1. **What are Sequence-to-sequence models?**

**Seq2Seq model or Sequence-to-Sequence model**, **is a machine learning architecture designed for tasks involving sequential data.** It takes an input sequence, processes it, and generates an output sequence. The architecture consists of two fundamental components: **an encoder**and a **decoder**. Seq2Seq models have significantly improved the quality of **machine translation systems** making them an important technology. The article aims to explore the fundamentals of the seq2seq model and its applications along with its advantages and disadvantages.

**What is Seq2Seq model?**

The seq2Seq model is a kind of machine learning model that takes sequential data as input and generates also sequential data as output. Before the arrival of Seq2Seq models, the[machine translation](https://www.geeksforgeeks.org/nlp-bleu-score-for-evaluating-neural-machine-translation-python/) systems relied on statistical methods and phrase-based approaches. The most popular approach was the use of **phrase-based statistical machine translation (SMT)** systems. That was not able to handle long-distance dependencies and capture global context.

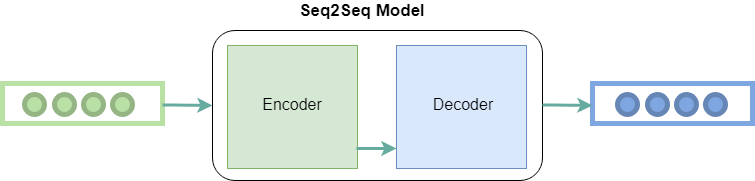
Seq2Seq models addressed the issues by leveraging the power of neural networks, especially [recurrent neural networks (RNN)](https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/). The concept of seq2seq model was introduced in the paper titled “*Sequence to Sequence Learning with Neural* *Networks*” by Google. The architecture discussed in this research paper is fundamental framework for natural language processing tasks. The seq2seq models are encoder-decoder models. The encoder processes the input sequence and transforms it into a fixed-size hidden representation. The decoder uses the hidden representation to generate output sequence. The encoder-decoder structure allows them to handle input and output sequences of different lengths, making them capable to handle sequential data. Seq2Seq models are trained using a dataset of input-output pairs, where the input is a sequence of tokens, and the output is also a sequence of tokens. The model is trained to maximize the likelihood of the correct output sequence given the input sequence.

The advancement in neural networks architectures led to the development of more capable seq2seq model named [transformers](https://www.geeksforgeeks.org/getting-started-with-transformers/). “*Attention is all you need!*” was a research paper that first introduced the transformer model in the era of [Deep Learning](https://www.geeksforgeeks.org/introduction-deep-learning/) after which language-related models have taken a huge leap. The main idea behind the transformers model was that of attention layers and different encoder and decoder stacks which were highly efficient to perform language-related tasks.

Seq2Seq models have been widely used in [NLP](https://www.geeksforgeeks.org/natural-language-processing-overview/)tasks due to their ability to handle variable-length input and output sequences. Additionally, the **attention mechanism** is often used in Seq2Seq models to improve performance and it allows the decoder to focus on specific parts of the input sequence when generating the output.

**What is Encoder and Decoder in Seq2Seq model?**

In the seq2seq model, the Encoder and the Decoder architecture plays a vital role in converting input sequences into output sequences. Let’s explore about each block:



*Encoder and Decoder Stack in seq2seq model*

**Encoder Block**

The main purpose of the encoder block is to process the input sequence and capture information in a fixed-size context vector.

**Architecture:**

* The input sequence is put into the encoder.
* The encoder processes each element of the input sequence using neural networks (or transformer architecture).
* Throughout this process, the encoder keeps an internal state, and the ultimate hidden state functions as the context vector that encapsulates a compressed representation of the entire input sequence. This context vector captures the semantic meaning and important information of the input sequence.

The final hidden state of the encoder is then passed as the context vector to the decoder.

**Decoder Block**

The decoder block is similar to encoder block. The decoder processes the context vector from encoder to generate output sequence incrementally.

