ML – Attention mechanism

Let's take a look at hearing and a case study of selective attention in the context of a crowded cocktail party. Assume you're at a social gathering with a large number of people speaking at the same time. You're also talking with a friend, but the background noise is not recognized. You are only paying attention to your friend's voice and grasping their words while filtering out background noise. In this scenario, our auditory system employs selective attention to focus on the relevant auditory information. The neurological system of our brain improves the representation of speech by prioritizing relevant sounds and ignoring background noises.

A computer method for prioritizing specific information in a given context is called the attention mechanism of deep learning. During translation or question-answering activities, attention is used in natural language processing to align pertinent portions of the source phrase. Without necessarily relying on reinforcement learning, attention mechanisms allow neural networks to give various weights to various input items, boosting their ability to capture crucial information and improve performance in a variety of tasks. Google Streetview's house number identification is an example of an attention mechanism in Computer vision that enables models to systematically identify particular portions of an image for processing.

Attention Mechanism

An attention mechanism is an Encoder-Decoder kind of neural network architecture that allows the model to focus on specific sections of the input while executing a task. It dynamically assigns weights to different elements in the input, indicating their relative importance or relevance. By incorporating attention, the model can selectively attend to and process the most relevant information, capturing dependencies and relationships within the data. This mechanism is particularly valuable in tasks involving sequential or structured data, such as natural language processing or computer vision, as it enables the model to effectively handle long-range dependencies and improve performance by selectively attending to important features or contexts.

Recurrent models of visual attention use reinforcement learning to focus attention on key areas of the image. A recurrent neural network governs the peek network, which dynamically selects particular locations for exploration over time. In classification tasks, this method outperforms convolutional neural networks. Additionally, this framework goes beyond image identification and may be used for a variety of visual reinforcement learning applications, such as helping robots choose behaviours to accomplish particular goals. Although the most basic use of this strategy is supervised learning, the use of reinforcement learning permits more adaptable and flexible decision-making based on feedback from past glances and rewards earned throughout the learning process.

The application of attention mechanisms to image captioning has substantially enhanced the quality and accuracy of generated captions. By incorporating attention, the model learns to focus on pertinent image regions while creating each caption word. The model can synchronize the visual and textual modalities by paying attention to various areas of the image at each time step thanks to the attention mechanism. By focusing on important objects or areas in the image, the model is able to produce captions that are more detailed and contextually appropriate. The attention-based image captioning models have proven to perform better at catching minute details, managing complicated scenes, and delivering cohesive and educational captions that closely match the visual material.

The attention mechanism is a technique used in machine learning and natural language processing to increase model accuracy by focusing on relevant data. It enables the model to focus on certain areas of the input data, giving more weight to crucial features and disregarding unimportant ones. Each input attribute is given a weight based on how important it is to the output in order to accomplish this. The performance of tasks requiring the utilization of the attention mechanism has significantly improved in areas including speech recognition, image captioning, and machine translation.

How Attention Mechanism Works

An attention mechanism in a neural network model typically consists of the following steps:

1. Input Encoding: The input sequence of data is represented or embedded using a collection of representations. This step transforms the input into a format that can be processed by the attention mechanism.
2. Query Generation: A query vector is generated based on the current state or context of the model. This query vector represents the information the model wants to focus on or retrieve from the input.
3. Key-Value Pair Creation: The input representations are split into key-value pairs. The keys capture the information that will be used to determine the importance or relevance, while the values contain the actual data or information.
4. Similarity Computation: The similarity between the query vector and each key is computed to measure their compatibility or relevance. Different similarity metrics can be used, such as dot product, cosine similarity, or scaled dot product.
$$Score(s,i) = \begin{cases} h_s^{(1)} \cdot y_i & \text{Dot Product} \\ (h_s^{(2)})^T W y_i & \text{General} \\ v^T \tanh\left(W \left[\frac{h_s}{y_i}\right]\right) & \text{concat} \end{cases}$$
where,
   - hs: Encoder source hidden state at position s
   - yi: Encoder Target hidden state at the position i
   - W: Weight Matrix
   - v : Weight vector
5. Attention Weights Calculation: The similarity scores are passed through a softmax function to obtain attention weights. These weights indicate the importance or relevance of each key-value pair.
$$\text{Attention Weight } (\alpha(s,i)) = \text{softmax}(\text{Similarity Scores}(s,i))$$
6. Weighted Sum: The attention weights are applied to the corresponding values, generating a weighted sum. This step aggregates the relevant information from the input based on their importance determined by the attention mechanism.
$c_t = \sum_{i=1}^{T_s} \alpha(s,i) h_j^{(1)}$
Here,
   - Ts: Total number of key-value pairs (source hidden states) in the encoder.
7. Context Vector: The weighted sum serves as a context vector, representing the attended or focused information from the input. It captures the relevant context for the current step or task.

