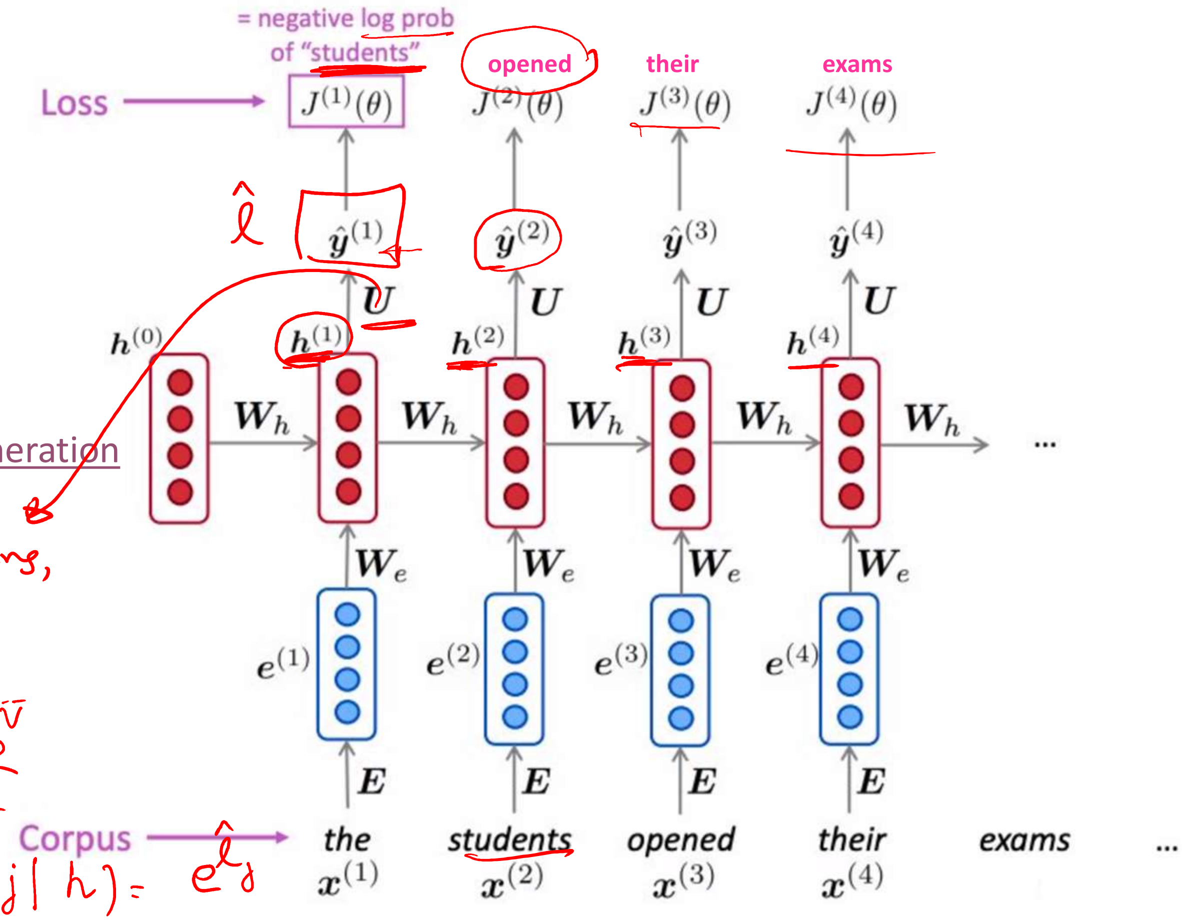
Input: Text, Task: Predicting next word in a sentence

- Input sequence: $x_1, x_2, \ldots, x_n$
- Each x_t instead of a real-valued vector is a discrete word.
- The task is to predict the following word $x_{\{n+1\}}$
- Output sequence: $y_1, y_2, \ldots, y_n$ can be written as $y_t = x_{t+1}$

Learning to output the next word

Find parameters that maximize the probability of predicting the next word in a sentence



https://www.tensorflow.org/tutorials/text/text_generation

$h \in \mathbb{R}^{128}$ $\quad V = [1, \dots 65]$

Software matrix, output embedding, last linear layer = U

$U: h^{(t)} \Rightarrow \mathbb{R}^{V}$ $\quad \hat{\ell}$

$hU \rightarrow$ logits $= \mathbb{R}^{V}$

$U^T = 128 \times 65$ $\quad \Pr(\hat{y} = j \mid h) = e^{\hat{\ell}_j}$

Image src: https://towardsdatascience.com/introduction-to-rnns-sequence-to-sequence-language-translation-and-attention-fc43ef2cc3fd

$h^{(1)}, h^{(2)}, \ldots h^{(n)}$ — RNN states.

