Attention Mechanisms from Basics

Example Sentence

"The cat that was sitting on the mat, which was next to the dog, jumped"

Types of Attention Mechanisms

The four types of attention mechanisms are:

- **Simplified Self Attention**
- **Self Attention**
- **Causal Attention**
- **Multi-Head Attention**

Details of Attention Mechanisms

- **Simplified Self Attention**
  - A simplified self-attention technique to introduce the broad idea.
- **Self Attention**
  - Self-attention with trainable weights, forming the basis of the mechanism used in Large Language Models (LLMs).
- **Causal Attention**
  - A type of self-attention used in LLMs that allows the model to only consider previous and current inputs in a sequence.
- **Multi-Head Attention**
  - An extension of self-attention and causal attention that enables the model to simultaneously attend to information from different representation subspaces.

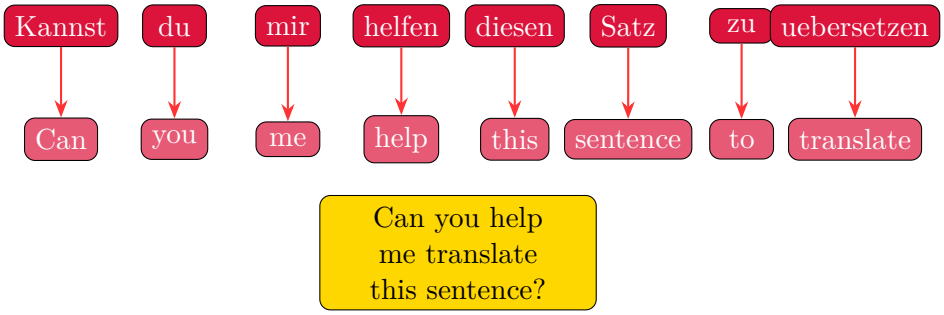
Additional Note

- LLMs attend to various input data in parallel.

The Problem with Modeling Long Sequences

Challenges with Architectures without Attention Mechanisms

Consider a language translation model translating a German sentence to English.



Observation:

- Word-by-word translation results in a grammatically incorrect sentence: *"Can you me help this sentence to translate"*.
- Correct translation: *"Can you help me translate this sentence?"*

Hindi Translation:

- क्या आप इस वाक्य का अनुवाद करने में मेरी मदद कर सकते हैं?

Note:

- Word-by-word translation does not work effectively for accurate language translation.

Translation Process

Contextual Understanding and Grammar Alignment

The translation process requires **contextual understanding** and **grammar alignment**. Certain words in the generated translation require access to words that appear earlier or later in the original sentence.

Addressing Word-by-Word Translation Issues

To address this issue that we cannot translate text word by word, it is common to use a neural network with two submodules: **Encoder** and **Decoder**.

Sequence to Sequence Model

Looking Under the Hood

Under the hood, the model is composed of an encoder and a decoder. The encoder processes each item in the input sequence, and compiles the information it captures into a vector (called the context). After processing the entire input sequence, the encoder sends the context over to the decoder, which begins producing the output sequence item by item.

Example Translation with Diagram

German Input Sentence:

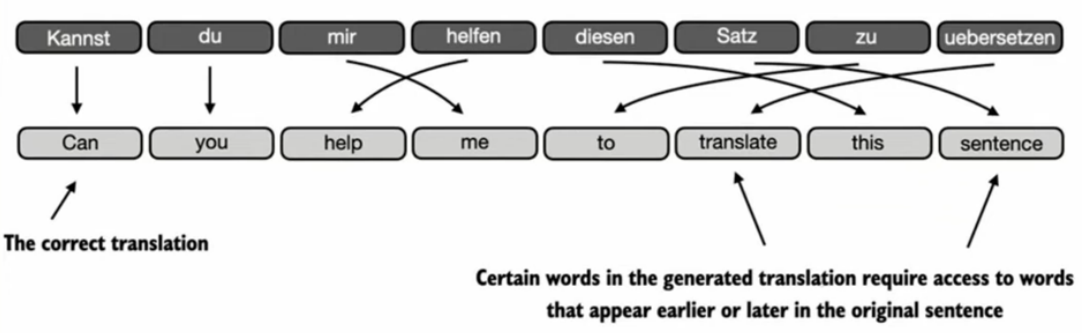
- Kannst
- du
- mir
- helfen
- diesen
- Satz
- zu
- uebersetzen

