Investigation into the Functionality of Key, Query, Value Sub-modules of a Transformer

NYU DATA SCIENCE

Ziyi Xie and Letian Jiang, Instructed by Naomi Saphra

Introduction

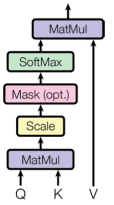
The attention mechanism contains three major components: Key (K), Query (Q), and Value (V) (Vaswani et al. 2017) . Those three components/submodules are used to calculate the Scaled Dot-Product Attention scores.

We will use two methods to analyze how limiting the access of K,Q,V to linguistic information at different layers can affect the overall efficiency of training and performance of a transformer model.

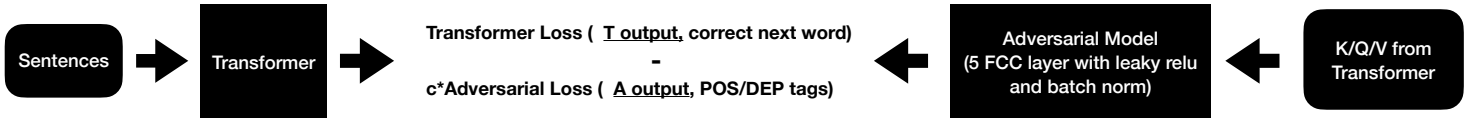
Model, Data & Evaluation

Transformer Model: 5 layer 5 head transformer encoder, linear decoder
Data: Wikitext2 (Logan et al. 2020) ~2.5 M masked word prediction pairs
Task: Next word prediction
Evaluation: Cross entropy loss, accuracy

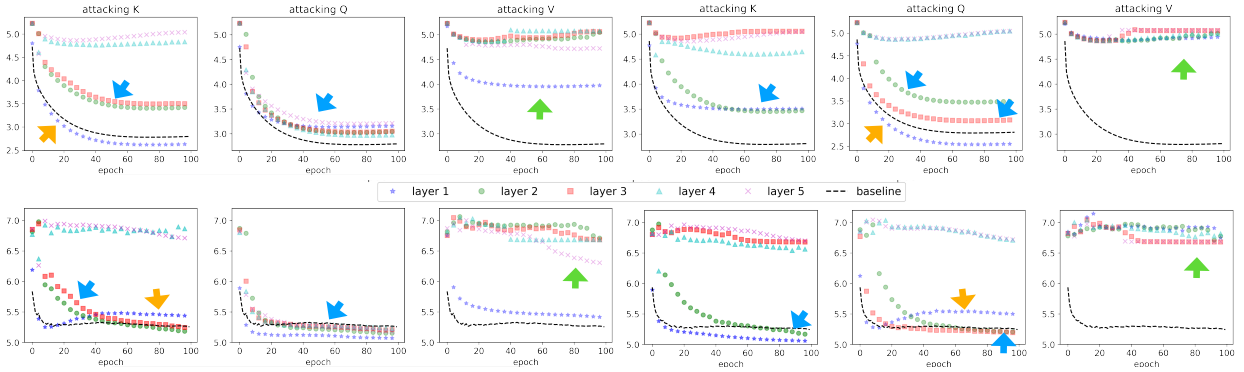
Scaled Dot-Product Attention



Method 1: Adversarial Removal of Linguistic Information (Yaroslav et al. 2019)



Remove **Part of Speech** | Remove **Sentence Dependency**



Findings form loss plot

Better train loss results in overfitting

Worse train loss but more robust model

Attacking V keeps train and evaluation loss high

Discussion

❓ Transformer is able to recover from attacking K and Q, but is not able to recover from attacking V

🧠 V is the last element being multiplied to produce the attention score, therefore more vulnerable

❓ Some similarity between the training plots of attacking Q in POS and V in DEP (vice versa)

🧠 K and Q are interchangeable in Scaled Dot-Product Attention

Method 2: Rewinding & Freezing Submodule Weights

Experiment 1: Rewinding module weights and Criticality:

The robustness characteristics of Transformer submodule to parameter perturbation is tested by rewinding model weights to a convex combination. Convex combination (Chatterji et al. 2020) is defined as:

