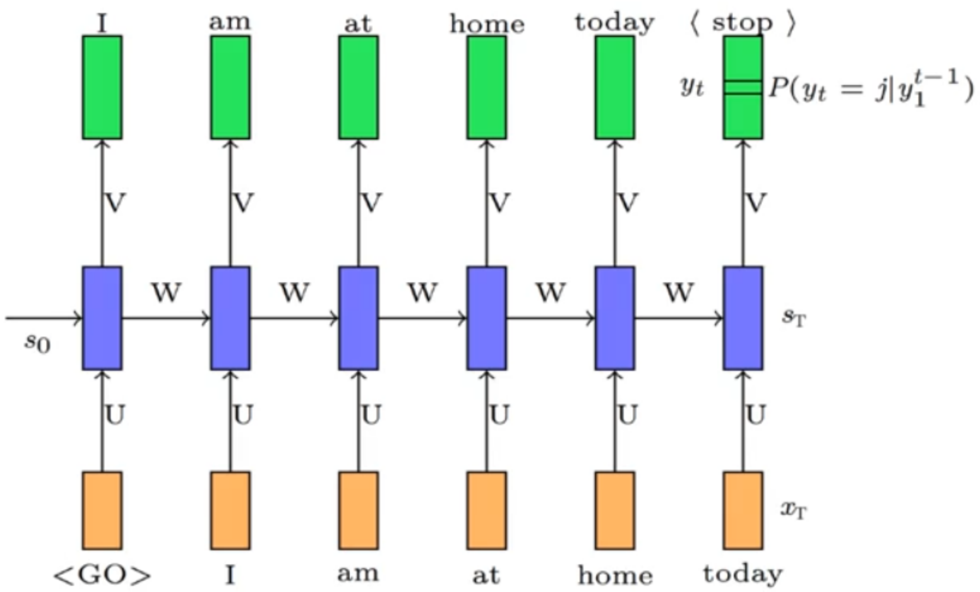
Deep Learning : Encoder-Decoder Model

राष्ट्रीय प्रौद्योगिकी संस्थान सिक्किम
NATIONAL INSTITUTE OF TECHNOLOGY SIKKIM

Course Instructor:
Dr. Bam Bahadur Sinha
Assistant Professor
Computer Science & Engineering
National Institute of Technology Sikkim



- We are interested in

$$P(y_t = j|y_1, y_2 \ldots y_{t-1})$$

where $j \in V$ and V is the set of all vocabulary words

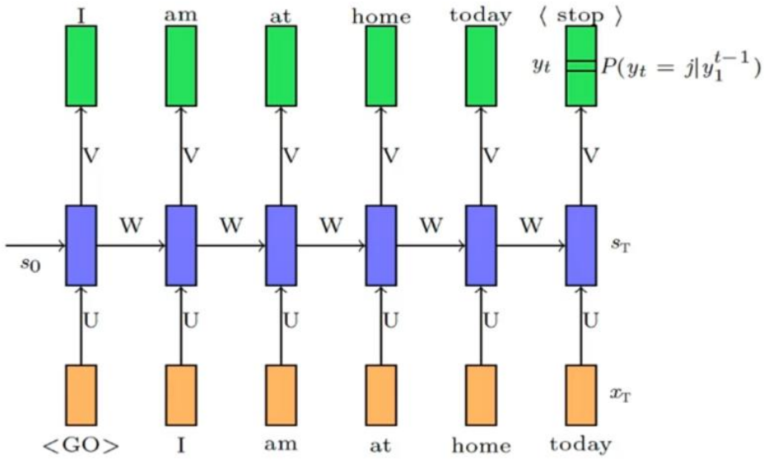
- Using an RNN we compute this as

$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- In other words we compute

$$\begin{aligned} P(y_t = j|y_1^{t-1}) &= P(y_t = j|s_t) \\ &= softmax(Vs_t + c)_j \end{aligned}$$

Language Modelling



Data:

India, officially the Republic of India, is a country in South Asia. It is the seventh-largest country by area, ...

