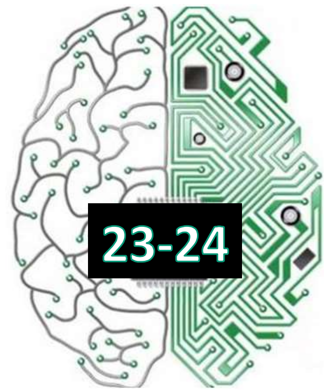
Open Elective Course [OE]
Course Code: CSO507
Winter 2023-24

Lecture#

Deep Learning

Unit-5: Sequence Modeling with Recurrent Neural Network (RNN)_Part-IV&V

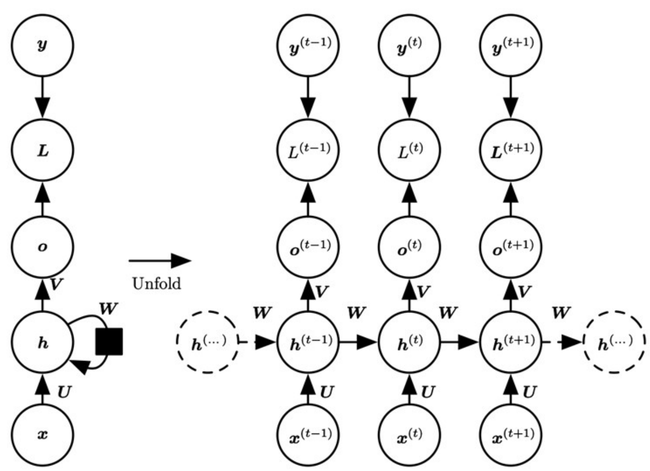


Course Instructor:
Dr. Monidipa Das
Assistant Professor
Department of Computer Science and Engineering
Indian Institute of Technology (Indian School of Mines) Dhanbad, Jharkhand 826004, India

Vanishing/Exploding Gradient Problem

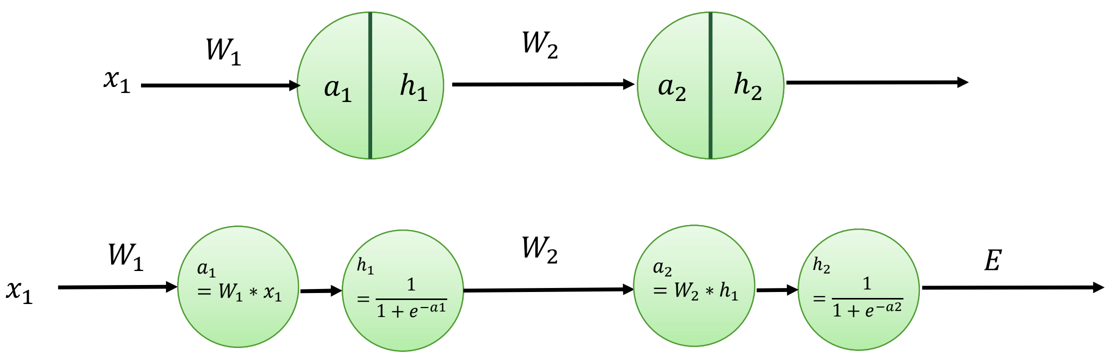
2

- Backpropagated errors multiply at each layer, resulting in exponential decay (if derivative is small) or growth (if derivative is large).
- Makes it very difficult train deep networks, or simple recurrent networks over many time steps.



Vanishing/Exploding Gradient Problem

3



$$W_i = W_i - \eta \frac{\partial E}{\partial W_i}$$

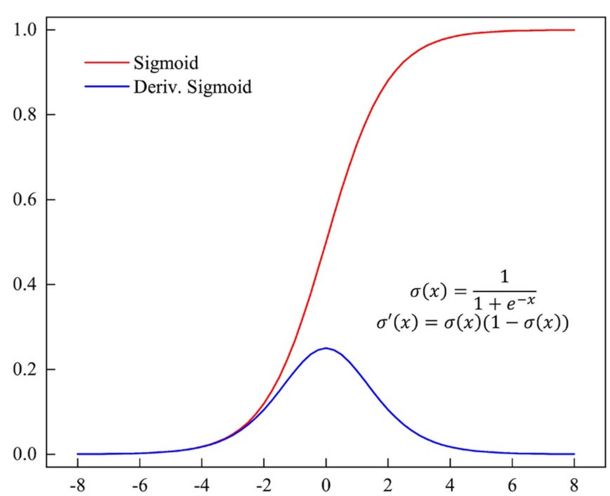
$$\frac{\partial E}{\partial w_2} = \frac{\partial E}{\partial h_2} * \frac{\partial h_2}{\partial a_2} * \frac{\partial a_2}{\partial W_2}$$

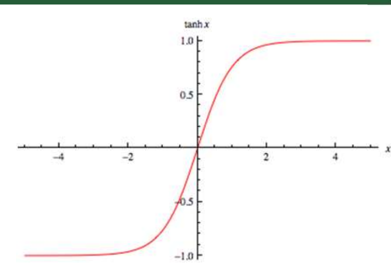
$$\frac{\partial E}{\partial w_1} = \frac{\partial E}{\partial h_2} * \frac{\partial h_2}{\partial a_2} * \frac{\partial a_2}{\partial h_1} * \frac{\partial h_1}{\partial a_1} * \frac{\partial a_1}{\partial w_1}$$

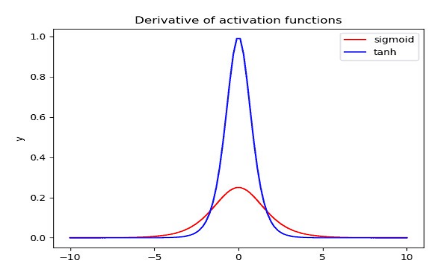
Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Activation Functions and Derivatives

