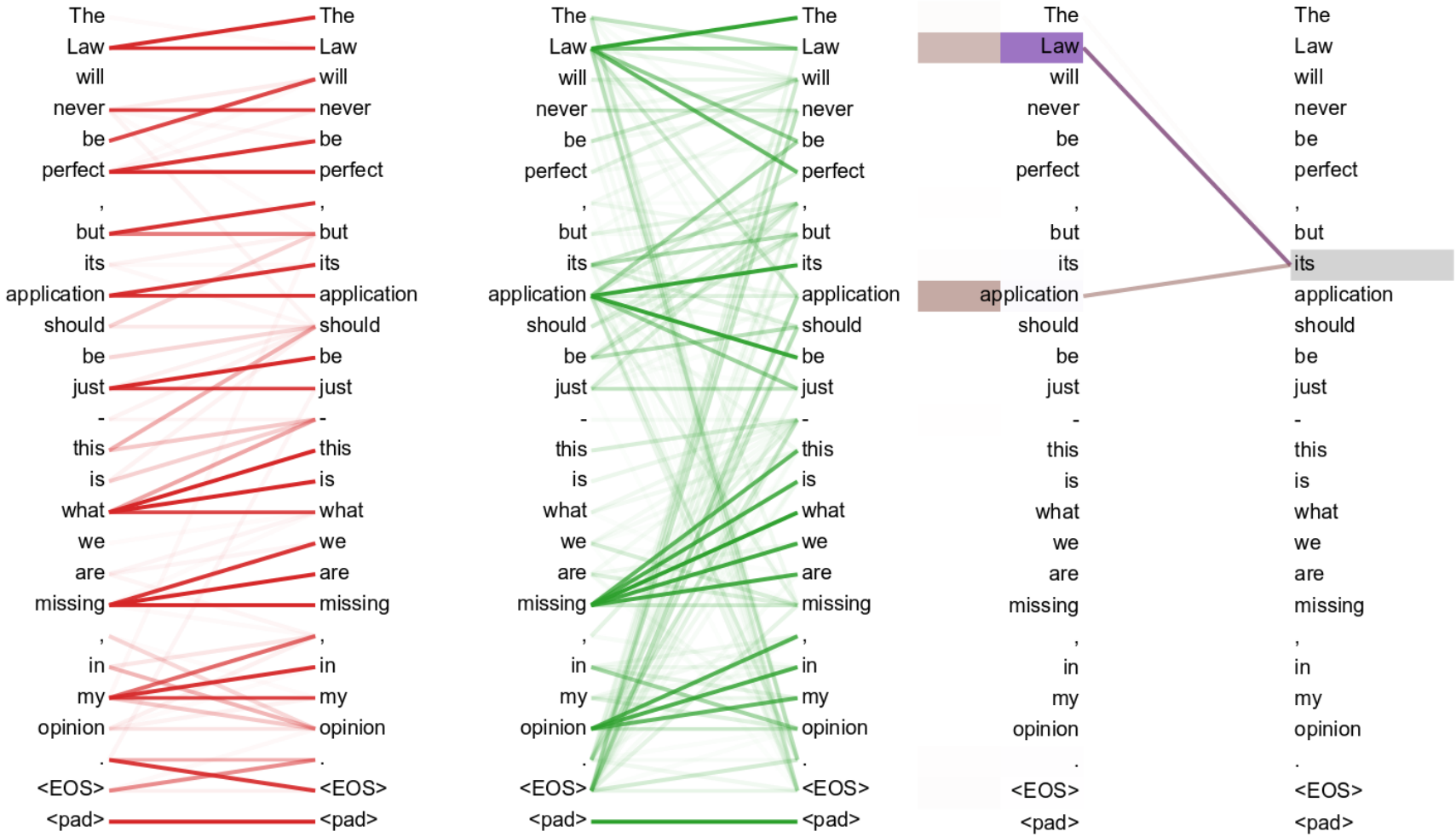
Self-Attention mechanism

- For each position (source word), the self-attention mechanism can look at all positions in the previous layer
- Residual connections boost the attention to the same position in the previous layer
- Typically 16 heads, each with different weights



Heads with syntactic properties

Original paper self-attention visualization (Vaswani, 2017):

