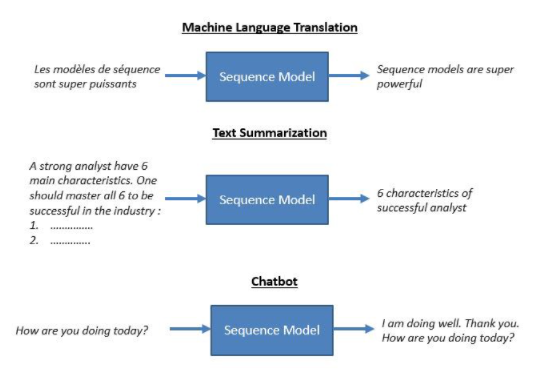
**Assignment 05**

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

**Ans:** Sequence to Sequence (often abbreviated to seq2seq) models is a special class of Recurrent Neural Network architectures .Seq2Seq (Sequence-to-Sequence) is a type of model in machine learning that is used for tasks such as [machine translation](https://www.geeksforgeeks.org/machine-translation-of-languages-in-artificial-intelligence/), text summarization, and image captioning. The model consists of two main components:

* Encoder
* Decoder

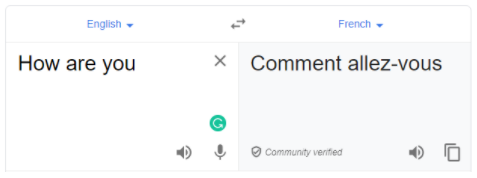
Seq2Seq models are trained using a dataset of input-output pairs, where the input is a sequence of tokens and the output is also a sequence of tokens. The model is trained to maximize the likelihood of the correct output sequence given the input sequence.



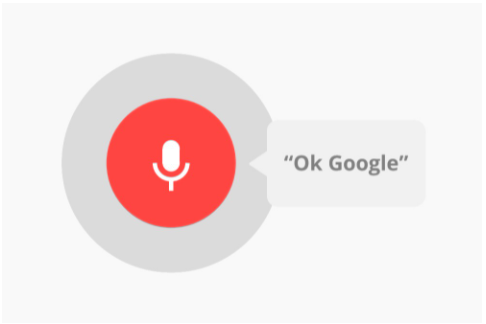
**Use Cases of the Sequence to Sequence Models:** Sequence to sequence models lies behind numerous systems that you face on a daily basis. For instance, seq2seq model powers applications like Google Translate, voice-enabled devices, and online chatbots.

The following are some of the applications:

* Machine translation — a 2016 paper from Google shows how the seq2seq model’s translation quality “approaches or surpasses all currently published results”.



* Speech recognition — another Google paper that compares the existing seq2seq models on the speech recognition task.



These are only some applications where seq2seq is seen as the best solution. This model can be used as a solution to any sequence-based problem, especially ones where the inputs and outputs have different sizes and categories.

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

**Ans** Vanilla RNNs are the simplest form of RNNs, where each hidden unit receives inputs from the current and previous time steps. Vanilla RNNs can learn short-term dependencies, but they struggle with long-term dependencies, meaning they forget or ignore information that is too far back in the sequence. This is due to the problem of [vanishing or exploding gradients](https://www.linkedin.com/advice/3/how-do-you-deal-vanishing-exploding?trk=article-ssr-frontend-x-article_little-text-block), which occurs when the backpropagation algorithm fails to update the weights of the network effectively.

