Encoder Decoder with Attention Mechanism

Attention: because it adds attention weights to h_n

Unlike vanila Encoder Decoder, Encoder Decoder with attention mechanism gets access to both

- Context vector
- Encoded hidden states (adds weight to term to prioritize).

Imp

1 Core Concept

- In vanilla Encoder–Decoder, the **decoder only gets the fixed-length context vector** (summary).
- **Attention mechanism** fixes this by letting the decoder **look at all encoder hidden states** at every step.
- Instead of relying on a single summary, the decoder dynamically decides **"which parts of the input to focus on"** for generating each output token.

👉 Think of Attention as giving the decoder **"a spotlight"** that it can move across different words in the input while generating the output.

Main Idea: At the decoder level, it interacts with each token from hidden states

2 How It Works (Step by Step)

1. **Encoder** processes input → produces hidden states for each token: $h_1, h_2, \ldots, h_n$.
2. At each decoder step t:
   - The decoder computes **alignment scores** between its current hidden state s_t and each encoder hidden state h_i.
   - These scores are normalized (via **softmax**) → form **attention weights** α_i.
   - Weighted sum of encoder states = **context vector (c_t)** for that step.
3. The decoder uses both c_t (focused context) + previous token to generate the next word.

3 Benefits Over Vanilla Encoder–Decoder

Aspect	Vanilla Encoder–Decoder	With Attention
Context	Single fixed vector	Dynamic, step-wise context
Long sequences	Struggles (info bottleneck)	Handles long inputs better
Output quality	Often generic/incomplete	More accurate, context-aware
Interpretability	Hard to see what model used	Attention weights show "which words were focused on"

✅ One-Line Interview Answer:

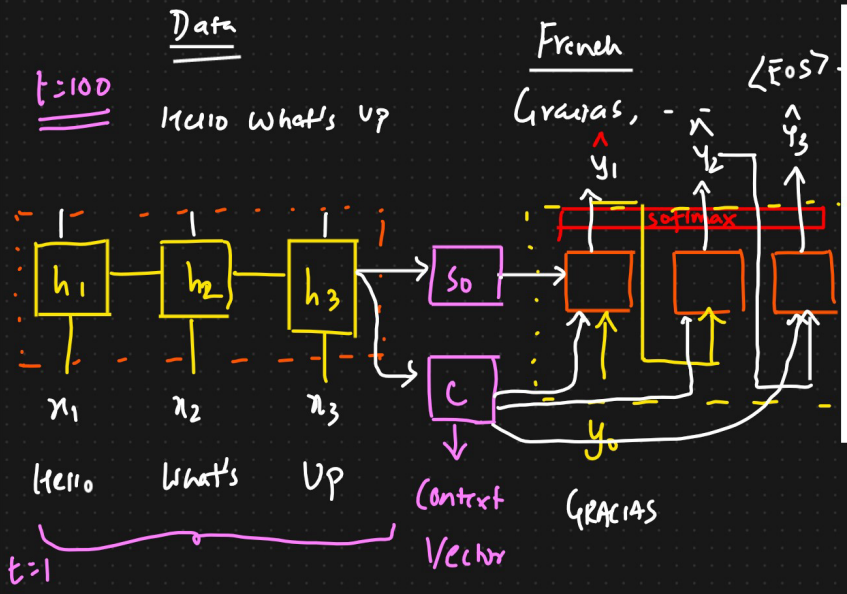
"Encoder–Decoder with Attention improves Seq2Seq by allowing the decoder to look at all encoder hidden states and assign dynamic importance weights at each step, overcoming the information bottleneck of a fixed context vector."

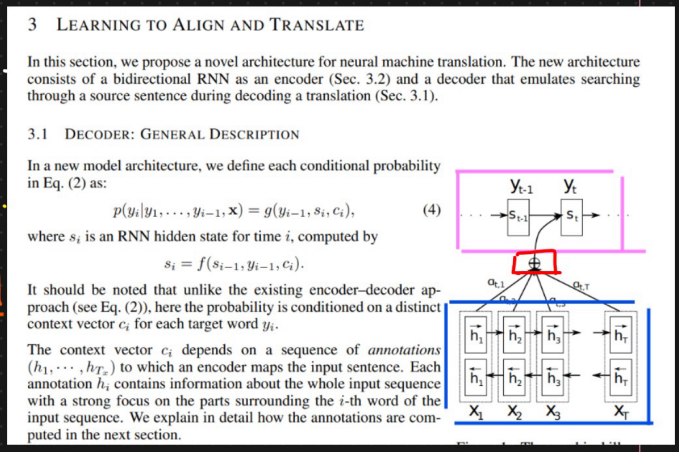
In short: In Attention Mechanism, decoder has access to all the hidden states and builds a new context vector.

