A quick guide to Attention Mechanisms

José Hilario

Backbone

Attention

Attention mechanisms map a query and key-value pairs to an output. The output can be seen as a convex combination of the values, where the weight assigned to each value is obtained by a similarity function of the query with the key corresponding to the value. We can think of it as a form of soft-retrieval; we have: 1) a query for which we want to retrieve its value; and, 2) keys associated to values, to which we will match the query:

$$y = Attention(q, K, V) = \sum_i Similarity(q, k_i) \cdot v_i \qquad (1)$$

Self-attention

Relates elements from an input set between each other. In self-attention, the queries, the keys and the values, all come from the same source of information. Considering a group of n input vectors $X = (x_1, \cdots, x_n)$, we obtain with self-attention $Y = (y_1, \cdots, y_n)$, where:

$$y_i = Attention(x_i, X, X) = \sum_j Similarity(x_i, x_j) \cdot x_j \qquad (2)$$

Remark. *The resulting self-attention is permutation equivariant:* $Attention(\pi(X)) = \pi(Attention(X))$.

Matrices

We desire to learn representation spaces suitable for comparing the queries and keys, as we may not assume that their initial representations can be directly compared. Similarly, we wish to impart more power unto the attention mechanism by allowing it to project the values into a space that is more suitable to produce the final values obtained by attention.

Considering $X, Y \in \mathbb{R}^{n \times d_{kin}}$, $Z \in \mathbb{R}^{n \times d_{vin}}$, we let $W^Q, W^K \in \mathbb{R}^{d_{kin} \times d_k}$, $W^V \in \mathbb{R}^{d_{vin} \times d_v}$. Then,

$$Q = X \cdot W^Q, \quad K = Y \cdot W^K, \quad V = Z \cdot W^V \qquad (3)$$

where $Q, K \in \mathbb{R}^{n \times d_k}$, $V \in \mathbb{R}^{n \times d_v}$.

Scaled Dot-Product Attention

The similarity function of choice by attention mechanisms is the dot product, followed by the $Softmax$ function (used to normalize the weights into probabilistic form). Considering previously defined cardinalities, the attention values are obtained as follows:

$$Attention(Q, K, V) = Softmax\left(\frac{Q \cdot K^\top}{\sqrt{d_k}}\right) \cdot V \qquad (4)$$

Multi-Head Attention

Multi-Head Attention (MHA) allows the model to jointly attend to information from different representation subspaces: instead of performing a single attention function with d_k-dimensional keys and queries and d_v-dimensional values, the queries, keys and values are projected with h different learned projections, each with size $\frac{d_k}{h}$ for the queries and the keys, and size $\frac{d_v}{h}$ for the values. Attention is performed in parallel across the h heads, yielding $\frac{d_v}{h}$-dimensional output values. These are concatenated and once again projected with learned projection W^O, resulting in the final d_v-dimensional values. Let $O_i = Attention(X \cdot W_i^Q, Y \cdot W_i^K, Z \cdot W_i^V)$, then:

$$MHA(X, Y, Z) = Concat[O_1, \cdots, O_h] \cdot W^O \qquad (5)$$

Transformer

Why Transformers?

Challenges with Recurrent Neural Network (RNN):

- *(Long-range dependencies)* Usually solved using attention mechanisms.
- *(Gradient vanishing and explosion)* When the gradient is passed back through many time steps, it tends to grow or vanish.
- *(Large number of training steps)* RNNs need to be unrolled for as many steps as corresponds in the sequence. The optimization tends to be difficult because of the parameter sharing in the unfolded network.
- *(Recurrence prevents parallel computation)* Due to the sequential nature of RNNs.

Properties of the *Transformer*:

- *(Facilitate long-range dependencies)* Connections can be drawn among any parts of the sequence. Long-range dependencies have the same prior likelihood of being used than short-range dependencies.
- *(No gradient vanishing and explosion)* Instead of having a number of computations that grows linearly with the length of the sequence, we do the computation for the entire sequence simultaneously, passing it through a constant amount of layers.
- *(Fewer training steps)* Easier to optimize.
- *(No recurrence)* Faster training due to parallelization is possible using *Teacher-Forcing* (note that decoder inference still needs to be sequential).

Architecture

Encoder-decoder architecture designed for machine translation. In contrast to RNN-based encoders in which sentences are processed word by word, the *Transformer Encoder* processes all words in parallel. The input of the *Transformer* is a vector of word embeddings representing the input sentence, each of the embeddings injected with a positional encoding value containing information about the position of the word in the sentence.