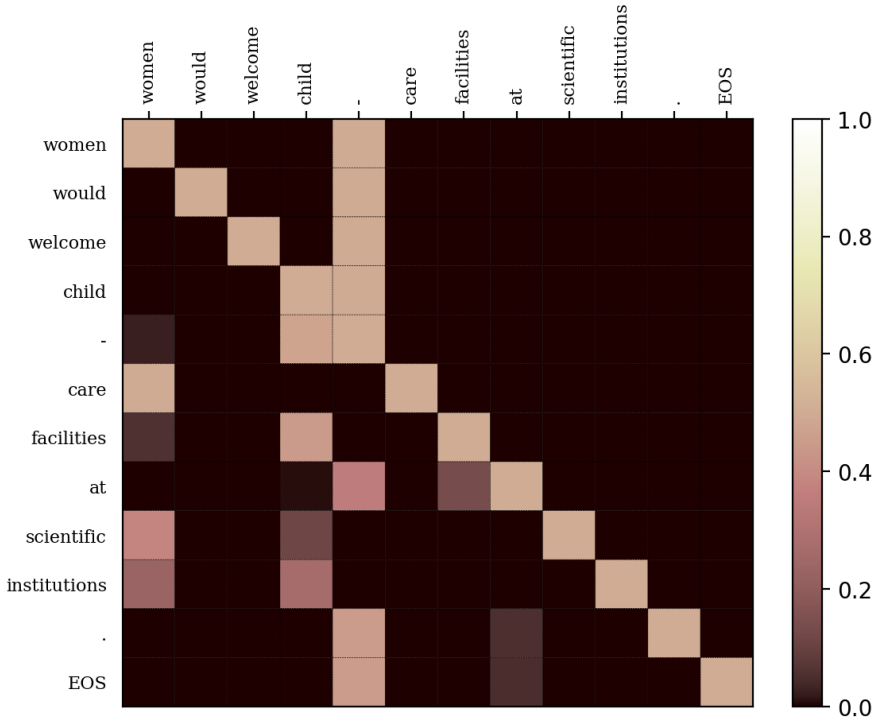


Experiments

- We trained Transformer in its standard setting (6 layers, 16 heads, ...)
- Europarl parallel corpus
  - English→Czech
  - English→German
  - English→French
  - English→Finnish
- BPE, dictionary size 100k
- We translated 1000 test sentences and extract encoder self-attentions weights

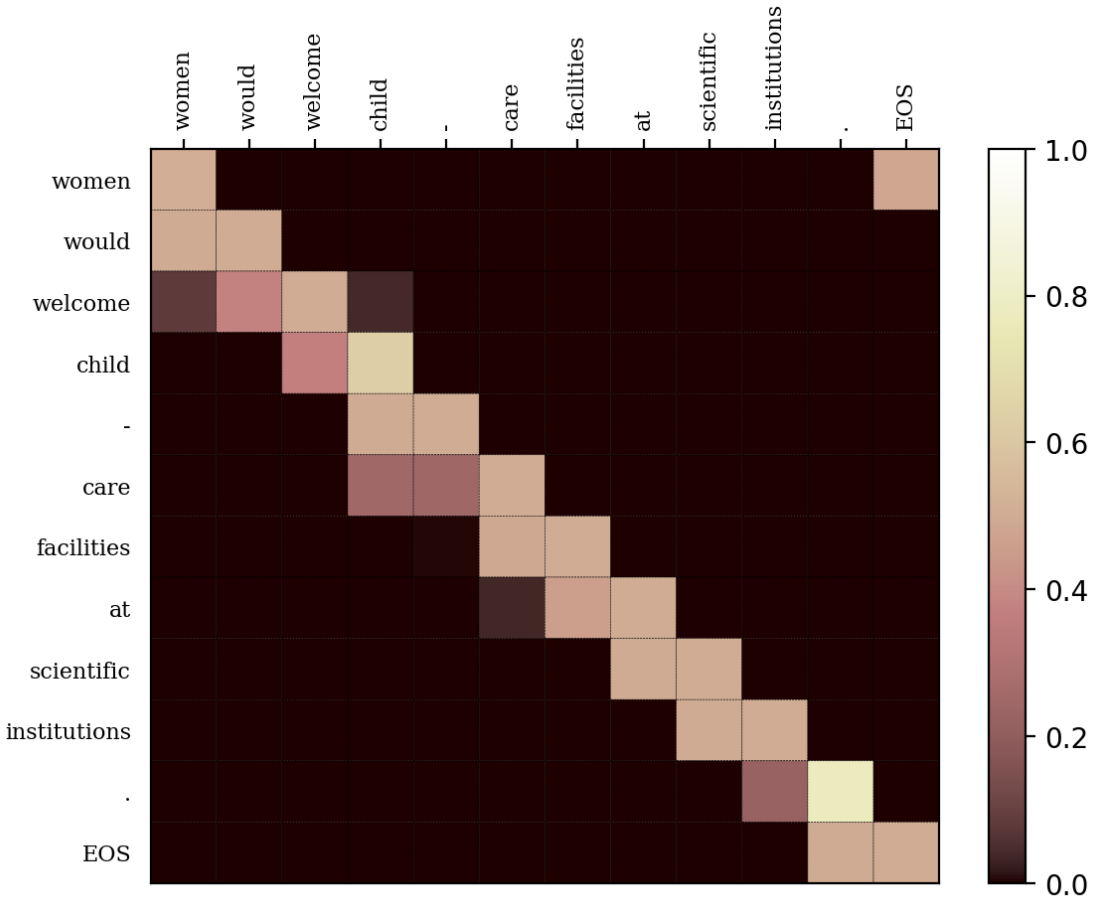
Visualisation of one self-attention head