**3. What is Gradient clipping?**

**Ans:** Gradient clipping is a technique to prevent exploding gradients in very deep networks, usually in recurrent neural networks tha is Gradient Clipping handles one of the most difficult challenges in Backpropagation for Neural Networks: computing gradients Gradient clipping reduces the amplitude of the gradient and improves the behavior of stochastic gradient descent (SGD) near cliffs:

* In recurrent networks, high cliffs are typical in the area where the recurrent network behaves roughly linearly.
* SGD without gradient clipping exceeds the landscape minimum, whereas SGD with gradient clipping falls below it.
* gradient clipping is a technique that handles or avoids exploding gradients.
* Gradient clipping will ‘clip’ the gradients or cap them to a Threshold value to prevent the gradients from getting too large.
* The basic principle of gradient clipping is to rescale the size and value of the gradient, bringing it down to the appropriate scale.
* If the gradient gets too large, we rescale it to keep it appropriate. More precisely, if ‖**g**‖ ≥ *c*, then
* **g** ↤ *c*· **g**/‖**g**‖
* where *c* is a hyperparameter, **g**is the gradient, and ‖**g**‖ is the norm of **g**. Since **g**/‖**g**‖ is a unit vector, after rescaling the new **g**will have norm *c*. Note that if ‖**g**‖ < *c*, then we don’t need to do anything.
* Gradient clipping ensures the gradient vector **g** has norm at most *c*.
* The clipping method helps gradients to have a reasonable functionality and be consistent in the data training process.

**4. Explain Attention mechanism**

**Ans:** The attention mechanism was introduced to improve the performance of the encoder-decoder model for machine translation. The idea behind the attention mechanism was to permit the decoder to utilize the most relevant parts of the input sequence in a flexible manner, by a weighted combination of all the encoded input vectors, with the most relevant vectors being attributed the highest weights.

The attention mechanism was introduced by Bahdanau et al. (2014), to address the bottleneck problem that arises with the use of a fixed-length encoding vector, where the decoder would have limited access to the information provided by the input. This is thought to become especially problematic for long and/or complex sequences, where the dimensionality of their representation would be forced to be the same as for shorter or simpler sequences.

that Bahdanau et al.’s *attention mechanism* is divided into the step-by-step computations of the *alignment scores*, the *weights,* and the *context vector*:

1. **Alignment scores**: The alignment model takes the encoded hidden states, ℎ, and the previous decoder output, −1, to compute a score, that indicates how well the elements of the input sequence align with the current output at the positionThe alignment model is represented by a function,which can be implemented by a feedforward neural network:
2. **Weights**: The weights are computed by applying a softmax operation to the previously computed alignment scores:
3. **Context vector**: A unique context vector, is fed into the decoder at each time step. It is computed by a weighted sum of all,  encoder hidden states:

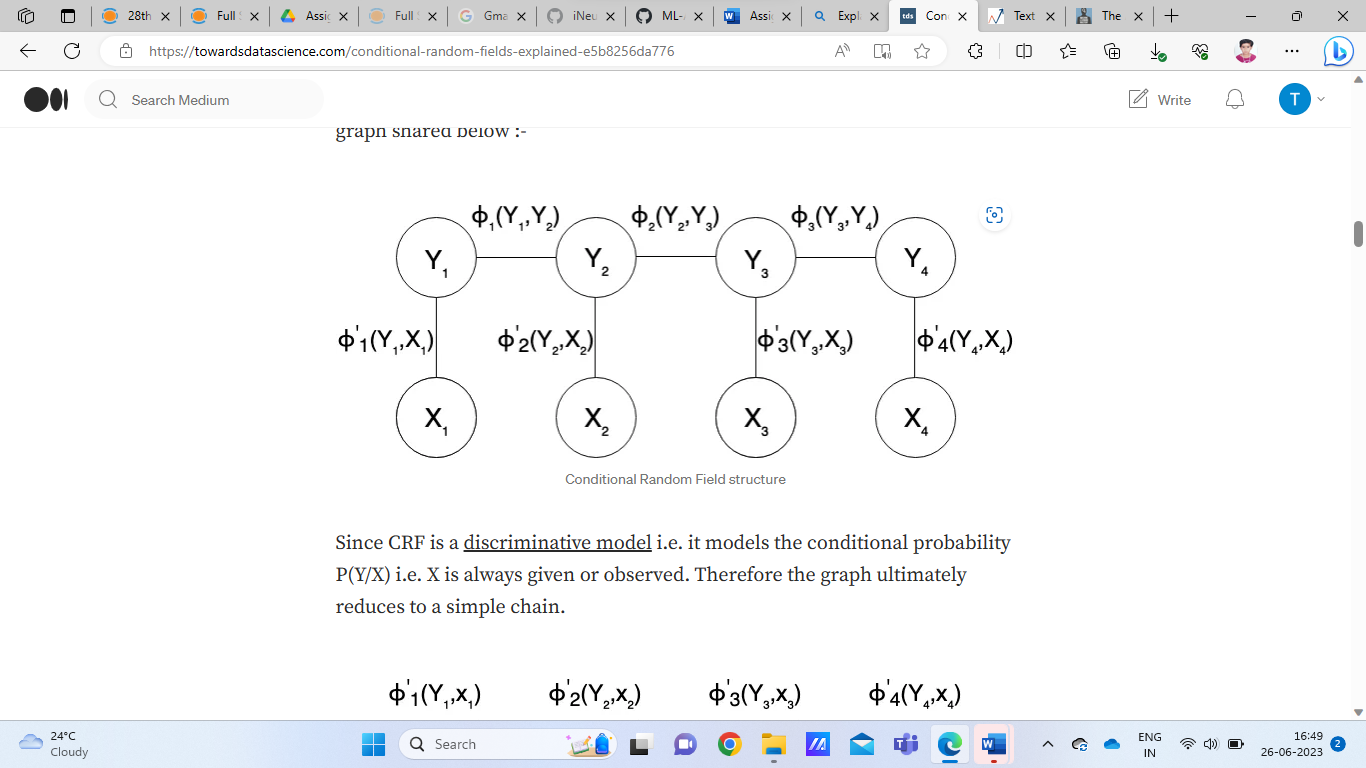
**5. Explain Conditional random fields (CRFs)**

# **Ans:** Conditional random fields (CRFs) are a class of statistical modeling methods often applied in pattern recognition and machine learning and used for structured prediction. Whereas a classifier predicts a