- **Data:** All sentences from any large corpus (say wikipedia)
- **Model:**

$$s_t = \sigma(Ws_{t-1} + Ux_t + b)$$
$$P(y_t = j|y_1^{t-1}) = softmax(Vs_t + c)_j$$

- **Parameters:** U, V, W, b, c
- **Loss:**

$$\mathscr{L}(\theta) = \sum_{t=1}^{T} \mathscr{L}_t(\theta)$$
$$\mathscr{L}_t(\theta) = -\log P(y_t = \ell_t|y_1^{t-1})$$

Encoder Decoder Model

$$s_t = \sigma(U x_t + W s_{t-1} + b)$$

$$\tilde{s}_t = \sigma(W(o_t \odot s_{t-1}) + U x_t + b)$$
$$s_t = i_t \odot s_{t-1} + (1 - i_t) \odot \tilde{s}_t$$

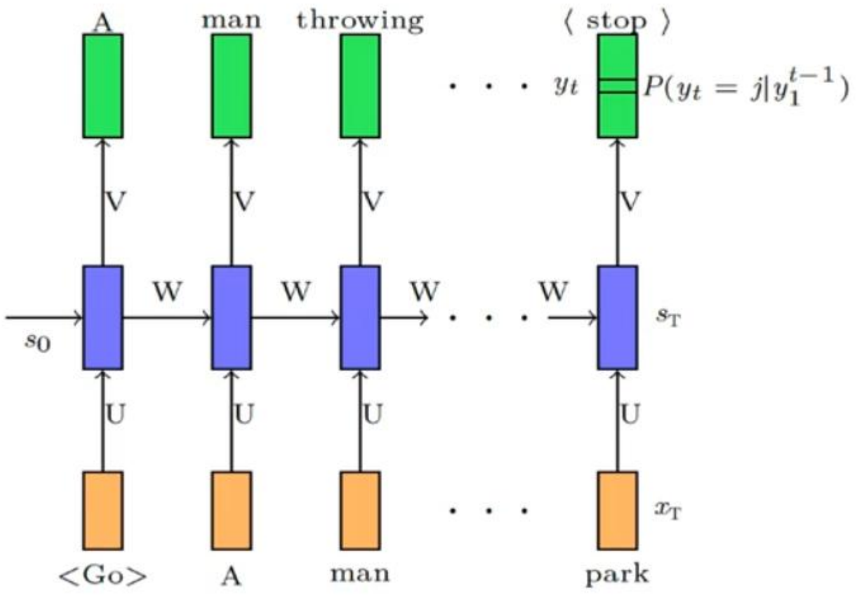
$$\tilde{s}_t = \sigma(W h_{t-1} + U x_t + b)$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t$$
$$h_t = o_t \odot \sigma(s_t)$$