- square matrix
- each row shows attention distribution of one word across all the words in the previous layer
- attention to the same position (diagonal) is strong due to residual connections
- we have 6 layers x 16 heads = 96 heads in total



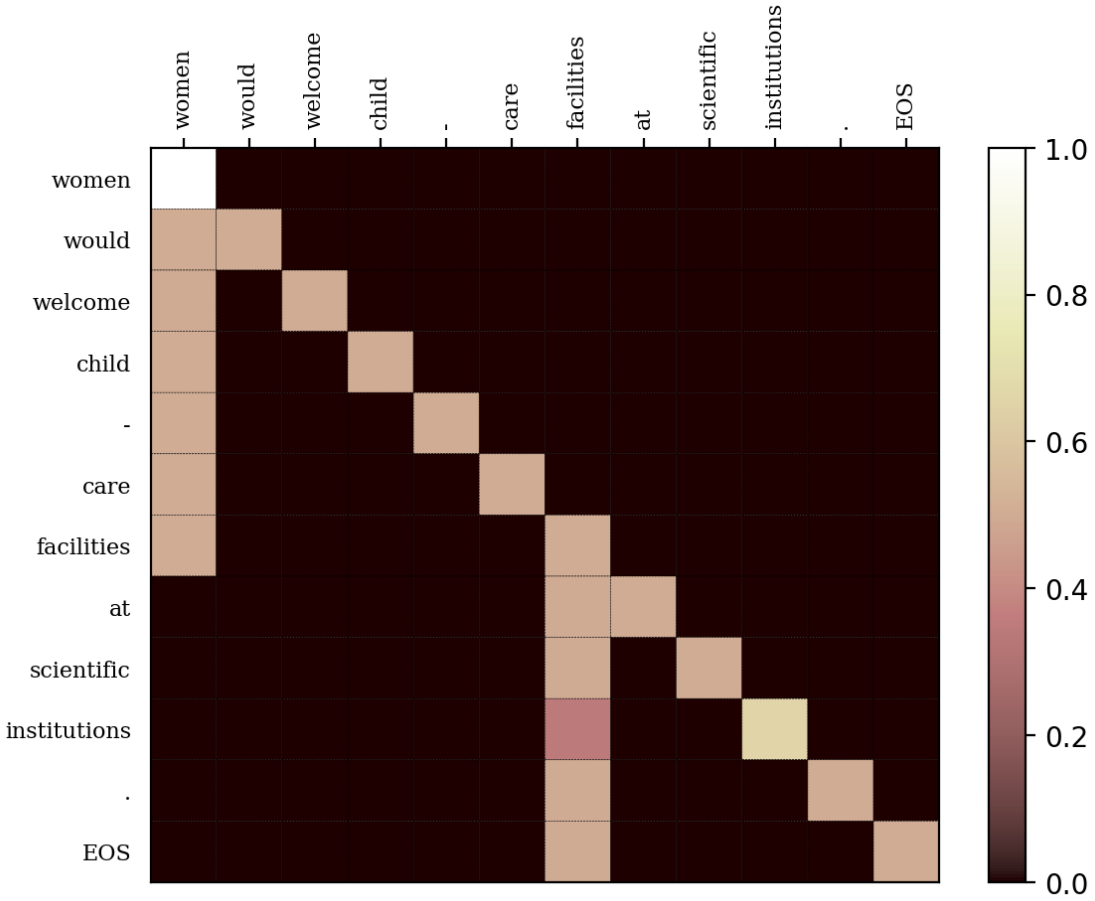
Analysis of attention heads

- $\sim$ 15% of all heads (mainly in the first layer) look at the previous, at the same, or at the next word.