8. Integration with the Model: The context vector is combined with the model's current state or hidden representation, providing additional information or context for subsequent steps or layers of the model.
9. Repeat: Steps 2 to 8 are repeated for each step or iteration of the model, allowing the attention mechanism to dynamically focus on different parts of the input sequence or data.

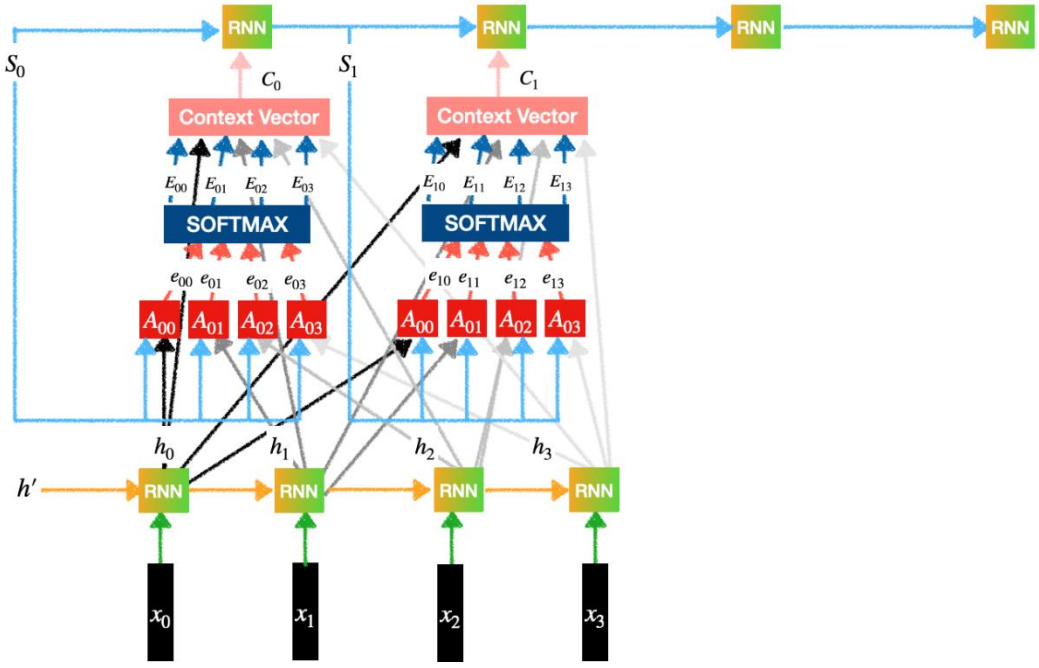
By incorporating an attention mechanism, the model can effectively capture dependencies, emphasize important information, and adaptively focus on different elements of the input, leading to improved performance in tasks such as machine translation, text summarization, or image recognition.

Attention Mechanism Architecture for Machine Translation

The attention mechanism architecture in machine translation involves three main components: Encoder, Attention, and Decoder. The Encoder processes the input sequence and generates hidden states. The Attention component computes the relevance between the current target hidden state and the encoder's hidden states, generating attention weights. These weights are used to compute a context vector that captures the relevant information from the encoder's hidden states. Finally, the Decoder takes the context vector and generates the output sequence. This architecture allows the model to focus on different parts of the input sequence during the translation process, improving the alignment and quality of the translations. We can observe 3 sub-parts or components of the Attention Mechanism architecture :

- Encoder
- Attention
- Decoder

Consider the following Encoder-Decoder architecture with Attention.