**Architecture:**

* In the training phase, the decoder receives both the context vector and the desired target output sequence (ground truth).
* During inference, the decoder relies on its own previously generated outputs as inputs for subsequent steps.

The decoder uses the context vector to comprehend the input sequence and create the corresponding output sequence. It engages in autoregressive generation, producing individual elements sequentially. At each time step, the decoder uses the current hidden state, the context vector, and the previous output token to generate a probability distribution over the possible next tokens. The token with the highest probability is then chosen as the output, and the process continues until the end of the output sequence is reached.

**RNN based Seq2Seq Model**

The decoder and encoder architecture utilizes RNNs to generate desired outputs. Let’s look at the simplest seq2seq model.

For a given sequence of inputs , a RNN generates a sequence of outputs through iterative computation based on the following equation:

Here,

* represents hidden state at time step t
* represents input at time step t
* represents the weight matrix for the input
* represents hidden state from the previous time step (t-1)
* represents the sigmoid activation function.
* represents output at time step t
* represents the wight matrix for the output
* T is the length of the sequence.

Recurrent Neural Networks can easily map sequences to sequences when the alignment between the inputs and the outputs are known in advance. Although the vanilla version of RNN is rarely used, its more advanced version i.e. [LSTM](https://www.geeksforgeeks.org/long-short-term-memory-networks-explanation/) or [GRU](https://www.geeksforgeeks.org/gated-recurrent-unit-networks/) is used. This is because RNN suffers from the problem of vanishing gradient. LSTM develops the context of the word by taking 2 inputs at each point in time. One from the user and the other from its previous output, hence the name recurrent (output goes as input).

**Advantages of seq2seq Models**

* **Flexibility**: Seq2Seq models can handle a wide range of tasks such as machine translation, text summarization, and image captioning, as well as variable-length input and output sequences.
* **Handling Sequential Data:**Seq2Seq models are well-suited for tasks that involve sequential data such as natural language, speech, and time series data.
* **Handling Context:** The encoder-decoder architecture of Seq2Seq models allows the model to capture the context of the input sequence and use it to generate the output sequence.
* **Attention Mechanism:** Using attention mechanisms allows the model to focus on specific parts of the input sequence when generating the output, which can improve performance for long input sequences.

**Disadvantages of seq2seq Models**

* **Computationally Expensive:** Seq2Seq models require significant computational resources to train and can be difficult to optimize.
* **Limited Interpretability:** The internal workings of Seq2Seq models can be difficult to interpret, which can make it challenging to understand why the model is making certain decisions.
* **Overfitting**: Seq2Seq models can overfit the training data if they are not properly regularized, which can lead to poor performance on new data.
* **Handling Rare Words:** Seq2Seq models can have difficulty handling rare words that are not present in the training data.
* **Handling Long input Sequences:** Seq2Seq models can have difficulty handling input sequences that are very long, as the context vector may not be able to capture all the information in the input sequence.

**Applications of Seq2Seq model**

Throughout the article, we have discovered the machine translation is the real-world application of seq2seq model. Let’s explore more applications:

* **Text Summarization:** The seq2seq model effectively understands the input text which makes it suitable for news and document summarization.
* **Speech Recognition:** Seq2Seq model, especially those with attention mechanisms, excel in processing audio waveform for ASR. They are able to capture spoken language patterns effectively.
* **Image Captioning:**The seq2seq model integrate image features from CNNs with textual generation capabilities for image captioning. They are capable to describe images in a human readable format.

