Neural Networks and Deep Learning Lecture 10

Wei WANG

cs5242@comp.nus.edu.sg

NUS National University of Singapore

School of Computing

Administrative

- Project
  - Kaggle submission, 9 Apr 2018 (Week 12), 17:00
  - Report and code, 16 Apr 2018 (Week 13), 17:00
  - Submission
    - Report
      - ~ 4 pages with the follow sections
        - task description, model introduction, experiment setting and result analysis
      - Evaluation based on clarity, completeness and findings. NOT based on number of pages.
    - Code
      - See the next slide for the submission folder structure
- Choose 2 submissions for private leader board ranking on kaggle
  - Final evaluation is based on the private leader board
- Rename your kaggle group name as Group ID (e.g. Group 08).

- Code folder structure
  - report.pdf (including the group member information)
  - workspace
    - train.py or train.sh
    - predict.py or predict.sh
    - readme.txt (for any other things to be noted for evaluation, e.g. the running environment, GPU, CUDA, Cudnn versions, memory requirement)
    - requirements.txt (third party library name and version)
    - test.csv (to be generated by predict.py or predict.sh for submission)
    - data

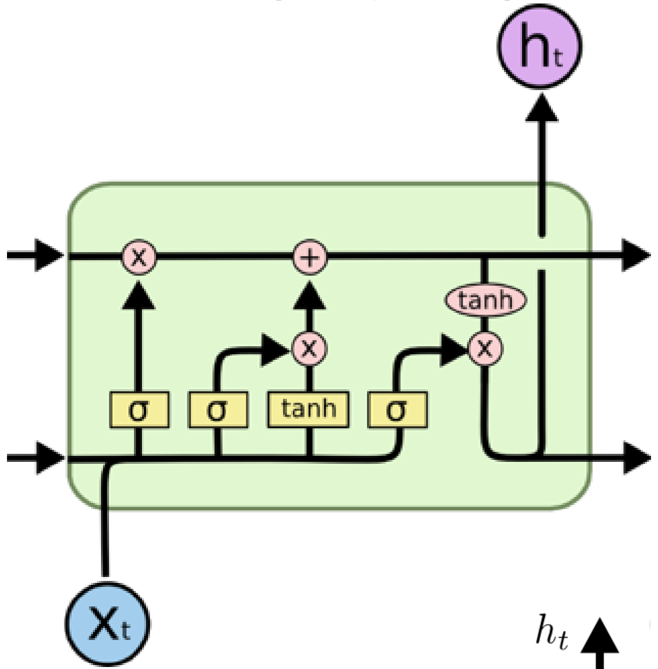
- **NOTE**:
  - You can have other code files or folders in workspace
  - Before you zip the submission folder, **remove the data folder** and other intermediate data (including the checkpoint of the models and word vectors).
  - Do not use absolute path like "/home/wangwei/cs5242/project1/workspace/data". Use relative path like "./data/"