$$s_t = \mathrm{RNN}(s_{t-1}, x_t)$$

$$s_t = \mathrm{GRU}(s_{t-1}, x_t)$$

$$h_t, s_t = \mathrm{LSTM}(h_{t-1}, s_{t-1}, x_t)$$

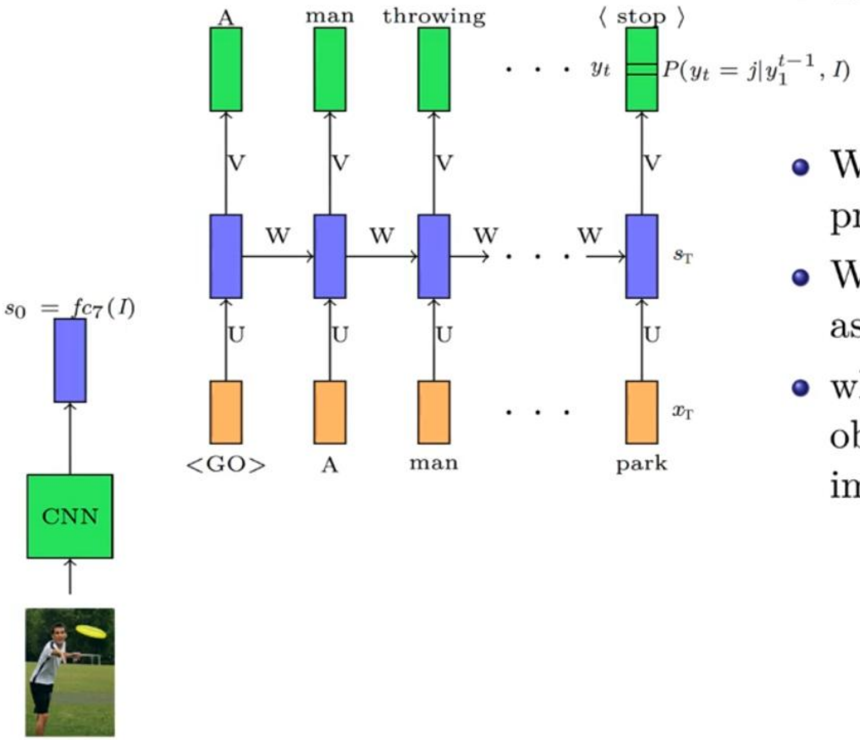
Compact way of writing functions computed using RNNs, GRU and LSTM



A man throwing
a frisbee in a park

- So far we have seen how to model the conditional probability distribution $P(y_t|y_1^{t-1})$
- More informally, we have seen how to generate a sentence given previous words
- What if we want to generate a sentence given an image?
- We are now interested in $P(y_t|y_1^{t-1}, I)$ instead of $P(y_t|y_1^{t-1})$ where I is an image
- Notice that $P(y_t|y_1^{t-1}, I)$ is again a conditional distribution