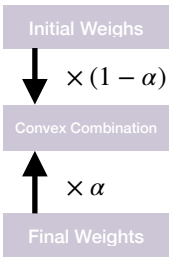
$$\theta^{\alpha_n}_{n,att} = (1-\alpha_n)\theta^{0}_{n,att} + \alpha_n \theta^{f}_{n,att}, \quad \alpha_n \in [0,1]$$

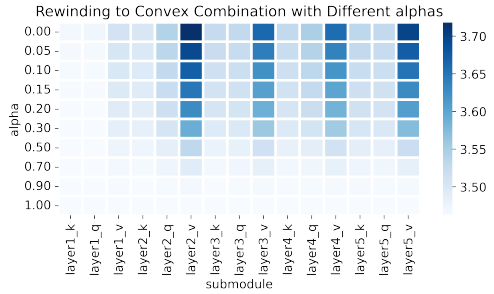
where $\theta^{0}_{n,att}$ and $\theta^{f}_{n,att}$ are the initial and final weight of self-attention component "att" at n-th layer. The evaluation metric **criticality** score (Zhang et al. 2019) is defined as:

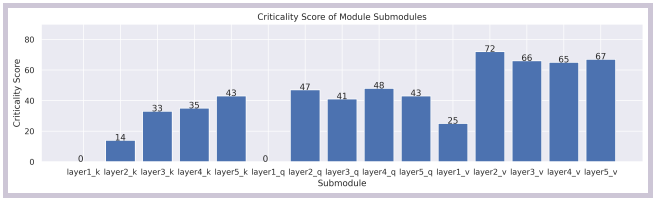
$$\text{Criti}_n = \min \alpha_n \quad \text{s.t. Loss}\left(\text{model with } \theta^{f}_{n,att}\right) - \text{Loss}\left(\text{model with } \theta^{\alpha_n}_{n,att}\right) < \epsilon$$

Criticality score describes the minimum α to maintain the model performance drop within threshold ϵ, which is set to be 1% of the final loss in this experiment. A higher criticality score indicates the submodule information is important during the evaluation.

Observations:
- Layer 1 has smaller criticality scores among all layers
- V (value) component in attention has larger criticality scores among all components
- Upper encoder-attention layers are more important than lower encoder-attention layers





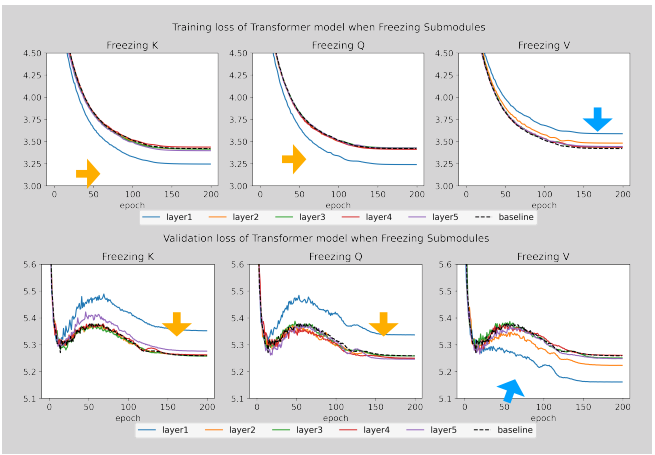


Experiment 2: Freezing Model Weights and Performance Curve:

In this method, self-attention weights of one submodule is frozen at a time thought the training: we set a submodule weights to be equal to initialization weight and keep gradient to be zero all the time.

The model training curve and validation curve are shown in the right figures.

Observations:
- Freezing layer 1 has the most performance perturbation: freezing K and Q causes overfitting but freezing V causes underfitting.
- Information can be still learned if we freeze most of other submodules



Conclusion:

- Control information in different layers can change model robustness
- V is important in and limiting information on K and Q can be recovered
- Layer 1 is has small criticality score but causes huge perturbation in learning process (method 1 and freezing), one possible explanation is that layer1 learns little about contextual information and is not important during the evaluation

Future Work

- More linguistic tasks or properties
- Look at other parts of the Transformer: feed-forward, decoder self-attention, encoder-decoder attention
- Include uncertainty from random initialization

Acknowledgements

We would like to sincerely thank Naomi Saphra for guidance, inspiration and mentorship made in this project. We would also like to thank Julia Kempe, Najoung Kim, Elena Sizikova, and Wenda Zhou for their help in the CDS Capstone Course.