⑧ Attention Mechanism → Seq2Seq Network

Longer paragraph → {Context Vector}
+
{Context}

Attention Mechanism | Seq2Seq Networks





3 LEARNING TO ALIGN AND TRANSLATE

In this section, we propose a novel architecture for neural machine translation. The new architecture consists of a bidirectional RNN as an encoder (Sec. 3.2) and a decoder that emulates searching through a source sentence during decoding a translation (Sec. 3.1).

3.1 DECODER: GENERAL DESCRIPTION

In a new model architecture, we define each conditional probability in Eq. (2) as:

$$p(y_i|y_1,\dots,y_{i-1},\mathbf{x}) = g(y_{i-1}, s_i, c_i), \quad (4)$$

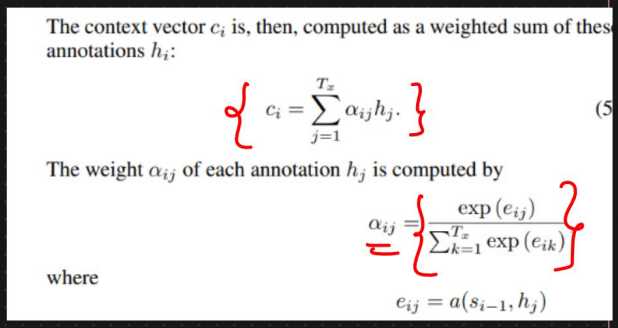
where s_i is an RNN hidden state for time i, computed by

$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

It should be noted that unlike the existing encoder–decoder approach (see Eq. (2)), here the probability is conditioned on a distinct context vector c_i for each target word y_i.

The context vector c_i depends on a sequence of *annotations* $(h_1,\cdots,h_{T_x})$ to which an encoder maps the input sentence. Each annotation h_i contains information about the whole input sequence with a strong focus on the parts surrounding the i-th word of the input sequence. We explain in detail how the annotations are computed in the next section.

Encoder Decoder Architecture



The context vector c_i is, then, computed as a weighted sum of these annotations h_i:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \quad (5$$

The weight α_{ij} of each annotation h_j is computed by

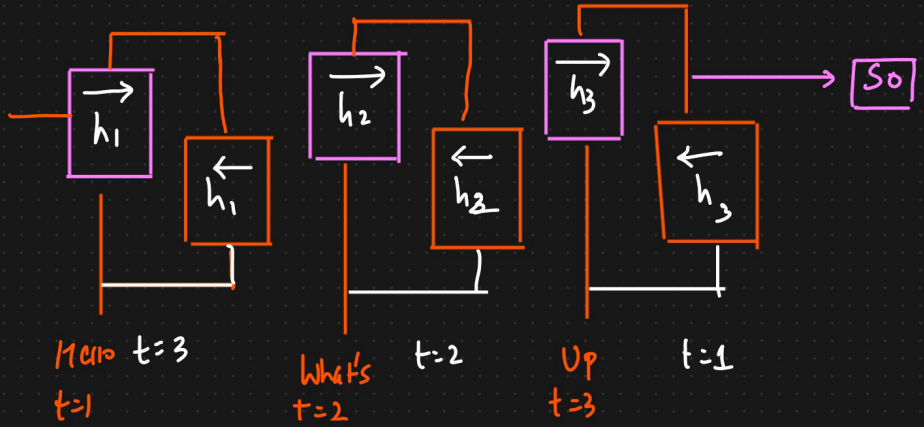
$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}$$