1. **What are the Problem with Vanilla RNNs?**

**Vanilla RNNs have limited representational power, which can make it difficult for them to learn complex patterns in sequential data.**

**Due to the limitations of short-term memory and vanishing gradients, vanilla RNNs have difficulty in learning long sequences.**

1. **What is Gradient clipping?**

**Gradient Clipping** is the process that helps maintain numerical stability by preventing the gradients from growing too large. When training a neural network, the loss gradients are computed through backpropagation. However, if these gradients become too large, the updates to the model weights can also become excessively large, leading to numerical instability. This can result in the model producing NaN (Not a Number) values or overflow errors, which can be problematic. This problem is often referred to as ‘**gradient exploding’**, it could be solved by clipping the gradient to the value that we want it to be. Let’s thoroughly discuss gradient clipping.

**Gradient Clipping in Deep Learning**

Gradient Clipping is a technique used during the training of [neural networks](https://www.geeksforgeeks.org/neural-networks-a-beginners-guide/) to address the issue of [exploding gradients](https://www.geeksforgeeks.org/vanishing-and-exploding-gradients-problems-in-deep-learning/). When the gradients of the [loss function](https://www.geeksforgeeks.org/ml-common-loss-functions/)concerning the parameters become too large, it can cause the model’s weights to be updated by huge amounts, leading to numerical instability and a slow or even halted convergence of the training process. By using Gradient Clipping, we can maintain numerical stability by preventing the gradients from growing too large, thus improving the model’s overall performance.

Gradient clipping is a very effective technique that helps address the exploding gradient problem during training. By limiting the magnitude of the gradients, it helps to prevent them from growing unchecked and becoming too large. This ensures that the model learns more effectively and prevents it from getting stuck in a local minima. The clip value or clip threshold is an important parameter that determines how aggressively the gradients are scaled down.

**How does Gradient Clipping work?**

Let’s discuss the step-by-step description of gradient clipping:

1. **Calculate Gradients**:  
   When the model is learning, it’s like a student taking an exam. [Backpropagation](https://www.geeksforgeeks.org/backpropagation-in-data-mining/) is like a teacher grading the exam and giving feedback to the student. It calculates the gradients of the model’s parameters with respect to the loss function, helping the model learn and improve its performance. So, think of backpropagation as a helpful teacher guiding the model to success!
2. **Compute Gradient Norm**:  
   To measure the magnitude of the gradients, we can use different types of norms such as the L2 norm (also known as the Euclidean norm) or the L1 norm. These norms help us to quantify the size of the gradients and understand how fast the parameters are changing. The L2 norm calculates the square root of the sum of the squares of the individual gradients, while the L1 norm calculates the sum of the absolute values of the gradients. By measuring the norm of the gradients, we can monitor the training process and adjust the learning rate accordingly to ensure that the model is converging efficiently.
3. **Clip Gradients**:  
   If the computed gradient norm exceeds the predefined clip threshold, the gradients are scaled down to ensure that the norm does not exceed this threshold. The scaling factor is determined by dividing the clip threshold by the gradient norm.
   * The clipped gradients become,.
4. **Update Model Parameters**:  
   The clipped gradients are used to update the model parameters. By using the clipped gradients to update the model parameters, we can prevent the weights from being updated by excessively large amounts, which can lead to numerical instability and slow down the training process. This helps to ensure that the model is learning effectively and converging towards a good solution.

The **clip\_threshold**discussed here is a type of hyperparameter whose value could be determined by experimenting on the dataset present in front of us.

**Types of Gradient Clipping Techniques**

There are two different gradient clipping techniques that are used, gradient clipping by value and gradient clipping by norm, let’s discuss them thoroughly.

**Clipping by Value:**

‘Clipping by value’ is the most straightforward and effective gradient clipping technique, in this method the gradients are individually clipped so that they lie in the predefined range that is mentioned. This technique is done elementwise, so each component of the gradient vector is clipped individually. In this gradient clipping technique, the minimum and maximum thresholds are defined, and the range is set accordingly so that the gradient’s value lies in between the minimum and maximum value.

After the computation of the gradients through backpropagation, inspection of the gradient component is done, if the gradient component is greater than the maximum threshold it’s value is set to maximum threshold and if the gradient component is lower than the minimum threshold value mentioned then the value of the gradient component is set to minimum threshold value and if the value of the gradient component lies in between the range of minimum and maximum threshold value then the gradient component is set as it is and not changed.