4





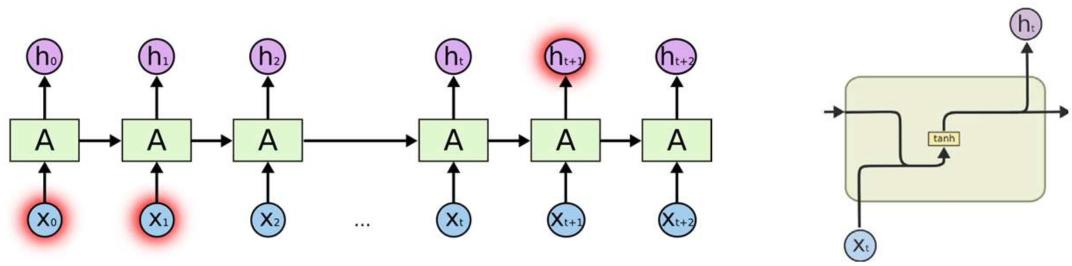


Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Long Distance Dependencies

5

- It is very difficult to train simple recurrent networks (SRNs) to retain information over many time steps
- This makes it very difficult to learn SRNs that handle long-distance dependencies, such as subject-verb agreement.



6

Gated RNNs

Long Short-Term Memory (LSTM)
Gated Recurrent Unit (GRU)

Gated RNNs

7

- The most effective sequence models used in practical applications are called gated RNNs.

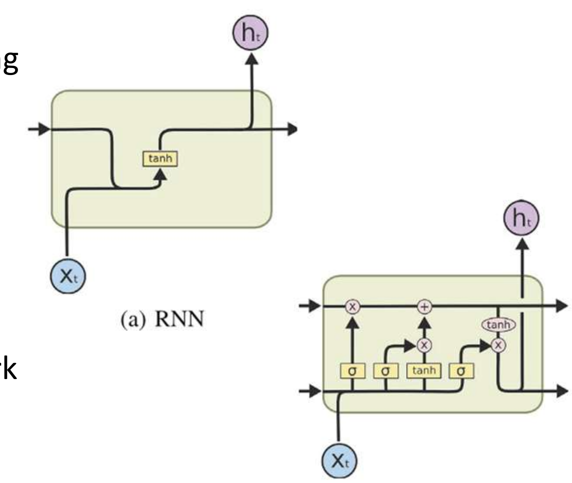
- These include the long short-term memory (LSTM) and networks based on the gated recurrent unit (GRU)
  - Create paths through time that have derivatives that neither vanish nor explode
  - *Accumulate* information such as evidence for a particular feature or category,
  - *Forget* the old state and start over.

Long Short Term Memory

8

- LSTM networks, add additional gating units in each memory cell.
  - Forget gate
  - Input gate
  - Output gate

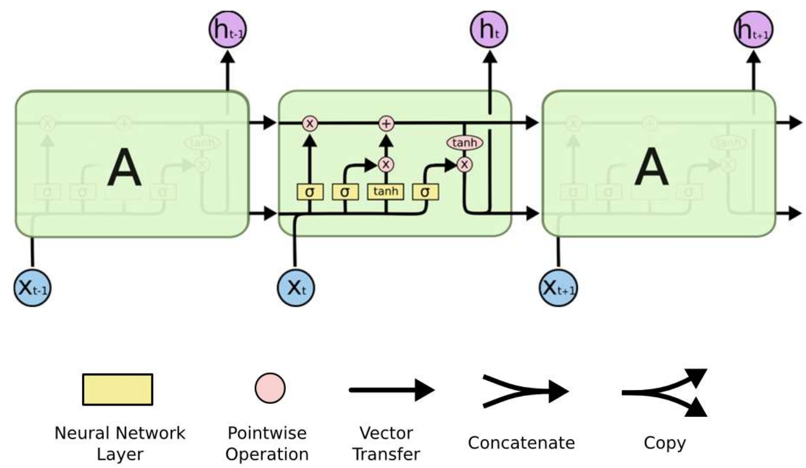
- Prevents vanishing/exploding gradient problem and allows network to retain state information over longer periods of time.



(a) RNN

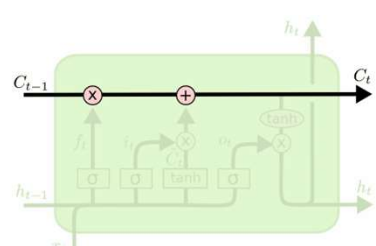
(b) LSTM

LSTM Network Architecture



Cell State

- Maintains a vector C_t that is the same dimensionality as the hidden state, h_t
- Information can be added or deleted from this state vector via the forget and input gates.



Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Cell State Example

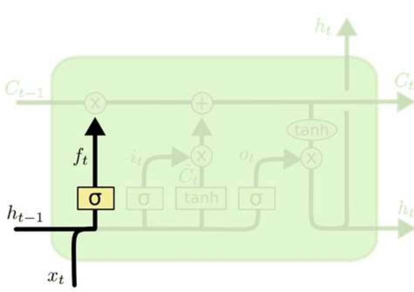
11

- Want to remember person & number of a subject noun so that it can be checked to agree with the person & number of verb when it is eventually encountered.
- Forget gate will remove existing information of a prior subject when a new one is encountered.
- Input gate "adds" in the information for the new subject.

Forget Gate

12

- Forget gate computes a 0-1 value using a logistic sigmoid output function from the input, x_t, and the current hidden state, h_t:
- Multiplicatively combined with cell state, "forgetting" information where the gate outputs something close to 0.

