Analyzing Multi-Head Self-Attention

MICKUS, Timothee

General layout of this presentation:

1. Self-attentive networks & why we want to analyze them
2. Math tools for analyzing SANNS: *Layer-wise Relevance Propagation*, head confidence & L_0 based pruning
3. Voita et al.'s findings

NB:

- SANN = Self attentional neural networks, viz. "Transformer architectures"

Self-attentive networks & how to analyze them

Analyzing SANN

Actually, we talked about this before

'Cutting edge technology':

- BERT (Devlin et al., 2018) on GLUE Wang et al. (2019) for NLU / embedding
  Compare Infersent (Conneau et al., 2017) (76) vs. BERT Large (89)
- Vanilla Transformer (Vaswani et al., 2017) with NMT
  Significant gains in terms of BLEU
- GPT (Radford, 2018) / Transformer XL (Dai et al., 2019) for LM
  Perplexity down to under 1 bit per character using Transformer XL
- some, like BERT (Devlin et al., 2018) or GPT-2 (Radford et al., 2019) basically turn out to be broadly applicable to almost any NLP task

But we don't really know how, or why that works.

Analyzing SANN

Attention (?)

Some counter-intuitive facts:

- BERT embeddings basically behave like sentence representations rather than word embeddings (sensitive to order, not usable for formal analogy, wdely used in NLU setups)
- Attention over a single item devolves into a linear transformation:

  $$\mathit{Softmax}\left(q \cdot K^T\right) V = \frac{e^{q \cdot k_i}}{\sum_{k_j \in K} e^{q \cdot k_j}} V$$

  Since $K = \{k_i\} = \{k_j\}$, this simplifies to:

  $$\mathit{Softmax}\left(q \cdot K^T\right) V = \frac{e^{q \cdot k}}{e^{q \cdot k}} V = V$$

  which means that SANN don't behave similarly wrt. sequences and singletons
- SANN like the Transformer use multi-heads rather than vanilla attention, which likewise devolves into a linear transformation

Analyzing SANN

Not all hope is lost

Some work has been done in order to understand the behavior of SANNs, eg.: Raganato and Tiedemann (2018) probe encoder representations as computed by NMT vanilla Transformers (Vaswani et al., 2017) from English to 7 target languages.

- Attention weight visualisations:
  four patterns shared across languages: to the word itself, to the previous word, to the next word, to the last token in the sentence.
- Induced tree structures:
  Attention weights for a given head can be seen as a weighted graph. The induced tree is tested on an UD treebank, given gold segmentation, tokenization and root. Average UAS F1-score is similar to a left-branching baseline, best scores are comparable to a right-branching baseline.
- Probing sequence labeling tasks: POS-tag, Chunking, NER, and Semantic tagging
  POS best scores are found in the first 3 layers, Chunking in the first 4, NER in layers layers $3 \sim 5$, Semantic tagging in the last 3 layers. Precision ranges from 70% to 90%, error rate from 2% to 50%
- Transfer learning capacities:
  Using the EN→DE encoder for EN→TR boosts performances by 1 BLEU

Analyzing SANN

Where we are

- Over the last few years, some progress has been made on the understanding of the mechanical behavior of SANNs:
  - Raganato and Tiedemann (2018) showed that NMT Transformers encoders don't do everything out of the box, can't really be said to do 'syntax' & require layers and resources to tackle semantics.
  - Other works suggest interesting capacities: Voita et al. (2018) study how NMT Transformers exploit contextual information for anaphora resolution
  - Tang, Sennrich, and Nivre (2018) show attention mechanisms focus more on ambiguous token
- All of these works suggest interesting properties ("contextualization" is not just an empty word)
- But generally, there's no formal framework and the observations stem from *ad-hoc* tests, which makes it difficult to separate what's to be blamed on the probing task from what's the actual capacities of the SANN.