**Clipping by Norm:**

In the ‘clipping by norm’ technique of gradient clipping the gradients are clipped if their norm (or their size) is greater than the specified threshold value. In contrast to the ‘clipping by value’ here in this case the values of the gradients greater than or less than the threshold values are not set to the threshold values. It makes sure that the norm of the updated gradients remains small and manageable, and the learning process is more stable. There are different types of ‘clipping by norm’ techniques let’s explore them one by one.

* **L2 Norm Clipping**:  
  In this form of norm clipping technique the gradient value is clipped down if it’s L2 norm (Euclidean norm) exceeds the predefined threshold value. The L2 norm or the Euclidean norm is calculated as the square root of the squared values of its components. Considering gradient vector as **g** = [] whereis the gradient with respect to theparameter of the model and n is the total number of model parameters. Therefore, the L2 norm is represented as:  
    
  Now, if the L2 norm exceeds the threshold value the upgraded gradient after clipping of the components becomes:
* **L1 Norm Clipping**:  
  L1 norm technique of gradient is similar to the L2 norm gradient clipping technique, in this technique if the L1 norm exceeds the threshold that we defined in alignment with our specific requirements. L1 norm of a gradient is the sum of the absolute values of all its components. Therefore, the L1 norm is represented as:  
    
  If the L1 norm exceeds the threshold value, the upgraded gradient after clipping becomes:

Gradient clipping by norm provides a more global control over the gradients, and it is often used to address the exploding gradient problem.

**Necessity of Gradient Clipping**

Gradient Clipping is a crucial step in the training of neural networks since it helps in addressing the issue of exploding gradients. The exploding gradients problem arises when the gradients in the backpropagation process becomes excessively large in value, causing instability in training of model. Now we will see some of the most important points which explains the necessity of gradient clipping in neural network model training.

1. **Stability of Training**: During the training of neural network, the optimization algorithm adjusts the value of model parameters with the help of obtained gradients. If the gradients are obtained to be too large the weights of the model updates by a large value causing the model to oscillate and diverge instead of converging to an optimal solution. Whereas gradient clipping limits the size of the gradient and eliminating this issue of instability in the model.
2. **Improving Generalization**: Large gradients might cause the model to overfit to the training data, which in turn might capture more noise and makes the model bad at generalization. Gradient clipping removes this hindrance and makes the model generalize better on new data preventing extreme updates.
3. **Convergence to Optimal Solution**: Exploding gradient prevents the model to converge to optimal solution and instead produces more unstable modelling of data. By clipping the gradient values, we can suspend the possibility of instability, the model gets better at navigating the parameter space enabling consistent progress toward optimal solution.
4. **Compatibility with Activation Function**: Some of the [activation functions](https://www.geeksforgeeks.org/activation-functions-neural-networks/) such as ‘Sigmoid’ and ‘tanh’ functions are sensitive to large input. Gradient clipping ensures the gradient passed through the activation function is within a reasonable range which also helps in removing undesirable behavior like saturation.
5. **Mitigating Vanishing Gradient Problem**: Sometimes the gradient of the loss function with respect to the weights become extremely small which causes the weight to stop updating or even halt the process of training the model. Norm-based gradient clipping helps in preventing the vanishing gradient problem by maintaining the range of value for the gradient that is effective for training the model.
6. **Explain Attention mechanism**

# ML – Attention mechanism

**Last Updated :**28 Nov, 2023

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](https://www.geeksforgeeks.org/deep-learning-tutorial/). During translation or question-answering activities, attention is used in [natural language processing](https://www.geeksforgeeks.org/natural-language-processing-nlp-tutorial/) to align pertinent portions of the source phrase. Without necessarily relying on [reinforcement learning](https://www.geeksforgeeks.org/what-is-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](https://www.geeksforgeeks.org/computer-vision-introduction/) that enables models to systematically identify particular portions of an image for processing.