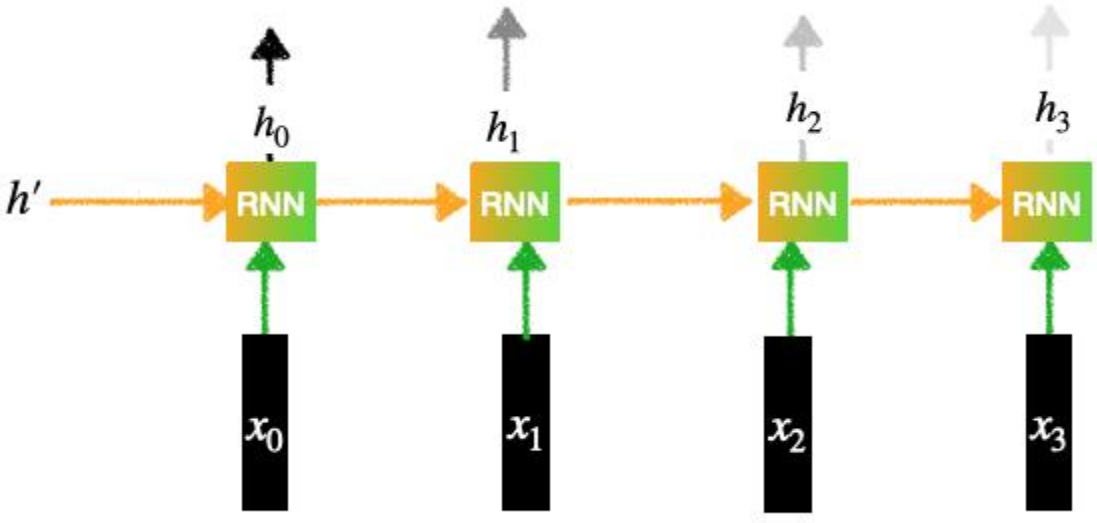


Encoder-Decoder with Attention

Encoder:

The encoder applies recurrent neural networks (RNNs) or transformer-based models to iteratively process the input sequence. The encoder creates a hidden state at each step that contains the data

from the previous hidden state and the current input token. The complete input sequence is represented by these hidden states taken together.



Encoder

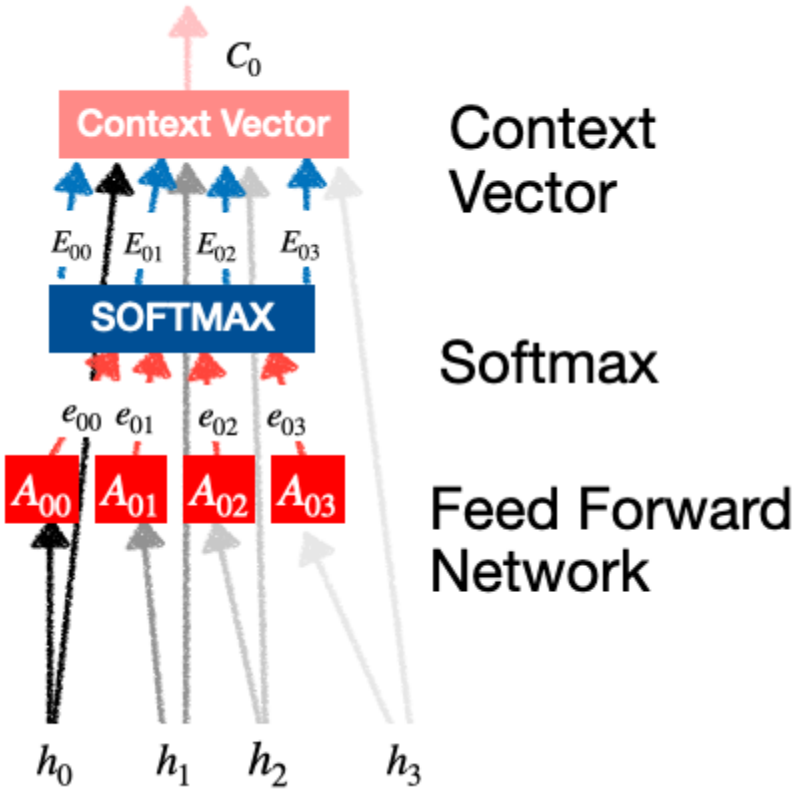
Contains an RNN layer (Can be LSTMs or GRU):