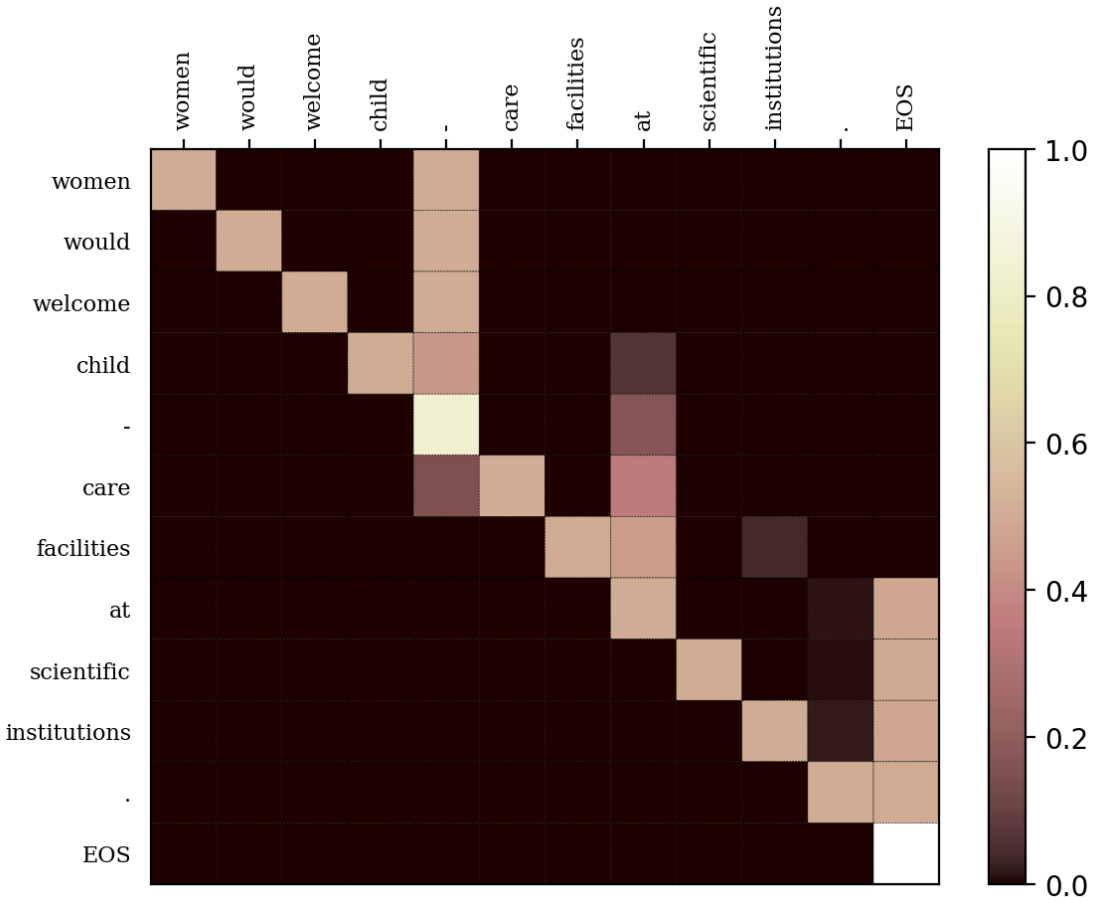
Analysis of attention heads

- $\sim$ 15% of all heads form continuous phrases looking at their first word.