label for a single sample without considering "neighbouring" samples, a CRF can take context into account. **Conditional Random Field Model**

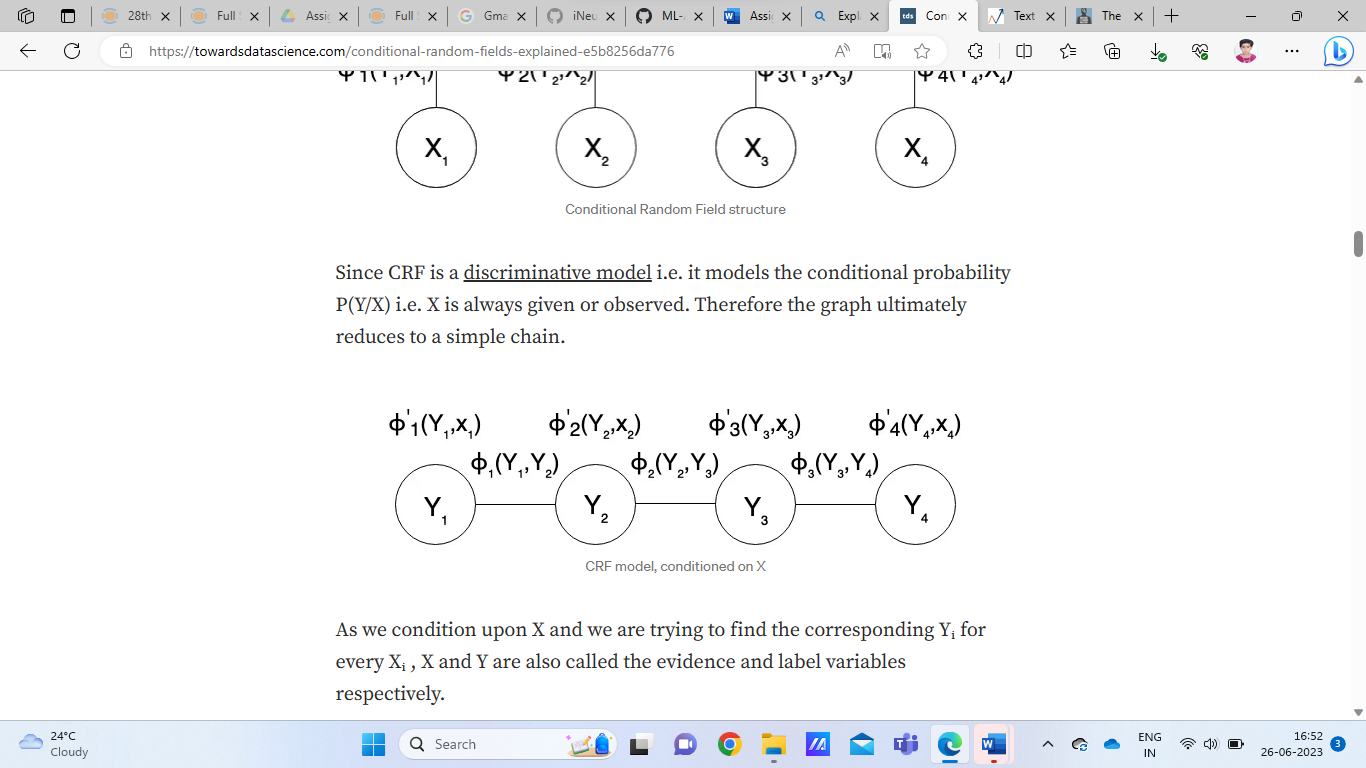
For a moment, let us assume a Markov Random Field and divide it into two sets of random variables Y and X respectively.

