VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**Nguyen Tan Bao - 521H0438**

**Tang Thieu Phong – 521H0136**

**Ho Huu An – 521H0489**

**FINAL REPORT**

**INTRODUCTION TO NATURAL LANGUAGE PROCESSING**

**HO CHI MINH CITY, 2024**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**Nguyen Tan Bao - 521H0438**

**Tang Thieu Phong - 521H0136**

**Ho Huu An – 521H0489**

**FINAL REPORT**

**INTRODUCTION TO NATURAL LANGUAGE PROCESSING**

Instructor

**Assoc. Prof.**  **Le Anh Cuong**

**HO CHI MINH CITY, 2024**

**ACKNOWLEDGMENTS**

We would like to sincerely thank to Assoc. Prof. Le Anh Cuong for teaching the subject of introduction to natural processing language

*Ho Chi Minh city, 19/07/2024*

*Author*

*Ho Hu An*

*Tang Thieu Phong*

*Nguyen Tan Bao*

**COMPLETED WORKS**

**AT TON DUC THANG UNIVERSITY**

I would like to assure you that this is my own research work and with the scientific guidance of PGS. Le Anh Cuong. The research contents and results in this project are honest and have not been published in any form before. The data in the tables for analysis, comment and evaluation collected by the author himself from different sources are clearly stated in the references.

In addition, the Project also uses a number of comments, evaluations as well as data of other authors, other agencies and organizations with citations and annotations of the source.

**If any fraud is detected, I would like to take full responsibility for the content of my Project.** Ton Duc Thang University is not involved in the copyright and copyright violations caused by me in the process of implementation (if any)**.**

.

*Ho Chi Minh city, 8/8/2024*

*Author*

*Bao*

*Nguyen Tan Bao*

**MỤC LỤC**

[CHAPTER 1: Attention Mechanism in Sequence-to-Sequence Models 1](#_Toc30587)

[1.1 Model base on LSTM (LSTM-based Seq2Seq) 1](#_Toc15369)

[1.1.1 Model Architecture 1](#_Toc14073)

[1.1.2 Attention in LSTM: 2](#_Toc2190)

[1.1.3 Attention Formula: 2](#_Toc14355)

[1.2 Transformer Model 3](#_Toc14954)

[1.2.1 Model Architecture: 3](#_Toc18341)

[1.3 Steps to implement the algorithm 4](#_Toc20063)

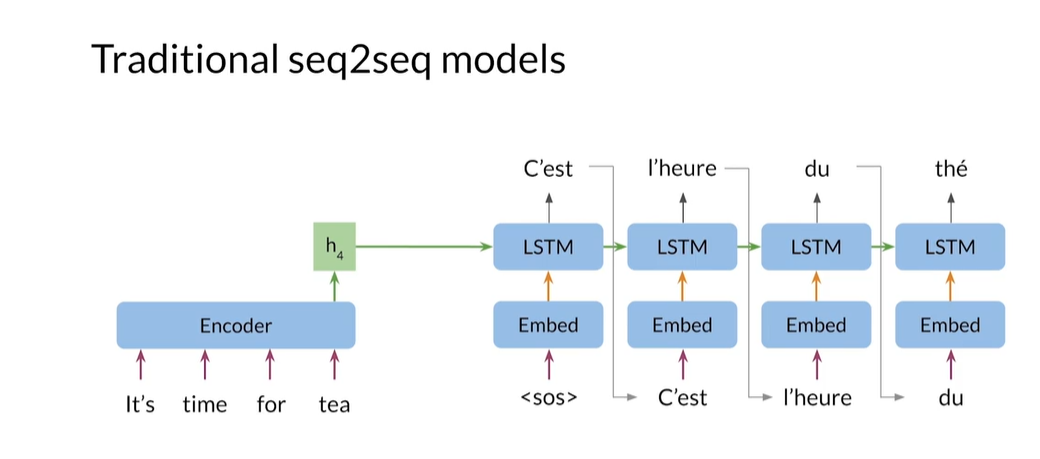
[1.4 Illustrative example 4](#_Toc11611)

[REFERENCES 6](#_Toc31644)