Tools for analyzing SANNs

LRP

LRP: Layer-wise relevance Propagation

- Works like Raganato and Tiedemann (2018), Voita et al. (2018), and Tang, Sennrich, and Nivre (2018) mostly consists of on-the-spot tests. If we are to study SANNs systematically, we may require a more formal approach.
- **Layer-wise Relevance Propagation** (LRP) is a mathematical tool to define how much a given neuron contributes to a given output.
  - If a NN θ classifies an item y to a class C, it assigns a score: $Pr(x \in C|\theta)$.
  - If we consider solely the last layer L of dimension d, we can write this probability as a function $f : \mathbb{R}^d \rightarrow \mathbb{R}^1$,
  - we can approximate f as a weighted sum of each neuron: $f(x) \approx \sum_d L_d$.
  - This gives us a score of how **relevant** each neuron L_d of the **layer** L is to the prediction $x \in C$
  - we can use the same mechanism to **propagate** this score to previous layers, by computing how relevant each neuron in the previous layer L^{-1} is to a given neuron L_d in layer L
  - if we treat the input as the first layer of computation, we may blame the classification on specific parts of the input

LRP

From pixels to translation

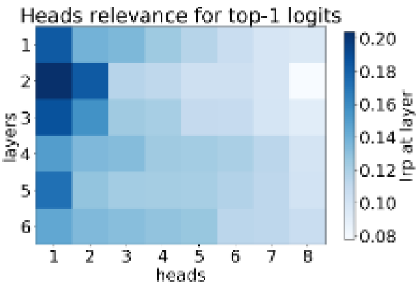
LRP is a fairly new technique, but it has been very quickly applied to NLP also.

- The tool was first proposed in Bach et al. (2015); in the original setup it is used for image classification. The general idea was to estimate how each pixel of the input contributes to the overall classification; cf. also Binder et al. (2016) for dealing with input normalization.
- The first use of LRP in NLP is that of Ding et al. (2017). Ding et al. adapt LRP to the seq2seq NMT model of Bahdanau, Cho, and Bengio (2014)
- Voita et al. (2019) (today's paper) apply LRP to Transformers, and focus on how the multi-head attention mechanism works.

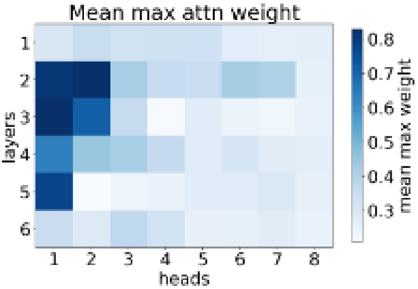
Confidence

How certain is your head?

- Voita et al. (2019) define a head's confidence as the average of its maximum attention weight, excluding the <eos> symbol.
- this, along with LRP, allows us to distinguish which heads are actually used:



(a) LRP



(b) confidence

Pruning heads

How to drop heads: relaxed L_0 penalty

- Technique borrowed from Louizos, Welling, and Kingma (2018), using a variation of the reparametrization trick (Kingma and Welling, 2013)
- the idea is to encourage the model to "turn off" heads, using a L_0 loss criterion and specific scalar gates.
  - The multi-head attention concatenation is re-written to include one scalar gate g_h per head H_h : $(\bigoplus_h g_h \cdot H_h) \cdot W^O$
  - one wants to minimize the L_0 loss to maximize the number of gates set to 0: $L_0(g_1, \ldots, g_h) = \sum_h (1 - [[g_h = 0]])$, but it's not differentiable
  - so the indicator function is replaced with a Hard Concrete Distribution ϕ_h that assigns the most of its mass to 0 and 1.
  - Which gives the regularization term : $L_C(\phi) = \sum_h (1 - P(g_h = 0|\phi_h))$
- Final training objective is thus $L(\theta, \phi) = L_{xent}(\theta, \phi) + L_C(\phi)$, with $L_{xent}(\theta, \phi)$ the original cross-entropy loss
- as the model does converge to solutions where gates are either 1 or 0, at test time the head H_h is ignored iff. $P(g_h = 0|\phi_h) > P(g_h = 1|\phi_h)$; so the model can be treated as a Transformer with fewer heads.

Less heads, same performances!

What can your head do?