$\Downarrow$ need to convert

$p^{(1)}(\hat{y}_1) \ldots p(y_n^{(n)}) \leftarrow$ probability distribution over characters.

$$h^{(t)} \bar{U} = \ell^{(t)} \in \mathbb{R}^{|V|}$$

last linear layer parameters.

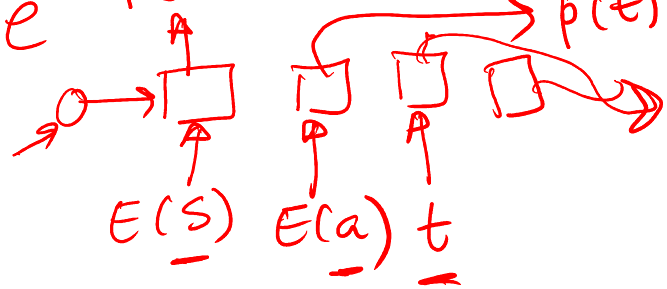
$$\underbrace{\{[-0.1,\ 2,\ -3,\ \ldots\ ]}_{|V|}\} \rightarrow \text{for example.}$$

$$p^{(t)}(y_t = j \mid \ell^{(t)}) = \frac{e^{\ell_j^{(t)}}}{\sum_{i=1}^{|V|} e^{\ell_i^{(t)}}}$$

$$-\log p^{(t)}(y_t = x_{t+1} \mid \ell^{(t)})$$

$p(h) = 0.3$ $\quad p(a) = 0.2$ $\ldots$

$p(t) = 0.4$, $p(r) = 0.3$



$p(\text{" "}) = 0.9$

$p(\text{"z"}) = 0.05$

The sequence prediction task

- Given a complex input **x**
  - Example: sentence(s), image, audio wave
- Predict a sequence **y** of discrete tokens $y_1, y_2, .., y_n$
  - Typically a sequence of words.
  - A token can be any term from a huge discrete vocabulary
  - Tokens are inter-dependent
    - Not n independent scalar classification task.

Translation

Context: **x**

Predicted sequence: **y**

Where can I find healthy and traditional Indian food? →

स्वस्थ और पारंपरिक भारतीय भोजन कहां मिल सकता है?

- Pre-DL translation systems were driven by transfer grammar rules painstakingly developed by linguists and elaborate phrase translation
- Whereas, modern neural translation systems are scored almost 60% better than these domain-specific systems.

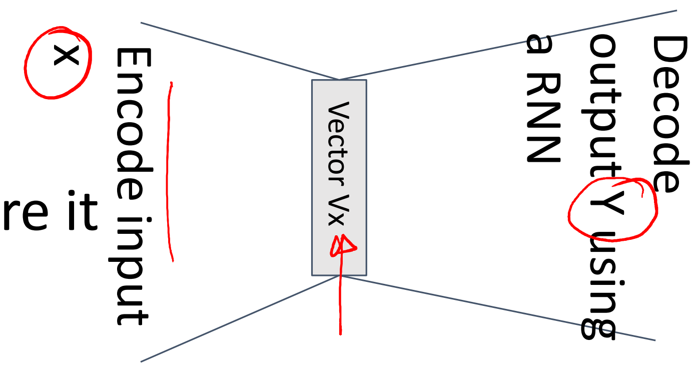
Sequence to Sequence learning

$\mathbf{x} : x_1, x_2, \ldots, x_n$ → Seq2Seq Learning → $\mathbf{y} : y_1, y_2, \ldots, y_T$

- Input sequence **x**
  - Example: sentence(s), audio wave
- Predicted sequence **y** has discrete tokens
  - Typically a sequence of words from a huge discrete vocabulary
- A common Model for Seq2Seq learning --- encoder-decoder model.