[Conditional Random Field](https://en.wikipedia.org/wiki/Conditional_random_field#Description) is a special case of Markov Random field wherein the graph satisfies the property : “When we condition the graph on X globally i.e. when the values of random variables in X is fixed or given, all the random variables in set Y follow the Markov property p(Yᵤ/X,Yᵥ, u≠v) = p(Yᵤ/X,Yₓ, Yᵤ~Yₓ), where Yᵤ~Yₓ signifies that Yᵤ and Yₓ are neighbors in the graph.” A variable’s neighboring nodes or variables are also called the [Markov Blanket](https://en.wikipedia.org/wiki/Markov_blanket) of that variable.

One such graph that satisfies the above property is the chain-structured graph shared below :-

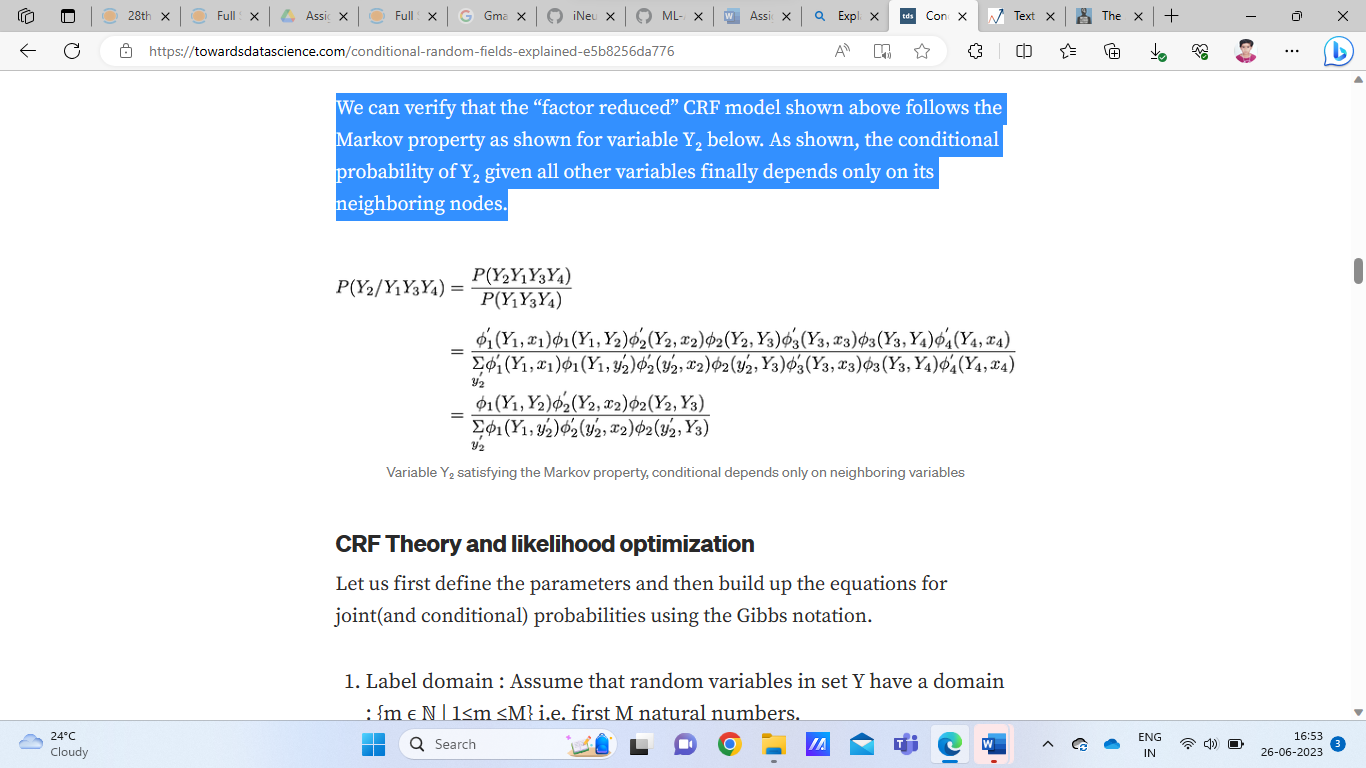


Since CRF is a [discriminative model](https://en.wikipedia.org/wiki/Discriminative_model) i.e. it models the conditional probability P(Y/X) i.e. X is always given or observed. Therefore the graph ultimately reduces to a simple chain.



As we condition upon X and we are trying to find the corresponding Yᵢ for every Xᵢ , X and Y are also called the evidence and label variables respectively.

We can verify that the “factor reduced” CRF model shown above follows the Markov property as shown for variable Y₂ below. As shown, the conditional probability of Y₂ given all other variables finally depends only on its neighboring nodes.

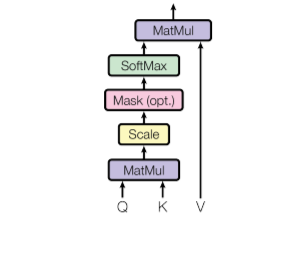


 The joint(and conditional) probabilities using the Gibbs notation are calculated and then the training problem reduces to maximizing the log likelihood wrt all model parameters Wcc’ and W’cs.

**6. Explain self-attention**

**Ans:** Self-attention was proposed by researchers at Google Research and Google Brain Self Attention, also called intra Attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. It has been shown to be very useful in machine reading, abstractive summarization, or image description generation.

**Self-attention mechanism:**



*Self-Attention*

The attention mechanism allows output to focus attention on input while producing output while the self-attention model allows inputs to interact with each other (i.e calculate attention of all other inputs wrt one input.

* The first step is multiplying each of the encoder input vectors with three weights matrices (W(Q), W(K), W(V)) that we trained during the training process. This matrix multiplication will give us three vectors for each of the input vector: the key vector, the query vector, and the value vector.
* The second step in calculating self-attention is to multiply the Query vector of the current input with the key vectors from other inputs.
* In the third step, we will divide the score by square root of dimensions of the key vector (dk). In the paper the dimension of the key vector is 64, so that will be 8. The reason behind that is if the dot products become large, this causes some self-attention scores to be very small after we apply softmax function in the future.
* In the fourth step, we will apply the softmax function on all self-attention scores we calculated wrt the query word (here first word).
* In the fifth step, we multiply the value vector on the vector we calculated in the previous step.
* In the final step, we sum up the weighted value vectors that we got in the previous step, this will give us the self-attention output for the given word.