1. Let's, there are 4 words sentence then inputs will be: x_0, x_1, x_2, x_3
2. Each input goes through an Embedding Layer, It can be RNN, LSTM, GRU or transformers
3. Each of the inputs generates a hidden representation.
4. This generates the outputs for the Encoder: h_0, h_1, h_2, h_3

Attention:

The attention component computes the importance or relevance of each encoder's hidden state with respect to the current target hidden state. It generates a context vector that captures the relevant information from the encoder's hidden states. The attention mechanism can be represented mathematically as follows:

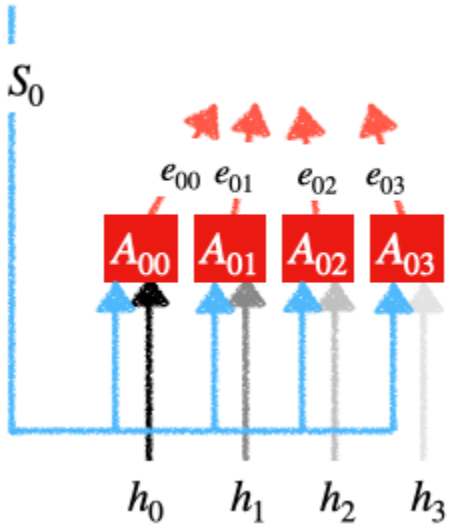
- Our goal is to generate the context vectors.
- For example, context vector C_1 tells us how much importance/ attention should be given the inputs: x_0, x_1, x_2, x_3.
- This layer in turn contains 3 subparts:
    - Feed Forward Network
    - Softmax Calculation
    - Context vector generation



attention

Feed Forward Network:

The feed-forward network is responsible for transforming the target hidden state into a representation that is compatible with the attention mechanism. It takes the target hidden state h(t-1) and applies a linear transformation followed by a non-linear activation function (e.g., ReLU) to obtain a new representation



Feed-Forward-Network

Each A_{00}, A_{01}, A_{02}, A_{03} is a simple feed-forward neural network with one hidden layer. The input for this feed-forward network is:

- Previous Decoder state
- The output of Encoder states.

Each unit generates outputs: e_{00}, e_{01}, e_{02}, e_{03}.i.e

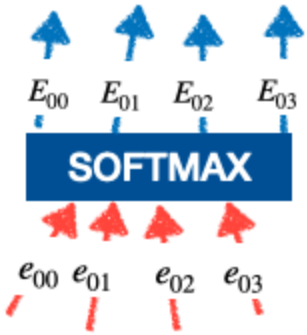
e0i = g (S_0, h_i)

Here,

- g can be any activation function such as sigmoid, tanh, or ReLu.

Attention Weights or Softmax Calculation:

A softmax function is then used to convert the similarity scores into attention weights. These weights govern the importance or attention given to each encoder's hidden state. Higher weights indicate higher relevance or importance.