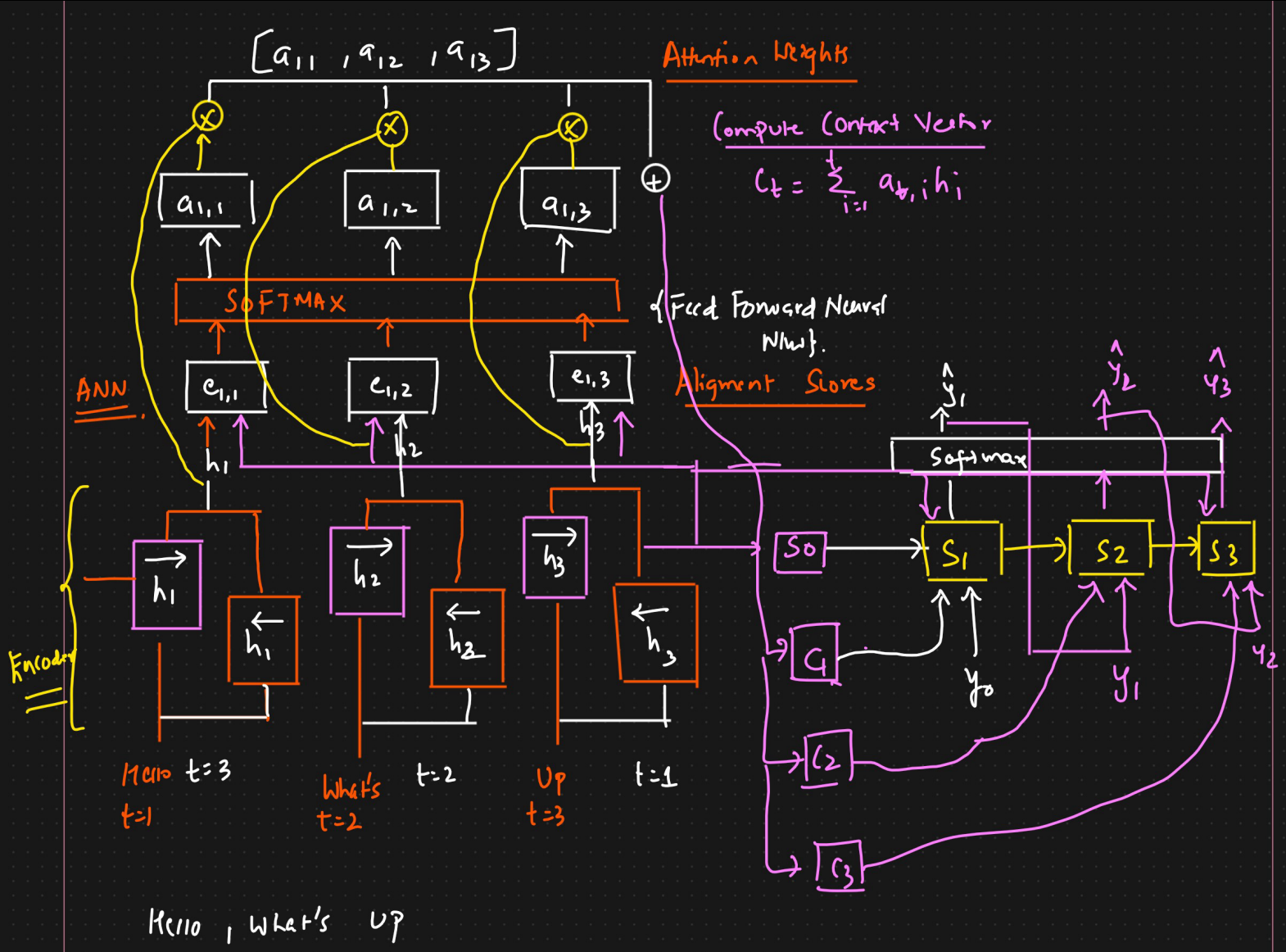
where

$$e_{ij} = a(s_{i-1}, h_j)$$

Attention Mechanism

https://erdem.pl/2021/05/introduction-to-attention-mechanism




[a11 , a12 , a13]
Attention Weights
Compute Context Vector
$c_t = \sum_{i=1}^{t} a_{t,i} h_i$
a1,1
a1,2
a1,3
SOFTMAX
{Feed Forward Neural Nwk}.
Alignment Scores
e1,1
e1,2
e1,3
ANN
h1
h2
h3
Softmax
y1
y2
y3
Encoder
h1
h2
h3
S0
S1
S2
S3
C
C2
C3
y0
y1
y2
Hello t=3
t=1
What's t=2
t=2
Up t=3
t=1
Hello , What's Up