The above procedure is applied to all the input sequences. Mathematically, the self-attention matrix for input matrices (Q, K, V) is calculated as:

A screenshot of a computer

Description automatically generated

**7. What is Bahdanau Attention?**

**Ans:** mechanism has inherited its name from the first author of the paper in which it was published.

Bahdanau et al. (2014) argue that encoding of a variable-length input into a fixed-length vector squashes the information of the source sentence, irrespective of its length, causing the performance of a basic encoder-decoder model to deteriorate rapidly with an increasing length of the input sentence. The approach they propose, on the other hand, replaces the fixed-length vector with a variable-length one, to improve the translation performance of the basic encoder-decoder model.

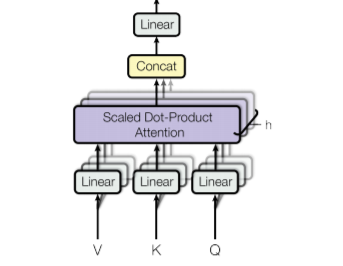
*Attention mechanism* is divided into the step-by-step computations of the *alignment scores*, the *weights,* and the *context vector*:

1. **Alignment scores**: The alignment model takes the encoded hidden states, ℎ, and the previous decoder output, −1, to compute a score, that indicates how well the elements of the input sequence align with the current output at the positionThe alignment model is represented by a function,which can be implemented by a feedforward neural network:
2. **Weights**: The weights are computed by applying a softmax operation to the previously computed alignment scores:
3. **Context vector**: A unique context vector, is fed into the decoder at each time step. It is computed by a weighted sum of all,  encoder hidden states:

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

**Ans:** Language modeling (LM) is the use of various statistical and probabilistic techniques to determine the probability of a given sequence of words occurring in a sentence. Language models analyze bodies of text data to provide a basis for their word predictions. They are used in natural language processing (NLP) applications, particularly ones that generate text as an output. Some of these applications include , machine translation and question answering.

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

**Ans:** Multi-head Attention is a module for attention mechanisms which runs through an attention mechanism several times in parallel. The independent attention outputs are then concatenated and linearly transformed into the expected dimension. Intuitively, multiple attention heads allows for attending to parts of the sequence differently (e.g. longer-term dependencies versus shorter-term dependencies).   


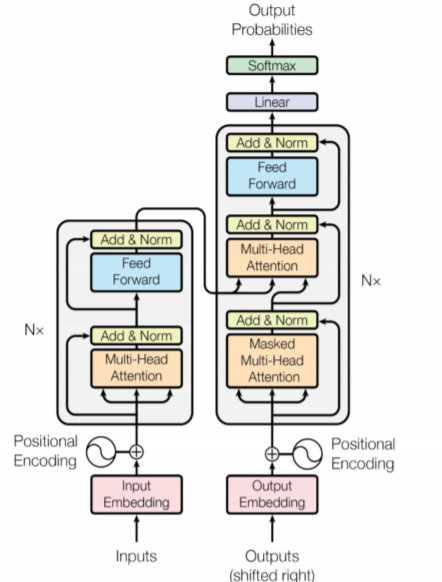
*Multi-headed attention*

In the attention paper, the authors proposed another type of attention mechanism called multi-headed attention. Below is the step-by-step process to calculate multi-headed self-attention:

* Take each word of input sentence and generate the embedding from it.
* In this mechanism, we created *h* (h = 8) different attention heads, each head has different weight matrices (W(Q), W(K), W(V)).
* In this step, we multiply the input matrix with each of the weight matrices (WQ, WK, WV) to produce the key, value, and query matrices for each attention head.
* Now, we apply the attention mechanism to these query, key, and value matrices, this gives us an output matrix from each attention head.
* In this step, we concatenate the output matrix obtained from each attention heads and dot product with the weight WO to generate the output of the multi-headed attention layer.