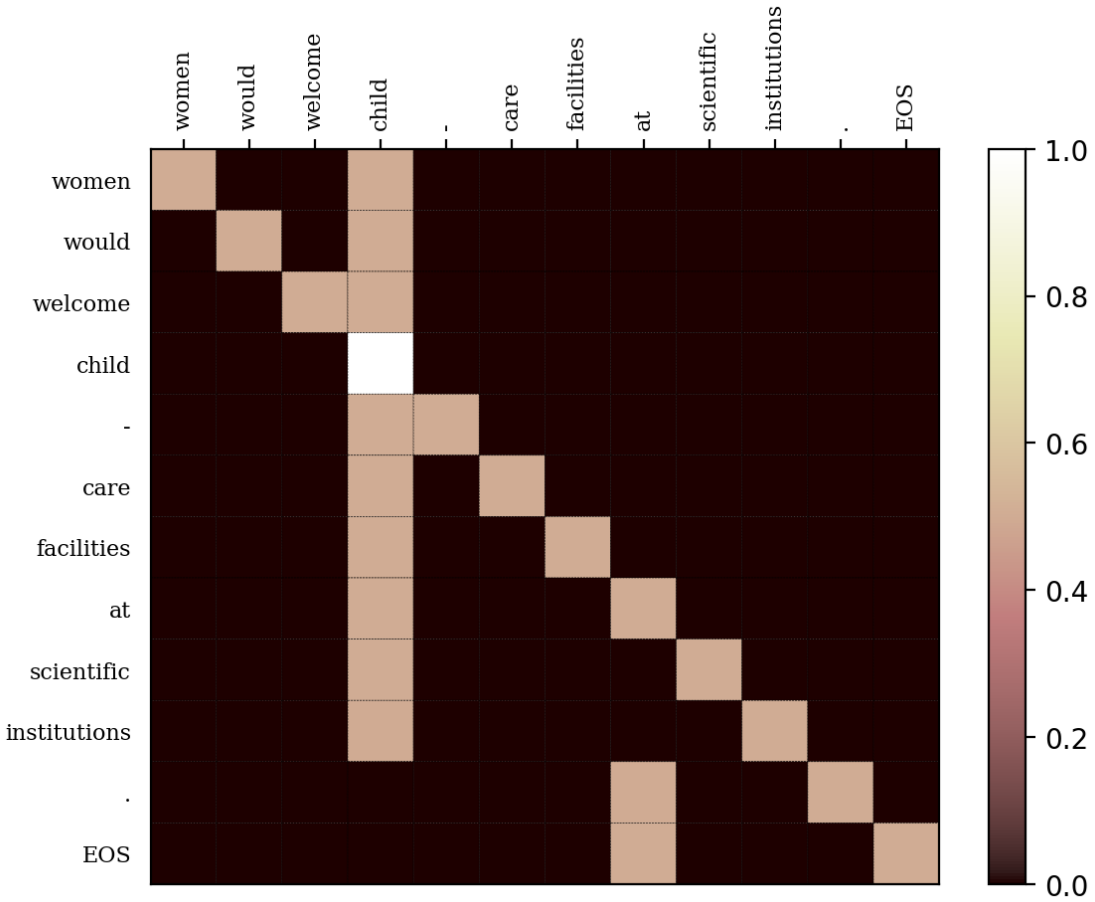
Analysis of attention heads

- $\sim$ 15% of all heads form continuous phrases looking at their last word.