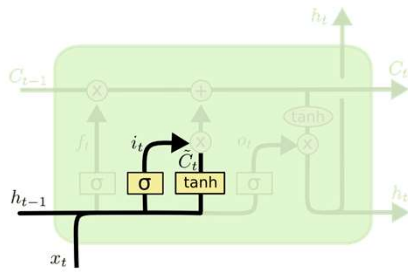


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

Input Gate

13

- First, determine which entries in the cell state to update by computing 0-1 sigmoid output.
- Then determine what amount to add/subtract from these entries by computing a tanh output (valued –1 to 1) function of the input and hidden state.

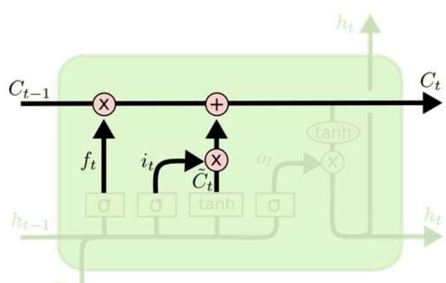


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Updating the Cell State

14

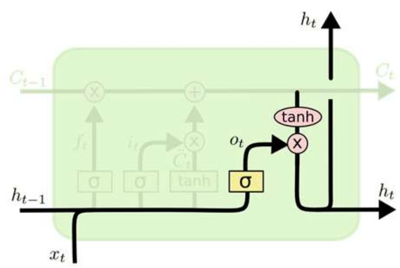
- Cell state is updated by using component-wise vector multiply to "forget" and vector addition to "input" new information.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

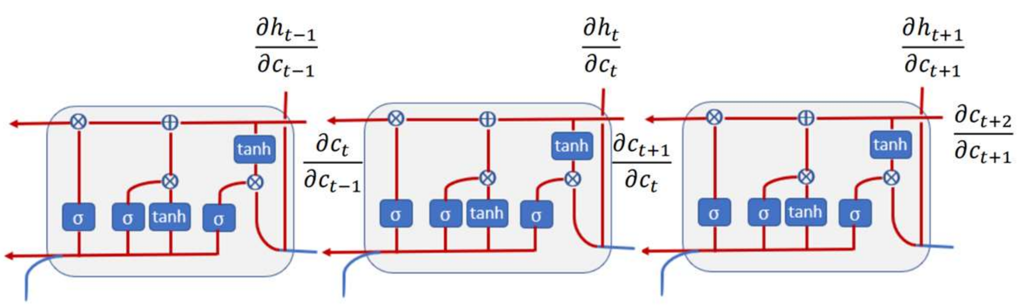
Output Gate

- Hidden state is updated based on a "filtered" version of the cell state, scaled to –1 to 1 using tanh.
- Output gate computes a sigmoid function of the input and current hidden state to determine which elements of the cell state to "output".



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Backpropagation in LSTM



$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$$

$$\frac{\partial E_t}{\partial W} = \frac{\partial E}{\partial h_t}\frac{\partial h_t}{\partial c_t}\frac{\partial c_t}{\partial c_{t-1}} \cdots \frac{\partial c_2}{\partial c_1}\frac{\partial c_1}{\partial W} = \frac{\partial E}{\partial h_t}\frac{\partial h_t}{\partial c_t}\left(\prod_{t=2}^{T}\frac{\partial c_t}{\partial c_{t-1}}\right)\frac{\partial c_1}{\partial W}$$

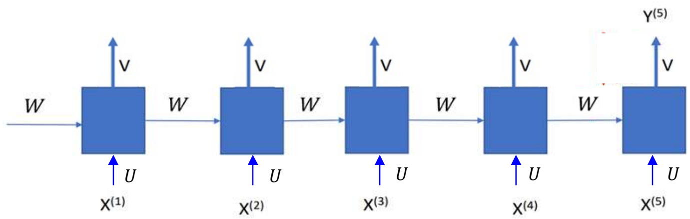
How does this contribute to handle vanishing gradient issue?

Backpropagation in Vanilla RNN

17

for $t = 1, \dots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$



$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$$

Main cause of the vanishing gradient issue

$$\frac{\partial E_t}{\partial W} = \sum_{i=0}^{t-1} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \boxed{\frac{\partial h_t}{\partial h_i}} \frac{\partial h_i}{\partial W}$$

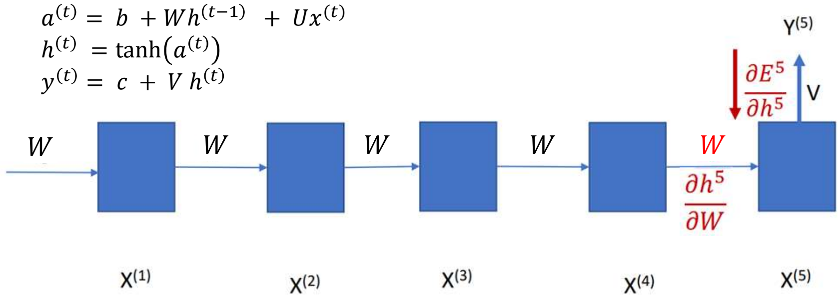
$$\boxed{\frac{\partial h_t}{\partial h_i} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdots \frac{\partial h_{i+1}}{\partial h_i} = \prod_{k=i}^{t-1} \frac{\partial h_{k+1}}{\partial h_k}}$$

Backpropagation in Vanilla RNN

18

for $t = 1, \dots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$



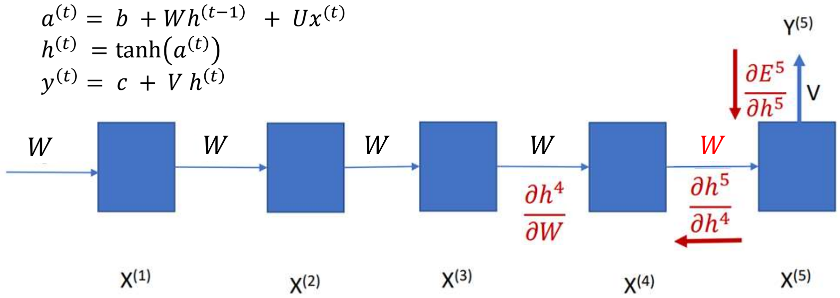
$$\frac{\partial E^5}{\partial W} = \frac{\partial E^5}{\partial y^5} \frac{\partial y^5}{\partial h^5} \frac{\partial h^5}{\partial W}$$