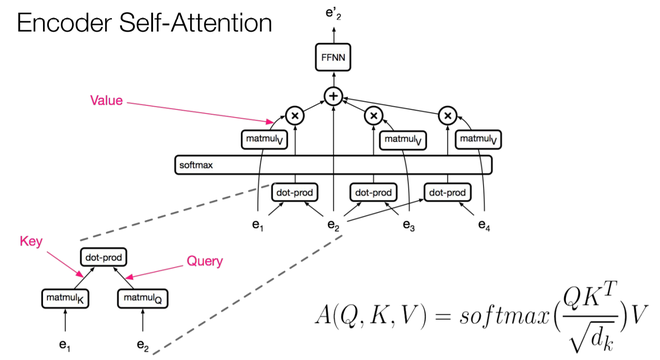
Mathematically multi-head attention can be represented by:

**Attention in Transformer architecture:**



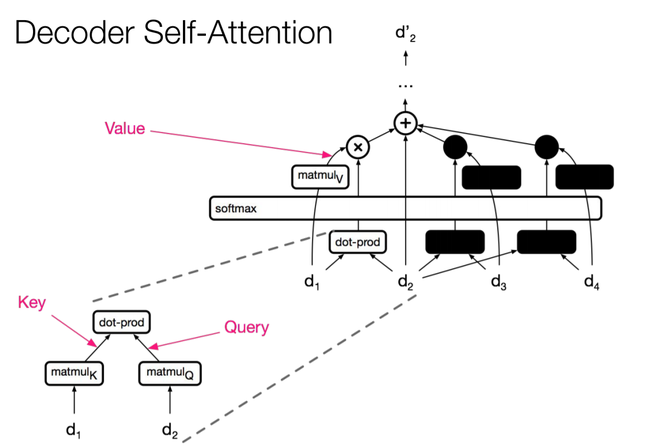
**–**The transformer architecture uses attention model uses multi-headed attention at three steps:

* The first is encoder-decoder attention layers, in this type of layer the queries come from the previous decoder layer while the keys and values come from the encoder output. This allows each position in the decoder to give attention to all the positions of the input sequence.
* The second type is the self-attention layer contained in the encoder, this layer receives key, value, and query input from the output of the previous encoder layer. Each position in the encoder can get attention score from every position in the previous encoder layer.

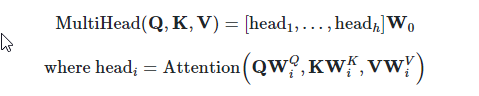


*Self-attention in Encoder*

* The third type is the self-attention in the decoder, this is similar to self-attention in encoder where all queries, keys, and values come from the previous layer. The self-attention decoder allows each position to attend each position up to and including that position. The future values are masked with (-Inf). This is known as masked-self attention.



*Self-attention in Decoder*



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

**Ans:** Neural Machine Translation (NMT) is a standard task in NLP that involves translating a text from a source language to a  target language. BLEU (Bilingual Evaluation Understudy) is a score used to evaluate the translations performed by a machine translator. In this article, we’ll see the mathematics behind the BLEU score and its implementation in Python.

## BLEU Score

As stated above BLEU Score is an evaluation metric for Machine Translation tasks. It is calculated by comparing the n-grams of machine-translated sentences to the n-gram of human-translated sentences. Usually, it has been observed that the BLEU score decreases as the sentence length increases. This, however, might vary depending upon the model used for translation.

A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.

The score was developed for evaluating the predictions made by automatic machine translation systems. It is not perfect, but does offer 5 compelling benefits:

* It is quick and inexpensive to calculate.
* It is easy to understand.
* It is language independent.
* It correlates highly with human evaluation.
* It has been widely adopted.