## Attention Mechanism

An attention mechanism is an [Encoder-Decoder](https://www.geeksforgeeks.org/difference-between-encoder-and-decoder/) kind of [neural network architecture](https://www.geeksforgeeks.org/artificial-neural-networks-and-its-applications/) 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](https://www.geeksforgeeks.org/what-is-reinforcement-learning/) to focus attention on key areas of the image. A [recurrent neural network](https://www.geeksforgeeks.org/introduction-to-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](https://www.geeksforgeeks.org/image-caption-generator-using-deep-learning-on-flickr8k-dataset/) 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](https://www.geeksforgeeks.org/machine-learning/) and [natural language processing](https://www.geeksforgeeks.org/natural-language-processing-nlp-tutorial/) 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.  
     
   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.
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.  
   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.

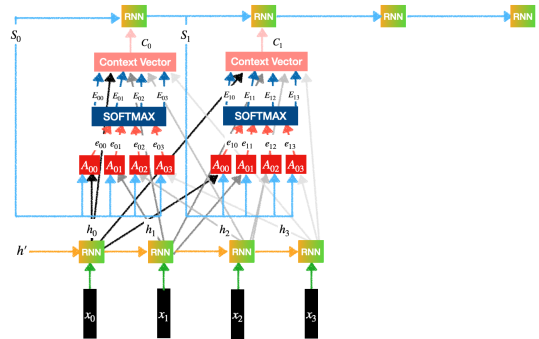
By incorporating an attention mechanism, the model can effectively capture dependencies, emphasize important information, and adaptively focus on different elements of the input, leading to improved performance in tasks such as machine translation, text summarization, or image recognition.

### 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

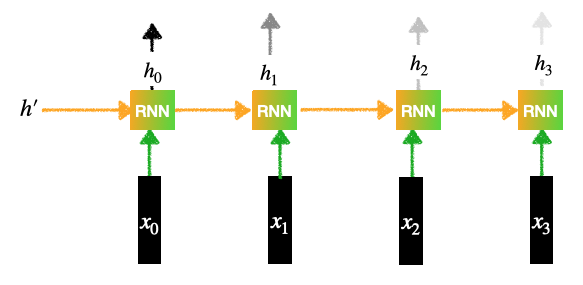
Consider the following Encoder-Decoder architecture with Attention.



*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*

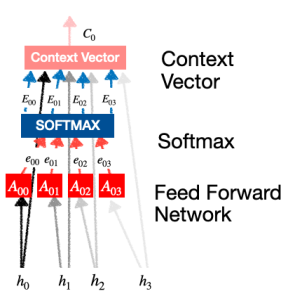
**Contains an RNN layer (Can be LSTMs or GRU):**

1. Let’s, there are 4 words sentence then inputs  will be:
2. Each input goes through an Embedding Layer, It can be RNN, LSTM, GRU or trnasformers
3. Each of the inputs generates a hidden representation.
4. This generates the outputs for the Encoder:

### ****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:

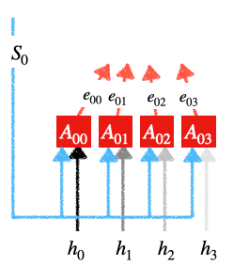
* Our goal is to generate the context vectors.
* For example, context vector tells us how much importance/ attention should be given the inputs: .
* This layer in turn contains 3 subparts:
  + Feed Forward Network
  + Softmax Calculation
  + Context vector generation



*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 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: .i.e

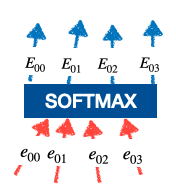
.

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.

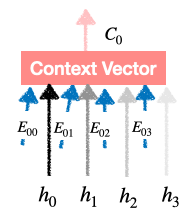


*softmax calculation*

These are called the attention weights. It decides how much importance should be given to the inputs .