Backpropagation in Vanilla RNN

19

for $t = 1, \ldots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$



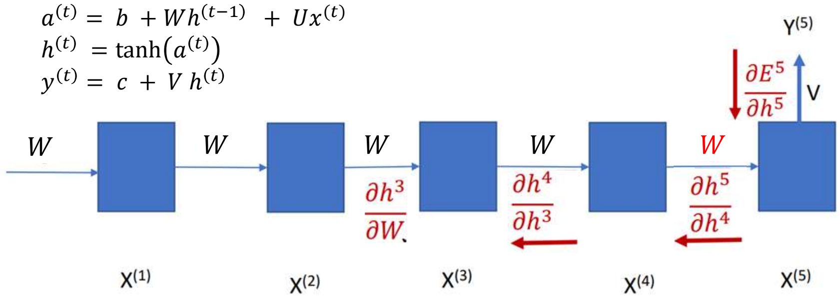
$$\frac{\partial E^5}{\partial W} = \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial W}$$

Backpropagation in Vanilla RNN

20

for $t = 1, \ldots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$

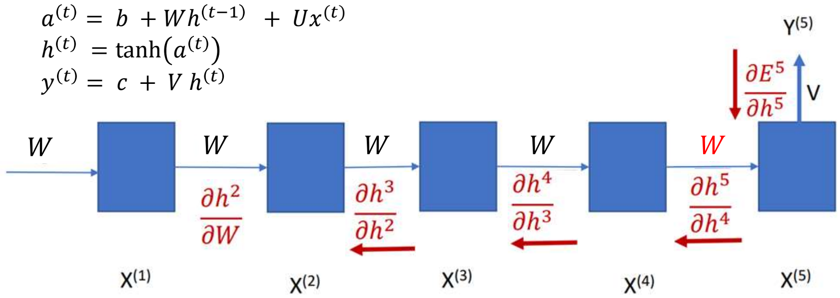


$$\frac{\partial E^5}{\partial W} = \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial W}$$

Backpropagation in Vanilla RNN

for $t = 1, \dots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$

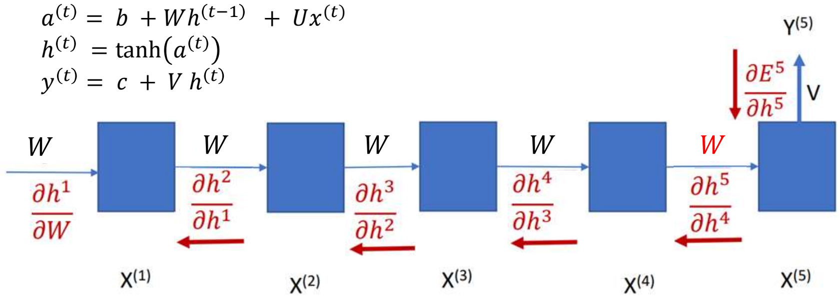


$$\frac{\partial E^5}{\partial W} = \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial h^2}\frac{\partial h^2}{\partial W}$$

Backpropagation in Vanilla RNN

for $t = 1, \dots, \tau$:

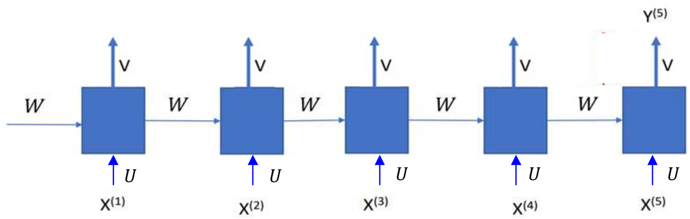
$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$



$$\frac{\partial E^5}{\partial W} = \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial h^2}\frac{\partial h^2}{\partial W} + \frac{\partial E^5}{\partial y^5}\frac{\partial y^5}{\partial h^5}\frac{\partial h^5}{\partial h^4}\frac{\partial h^4}{\partial h^3}\frac{\partial h^3}{\partial h^2}\frac{\partial h^2}{\partial h^1}\frac{\partial h^1}{\partial W}$$

Backpropagation in Vanilla RNN

23



for $t = 1, \dots, \tau$:

$$a^{(t)} = b + Wh^{(t-1)} + Ux^{(t)}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right)$$
$$y^{(t)} = c + V\,h^{(t)}$$

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$$

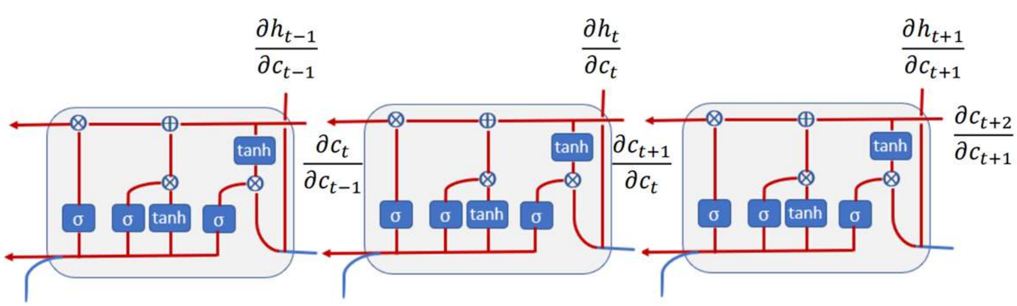
Main cause of the vanishing gradient issue

$$\frac{\partial E_t}{\partial W} = \sum_{i=0}^{t-1} \frac{\partial E_t}{\partial y_t} \frac{\partial y_t}{\partial h_t} \boxed{\frac{\partial h_t}{\partial h_i}} \frac{\partial h_i}{\partial W}$$

$$\boxed{\frac{\partial h_t}{\partial h_i} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdots \frac{\partial h_{i+1}}{\partial h_i} = \prod_{k=i}^{t-1} \frac{\partial h_{k+1}}{\partial h_k}}$$