Encoder Decoder Model

- Encode x into a fixed-D real vector X
- Decode Y a token at a time by conditioning the generation of the next token on the tokens before it using a RNN



parameters

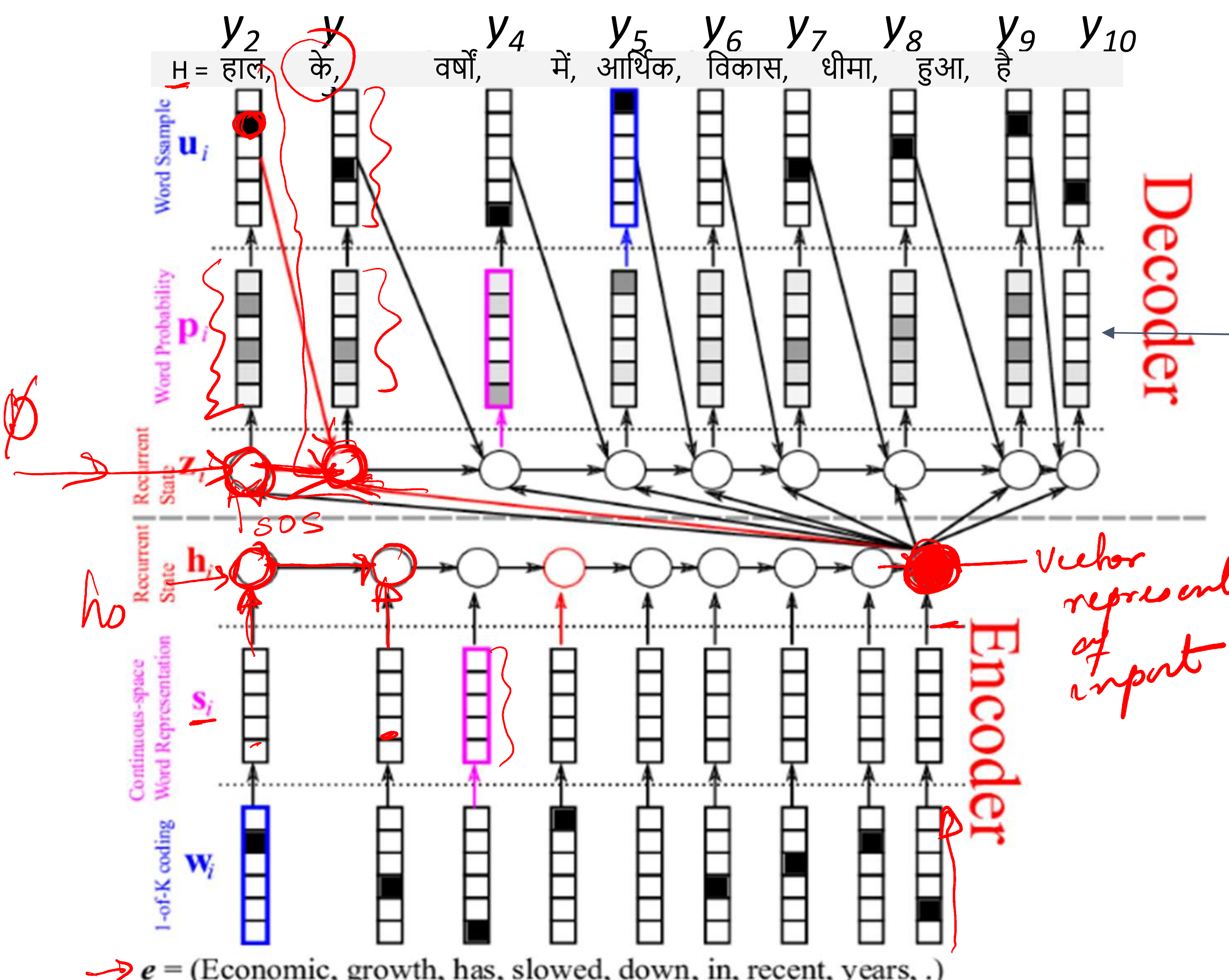
$$P(\mathbf{y}|\mathbf{x},\theta) = \prod_{t=1}^{n} P(y_t|y_1,\ldots,y_{t-1},\mathbf{x},\theta)$$

$y_1, y_2 - y_T$

chain rule of probability

Auto-regressive model

Encoder-decoder for sequence to sequence learning



From https://devblogs.nvidia.com/parallelforall/introduction-neural-machine-translation-gpus-part-2/

Predicted sequence: Choose high probability token and feed to next step.

$$\Pr(y_t | y_1, \ldots, y_{t-1}, \mathbf{x})$$

RNN to generate **y**

RNN e.g. LSTMs to summarize x token-by-token

Embedding layer to convert each word to a fixed-D real vector

Context: **x**

Attention

- Attention
  - [Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention) – Jay Alammar – Visualizing machine learning one concept at a time. (jalammar.github.io)]

Inference

- Greedy algorithm

- Beam-search

Transformers

- RNNs require sequential computation of state, which makes training slow
- Transformers replace “state” with multi-layered self-attention.
- All positions in a sequence can be trained in parallel.

- https://jalammar.github.io/illustrated-transformer/