Image Captioning using Encoder-Decoder Model

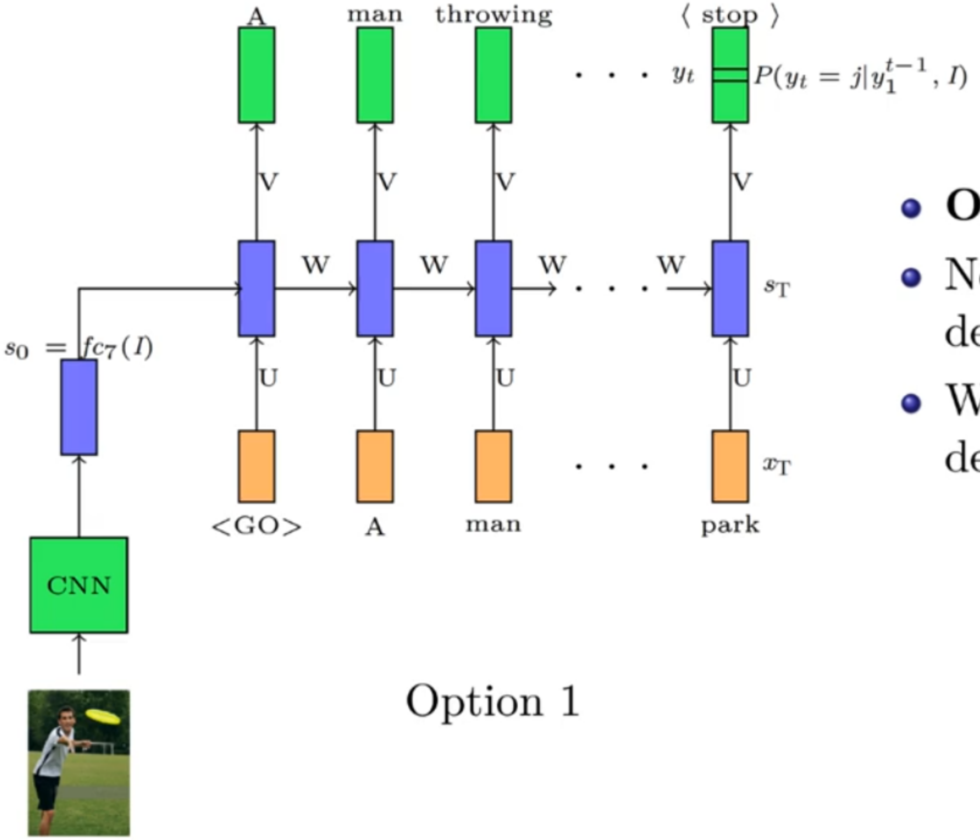


- Earlier we modeled $P(y_t|y_1^{t-1})$ as

$$P(y_t|y_1^{t-1}) = P(y_t = j|s_t)$$

- Where s_t was a state capturing all the previous words
- We could now model $P(y_t = j|y_1^{t-1}, I)$ as $P(y_t = j|s_t, f_{c_7}(I))$
- where $fc_7(I)$ is the representation obtained from the fc_7 layer of an image

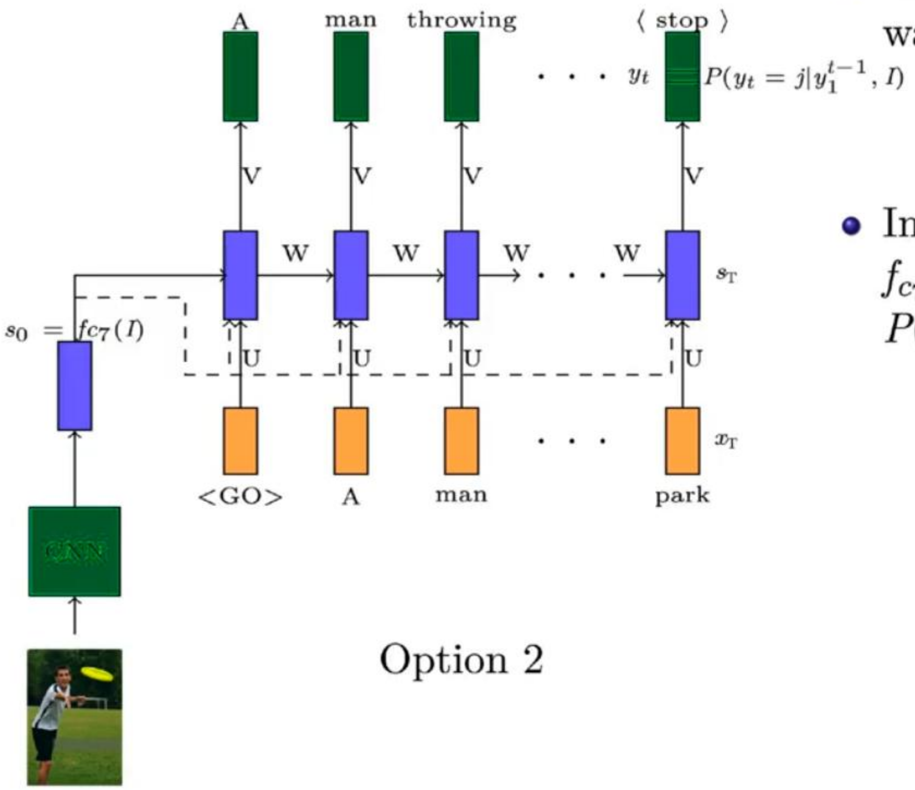
Image Captioning using Encoder-Decoder Model



Option 1

- **Option 1:** Set $s_0 = f_{c_7}(I)$
- Now s_0 and hence all subsequent s_t's depend on $f_{c_7}(I)$
- We can thus say that $P(y_t = j)$ depends on $f_{c_7}(I)$