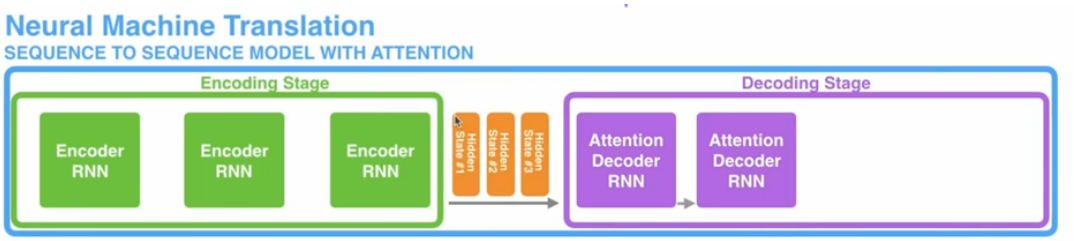
Translation Output (Word-by-Word):

- Can
- you
- me
- help
- this
- sentence
- to
- translate

Correct Translation:

- Can you help me translate this sentence?





Introduction to Recurrent Neural Networks (RNNs) in Translation

Before Transformers

(c) Before transformers, Recurrent Neural Networks (RNNs) were the most popular encoder-decoder architecture for language translation.

RNN Mechanism

(d) RNN!: Output from the previous step is fed as input to the current text. (e) Here's how the encoder-decoder RNN works!

- **Input text**
- **Encoder (processes input text sequentially)**
- **Updates hidden (internal) values state at each step in hidden layers**

Encoder-Decoder Process

Role of Hidden States

- Final hidden state: The encoder tries to capture the sentence meaning. - Decoder uses this final hidden state to generate the translated sentence (one word at a time). - (The decoder also updates its hidden state at each time step.)

Diagram Description

- **Translated English Sentence:** - Outputs: Can → you → help → ... - Hidden states: Transferred from encoder to decoder. - **German Input Sentence to Translate:** - Kannst → du → mir → ... - **Labels:** - Hidden states of a neural network. - A memory cell (hidden state) memorizing the entire input.

Process Overview

(f) The encoder processes the entire input text into a hidden state (memory cell). The decoder takes this hidden state to produce an output. - Think of this as an embedding vector. - The decoder takes this hidden state to produce an output.

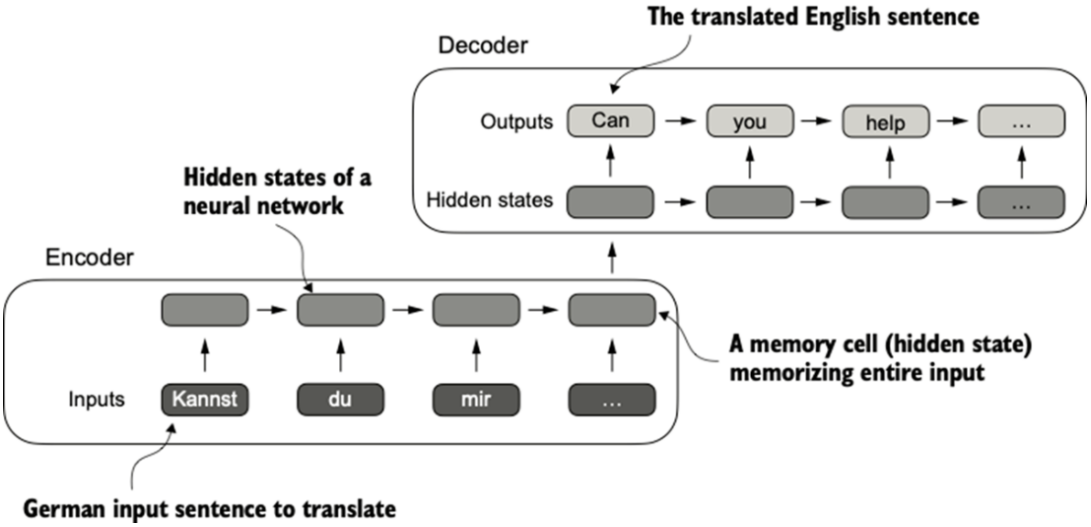


Figure 1: RNN

Limitations of RNNs

Context Loss in Decoding

(g) Big issue: RNN can't directly access earlier hidden states from the encoder during the decoding phase. - It relies solely on the current hidden state. - This leads to a loss of context, especially in complex sentences where dependencies might span long distances. - The encoder compresses the entire input sequence into a single hidden state vector. If the sentence is very long, it becomes very difficult for the RNN to capture all information in a single vector.

