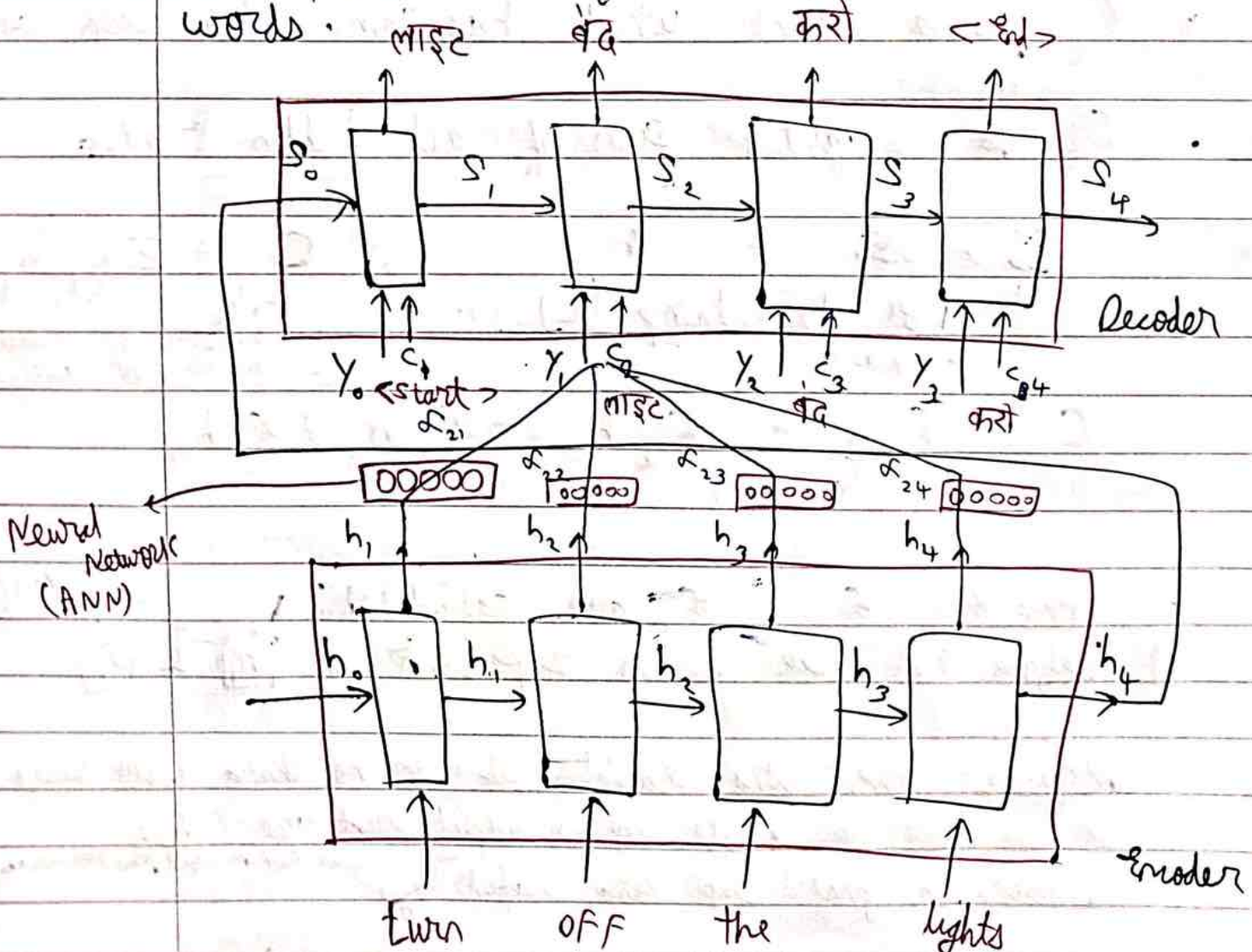
* Attention Mechanism

- The Encoder-Decoder model worked fine, but when we give a long paragraph it become diffcult to translate long para's, because it cannot pertain context vector.
- Imagine, we have given a task to convert a long English para into hindi, ... it becomes diffcult, ... BUT! if you translate along with reading, then you might be able to translate well, all it requires the "attention" to each words.



__/__/____

- at timestamp $t=2$ (In Decoder block)
→ $[Y_1, S_1]$ ~ Input

Vanilla encoder decoder	attention
timestamp = t	timestamp = t
Input @ decoder looks → (Y_{t-1}, S_{t-1})	(Y_{t-1}, S_{t-1}, C_t) ↳ attention input

- C_t is a vector which has same dimension as h_t
- C_t is weighted sum of all hidden states

For timestamp $(t) =)$

$$C_1 = \alpha_1 h_1 + \alpha_2 h_2 + \alpha_3 h_3 + \alpha_4 h_4$$

(scalars; ↳ vectors)

$$C_{t_1 t_2} = \sum \alpha_{t_1 t_2 t_1} h_{t_1}$$

t_1: timestamp for decoder.
t_2: timestamp for encoder.

- How $\alpha_1, \alpha_2 \ldots \alpha_n$ are calculated? Yes!!

1) Before that, see α is dependent on h_1 & S_1

Why S_1?

Because:- Given, jitna translation ab tak ho chuka usse basis pe ye batao ki agle step me output print krne k liye (decoder block me output print krna hota hai) encoder ka first step kitna useful hoga

$$\alpha_{2_1} \rightarrow F(h_1, S_1)$$

$$i.e \quad \alpha_{t_1 t_2} = f(h_{t_2}, S_{t_1-1})$$

2) Alpha (α) is a function of h & S

$$\therefore \alpha_{23} \rightarrow f(h_3, S_1)$$

What is this function?) → ANN!!

because ANN's are the best function approximater

3) When you provide enough data to ANN's., it will automatically adjusts it's weights to approximate the function, hence to determine value for (α). Brilliant idea, indeed.

4) Initially ANN gives randomly weightage & value (α) to everyone, but with loss function and backpropagation it will learn and adjust itself to give priority to (α) to a specific hidden state's (context vector)

* **Attention Mechanism**

- Calculates relevance bet" each input word and the current output step.
- Assigns weights (called ~~weighted sum~~ attention score) (C_i) to input tokens based on how important they are
- Produces a weighted sum of the ~~decoder~~ encoder outputs (called the context vector) for each decoder block cell
- Helps decoder focus on relevant words from the input sequence while generating each word in the output.