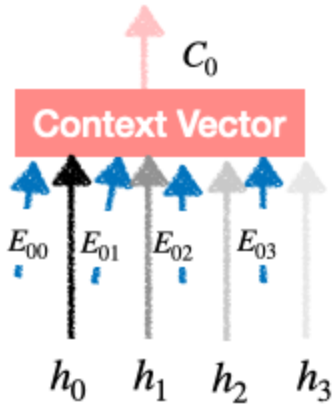
softmax calculation

$$E_{0i} = \frac{exp(e_{0i})}{\sum_{i=0}^{3} exp(e_{0i})}$$

These E_{00}, E_{01}, E_{02}, E_{03} are called the attention weights. It decides how much importance should be given to the inputs x_0, x_1, x_2, x_3 .

Contact Vector Generation:

Context Vector: The context vector is a weighted sum of the encoder's hidden states, where the attention weights serve as the weights for the summation. It represents a specific arrangement of the encoder's hidden states pertinent to generating the current token.



context vector generation

$C_0 = E_{00} * h_0 + E_{01} * h_1 + E_{02} * h_2 + E_{03} * h_3$.

We find C_1, C_2, C_3 in the same way and feed it to different RNN units of the Decoder layer. So this is the final vector which is the product of (Probability Distribution) and (Encoder's output) which is nothing but the attention paid to the input words.

Decoder:

The context vector is fed into the decoder along with the current hidden state of the decoder in order to predict the next token in the output sequence. Until the decoder generates the entire output sequence, this process is done recursively.
We feed these Context Vectors to the RNNs of the Decoder layer. Each decoder produces an output which is the translation for the input words.

Conclusions

The attention mechanism allows the decoder to dynamically focus on different segments of the input sequence based on their importance to the current decoding step. As a result, the model can handle lengthy input sequences with ease and capture the dependencies between various input and output sequence components. The attention mechanism is a crucial component of many cutting-edge sequence-to-sequence models since it significantly boosts the quality and fluency of the generated sequences.

Frequently Asked Questions (FAQs)

1. What is self attention?

Self-attention allows a model to weigh the importance of different parts of its input sequence when making predictions. It enables the model to focus selectively on relevant information, considering the context of each element in relation to others. This mechanism enhances the ability to capture long-range dependencies and improves performance in tasks like machine learning translations and natural language processing.

2. What is the applications of attention mechanism?

Attention mechanism is used in various Natural Language Processing and Computer vision tasks.

- *Machine Translation: Attention mechanisms have significantly improved the performance of machine translation models. It enable the model to focus on different parts of the source sentence when generating each word in the target sentence.*
- *In tasks like sentiment analysis, question answering, and named entity recognition, attention mechanisms helps models to focus on critical words contributing to sentiment expression.*
- *In text summarization, attention aids in selecting key information for concise summaries.*
- *Image Captioning: Attention mechanisms in image captioning models allow the model to focus on specific regions of an image while generating captions.*
- *Attention mechanisms have been applied to improve the accuracy of automatic speech recognition systems.*
- *In generative models like Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), attention mechanisms help the model capture dependencies between different parts of the input data, leading to more realistic and coherent generated samples.*
- *Object detection models have been known to employ attention processes in order to improve their localization and identification accuracy by strengthening their focus on relevant portions of an image.*

3. What are the different types of attention mechanism?

There are two types of attention mechanism:

- *Additive Attention computes attention scores by applying a feed-forward neural network to the concatenated query and key vectors.*
- *Dot-Product attention measures attention scores using dot product between the query and key vectors.*

4. What are the two main steps of attention mechanism?

The attention mechanism comprises two main steps:

- *Computing attention scores by measuring the relevance between a query element and all other elements in the input sequence, often using methods like dot-product or additive attention.*
- *Weighted summation is computed based on these attention scores, creating a context vector that emphasizes important input elements.*

These steps enable the model to selectively focus on relevant information.

5. How attention mechanism works?

Attention mechanisms operate by assigning weights to input elements based on their relevance to a specific context or query. The process involves calculating attention scores by comparing query and key vectors, applying a softmax function for normalization, and obtaining a weighted sum of input elements. This weighted sum, or context vector, captures crucial information for the model's decision-making. Attention mechanisms enhance the model's ability to selectively focus on pertinent details, enabling it to capture long-range dependencies and improve performance in various tasks, including natural language processing and computer vision.