Example with Long Dependencies

Sentence Example

- Handwritten text: 'The cat that was sitting on the mat, which was next to the dog, jumped' - French translation: 'Le chat qui était assis sur le tapis, qui était à côté du chien, a sauté'

Dependency Analysis

- Here, the key action "jumped" depends on the subject "cat" but also on understanding the longer dependencies ("that was sitting on the mat, next to the dog"). - The RNN decoder might struggle with this.

Related Notes

- (3) Capturing data dependencies with attention mechanisms. - (4) RNNs work fine for translating short sentences, but don't work for long texts as they don't have direct access to previous words in the input.

Improvements with Attention Mechanism

Shortcoming of RNNs

(b) One major shortcoming in this approach is that: RNNs must remember the entire encoded input in a single hidden state before passing it to the decoder.

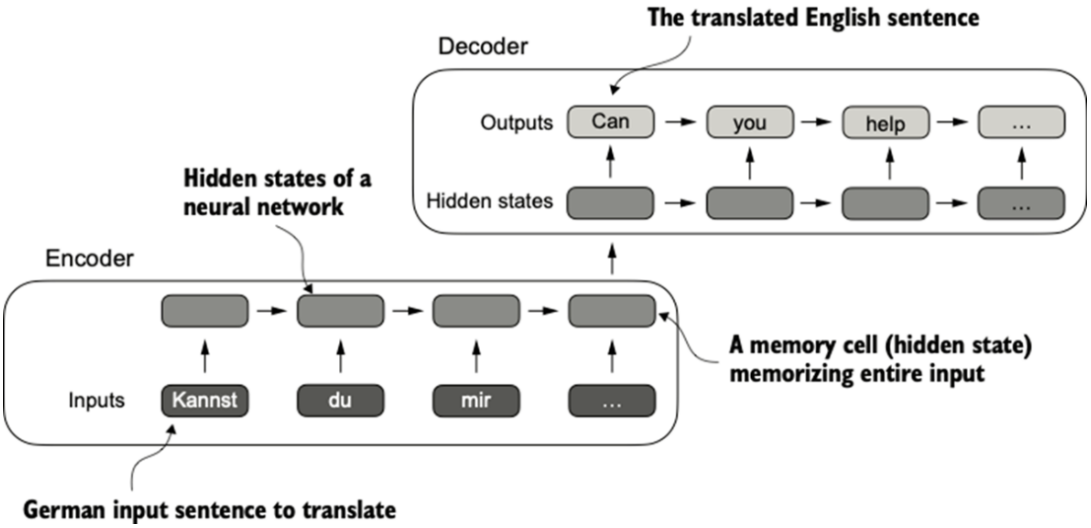


Figure 2: Bahdanau Attention

Bahdanau Attention Mechanism

(c) In 2014, researchers developed the so-called "Bahdanau attention mechanism for RNNs". - Modify the encoder-decoder RNN such that the decoder can selectively access different parts of the input sequence at each decoding step.

General Idea Behind the Bahdanau Mechanism

Using an attention mechanism, the text-generating decoder part of the network can access all input tokens selectively. This means that some input tokens are more important than others for generating a given output token. This importance is determined by the so-called attention weights.

Dynamic Focus

- Only 3 years later, researchers found that RNN architectures are not required for building deep neural networks for natural language processing and proposed the original transformer architecture, with a self-attention mechanism inspired by the Bahdanau attention mechanism.
- When the decoder is predicting "jumped," the attention mechanism allows it to focus on the part of the input that corresponds to "jumped."
- Dynamic focus on different parts of the input sequence allows models to learn long-range dependencies more effectively.

Example Sentence

”The cat that was sitting on the mat, which was next to the dog, jumped”

- At each decoding step, the model can look back at the entire input sequence and decide which parts are most relevant to generate the current word.

Self-Attention Mechanism

Definition

Self-attention is a mechanism that allows each position of the input sequence to attend to all positions in the same sequence when computing the representation of a sequence.

- Self-attention is a key component of contemporary LLMs based on the transformer architecture, such as the GPT series.
- Attending to different parts of the input with self-attention. In ”self-attention,” the ”self” refers to the mechanism’s ability to compute attention weights by relating different positions in a single input sequence.

Characteristics

- It learns the relationship between various parts of the input itself, such as words in a sentence.
- This is in contrast to traditional attention mechanisms, where the focus is on relationships between elements of two different sequences.