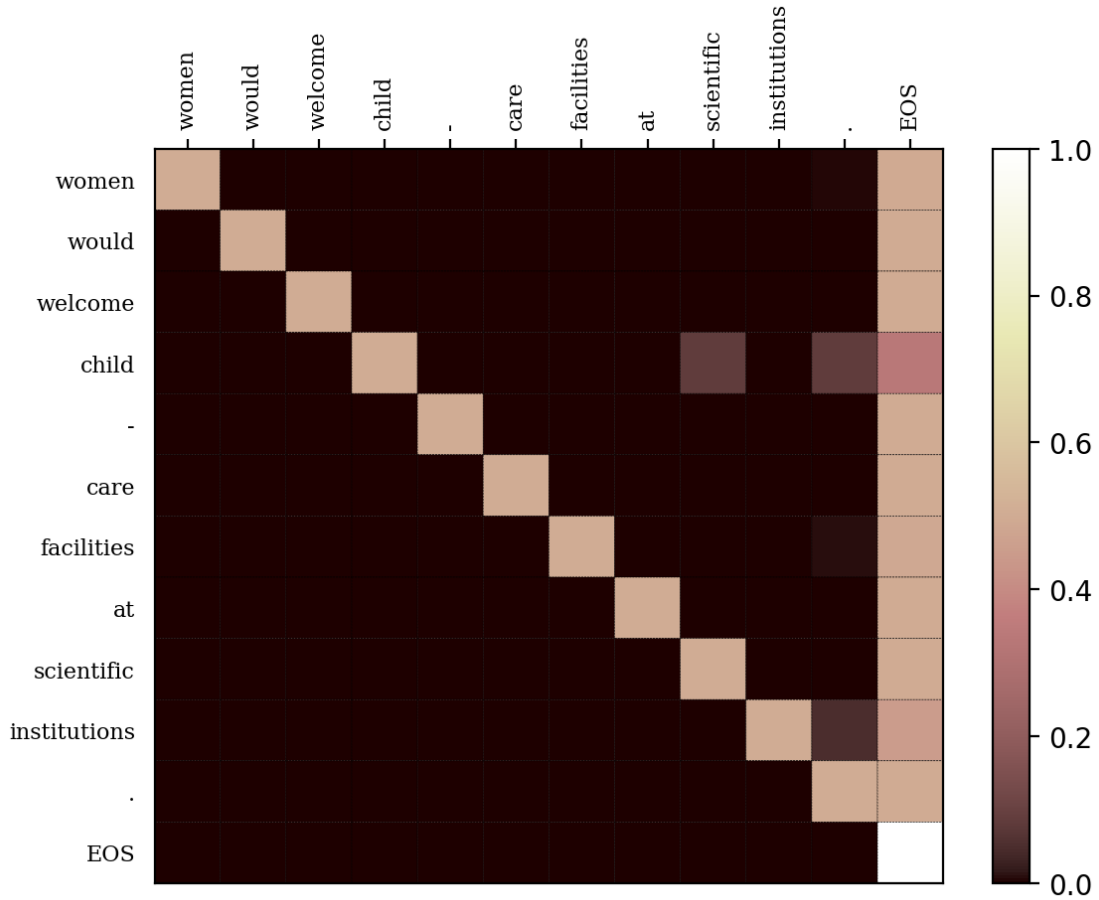
Analysis of attention heads

- $\sim$ 20% of all heads form continuous phrases looking at another word.



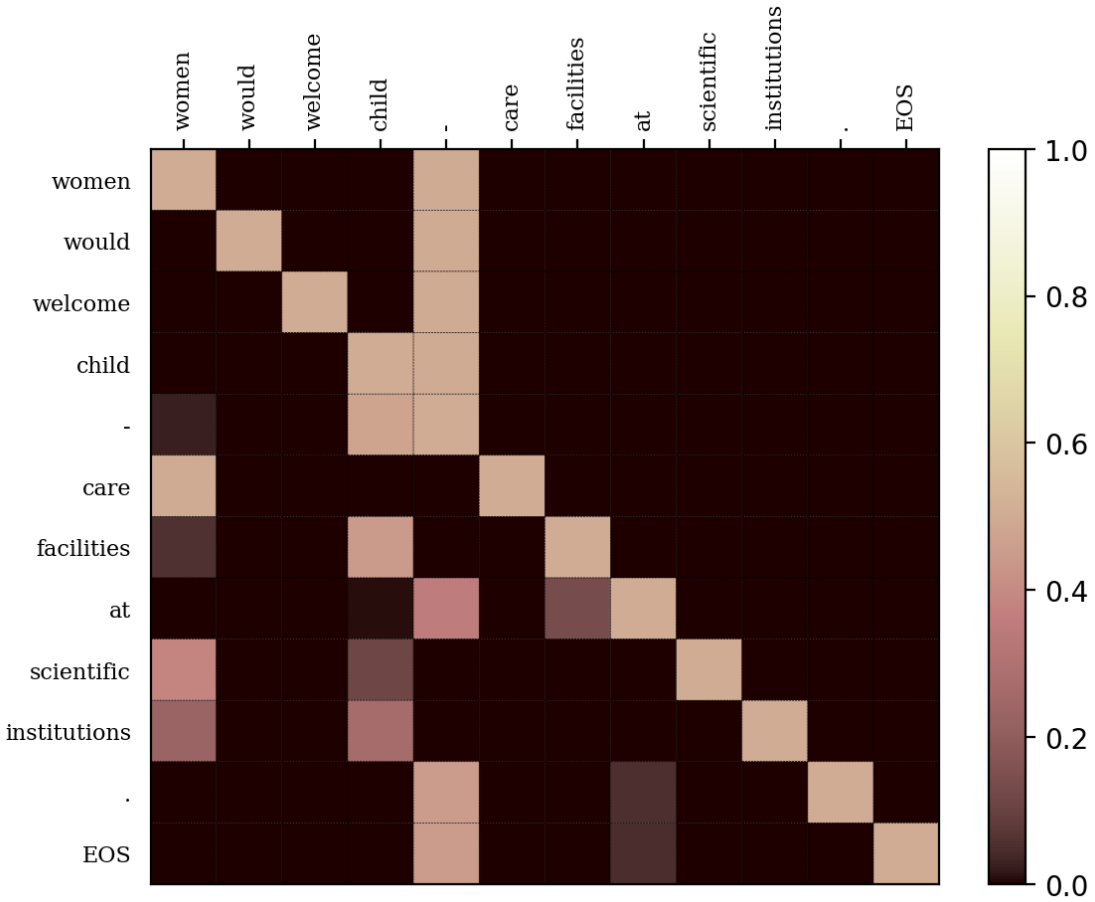
Analysis of attention heads

- In $\sim$ 5% of all heads, all the words look at the end of the sentence.



