You Just Want ‘Attention’

Sayar Ghosh Roy

The attention mechanism has become the architecture of choice for various types of tasks which, at some level, require the neural model to focus upon specific sections of the inputs. In this regard, we will look at a few novel offshoots based on the core principles governing the attention mechanism.

In the paper titled, “Attention-over-Attention Neural Networks for Reading Comprehension”, the idea of a two-way attention mechanism is introduced. The paper tries to formally solve the problem of reading comprehension in which the model tries to come up with an answer *A* for a query *Q*, with a document *D* as the passage for comprehension. Unlike a traditional global or local attention mechanism (where the attention weights are a function of the previous layer of embeddings, and all attention weights computed upto the last timestamp), the paper involves the computation of similarity scores between a word in the document with one in the query string. Thus, we can compute a matrix M with $M(i, j) = h^{T}_{doc}(i) \,.\, h_{query}(j)$ where $h_{doc}(i)$ and $h_{query}(j)$ are the generated embeddings for the i^{th} token in the document and the j^{th} token in query string respectively. Now, for a particular query word at index say ‘t’, a softmax over the sequence (M(1, t), M(2, t), … , M(|D|, t)) gives a probability distribution which acts as the attention vector for the t^{th} query word on the document D. If we call this as α(t), we can define α as the ordered concatenation of all α(i)’s which can be thought of as the query-to-document attention. Similarly, we proceed in the reverse direction to obtain query-level attentions for document words. We regard β as softmax over (M(t, 1), M(t, 2), … , M(t, |Q|)). Now, averaging individual attention preserves the normalizing condition and thus, we consider the arithmetic mean of all β’s as the averaged query level attention β. Finally, $s = \alpha^{T}\beta$ gives us the attended document-level attention where the contribution of each individual query word has been weighed independently. This is the attention-over-attention mechanism as it is attending upon the computed vector of attentions in matrix M. Note that, in evaluating attention scores for documents to queries and vice versa, it is the same matrix of similarities that becomes the base for further computation.

Finally, a sum of attentions is used, given the query-string and the document to compute the importance of that particular word, say w. This mechanism gives us the importance levels of every single word in our vocabulary which will aid in answering that particular query rooted on that individual document. The paper uses additional post-processing techniques such as checking the fluency and appropriateness of the answer string using various features such as local and global N-gram language models, trained on the individual testing document and the entire training corpus respectively and a word-class language model where sequences of word cluster heads are modelled based on the documents in the training set. The training is supervised and we maximize the log-likelihood of the correct answer.

Now, we come to the paper titled, "Bi-directional Attention Flow for Machine Comprehension" which tries to solve the same task and builds upon the findings of attention-over-attention. Unlike some other works in the field, the architecture here does not go for summarizing, or in other words, getting simpler representations of the underlying sequences as lower neuronal levels. The paper claims to represent context at various levels of granularity using a multi-staged hierarchical process, leading to a query-aware context representation. This idea is very similar to what we saw in AoA, and in fact, if you diligently review the related work stated in this paper, you will find that AoA is the closest approximation to their approach. BiDAF lists the memory-less attention mechanism as a key feature of their work. Specifically, the attention at each time step is a "function of only the query and the context paragraph at the current time step and does not directly depend on the attention at the previous time step." However, note that we saw the very same idea being used in AoA to create the similarity matrix M. The key notions and the main modular units are pretty much the same in both AoA and BiDAF, the differences lie within the modules themselves.

The first difference, though not a very significant one is in the creation of embeddings. While both the techniques go for word embeddings being combined into context-aware word representations, they differ in the choice of architectures. AoA uses a more traditional approach converting one-hot word vectors to compact continuous embeddings and sharing the same embedding generation matrix across document and query ends, and a more humble bi-GRU to generate context rich word embeddings. BiDAF goes for a more complex method where they leverage character embeddings using CNNs, pretrained GloVe embeddings and the concatenated character and word embeddings are passed into a two-layered highway network to yield the final token embeddings. Finally, a layer of bi-LSTMs constitutes the contextual embedding layer.

The key difference between AoA and BiDAF lies in the intricacies of the attention computation itself. We see the idea of using a similarity matrix once more. However, the computation of similarity itself is what changes. Instead of the simple dot product used in AoA, the similarity function is attributed as a learnable scalar function, and the implementation of the paper uses $w_{(S)}$ [h; u; h ◦ u] where h and u are the embeddings of a context word and a query word respectively, ◦ represents an element-wise multiplication and most importantly, we have a learnable parameter $w_{(S)}$. In the post similarity-matrix creation phase, the computation of the two-way attention scheme uses the same kind of information and computations as AoA. For context-to-query attention, a similar use case of the softmax layer as attention weights is witnessed and for the opposite query-to-context, another arbitrary trainable function β was suggested which in practice, was realised using a concatenation of h, ũ, h ◦ ũ and h ◦ h̃, where ũ and h̃ are the attended query and context representation respectively. There is a minor difference in computing the query-to-context specific attention weights: in BiDAF, the softmax is applied over the maximum values in each column of the similarity matrix.

