Paying attention to attention

Anonymous ACL submission

Abstract

This paper explores perspectives on attention (Bahdanau et al., 2016), a mechanism to process individual components of a sequence as a function of other components of the sequence. This paper explores attention as a method to create context-aware feature representations, explain model outputs, and route information within models.

1 Introduction

The attention mechanism was first described in (Bahdanau et al., 2016), where it was used to perform machine translation. Other works built on the attention mechanism to build state-of-the-art models to perform a variety of tasks including machine translation (Vaswani et al., 2017), text summarization (Liu and Lapata, 2019; Lin et al., 2018), text generation (Wiseman et al., 2017; Puduppully et al., 2019), and image captioning (Anderson et al., 2018). We will focus on attention through a few perspectives inspired by common uses in the reviewed papers. From the reviewed works, we observe few common perspectives of and uses of attention: a method to create contextualized feature representations as in transformers (Vaswani et al., 2017), a method to explain how input tokens contribute to outputs using attention maps, and finally a method to route input information, similar to methods seen in Switch Transformers (Fedus et al., 2022) and "mixture-of-experts" models (Shazeer et al., 2017; Zhang et al., 2024b).

2 Methodology

This section outlines the methods used to collect papers for review in this meta-analysis. To collect papers for the meta-analysis on the Attention mechanism in machine learning, I used ArXiv and Google Scholar to search for key terms like "attention", and further specifying queries like "additive attention," "scaled dot-product attention," and "mixture-of-experts" after encountering those terms in review. I prioritized highly cited papers, especially foundational works like (Bahdanau et al., 2016) and (Vaswani et al., 2017). I also considered some papers from the Seed42AI dataset on Huggingface (Lab, 2024), which provided recent works published in top conferences like CVPR and ACL. However, most of the papers reviewed were sourced from Google Scholar, or were cited in papers from those sourced using Google Scholar.

For each paper reviewed, I considered a few attributes:

1. Attention Type (e.g. self-attention, cross-attention, etc.)
2. Task Domain (e.g. text generation, text summarization, interpretability, etc.)
3. Use of Attention (e.g. create contextualized feature representations, interpret results, route tokens)
4. Key Takeaways (e.g. key results of the paper, observed from the abstract)

The model tasks among papers were fairly equally distributed between model tasks (text generation, text summarization), with a smaller minority focusing on interpretability. The majority of papers focused on attention's use as a method to create contextualized features, while some of the interpretability works focused on attention maps as a way to interpret the relation between model outputs and inputs.

3 Background

3.1 What is Attention?

In the original attention formulation in (Bahdanau et al., 2016), attention is computed using the following function:

$$\text{Attention}(q, k, v) = v^\top \tanh(W_q q + W_k k_i) \tag{1}$$
$$\text{where} \tag{2}$$
$$W_q \in \mathbb{R}^{n \times n} \tag{3}$$
$$W_k \in \mathbb{R}^{n \times 2n} \tag{4}$$
$$v_a \in \mathbb{R}^n \tag{5}$$

This form of attention is referred to as **additive attention**, because the linear combination of the query and key vectors is obtained as the sum of linear projections on the query and key vectors.

Another formulation of attention, called **scaled dot-product attention**, was introduced in (Vaswani et al., 2017). The attention mechanism in their work is a function that takes, as input, 3 vectors, $Q, K \in \mathbb{R}^{d_k}, V \in \mathbb{R}^{d_v}$, and produces another vector as a linear combination.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V \tag{6}$$

As described in (Bahdanau et al., 2016), the Q, K, V vectors are meant to represent a "query", "key" and "value" as typically referenced in database literature. Typically, the Q, K, V vectors are obtained as the output of some linear transformation on input vectors $Q', K', V' \in \mathbb{R}^{d_{model}}$.

$$Q = Q'W_Q, K = K'W_K, V = V'W_V \tag{7}$$
$$\text{where} \tag{8}$$
$$W_Q, W_K \in \mathbb{R}^{d_{model} \times d_k} \tag{9}$$
$$W_V \in \mathbb{R}^{d_{model} \times d_v} \tag{10}$$

Intuitively, the attention function can be understood as weighing each value component by how much the query and key components align with each other. Mathematically, the degree to which query and key components align with each other is just the dot product between the two, and we select components of the value according to the degree of alignment between the query and key vectors. Indeed, when comparing the two formulations of attention, there are clear similarities:

1. Both formulations apply a linear projection on the query and key vectors before combining them.

2. Both formulations also weigh the value vector using the combination of the query and key vectors.

3.2 Why is Attention so popular?

The attention mechanism was popularized as part of the transformer architecture, which has seen great success in sequence modeling tasks like text generation (Wiseman et al., 2017; Puduppully et al., 2019) and image captioning (Anderson et al., 2018). In the transformer paper (Vaswani et al., 2017), the attention mechanism is extended and formalized, as explained in the rest of this section.