# CHAPTER 1: Attention Mechanism in Sequence-to-Sequence Models

## Model base on LSTM (LSTM-based Seq2Seq)

### Model Architecture



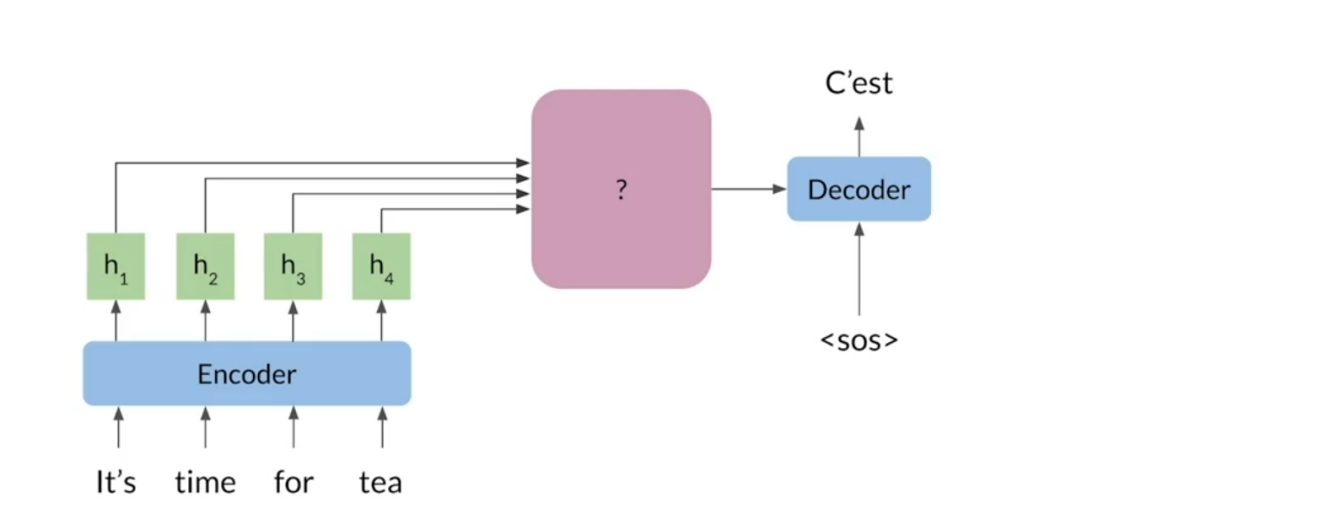
Traditional seek to seek models, use the final hidden states of the encoder as the initial hidden state of the decoder. This forces the encoder to store the meaning of the entire input sequence into this one hidden states.

* It consists of two main components: Encoder and Decoder.
* Encoder: Use LSTM to transform the input into a context vector.
* Decoder: Use LSTM to generate output from contextual vectors.
* Restrict:
* Fixed vector context for the entire sentence resulting in difficulty capturing information for long sentences (Vanishing Gradient)

### Attention in LSTM

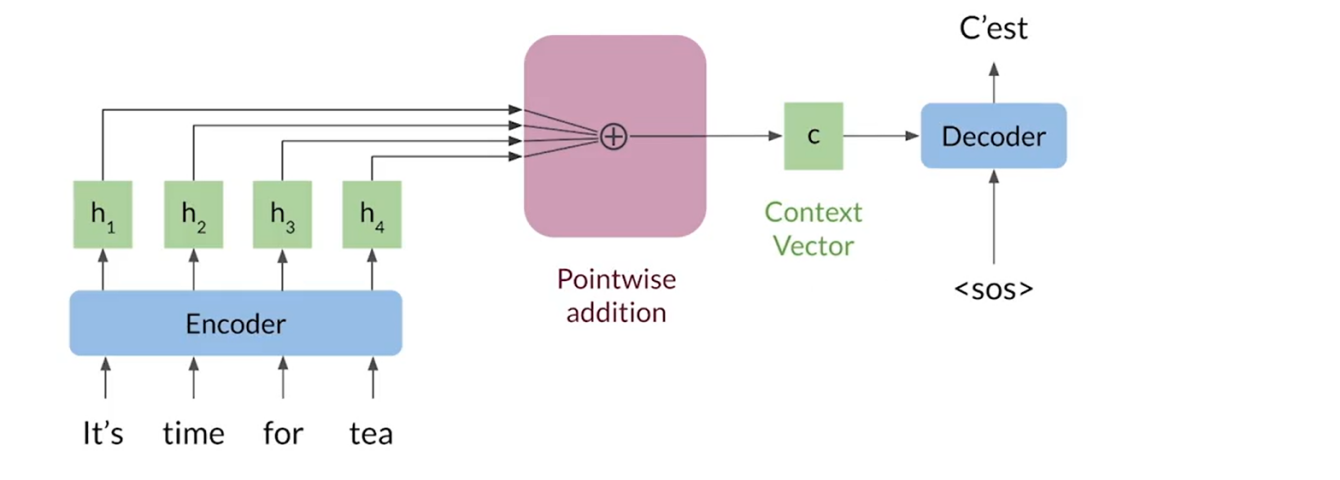
* Attention solves this problem by creating different weights for each part of the input instead of using a fixed vector.
* Idea: At each step of the decoder, the model will "pay attention" to different parts of the input instead of the entire sentence.