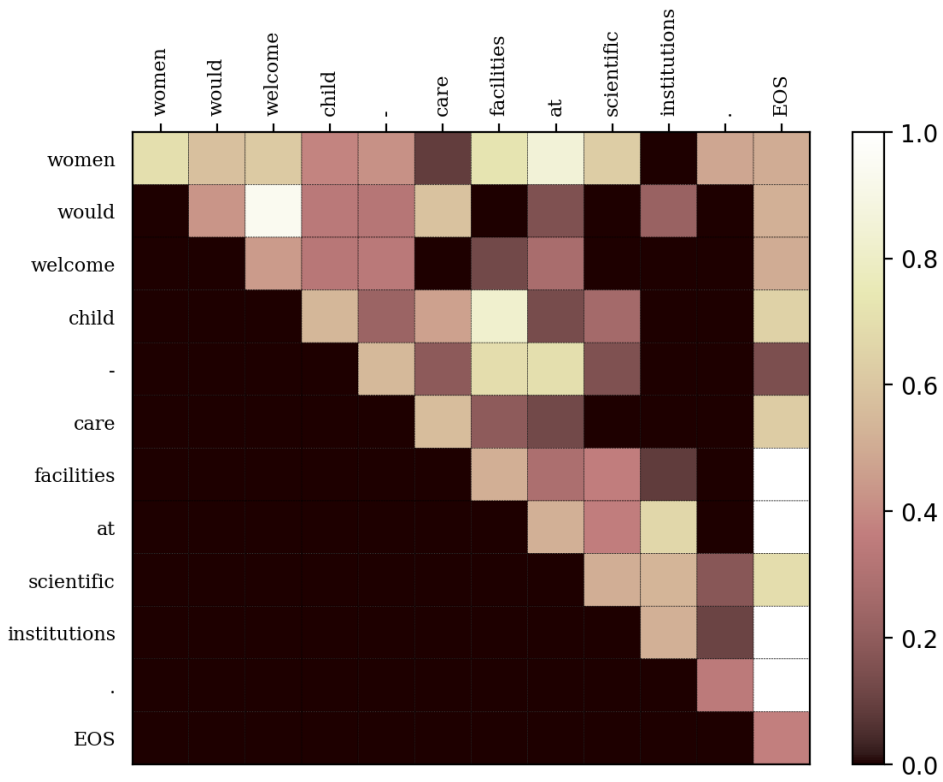
Analysis of attention heads

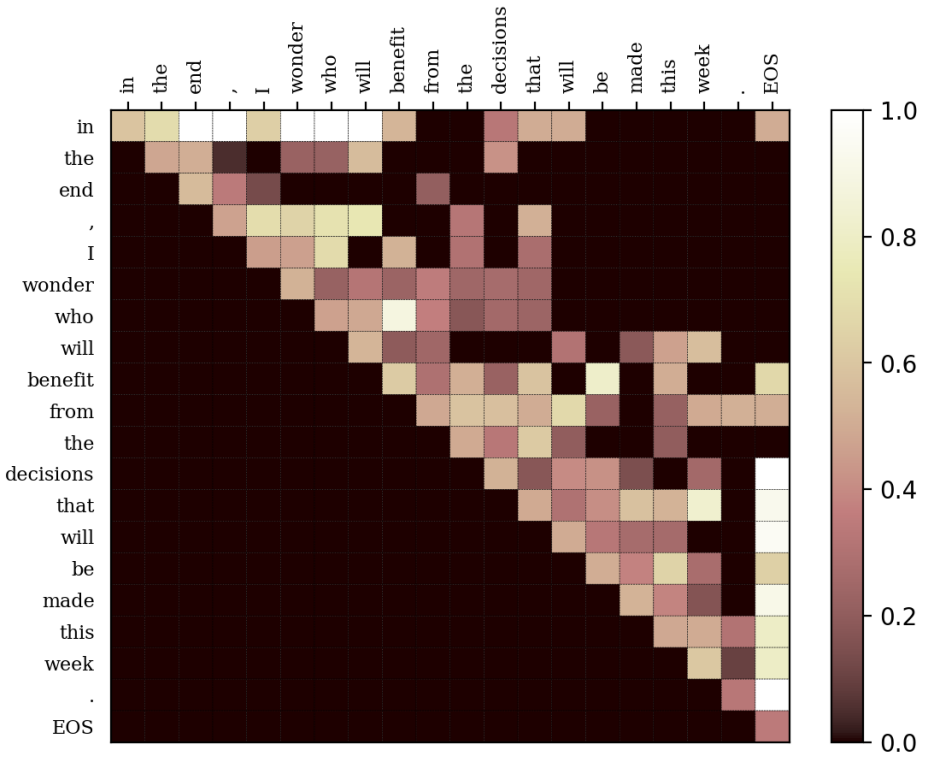
- Rest $\sim$ 30% of heads are more complicated.