#### ****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*

.

We find 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*](https://www.geeksforgeeks.org/natural-language-processing-nlp-tutorial/)*and*[*Computer vision*](https://www.geeksforgeeks.org/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.*

1. **Explain Conditional random fields (CRFs)**

# Conditional Random Fields (CRFs) for POS tagging in NLP

Part of Speech tagging is one of the tasks on which early Language models were tested for the GLUE score. In this article, we will learn about one such method which can be used for POS tagging. But before that let us understand what is POS tagging.

## What is POS tagging?

[Part-of-speech (POS) tagging](https://www.geeksforgeeks.org/nlp-part-of-speech-default-tagging/) is the process of assigning grammatical categories, such as nouns, verbs, adjectives, etc., to each word in a sentence. POS tagging is a fundamental task in [Natural Language Processing (NLP)](https://www.geeksforgeeks.org/natural-language-processing-nlp-tutorial/) and is used in various applications, such as [machine translation](https://www.geeksforgeeks.org/machine-translation-of-languages-in-artificial-intelligence/), [sentiment analysis](https://www.geeksforgeeks.org/what-is-sentiment-analysis/), and [text-to-speech synthesis](https://www.geeksforgeeks.org/convert-text-speech-python/).

Here’s an example of POS tagging for the sentence *“She likes to read books”:*

|  |  |
| --- | --- |
| **Word** | **POS Tag** |
| She | PRON |
| likes | VERB |
| to | PART |
| read | VERB |
| books | NOUN |

In this example, the word “She” is tagged as a pronoun, “likes” is tagged as a verb, “to” is tagged as a particle, “read” is tagged as a verb, and “books” is tagged as a noun. The POS tags provide information about the syntactic structure of the sentence, which can be used in downstream tasks, such as parsing or sentiment analysis.

## Conditional Random Fields

A Conditional Random Field (CRF) is a type of probabilistic graphical model often used in Natural Language Processing (NLP) and computer vision tasks. It is a variant of a Markov Random Field (MRF), which is a type of undirected graphical model.

* CRFs are used for structured prediction tasks, where the goal is to predict a structured output based on a set of input features. For example, in NLP, a commonly structured prediction task is Part-of-Speech (POS) tagging, where the goal is to assign a part-of-speech tag to each word in a sentence. CRFs can also be used for [Named Entity Recognition](https://www.geeksforgeeks.org/named-entity-recognition/) (NER), chunking, and other tasks where the output is a structured sequence.
* CRFs are trained using maximum likelihood estimation, which involves optimizing the parameters of the model to maximize the probability of the correct output sequence given the input features. This optimization problem is typically solved using iterative algorithms like gradient descent or L-BFGS.
* The formula for a Conditional Random Field (CRF) is similar to that of a Markov Random Field (MRF) but with the addition of input features that condition the probability distribution over output sequences.

Let X be the input features and Y be the output sequence. The joint probability distribution of a CRF is given by:

where:

* **Z(X)** is the normalization factor that ensures the distribution sums to 1 over all possible output sequences.
* **λk** are the learned model parameters.
* **fk(yi – 1, yi, xi**) are the feature functions that take as input the current output state **yi**, the previous output state**yi – 1**, and the input features **xi**.
* These functions can be binary or real-valued, and capture dependencies between the input features and the output sequence.

Here’s an example of using Conditional Random Fields (CRFs) for POS tagging in Python using the sklearn\_crfsuite library. First, you’ll need to install the sklearn\_crfsuite library using ‘pip’:

pip install sklearn-crfsuite

**‘sklearn-crfsuite’** is a Python library that provides an interface to the CRFsuite implementation of Conditional Random Fields (CRFs), a popular machine learning algorithm for sequence labeling tasks such as Part-Of-Speech (POS) tagging and named entity recognition (NER). The library is built on top of scikit-learn, a popular machine-learning library for [Python](https://www.geeksforgeeks.org/python-programming-language/).