Traditional sequence-to-sequence (Seq2Seq) models rely on the final hidden state of the encoder to initialize the hidden state of the decoder. This approach places the burden on the encoder to encapsulate the meaning of the entire input sequence into a single hidden state. An alternative to this method is to provide all the hidden states of the encoder to the decoder. However, this approach is computationally inefficient, as it requires storing the hidden states for every input step in memory.



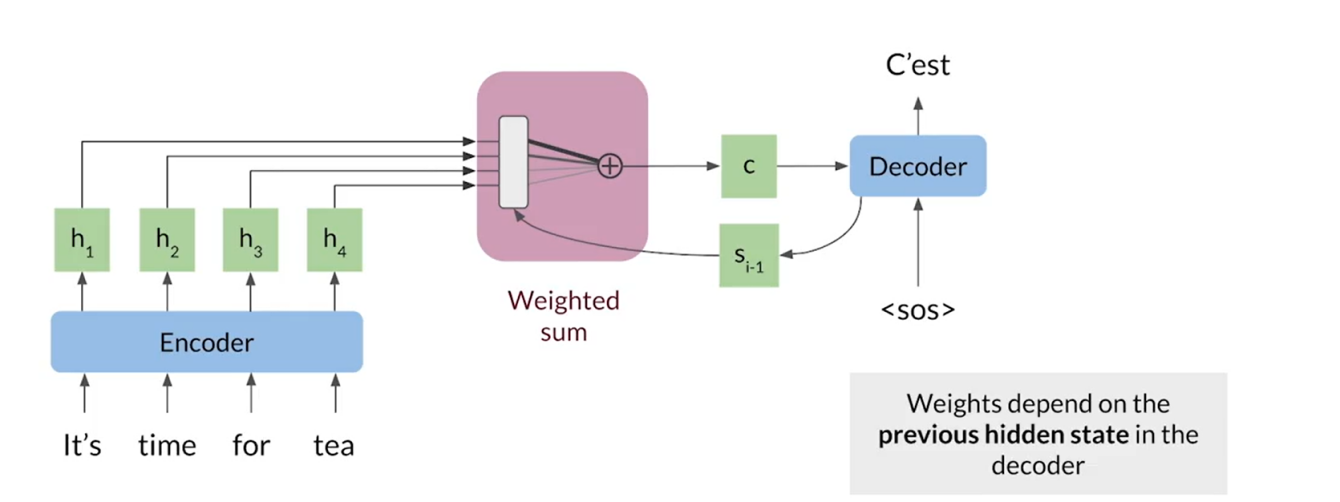
The process of retaining the hidden states for each input step in memory can quickly become inefficient. To address this issue, the hidden states are combined into a single vector, commonly referred to as the context vector

This combination is achieved through a point-wise addition operation. Since all the hidden vectors have the same size, their corresponding elements can be summed together to produce a new vector of the same dimensions. This approach allows the decoder to incorporate information from every step of the sequence. However, for tasks such as predicting the first word, the decoder primarily requires information from the initial input steps. Consequently, this method does not differ significantly from using only the final hidden states in models like LSTM or GRU.



The context vector encapsulates more information about the most significant words in the input sequence and less about the less relevant ones. To determine which words are important at each step, the calculation of weights plays a crucial role. The decoder's previous hidden state, denoted as wps, contains information about the preceding words in the output sequence. This enables a comparison between the decoder's state and each encoder state to identify the most relevant inputs.

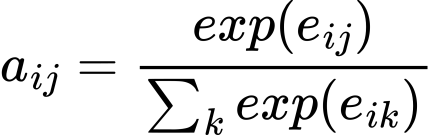
Intuitively, the decoder assigns weights in a way that prioritizes the most significant input words for the next prediction. By doing so, it determines which parts of the input sequence to focus on.



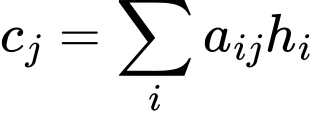
1. Calculate the critical score between the current state of the Decoder wps and each hidden status of encoder wps