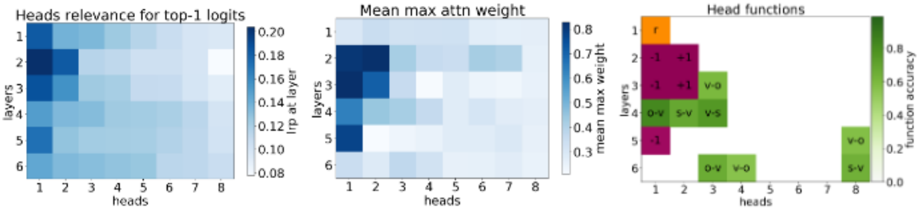
Identifying possible functions

- Voita et al. (2019) suggest heads might do three possible things; they study these :
    1. track relative position words (adjacency patterns in Raganato and Tiedemann (2018)): a head is deemed positional if 90% of the time the maximum attention weight is assigned to a specific relative position
    2. track syntactic relations: like Raganato and Tiedemann (2018), they study whether weight is assigned to an item in a dependency relation (nsubj, dobj, amod, advmod). Heads that have an accuracy of 10% higher than the baseline of predicting the most frequent relative position are deemed to track syntactic relations
    3. track rare words.
- Many heads don't really follow any of these patterns.

What does your head do?

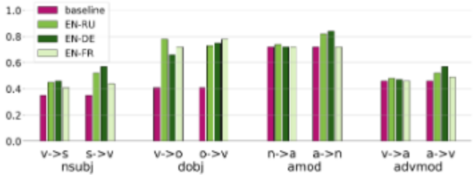
Studying the functions of relevant heads

- Roughly speaking, the relevant heads have identifiable functions:



(a) LRP (b) confidence (c) head functions

- The only head that tracks rare words is only detected with LRP.
- Syntactic dependencies are not equally easily identifiable:



Pruning heads

In the encoder: In terms of BLEU

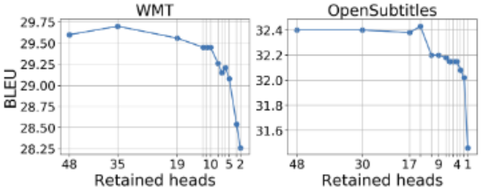


Figure 7: BLEU score as a function of number of retained encoder heads (EN-RU). Regularization applied by fine-tuning trained model.

- When all heads in a given layer are pruned, the residual connections ensures that some information is passed on; if all heads are pruned, the encoder devolves into an FFN
- We see that even with a very aggressive pruning in the encoder self-attention, BLEU only drops by at most 1.5

Pruning heads

In the encoder: In detail

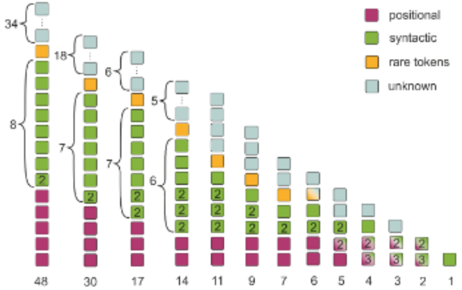


Figure 8: Functions of encoder heads retained after pruning. Each column represents all remaining heads after varying amount of pruning (EN-RU; Subtitles).

- The heads that are pruned first are those that have less clear roles

Pruning heads

Pruning all attentions: In terms of BLEU

	attention heads (e/d/d-e)	BLEU from trained	BLEU from scratch
WMT, 2.5m			
baseline	48/48/48	**29.6**	
sparse heads	14/31/30	29.62	29.47
	12/21/25	29.36	28.95
	8/13/15	29.06	28.56
	5/9/12	28.90	28.41
OpenSubtitles, 6m			
baseline	48/48/48	**32.4**	
sparse heads	27/31/46	32.24	32.23
	13/17/31	32.23	31.98
	6/9/13	32.27	31.84

Table 2: BLEU scores for gates in all attentions, EN-RU. Number of attention heads is provided in the following order: encoder self-attention, decoder self-attention, decoder-encoder attention.