Image Captioning using Encoder-Decoder Model



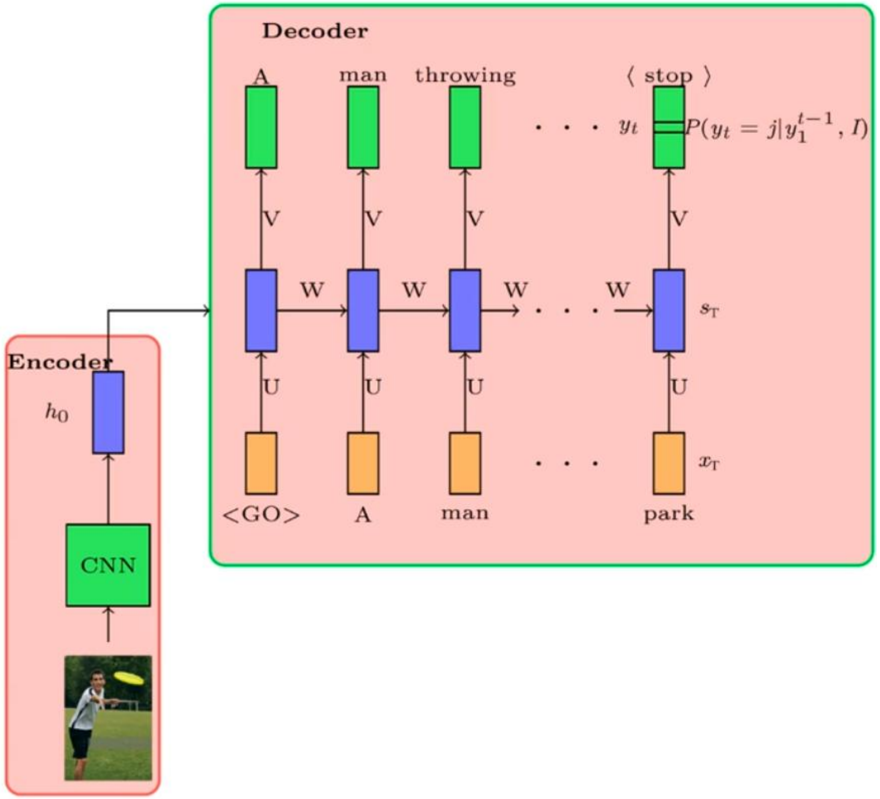
Option 2

- **Option 2:** Another more explicit way of doing this is to compute

$$s_t = RNN(s_{t-1}, [x_t, f_{c_7}(I)])$$

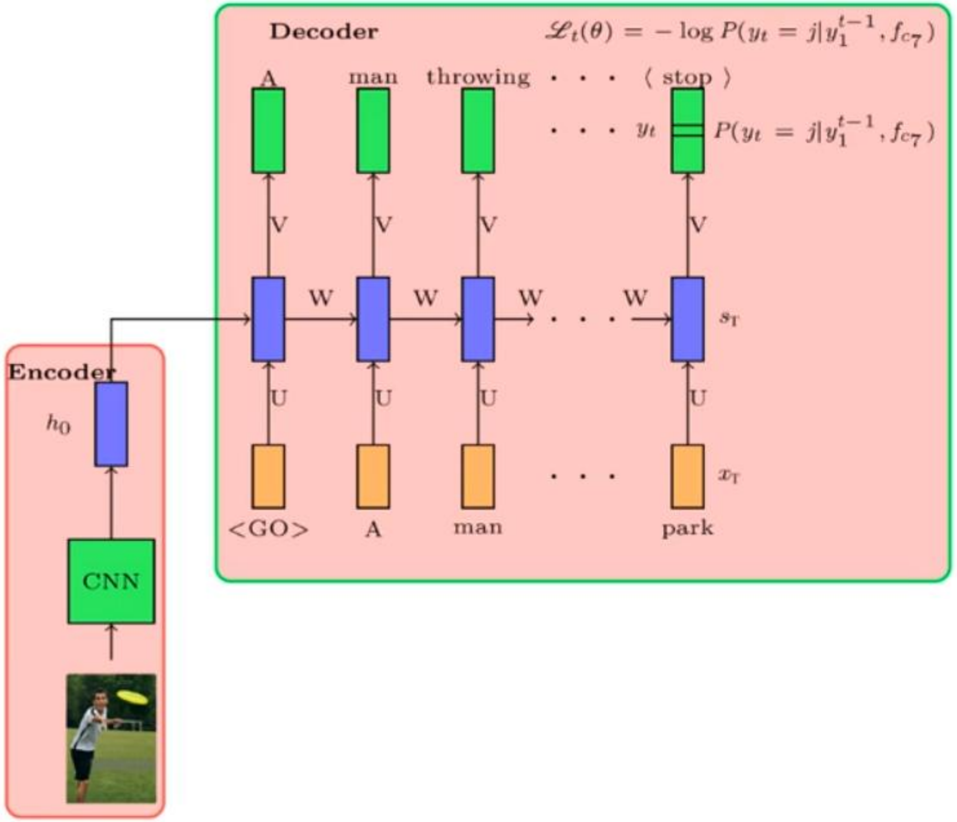
- In other words we are explicitly using $f_{c_7}(I)$ to compute s_t and hence $P(y_t = j)$