Backpropagation in LSTM

24



$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W}$$

$$\frac{\partial E_t}{\partial W} = \frac{\partial E}{\partial h_t} \frac{\partial h_t}{\partial c_t} \frac{\partial c_t}{\partial c_{t-1}} \cdots \frac{\partial c_2}{\partial c_1} \frac{\partial c_1}{\partial W} = \frac{\partial E}{\partial h_t} \frac{\partial h_t}{\partial c_t} \left(\prod_{t=2}^{T} \frac{\partial c_t}{\partial c_{t-1}} \right) \frac{\partial c_1}{\partial W}$$

How does this contribute to handle vanishing gradient issue?

Backpropagation in LSTM

25

$$c_t = c_{t-1} \otimes f_t \oplus i_t \otimes \widetilde{c_t}$$

$$\frac{\partial c_t}{\partial c_{t-1}} = \frac{\partial}{\partial c_{t-1}}(c_{t-1} \otimes f_t \oplus i_t \otimes \widetilde{c_t})$$

$$= \frac{\partial}{\partial c_{t-1}}[c_{t-1} \otimes f_t] \oplus [i_t \otimes \widetilde{c_t}]$$

$$= \frac{\partial c_{t-1}}{\partial c_{t-1}} f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \widetilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \widetilde{c_t}$$

$$= f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \widetilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \widetilde{c_t}$$

Backpropagation in LSTM

26

$$c_t = c_{t-1} \otimes f_t \oplus i_t \otimes \widetilde{c_t}$$

$$\frac{\partial c_t}{\partial c_{t-1}} = \frac{\partial}{\partial c_{t-1}}(c_{t-1} \otimes f_t \oplus i_t \otimes \widetilde{c_t})$$

$$= \frac{\partial}{\partial c_{t-1}}[c_{t-1} \otimes f_t] \oplus [i_t \otimes \widetilde{c_t}]$$

$$= \frac{\partial c_{t-1}}{\partial c_{t-1}} f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \widetilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \widetilde{c_t}$$

$$= f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \widetilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \widetilde{c_t}$$

$p \quad q \quad r \quad s$

Backpropagation in LSTM

Plus respective bias

$$p = \quad f_t$$

$$q = \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} = \sigma'(W_f \cdot [h_{t-1}, x_t]) \cdot W_f \cdot o_{t-1} \otimes tanh'(c_{t-1}) \cdot c_{t-1}$$

$$r = \frac{\partial \tilde{c_t}}{\partial c_{t-1}} i_t = tanh'(W_c \cdot [h_{t-1}, x_t]) \cdot W_c \cdot o_{t-1} \otimes tanh'(c_{t-1}) \cdot i_t$$

$$s = \frac{\partial i_t}{\partial c_{t-1}} \tilde{c_t} = \sigma'(W_i \cdot [h_{t-1}, x_t]) \cdot W_i \cdot o_{t-1} \otimes tanh'(c_{t-1}) \cdot \tilde{c}_t$$

Backpropagation in LSTM

$$c_t = c_{t-1} \otimes f_t \oplus i_t \otimes \tilde{c_t}$$

$$\frac{\partial c_t}{\partial c_{t-1}} = \frac{\partial}{\partial c_{t-1}}(c_{t-1} \otimes f_t \oplus i_t \otimes \tilde{c_t})$$

$$= \frac{\partial}{\partial c_{t-1}}[c_{t-1} \otimes f_t] \oplus [i_t \otimes \tilde{c_t}]$$

$$= \frac{\partial c_{t-1}}{\partial c_{t-1}} f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \tilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \tilde{c_t}$$

$$= f_t + \frac{\partial f_t}{\partial c_{t-1}} c_{t-1} + \frac{\partial \tilde{c_t}}{\partial c_{t-1}} i_t + \frac{\partial i_t}{\partial c_{t-1}} \tilde{c_t}$$

$p \quad q \quad r \quad s$

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = tanh(c_t) \otimes o_t$$

$$\frac{\partial E_t}{\partial W} = \frac{\partial E}{\partial h_t} \frac{\partial h_t}{\partial c_t} \left(\prod_{t=2}^{T} (p + q + r + s) \right) \frac{\partial c_1}{\partial W}$$

Additive update!
Helps in handling vanishing gradient issue

LSTM Training

29

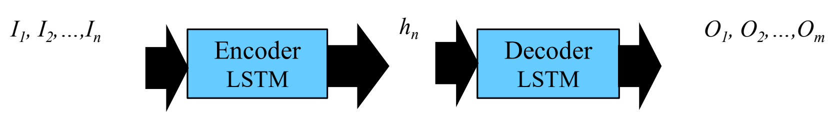
- Trainable with backprop derivatives such as:
  - Stochastic gradient descent with momentum
  - ADAM optimizer

- Each cell has many parameters (W_f, W_i, W_c, W_o)
  - Generally **requires lots of training data.**
  - Requires **lots of compute time that exploits GPU clusters**

Sequence to Sequence Transduction (Mapping)