1. **Explain self-attention**

**NLP models, especially transformer models, use a mechanism called self-attention, which is also referred to as scaled dot-product attention. When generating predictions, it enables the model to assign varying weights to distinct words within a sequence. The attention mechanism weighs words according to how relevant they are to the word that is being considered at that moment.**

1. **What is Bahdanau Attention?**

**Bahdanau attention, also known as Additive attention, is a type of attention mechanism that was first introduced by**[**D. Bahdanau**](https://scholar.google.de/citations?user=Nq0dVMcAAAAJ&hl=en)**, K. Cho, and Y. Bengio in their paper “**[**Neural Machine Translation by Jointly Learning to Align and Translate**](https://arxiv.org/abs/1409.0473)**” in 2015.**

**Unlike the Luong attention case, the authors of the paper propose a more complex way than just using a mathematical approach to compute the attention weights by conducting a linear combination of encoder and decoder states. Bahdanau technique employs and trains a**[**feed-forward neural network**](https://www.baeldung.com/cs/neural-networks-backprop-vs-feedforward)**. Moreover, the current state of the decoder, the previous state of the attention mechanism, and the current input are driven through the neural network.**

**Next, the attention weights are utilized to produce the weighted sum of the features of the input. This weighted sum is provided as additional information to the decoder, allowing it to focus on important input factors when producing a single output.**

1. **What is a Language Model?**

**Language models are a fundamental component of natural language processing (NLP) systems. A language model is a statistical model that assigns probabilities to sequences of words, allowing it to predict which word or sequence of words is most likely to occur next given the previous words.**

**Language models play a crucial role in many NLP applications:**

* **Predictive text input (auto-complete)**
* **Speech recognition**
* **Machine translation**
* **Spelling and grammar correction**
* **Generating human-like text**

**There are different types of language models in language modeling in NLP, from relatively simple n-gram models that consider only the last n words, to advanced neural network models like transformers that can capture long-range contextual dependencies.**

**What sets language models apart is their ability to quantify linguistic knowledge in a statistical framework that computers can process. This allows NLP systems to generate, understand, and translate natural language with increasing fluency and accuracy using language models in NLP.**

**Moreover, large language models pretrained on vast text data have emerged as a powerful basis for transfer learning to many downstream NLP model tasks in language modeling in NLP, substantially advancing the field’s capabilities.**

1. **What is Multi-Head Attention?**

The Multi-Head Attention is a central mechanism in Transformer just skip-joining in ResNet50 architecture. Sometimes there are multiple other points in the sequence to be attended to. Using the approach of finding an overall average will not distribute the weights around so that diverse value is given as weights as we would want. This brings about the idea of having an extension of creating individual attention mechanisms to multiple heads resulting in multiple attention mechanisms. The implementation now presents multiple different query-key-value triplets on a single feature.

The computations are performed in such a way that the attention module iterates over a number of times, organizing into parallel layers known as attention heads. Each separate head independently processes both the input sequence and the associated output sequence element. The cumulative scores from each head are then combined to obtain a final attention score, which incorporates every detail of the input sequence.

1. **What is Bilingual Evaluation Understudy (BLEU)**

# Bilingual Evaluation Understudy(BLEU)

**BLEU score measures the quality of predicted text, referred to as the candidate, compared to a set of references. There can be more than one correct/reference for one candidate in Sequence to Sequence tasks. Hence, it is important that the references are chosen carefully and all the possible references are included. BLEU score is a precision based measure and it ranges from 0 to 1. The closer the value is to 1, the better the prediction. It is not possible to achieve a value of 1 and usually a value higher than 0.3 is considered a good score.**

**In the next section, we see the calculation of BLEU on a single predicted sentence for illustration followed by the calculation for a corpus.**