Image Captioning using Encoder-Decoder Model



- A CNN is first used to **encode** the image
- A RNN is then used to decode (generate) a sentence from the encoding
- This is a typical **encoder decoder architecture**
- Both the encoder and decoder use a neural network
- Alternatively, the encoder's output can be fed to every step of the decoder

Encoder – Decoder Architecture



- **Task:** Image captioning
- **Data:** $\{x_i = image_i,\ y_i = caption_i\}_{i=1}^{N}$
- **Model:**
  - **Encoder:**

$$s_0 = CNN(x_i)$$

  - **Decoder:**

$$s_t = RNN(s_{t-1}, e(\hat{y}_{t-1}))$$

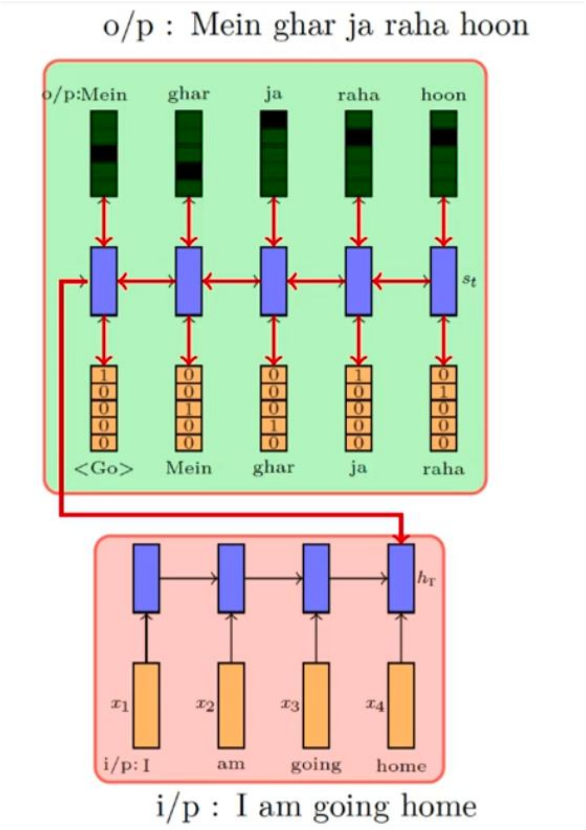
$$P(y_t | y_1^{t-1}, I) = softmax(Vs_t + b)$$

- **Parameters:** $U_{dec},\ V,\ W_{dec},\ W_{conv},\ b$
- **Loss:**

$$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_t(\theta) \quad = -\sum_{t=1}^{T} \log P(y_t = \ell_t | y_1^{t-1}, I)$$