30

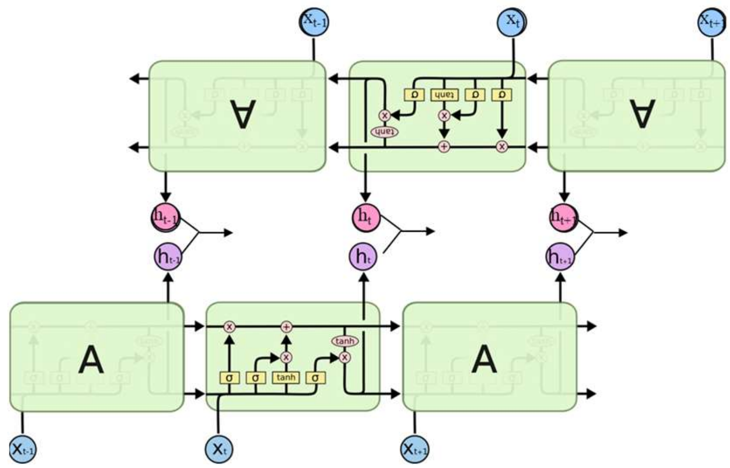
- Encoder/Decoder framework maps one sequence to a "deep vector" then another LSTM maps this vector to an output sequence.



- Train model "end to end" on I/O pairs of sequences.

Bi-directional LSTM (Bi-LSTM)

31



General Problems Solved with LSTMs

32

- **Sequence labeling**
  - Train with supervised output at each time step computed using a single or multilayer network that maps the hidden state (h_t) to an output vector (O_t).
- **Language modeling**
  - Train to predict next input ($O_t = I_{t+1}$)
- **Sequence (e.g. text) classification**
  - Train a single or multilayer network that maps the final hidden state (h_n) to an output vector (O).

Successful Applications of LSTMs

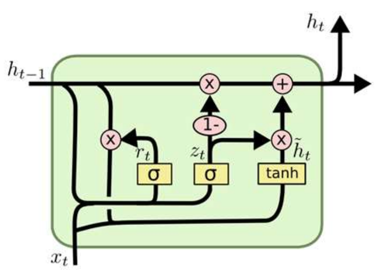
33

- **Speech recognition:** Language and acoustic modeling
- **Sequence labeling**
  - POS Tagging
  - NER
  - Phrase Chunking
- **Neural syntactic and semantic parsing**
- **Image captioning:** CNN output vector to sequence
- **Sequence to Sequence**
  - Machine Translation
  - Video Captioning (input sequence of CNN frame outputs)

Gated Recurrent Unit (GRU)

34

- Alternative RNN to LSTM that uses fewer gates
  - Update gate
  - Reset gate
  - Eliminates cell state vector



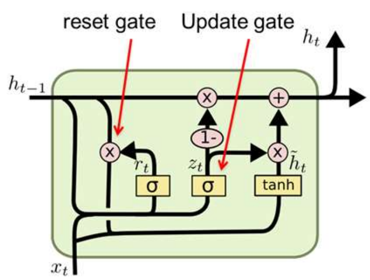
$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

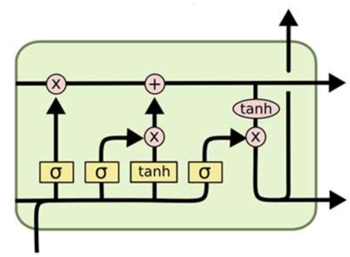
Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

GRU vs. LSTM

35

- GRU has significantly fewer parameters and trains faster.
- Experimental results comparing the two are still inconclusive, many problems they perform the same, but each has problems on which they work better.





Conclusions

36

- By adding “gates” to an RNN, we can prevent the vanishing/exploding gradient problem.
- Trained LSTMs/GRUs can retain state information longer and handle long-distance dependencies.
- Recent impressive results on a range of challenging NLP problems.

RNN Encoder-Decoder with **Attention Mechanism**

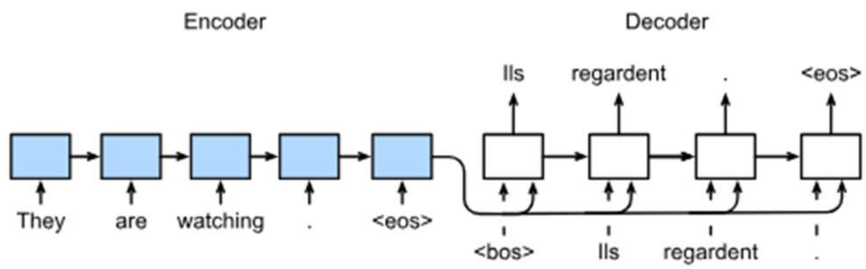
Attention

- For many applications, it helps to add “attention” to RNNs.
- Allows network to learn to attend to different parts of the input at different time steps, shifting its attention to focus on different aspects during its processing.
- Used in image captioning to focus on different parts of an image when generating different parts of the output sentence.
- In Machine Translation, allows focusing attention on different parts of the source sentence when generating different parts of the translation.

Attention in Language Processing

39

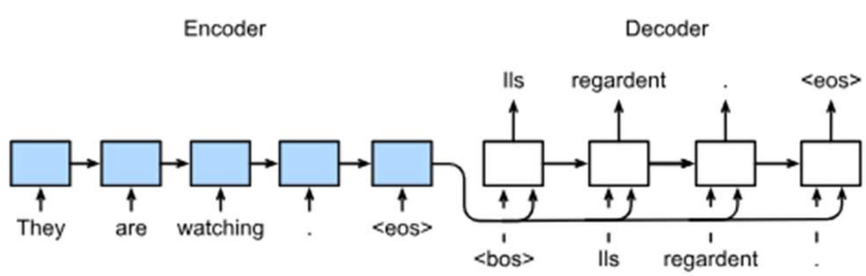
How are you doing? → Aap kaise Hain?