3.3 Attention in Transformer Models

The attention mechanism in transformer models is typically implemented as "multi-head" attention, where the outputs of several attention functions are concatenated to produce a vector that undergoes a linear projection. The specific equation, as explained in (Vaswani et al., 2017) is

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O \tag{11}$$
$$\text{where} \tag{12}$$
$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{13}$$
$$W_i^Q, W_i^K \in \mathbb{R}^{d_{model} \times d_k} \tag{14}$$
$$W_i^V \in \mathbb{R}^{d_{model} \times d_v} \tag{15}$$
$$W^O \in \mathbb{R}^{hd_v \times d_{model}} \tag{16}$$
$$d_k = d_v = d_{model}/h \tag{17}$$

In (Vaswani et al., 2017), this function is referenced as a single "Multi-Head Attention" block. We can visualize where Multi-head attention is used in the transformer block in Figure 1, the diagram from (Vaswani et al., 2017).

4 Contextualized Feature Representations

The attention mechanism can be thought of as a function to transform a sequence of raw token embeddings into a context-aware representation of the dependencies each token has with the rest of the sentence, quantified by the attention scores. **In this sense, attention is a mechanism to create contextualized feature representations from a sequence**. In the rest of this section, we explore some of the ways attention has been used to create context-aware feature representations for many sequence modeling tasks explained in the introduction.

4.1 Self-attention, Cross-Attention, Joint Attention

The terms "self-attention", "cross-attention" and "joint attention" are commonly mentioned in litera-

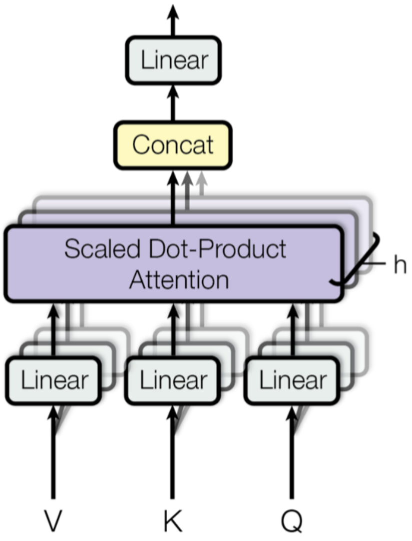

Figure 1: Each of the h heads is a separate execution of scaled dot-product attention. The output is concatenated and then undergoes a linear projection. This image sourced from (Vaswani et al., 2017).

ture (Vaswani et al., 2017; Lin et al., 2021), but are all similar applications of attention with different inputs.

4.1.1 Self-attention

Self-attention is the application of attention as explained in (Vaswani et al., 2017), where the input vectors Q, K, V are derived from the same input vector $\mathbf{x}$, that is, $Q = \mathbf{x}W^Q, K = \mathbf{x}W^K, V = \mathbf{x}W^K$. This attention function allows the model to process each input vector as a function of itself, rather than a simple linear projection (rather than using a fixed weighing of components of an input vector, the input vector itself can determine how much to weigh its components). Self-attention is used in the original attention work (Bahdanau et al., 2016), the transformers work (Vaswani et al., 2017), and more recent architectures like BERT (Devlin et al., 2019), PaLM (Chowdhery et al., 2022), *inter alia*.

4.1.2 Cross-attention

The term "cross-attention" is typically used in multi-modal settings, where the query vector comes from the encoder for one modality (e.g. text), while the key and value vectors come from the encoder for another modality (e.g. images). In this way, one modality can "attend" to another. In the function, this looks like calling Attention$(Q_{\text{text}}, K_{\text{image}}, V_{\text{image}})$.

4.1.3 Joint attention

The term "joint attention" typically refers to when a model architecture may be using cross-attention in settings where the keys come from one encoder (ex. text) as well as another attention block where the key vectors come from the other modality's encoder (ex. images). In the function, this looks like calling Attention$(Q_{\text{text}}, K_{\text{image}}, V_{\text{image}})$ as well as Attention$(Q_{\text{image}}, K_{\text{text}}, V_{\text{text}})$.

The original transformers paper is likely the most popular example of using attention as a mechanism to compute a representation of tokens that includes information from all other context tokens. This was particularly useful to develop many current large language models like BERT (Devlin et al., 2019), PaLM (Chowdhery et al., 2022), GPT-2 (Radford et al., 2019), and GPT-3 (Brown et al., 2020). In addition to scaling the given input features through methods like "self-attention" and "cross-attention" as described in (Vaswani et al., 2017), there is also significant work to use the same cross-attention mechanism for multiple modalities (using latent vectors encoded from a video with vectors encoded from text) (Palma Gomez et al., 2024; Zhang et al., 2024a).