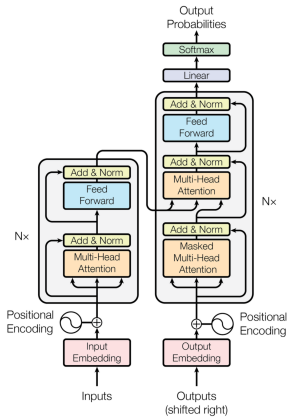

Figure 1: Transformer model by Vaswani et al. [5]

The *Transformer Encoder* consists of MHA, *Layer Normalization* and a Feed-Forward Neural Network (FFNN). The input to the encoder is a vector of embeddings of the words in the sentence. Since we wish to treat the input as a sequence rather than a set, we inject positional information using a *Positional Encoder*. Then, the MHA will compute the attention between all pairs of embeddings, producing for each word a better embedding containing additional relational information. The *Transformer Encoder* can be stacked: the first layer combines words with words, the second layer combines pairs with pairs, etc. Following the MHA is an *Add&Norm* layer, which passes a skip connection preceding MHA, and applies *Layer Normalization* by Ba et al. [1]. This is followed by a FFNN, which is again followed by an *Add&Norm* layer. The output of the encoder is a set of embeddings — one per word — each capturing the information of the original word embedding along with the attention given to the other words.

The *Transformer Decoder* is tasked to output a sequence of labels corresponding to translated words in the target dictionary. The decoder also uses a *Positional Encoder* at its input, and similar to the encoder, it can be stacked a number of times. In the *Transformer Decoder* we have a masked MHA, which performs self-attention on the embeddings of the target words at the input when applying the *Teacher-Forcing* training

strategy. When generating the word $\hat{y}_k$, the target words preceding in position $y_1, \cdots, y_{k-1}$, are used as input instead of the generated words $\hat{y}_1, \cdots, \hat{y}_{k-1}$; this makes training easier, and allows us to parallelize decoding for training. The mask restricts the attention of the decoder to the target words preceding the current target word. Formally,

$$MaskedAttn(Q, K, V) = Softmax\left(\frac{Q^\intercal \cdot K + M}{\sqrt{d_k}}\right) \cdot V \quad (6)$$

where M is a mask matrix filled with 0 and $-\infty$, ensuring the probability of masked elements will be zero after applying the $Softmax$ activation, and the distribution will be proper (adding up to 1). Then, another MHA block follows, which associates translated words to original input words. This is followed by a FFNN. Then the output of the *Transformer Decoder* is fed into a classifier which outputs a distribution over words in the target dictionary.

Computer Vision

Squeeze-and-Excitation Networks

The Squeeze-and-Excitation (S&E) block adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels. This block learns to use global information to emphasize informative features and suppress less useful ones. The representations produced by a Convolutional Neural Network (CNN) are enhanced by integrating the S&E block into the network which helps it capture spatial correlations between features.

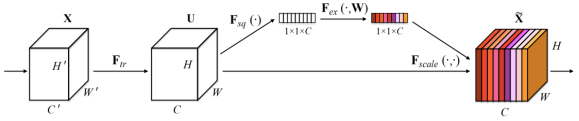

Figure 2: S&E block by Hu et al. [3]

(Squeeze: Global Information Embedding) Each convolutional filter operates with a local receptive field; each unit of the output U is unable to exploit contextual information outside of its region. To mitigate this problem, we squeeze global spatial information into a channel descriptor, using Global Average Pooling (GAP) to generate channel-wise statistics. The output U can be seen as a collection of local descriptors whose statistics are expressive for the whole image. A statistic $z \in \mathbb{R}^C$ is generated by shrinking U through its spatial dimensions H, W such that each c-th element of z is obtained by eq. 7.

(Excitation: Adaptive Recalibration) To make use of the information aggregated in the squeeze operation, a second operation aims to fully capture channel-wise dependencies. This operation learns non-linear interactions between channels, and a non-mutually-exclusive relationship, so that multiple channels are allowed to be emphasized. A *Sigmoid* activation is used in eq. 8, where $W_1 \in \mathbb{R}^{\frac{C}{r} \times C}, W_2 \in \mathbb{R}^{C \times \frac{C}{r}}$. The final output is obtained by rescaling U with the activations s, by eq. 9.

$$z_c = F_{sq}(u_c) = \frac{1}{H \cdot W} \cdot \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j) \quad (7)$$

$$s = F_{ex}(z, W) = Sigmoid(W_2 \cdot ReLU(W_1 \cdot z)) \quad (8)$$

$$\widetilde{x}_c = F_{scale}(u_c, s_c) = s_c \cdot u_c \quad (9)$$

Attention Augmented Convolutional Networks

The convolution operator is limited by its locality and lack of understanding of global contexts. The S&E block captures long-range dependencies by recalibrating feature maps; in Attention-Augmented Convolutions (AAConv): 1) an attention mechanism that attends jointly to spatial and feature sub-spaces is used; and, 2) additional feature maps are introduced (rather than only refining existing ones). AAConv are equivariant to translation, and can operate on inputs of different spatial dimensions: $AAConv(X) = Concat[Conv(X), MHA(X)]$.