- **Algorithm:** Gradient descent with backpropagation

Encoder – Decoder Architecture



- **Task:** Image captioning
- **Data:** $\{x_i = image_i,\ y_i = caption_i\}_{i=1}^{N}$
- **Model:**
  - **Encoder:**

$$s_0 = CNN(x_i)$$

  - **Decoder:**

$$s_t = RNN(s_{t-1}, e(\hat{y}_{t-1}))$$

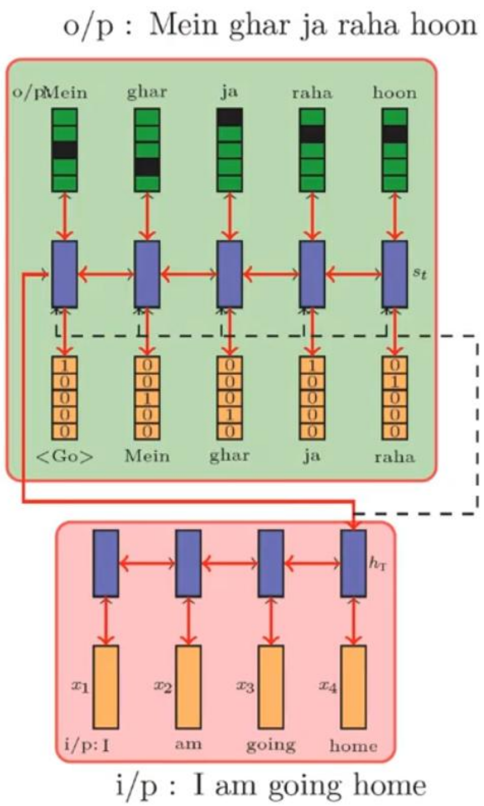
$$P(y_t|y_1^{t-1}, I) = softmax(Vs_t + b)$$

- **Parameters:** $U_{dec},\ V,\ W_{dec},\ W_{conv}, b$
- **Loss:**

$$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_t(\theta) \quad = -\sum_{t=1}^{T} \log P(y_t = \ell_t | y_1^{t-1}, I)$$

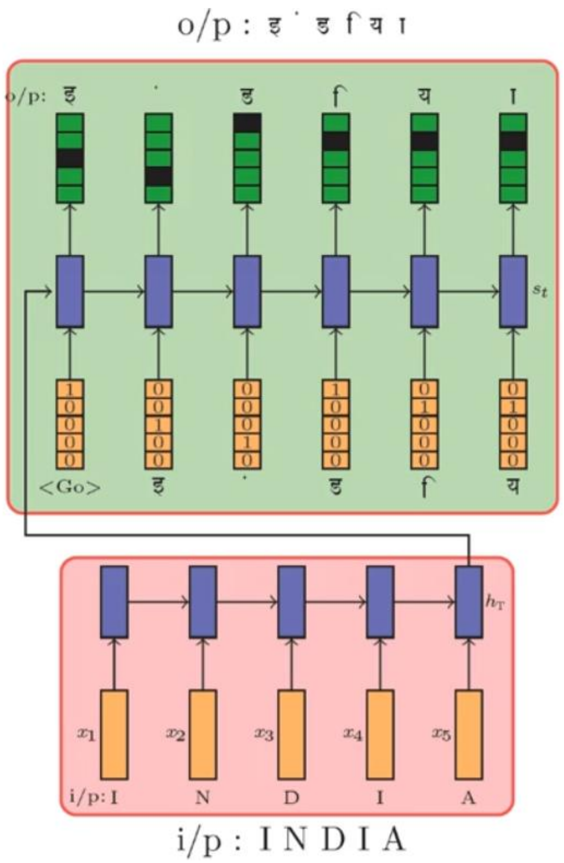
- **Algorithm:** Gradient descent with backpropagation

Encoder – Decoder Architecture – Option 1 (Machine Translation)



- **Task:** Machine translation
- **Data:** $\{x_i = source_i,\ y_i = target_i\}_{i=1}^{N}$
- **Model (Option 2):**
  - **Encoder:**
    $$h_t = RNN(h_{t-1}, x_{it})$$
  - **Decoder:**
    $$s_0 = h_T \quad (T \text{ is length of input})$$
    $$s_t = RNN(s_{t-1}, [h_T, e(\hat{y}_{t-1})])$$
    $$P(y_t|y_1^{t-1}, x) = softmax(Vs_t + b)$$
- **Parameters:** $U_{dec},\ V,\ W_{dec},\ U_{enc},\ W_{enc},\ b$
- **Loss:**
  $$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_t(\theta) = -\sum_{t=1}^{T} \log P(y_t = \ell_t|y_1^{t-1}, x)$$
- **Algorithm:** Gradient descent with backpropagation