From self-attentions to trees

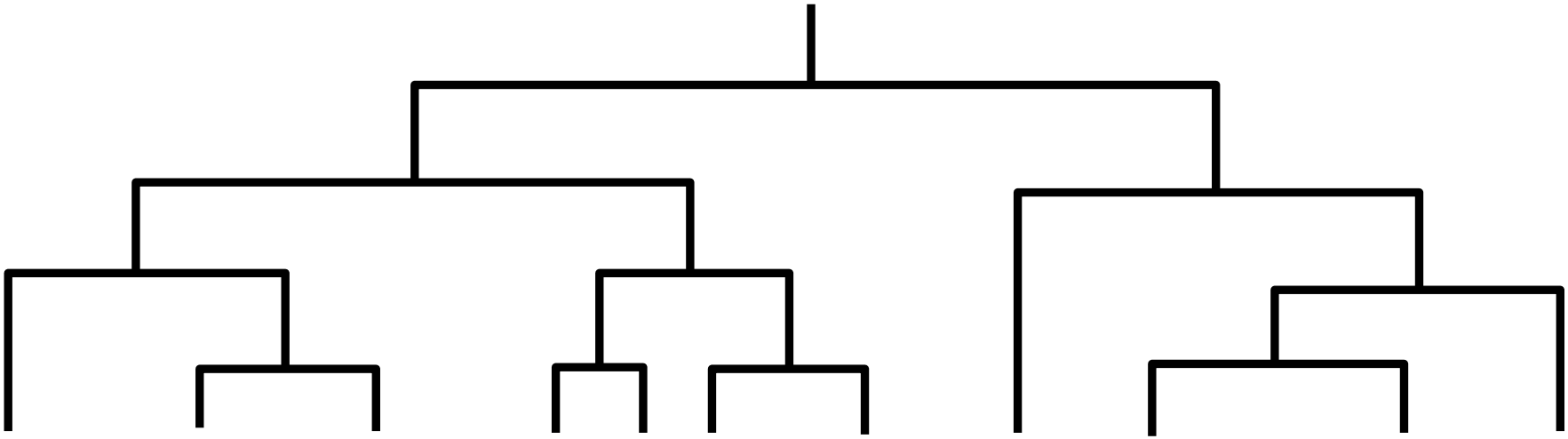
- We know that majority of heads form "continuous phrases".
- We collect all such continuous phrases **across all the heads and layers** in the encoder.
- For each possible phrase, we compute its score reflecting its frequency.





Constituency parsing using CKY

- We find the best phrase-tree (comprising the best scoring phrases) using CKY algorithm.
- Probability of joining two phrases together and probability of the two respective subtrees is weighted 1:1



Evaluation

1. Against Stanford parser trained on Penn Treebank
   - compute precision, recall, and F1 score on individual brackets
2. Against UDPipe parser trained on UniversalDependencies English treebank
   - convert dependency structures into phrase structures
   - compute the ratio of valid (non-crossing) brackets
   - compute precision, recall, F1 score
   - recall is very high since the phrase trees converted from dependency trees are very flat

Results

F1 scores

method	Stanford PTB	Universal Dependencies
EN left branching	11.1%	37.4%
EN right branching	12.1%	38.5%
EN left ballanced	22.0%	56.1%
EN right ballanced	26.4%	58.8%
EN-CS self-attentions	30.9%	63.8%
EN-DE self-attentions	30.3%	62.8%
EN-FR self-attentions	28.4%	60.8%
EN-FI self-attentions	30.6%	64.8%

Conclusions

We found that:

- In a majority of attention heads, the whole continuous phrases look at the same word in the previous layer.
- The phrases are often in agreement with the linguistic theories.
- If we extract such phrases and build a constituency tree, it is better than left/right branching and ballanced binary baselines.
- The encoder very well separates longer sentences into clauses. It easily finds commas and conjunctions.

Some kind of syntax is probably learned, but is quite different for the one annotated in treebanks.

Conclusions

Thank you for your attention!