Attention in Language Processing

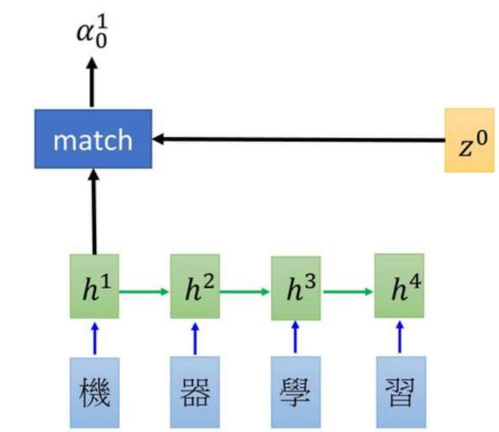
40

How are you doing? → Aap kaise hain?

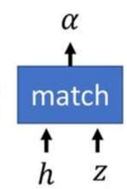


Attention-based Model

41



Jointly learned with other part of the network



What is match ?

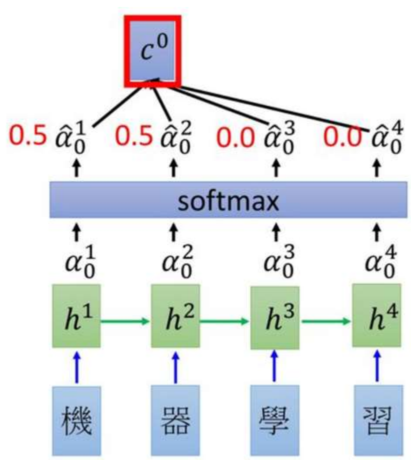
Design by yourself

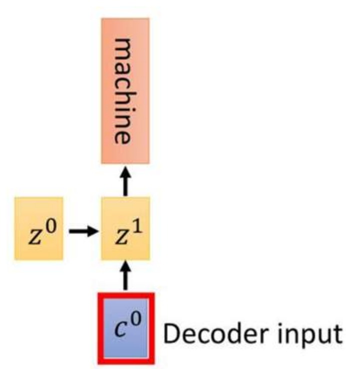
- Cosine similarity of z and h
- Small NN whose input is z and h, output a scalar
- $\alpha = h^T W z$

Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Attention-based Model

42





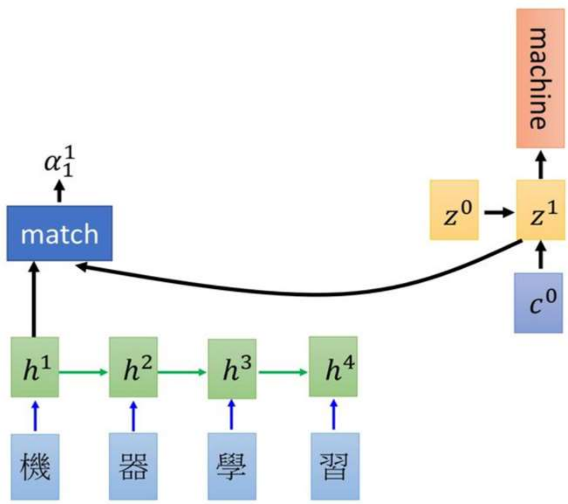
$$c^0 = \sum \hat{\alpha}_0^i h^i$$

$$= 0.5h^1 + 0.5h^2$$

Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

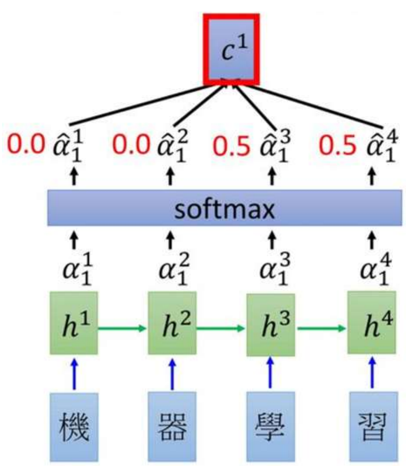
Attention-based Model

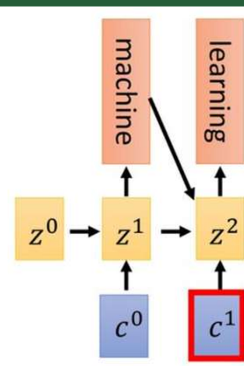
43



Attention-based Model

44

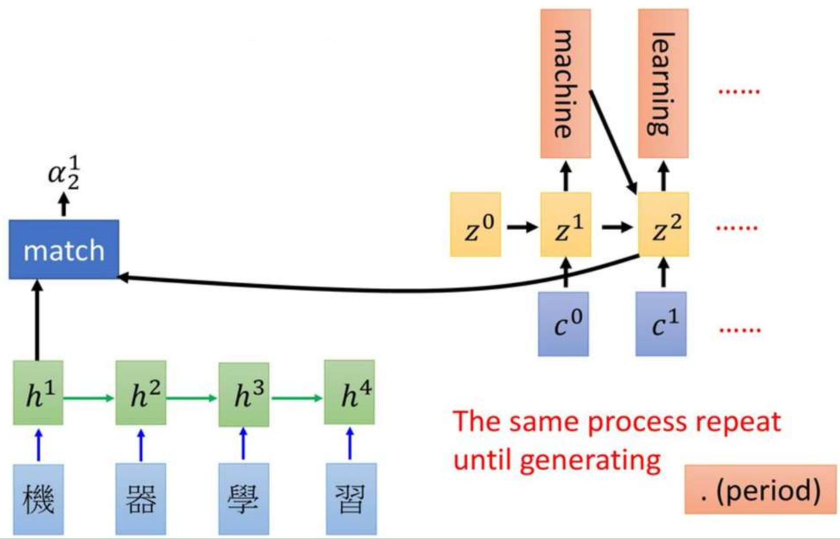




$$c^1 = \sum \hat{\alpha}_1^i h^i$$
$$= 0.5h^3 + 0.5h^4$$

Attention-based Model

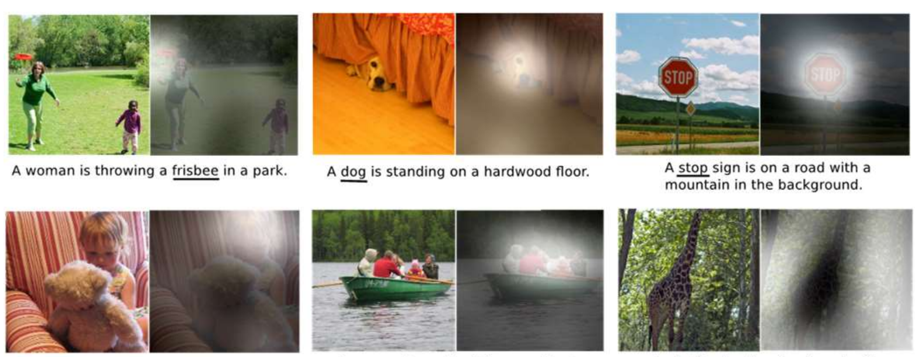
45



Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Attention in Image Caption Generation

46



A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015

Prof. Monidipa Das, Department of CSE, IIT (ISM) Dhanbad

Image caption generation using attention

47

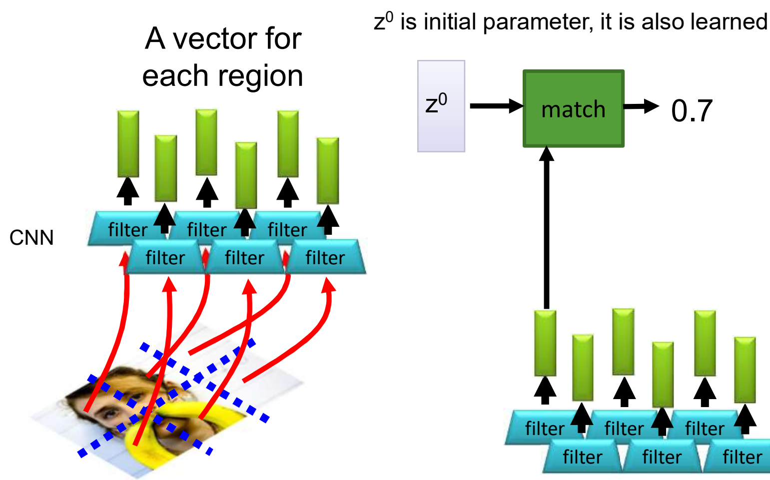


Image Caption Generation

48

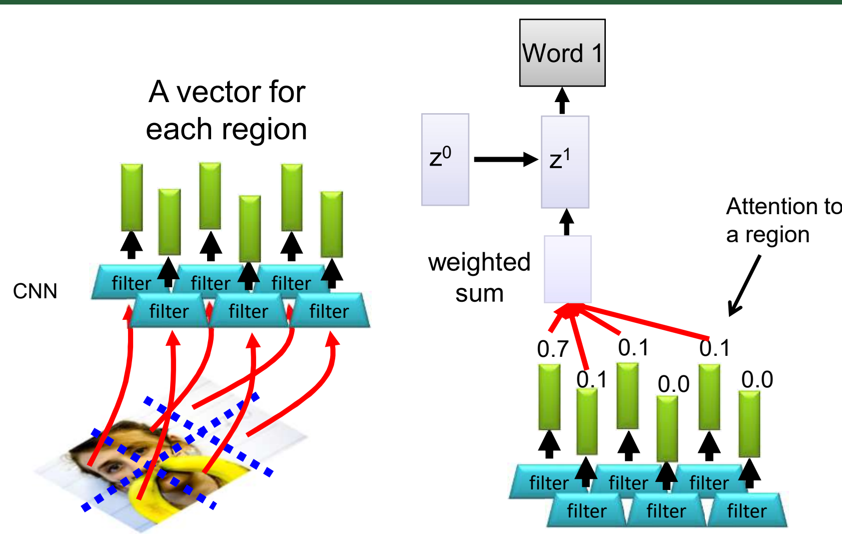
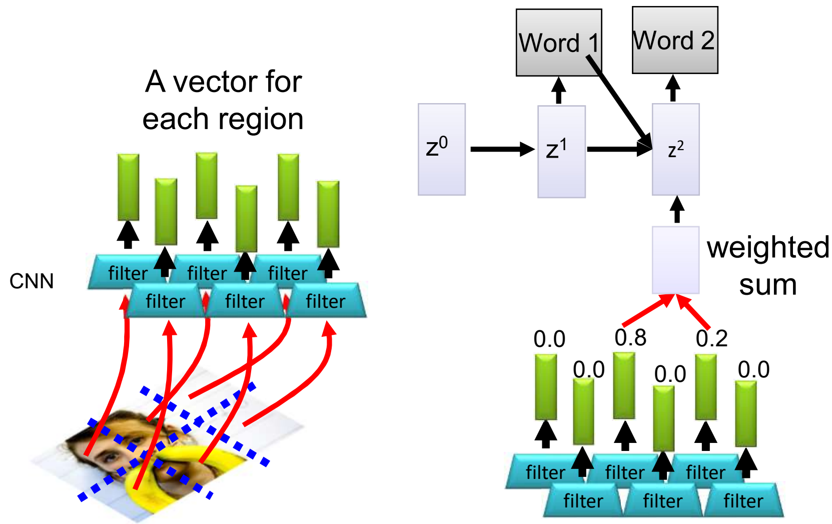


Image Caption Generation

49



50

Questions?