A modelling layer is required since the dimensions of query-to-context and context-to-query attentions may not be equal which implies that a specific operation is required to create a match. But more importantly, a final layer is intuitively necessary to capture the interaction among the context words, conditioned upon the entire query. Now, the output layer can be task specific and the modularity in lower-layers allows easy modifications to the output layer. For machine comprehension, we try to predict the initial and final indices from the document which will fill-in-the-blank in our answer. Therefore, we try to minimize the sum of the negative log-likelihoods of the true-end and true-start indices based on the gold standard data.

Similar schemes of this two-way attention were tried out for visual question answering where the model needs to attend-to specific pixels or larger sections in the image in order to answer a particular query, and these were able to produce pretty decent results. Machine Comprehension serves as the corresponding task on the side of pure natural language processing and endeavours at this end show that it is indeed the scheme of attention which is required to answer document specific queries.

If we carefully analyze the two papers, the only key difference in AoA and BiDAF is the free-flow of all attention vectors to the final recurrent layer in BiDAF as opposed to a summarised $s = \alpha^T\beta$ plus a sum attention mechanism leveraging the two-way attention in AoA. The authors of BiDAF claim that this will help remove the unnecessary noise induced due to early summarisation. Apart from that, BiDAF was the later work which was clearly ‘inspired’ by AoA and resorted to the use of more complex subunits to improve upon AoA. Use of character-CNNs, highway networks to make use of both character and word-embeddings, LSTMs as opposed to GRUs, more complicated functions for similarity and attention computations are all examples where BiDAF chose to go for a bigger model with a higher number of parameters to outperform a simpler system. At its core, however, the big picture presented by BiDAF is the same as that of AoA.

Following along the themes of simple versus complicated models and when to use what, let us quickly go through this paper on “Dynamic Capacity Networks” or DCNs. To have some background, let us recall Occam’s Razor from philosophy which states that “plurality should not be posited without necessity”. In simple terms, if we have two theories which sufficiently explain a particular phenomena, we should prefer the simpler one. This idea was also given a thumbs up by Albert Einstein who once said, “Everything should be made as simple as possible, but no simpler.” We have seen some examples in legacy image processing systems where a particular model is selected out of a few with varying complexities based on how difficult the particular unit of task-at-hand is. In that, we only go for a more complex model if the simpler model fails to solve the problem within a threshold possible error.

Now, if we come to the task of creating latent representations of say, an image, our final goal may require us to focus on only a few areas in the image. Therefore, it is futile to use the same amount of resources ‘staring at’ each image section when we could have easily ‘glanced over’ the less important parts and focussed on the key spots. This is what is achieved by DCNs.

We can divide a neural network into its bottom and top layers, say f and g respectively. Now, the function implemented by the Neural Network is effectively h(x) = g(f(x)). Where f(x) is the portion generating the underlying representation while g(.) operates on the intermediate to produce the final output. We make a distinction between two types of bottom layers: f_c and f_f, where the coarse f_c is much less computationally expensive than the fine f_f. We need to ensure that the intermediate outputs by both these types of bottom layers are in the very same latent space. This allows the same g(.) to act on outputs of f_c and f_f with no parameter tweaks. A metric of squared distance between the outputs from fine and coarse layers can be minimized to ensure just this. Note that this work was published about four years ago and various different schemes using discriminator based losses have been adopted since then to solve the latent space mismatch issue.

Given that we have two possible choices for bottom layers which can be used interchangeably, we need to know when to use what. This is where the hard attention mechanism comes into the picture. The hypothesis is to use fine representations for sections for which the distribution over the class label is most ‘sensitive’. A natural choice for quantifying sensitivity can be based upon the entropy metric and indeed, the saliency or importance M(i, j) of a particular section (i, j) of the image is the norm of the gradient of the entropy H w.r.t the coarse vector $c_{(i,\ j)} = f_c(X_{(i,j)})$. No true labels are used in the calculation of saliency and therefore, the same measure is available at the time of inference. If $o = g(f_c(x))$ is the vector output of the coarse model, then:

$$H = -\sum_{l=1}^{C} o^{(l)} \log(o^{(l)}) \text{ and } M(i,j) = ||\nabla_{c(i,j)} H||_2$$

Now, we select a set of (i, j) with the highest saliency values and apply f_f only on these $X_{(i,j)}$’s. We can denote I_s as the set of salient (i, j)’s. Clearly $I_s \subseteq [1, s_1] \times [1, s_2]$, such that $|I_s| = k$; with s_1 and s_2 being the spatial dimensions which depend upon the image size.

A small set of fine representations is generated: $f_f(X_s) = \{f_f(x_{i,j}) \text{ s.t } (i, j) \in I_s\}$ and we have the combined feature representation of $f_c(X)$ and $f_f(X_s)$ as $f_r(X)$ which is used for further downstream tasks. The model was trained on the image classification task using the cross-entropy objective function, which is equivalent to maximizing the log-likelihood of the correct labels. This reduction from $[1, s_1] \times [1, s_2]$ down to I_s as the set which requires a more complicated model greatly brings down the total cost of generating latent representations while sufficiently keeping up the performance of using a full-scale fine model f_f. This serves as the unique selling point of using DCNs.

Thus, we have seen approaches and use cases of using two-way attentions for machine comprehension tasks. We saw a more complex model building up on previous ideas to improve the system’s performance. Finally, we reviewed the idea of using a hard attention scheme whereby we can use a more complicated model only for the more salient areas, which brings down the computational cost involving the use of a complex model on the entire span of input.

Thank You