wps

1. Use the softmax function to normalize to weights wps

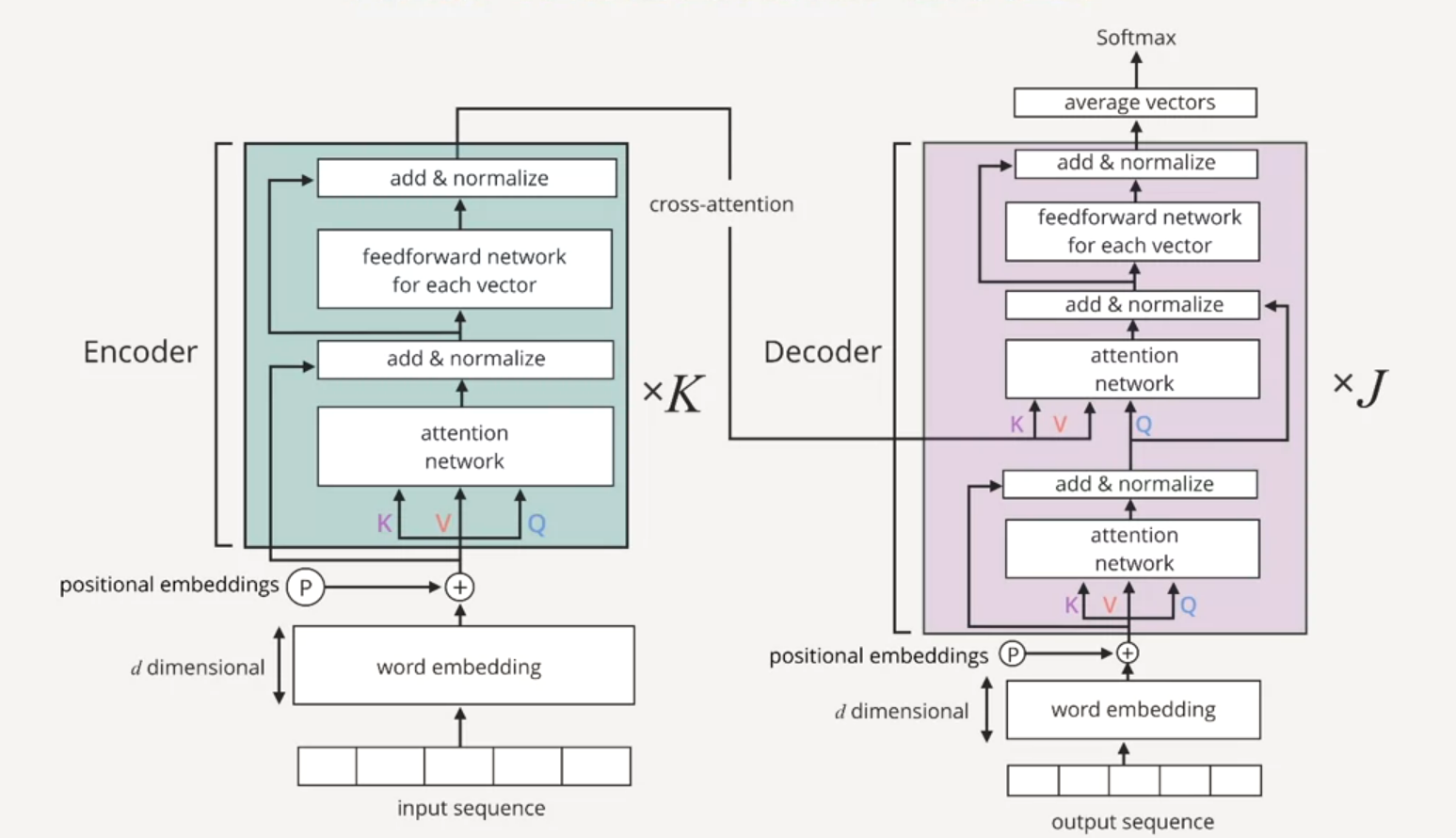


1. Summarize the Encoder's hidden states into a context vector wps



## Transformer Model

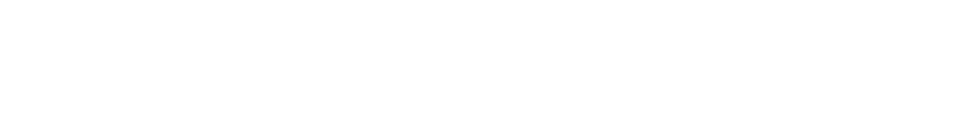
### Model Architecture:



* Transformer completely eliminates the use of RNN/LSTM, instead using basic Attention as the foundation.
* Comprise:
* Encoder: Processing input through Self-Attention and Feed-Forward Neural Networks.
* Decoder: Similar to Encoder but with the addition of Masked Multi-Head Attention to generate step-by-step output.

### Attention in Transformer

* Attention in Transformer is based on Self-Attention and Multi-Head Attention.
* Self-Attention: Allows every token in the input sentence to interact with each other and learn the relationship between them.
* Attention formula:



* Multi-Head Attention: Perform multiple independent Attention on different "heads", then combine the results.

# 

## GPT model (Generative Pretrained Transformer)

### Model Architecture

# REFERENCES