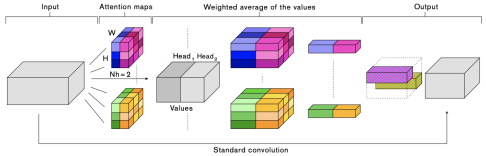

Figure 3: AAConv by Bello et al. [2]

Given an input tensor $X \in \mathbb{R}^{H \times W \times d_{in}}$, we flatten it into a matrix $X \in \mathbb{R}^{H \cdot W \times d_{in}}$ and perform MHA with h heads and with learned parameters for head i: $W_i^Q, W_i^K \in \mathbb{R}^{d_{in} \times \frac{d_k}{h}}$, $W_i^V \in \mathbb{R}^{d_{in} \times \frac{d_v}{h}}$; and $W^O \in \mathbb{R}^{d_v \times d_v}$.

(Two-Dimensional Positional Encoding) In the domain of computer vision, translation equivariance is a desirable property in primitive operations. Two-dimensional relative self-attention is implemented by independently adding relative height and width information. Attention from pixel $p_1 = (x_1, y_1)$ to $p_2 = (x_2, y_2)$ is defined as:

$$y_{p_1, p_2} = Softmax\left(\frac{q_{p_1}^\intercal \cdot \left(k_{p_2} + r^W_{x_2 - x_1} + r^H_{y_2 - y_1}\right)}{\sqrt{d_k}}\right) \cdot v_{p_2} \quad (10)$$

Stand-Alone Self-Attention in Vision Models

Capturing long-range interactions for CNNs is challenging because of their poor scaling properties with respect to large receptive fields. Attention mechanisms have the ability to directly model long-distance relationships, which makes it an appropriate suitor to enhance or supersede convolutions. Experiments suggest that convolutions may better capture low level features, while local attention layers may better integrate global information.

(Memory block) Analogous to a convolution, given a pixel $x_{ij} \in \mathbb{R}^{d_{in}}$, we first extract the memory block: a local region of pixels in positions $ab \in \mathcal{N}_k(i, j)$ with spatial extent k centered around x_{ij}. This form of local attention differs from prior work by Bello et al. [2] exploring attention in vision which performed global attention between all pixels, which can be used only after significant spatial downsampling because of its computational demands. For positional encoding, relative self-attention is used: the row and column offsets are associated with an embedding r_{a-i} and r_{b-j}, respectively, each with dimension $\frac{1}{2} d_v$. The row and column offset embeddings are concatenated to form $r_{a-i,b-j}$. The logit measuring the similarity between the query and an element in $\mathcal{N}_k(i, j)$ is modulated both by the content of the element and the relative distance of the element from the query. Spatial-relative attention is defined as:

$$y_{ij} = \sum_{ab \in \mathcal{N}_k(i,j)} Softmax_{ab}\left(\frac{q_{ij}^\intercal \cdot (k_{ab} + r_{a-i,b-j})}{\sqrt{d_k}}\right) \cdot v_{ab} \quad (11)$$

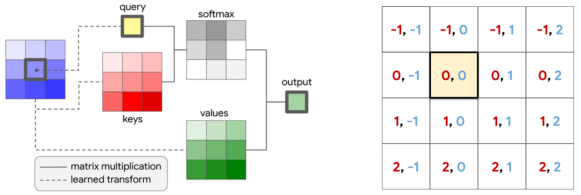

Figure 4: Stand-Alone Self-Attention Convolutions (SASAConv) by Ramachandran et al. [4]. Left, an example of a local attention layer over spatial extent $k = 3$; right, relative distances for relative encoding computed with respect to the highlighted pixel.

In practice we use h heads for h groups. Then, for group n, $x_{ij}^n \in \mathbb{R}^{\frac{d_{in}}{h}}$. Considering W_n^Q, W_n^K, $W_n^V \in \mathbb{R}^{\frac{d_{in}}{h} \times \frac{d_{out}}{h}}$, we have $q_{ij}^n = x_{ij}^n \cdot W_n^Q$, $k_{ab}^n = x_{ab}^n \cdot W_n^K$, $v_{ab}^n = x_{ab}^n \cdot W_n^V$.

A spatially-aware attention stem is proposed to bridge the gap with convolutions concerning learning local features at earlier layers. For more information, refer to [4].

References

[1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization, 2016.

[2] I. Bello, B. Zoph, Q. Le, A. Vaswani, and J. Shlens. Attention augmented convolutional networks. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3285–3294, 2019. doi: 10.1109/ICCV.2019.00338.

[3] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018. doi: 10.1109/CVPR.2018.00745.

[4] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens. Stand-alone self-attention in vision models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/3416a75f4cea9109507cacd8e2f2aefc-Paper.pdf.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.