5 Attention Maps

The attention mechanism has also been used in efforts to interpret and understand a language model's outputs by observing the attention scores that the model assigns other tokens within the input sequence for each token. The attention map is just a matrix $A \in \mathbb{R}^{n \times n}$ where A_{ij} describes how much token$_i$ "attends to" token$_j$, where a larger value corresponds to the two being more closely related.

$$A = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{18}$$

Indeed, we observe that A has the correct dimension to observe how each token may attend to every other token in the input sequence, since $Q, K \in \mathbb{R}^{d_{model} \times d_k}$, $A \in \mathbb{R}^{d_{model} \times d_{model}}$. These attention scores can help explain how the model is manipulating an input sequence to achieve the training objective. **In this sense, attention scores can be a mechanism to help explain how language models produce their outputs**. The attention maps can be visualized: Before training, we would expect to see a completely random attention map, where tokens randomly attend to other tokens in the sequence.

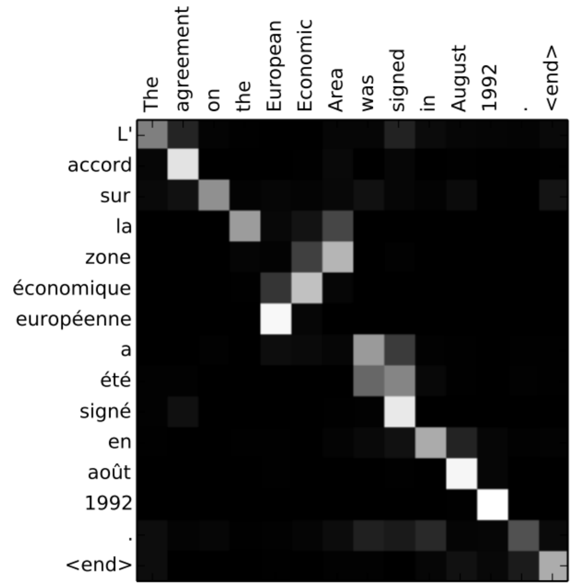

Figure 2: The attention map for the English-to-French translation task for the English sentence "The agreement of the European Ecnomic Area weas signed in August 1992." We observe that the tokens "European Economic Area" correctly attend to tokens in an order that's not simply left-to-right because the order of these words is reversed in French.(Alammmar, 2018)

5.1 A similar idea in Computer Vision

A similar idea to explain how input features are combined has been explored in computer vision through a technique called "saliency maps". Saliency maps depict the magnitude of the gradient on the input image, using the idea that pixels of an image that may significantly affect the model output can help explain how the model is processing the input image. An example saliency map and its input image can be seen in Figure 3. The idea of attention maps can also be applied to computer vision in the case of Vision Transformers introduced in (Dosovitskiy et al., 2021), which take 16x16 patches of pixels as a single "token", and then carry out self-attention in a similar manner with the input token representing the 16x16 patch of pixels. The generated attention maps can be used to directly infer which patches of the image are most related to each other. An example attention map that could be found in a Vision Transformer is in Figure 4.

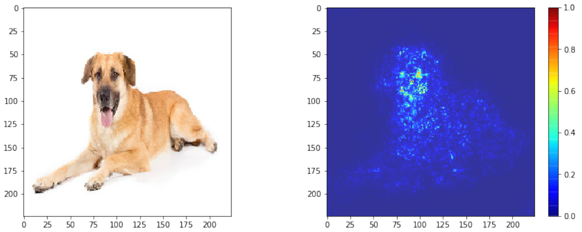

Figure 3: A saliency map for a CNN with a classification task to predict animals. We observe that the picture of the dog has high gradients applied to the dog's face, which suggests that the pixels that correspond to the dog's face have the largest impact on the predicted class, as one would expect. This image taken from (Rizwan, 2020)

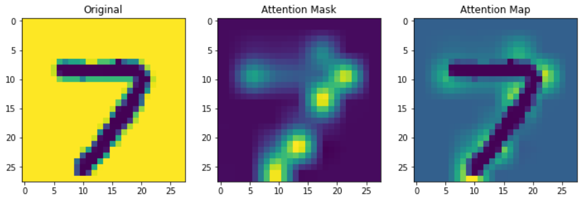

Figure 4: An attention map applied to an image of the number 7 from the MNIST dataset. This image taken from (jo1jun, 2021).