- the same pattern can be seen: even fairly aggressive pruning does not deteriorate BLEU scores by much
- these results hold even when training the model from scratch

Pruning heads

Pruning all attentions: In detail

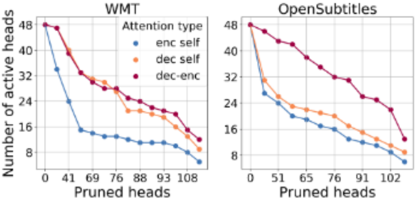


Figure 9: Number of active heads of different attention type for models with different sparsity rate

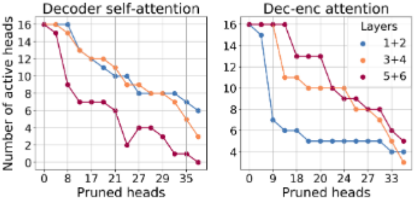


Figure 10: Number of active heads in different layers of the decoder for models with different sparsity rate (EN-RU, WMT)

- Clean distinction between context- and self-attention across models: context-attention preferred over self-attention (in WMT, note the effects of longer input)
- Opposite behavior of layers in decoder self- and context-attention: bottom layers do LM, top layers condition on source

In conclusion...

Conclusion

- LRP can tell apart useless heads from useful ones
- We can highlight the roles of individual heads
  - some heads exhibit recognizable behaviors: relative positional marking, syntactic contextualization, rare word attention
  - after pruning, heads mostly exhibit one of these recognizable behaviors
- using a relaxed L_0 penalty, two thirds of the heads can be removed while not loosing more than 1 BLEU point, suggesting that SANNs do not use their parameters optimally.

References I

Bach, Sebastian et al. (2015). “On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation”. In: *PLoS ONE* 10.7, e0130140. doi: 10.1371/journal.pone.0130140. url: http://dx.doi.org/10.1371%2Fjournal.pone.0130140.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *CoRR* abs/1409.0473. arXiv: 1409.0473. url: http://arxiv.org/abs/1409.0473.

Binder, Alexander et al. (2016). “Layer-wise Relevance Propagation for Neural Networks with Local Renormalization Layers”. In: vol. 9887. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 63–71. doi: 10.1007/978-3-319-44781-0_8.

Conneau, Alexis et al. (2017). “Supervised Learning of Universal Sentence Representations from Natural Language Inference Data”. In: *CoRR* abs/1705.02364. arXiv: 1705.02364. url: http://arxiv.org/abs/1705.02364.

Dai, Zihang et al. (2019). “Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context”. In: *CoRR* abs/1901.02860. arXiv: 1901.02860. url: http://arxiv.org/abs/1901.02860.

References II

Devlin, Jacob et al. (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805. arXiv: 1810.04805. url: http://arxiv.org/abs/1810.04805.

Ding, Yanzhuo et al. (2017). "Visualizing and Understanding Neural Machine Translation". In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, pp. 1150–1159. doi: 10.18653/v1/P17-1106. url: https://www.aclweb.org/anthology/P17-1106.

Kingma, Diederik P and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *arXiv e-prints*, arXiv:1312.6114, arXiv:1312.6114. arXiv: 1312.6114 [stat.ML].

Louizos, Christos, Max Welling, and Diederik P. Kingma (2018). "Learning Sparse Neural Networks through L_0 Regularization". In: *International Conference on Learning Representations*. url: https://openreview.net/forum?id=H1Y8hhg0b.

Radford, Alec (2018). "Improving Language Understanding by Generative Pre-Training". In:

Radford, Alec et al. (2019). "Language Models are Unsupervised Multitask Learners". In:

References III

Raganato, Alessandro and Jörg Tiedemann (2018). "An Analysis of Encoder Representations in Transformer-Based Machine Translation". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, pp. 287–297. url: https://www.aclweb.org/anthology/W18-5431.

Tang, Gongbo, Rico Sennrich, and Joakim Nivre (2018). "An Analysis of Attention Mechanisms: The Case of Word Sense Disambiguation in Neural Machine Translation". In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, pp. 26–35. url: https://www.aclweb.org/anthology/W18-6304.

Vaswani, Ashish et al. (2017). "Attention Is All You Need". In: *CoRR* abs/1706.03762. arXiv: 1706.03762. url: http://arxiv.org/abs/1706.03762.

Voita, Elena et al. (2018). "Context-Aware Neural Machine Translation Learns Anaphora Resolution". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 1264–1274. url: https://www.aclweb.org/anthology/P18-1117.

References IV

Voita, Elena et al. (2019). “Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned”. In: *CoRR* abs/1905.09418. arXiv: 1905.09418. url: http://arxiv.org/abs/1905.09418.

Wang, Alex et al. (2019). “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding”. In: In the Proceedings of ICLR.