Encoder – Decoder Architecture – Option 2 (Machine Translation)



- **Task:** Transliteration
- **Data:** $\{x_i = srcword_i, \;\; y_i = tgtword_i\}_{i=1}^{N}$
- **Model (Option 1):**
  - **Encoder:**

$$h_t = RNN(h_{t-1}, x_{it})$$

  - **Decoder:**

$$s_0 = h_T \quad (T \text{ is length of input})$$

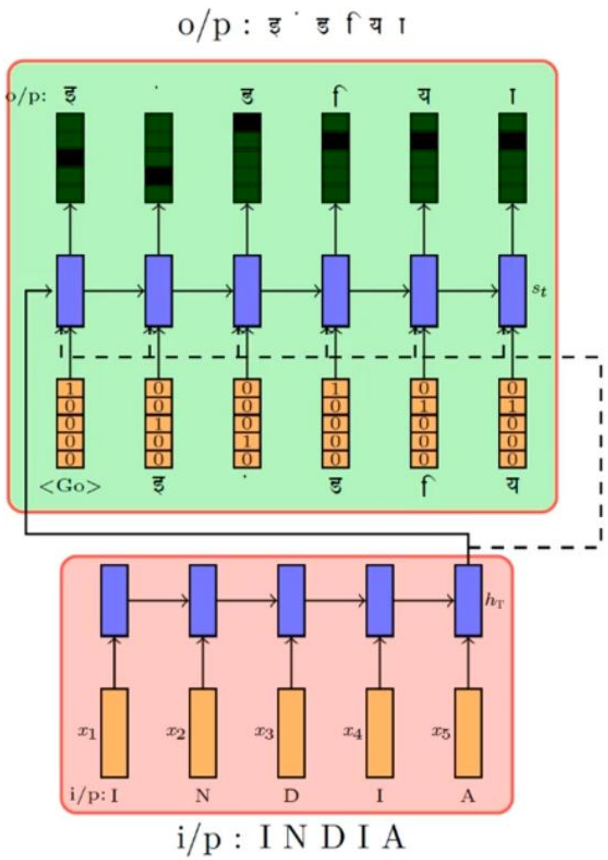
$$s_t = RNN(s_{t-1}, e(\hat{y}_{t-1}))$$

$$P(y_t|y_1^{t-1}, x) = softmax(Vs_t + b)$$

- **Parameters:** $U_{dec}, \; V, \; W_{dec}, \; U_{enc}, \; W_{enc}, b$
- **Loss:**

$$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_t(\theta) = -\sum_{t=1}^{T} \log P(y_t = \ell_t | y_1^{t-1}, x)$$

Encoder – Decoder Architecture – Option 1 (Transliteration)



- **Task:** Transliteration
- **Data:** $\{x_i = srcword_i,\ y_i = tgtword_i\}_{i=1}^{N}$
- **Model (Option 2):**
  - **Encoder:**

$$h_t = RNN(h_{t-1}, x_{it})$$

  - **Decoder:**

$$s_0 = h_T \quad (T \text{ is length of input})$$

$$s_t = RNN(s_{t-1}, [e(\hat{y}_{t-1}), h_T])$$

$$P(y_t | y_1^{t-1}, x) = softmax(Vs_t + b)$$

- **Parameters:** U_{dec}, V, W_{dec}, U_{enc}, W_{enc}, b
- **Loss:**

$$\mathscr{L}(\theta) = \sum_{i=1}^{T} \mathscr{L}_t(\theta) = -\sum_{t=1}^{T} \log P(y_t = \ell_t | y_1^{t-1}, x)$$

- **Algorithm:** Gradient descent with backpropagation

Encoder – Decoder Architecture – Option 2 (Transliteration)