5.2 Caveats to Attention Maps

However, the degree to which attention maps can help explain model outputs has been contested in (Jain and Wallace, 2019; Serrano and Smith, 2019). Jain and Wallace conducted an empirical study showing that, even after manipulating attention weights, several existing models (Bi-RNN, transformer) still output the same predictions. They also demonstrate that learned attention can be manipulated without significantly affecting model performance, which weakens its reliability as an explanation. The paper suggests that attention should not be treated as an interpretability tool and emphasizes the need for a more rigorous explanation method for models that use the attention mechanism. Pursuant to this lack of interpretability, (Brocki et al., 2024) attempts to extend attention maps for Vision Transformers by scaling attention scores with the relevance a particular token has on the model prediction. The resulting scaled attention map more closely contributes to model predictions, which the authors encourage as more applicable for exlpaining model outputs.

From the papers reviewed, it's unclear if the literature supports a clear consensus on whether raw attention maps can provably or empirically be used to interpret model predictions. There is work suggesting that modifying the attention weights can help them be better indicators for what parts of the input are more relevant to model outputs, but future research is required to devise a method with which to better explain model outputs.

6 Attention as soft-routing

Since the attention mechanism computes features for an input token given all tokens of an input sequence, attention can be thought of as performing a soft routing of tokens, where certain tokens are routed with greater or smaller probability to process another token. **In this sense, attention is performing a soft routing of tokens to each other.** This idea of routing is made more explicit in Switch Transformers (Fedus et al., 2022), in which an explicit router model selects which query, key and value linear projects a token should be processed with before being used in self-attention (Fedus et al., 2022). An interesting side effect of this architecture is that the parameter count may be increased dramatically (even to 1 trillion tokens (Fedus et al., 2022)) while only training a subset of the whole architecture at any one time. (Zhang et al., 2024b) develops a technique that uses attention directly in the routing, making the connection to routing information more explicit.

7 Future Research

The most contentious concept found in the reviewed papers is the use of attention maps to help explain model outputs. Given the empirical research showing that attention weights do not provide counterfactual evidence to show that they can be used to explain model outputs. This suggests a strong need for future research to devise a mechanism with which to relate model inputs to outputs, particularly in the case of attention.

8 Conclusion

The attention mechanism presents a variety of use-cases that help demonstrate the versatility of its applications in building contextual feature representations, a method to help explain how a language model may be generating its outputs with connections to computer vision, and finally a form of routing information within networks. However, future research is necessary to evaluate attention as a method to explain how input tokens relate to model outputs.

References

Jay Alammmar. 2018. Visualizing a neural machine translation model (mechanics of seq2seq models with attention).

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. *Preprint*, arXiv:1707.07998.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate. *Preprint*, arXiv:1409.0473.

Lennart Brocki, Jakub Binda, and Neo Christopher Chung. 2024. Class-discriminative attention maps for vision transformers. *Preprint*, arXiv:2312.02364.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, and 48 others. 2022. Palm: Scaling language modeling with pathways. *Preprint*, arXiv:2204.02311.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. *Preprint*, arXiv:2010.11929.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Preprint*, arXiv:2101.03961.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. *Preprint*, arXiv:1902.10186.

jo1jun. 2021. Vision_transformer.

Seed42 AI Lab. 2024. Ai-paper-crawl.

Hezheng Lin, Xing Cheng, Xiangyu Wu, Fan Yang, Dong Shen, Zhongyuan Wang, Qing Song, and Wei Yuan. 2021. Cat: Cross attention in vision transformer. *Preprint*, arXiv:2106.05786.

Junyang Lin, Xu Sun, Shuming Ma, and Qi Su. 2018. Global encoding for abstractive summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 163–169, Melbourne, Australia. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *Preprint*, arXiv:1908.08345.

Frank Palma Gomez, Ramon Sanabria, Yun-hsuan Sung, Daniel Cer, Siddharth Dalmia, and Gustavo Hernandez Abrego. 2024. Transforming LLMs into cross-modal and cross-lingual retrieval systems. In *Proceedings of the 21st International Conference on Spoken Language Translation (IWSLT 2024)*, pages 23–32, Bangkok, Thailand (in-person and online). Association for Computational Linguistics.

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with entity modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2023–2035, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Usman Rizwan. 2020. Saliency maps in tensorflow 2.0.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? *Preprint*, arXiv:1906.03731.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *Preprint*, arXiv:1701.06538.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Preprint*, arXiv:1706.03762.

Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.

Haowei Zhang, Jianzhe Liu, Zhen Han, Shuo Chen, Bailan He, Volker Tresp, Zhiqiang Xu, and Jindong Gu. 2024a. Visual question decomposition on multimodal large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1926–1949, Miami, Florida, USA. Association for Computational Linguistics.

Qizhen Zhang, Nikolas Gritsch, Dwaraknath Gnaneshwar, Simon Guo, David Cairuz, Bharat Venkitesh, Jakob Foerster, Phil Blunsom, Sebastian Ruder, Ahmet Ustun, and Acyr Locatelli. 2024b. Bam! just like that: Simple and efficient parameter upcycling for mixture of experts. *Preprint*, arXiv:2408.08274.