Recap

Greedy search VS beam search

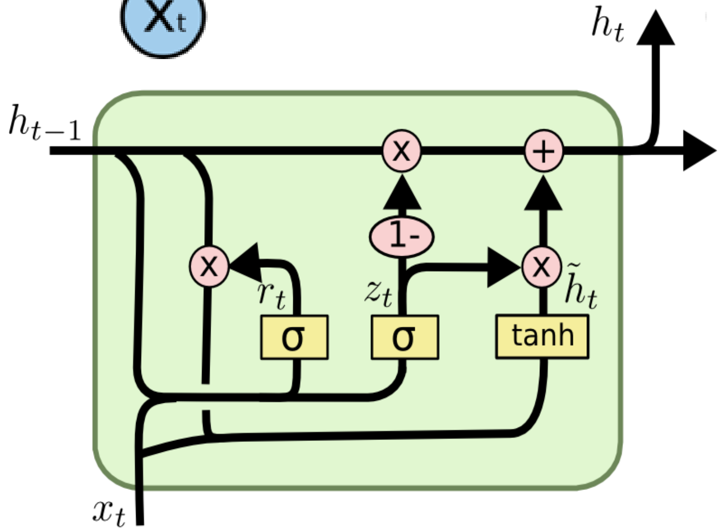
- For sampling data from a pretrained RNN model
  - Generating next words by given some seeding words
  - Generating the translation for machine translation
  - Generating the answers for question answering
- Greedy search
  - Always choose the word with the largest conditional probability for every t
    - $P(x_t|x_{t-1}, x_{t-2}, \dots, x_1)$
  - Drawback?
- Beam search
  - Choose the next word based on the joint probability
    - $P(x_t,\ x_{t-1}, x_{t-2}, \dots, x_1) = P(x_t|x_{t-1}, x_{t-2}, \dots, x_1) * P(x_{t-1}, x_{t-2}, \dots, x_1)$
  - Keep the top-k assignments based on the joint probability

LSTM VS GRU



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$h_t = o_t * \tanh\left(C_t\right)$$



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

[h, x] Concatenate h and x to share the same weight matrix; The following link learns one weight matrix for h and x respectively
https://github.com/DoctorTeeth/gru/blob/master/gru.py

Bi-directional RNN

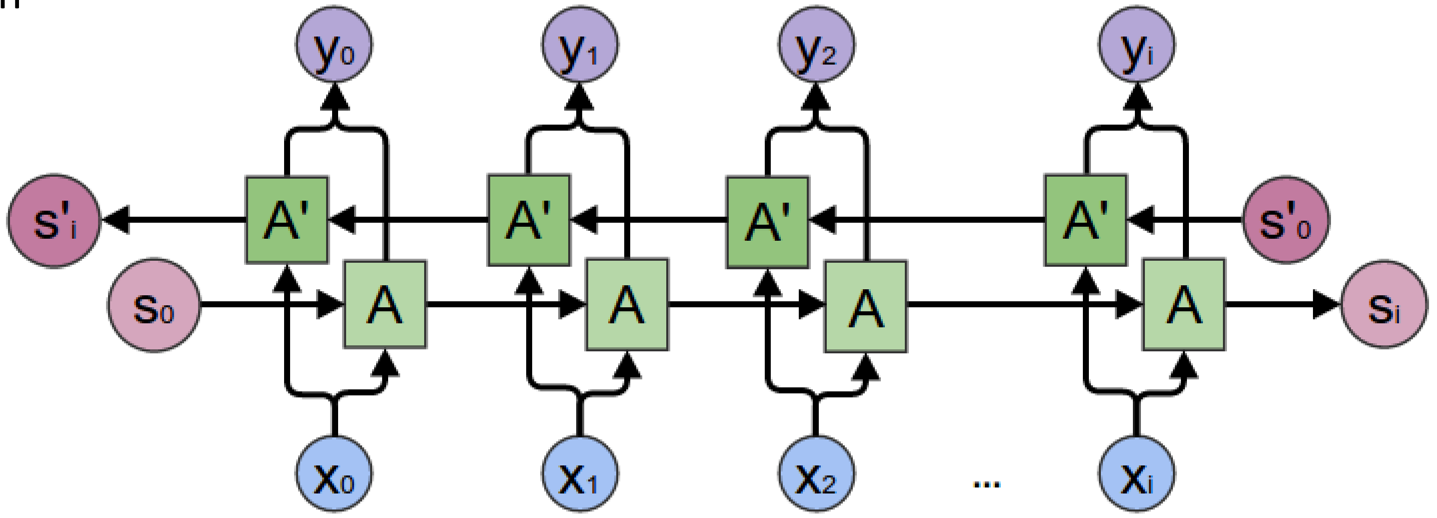
Why do we need it?

To have context information at each unit/time step→ better hidden state

Speech recognition

Handwritten recognition

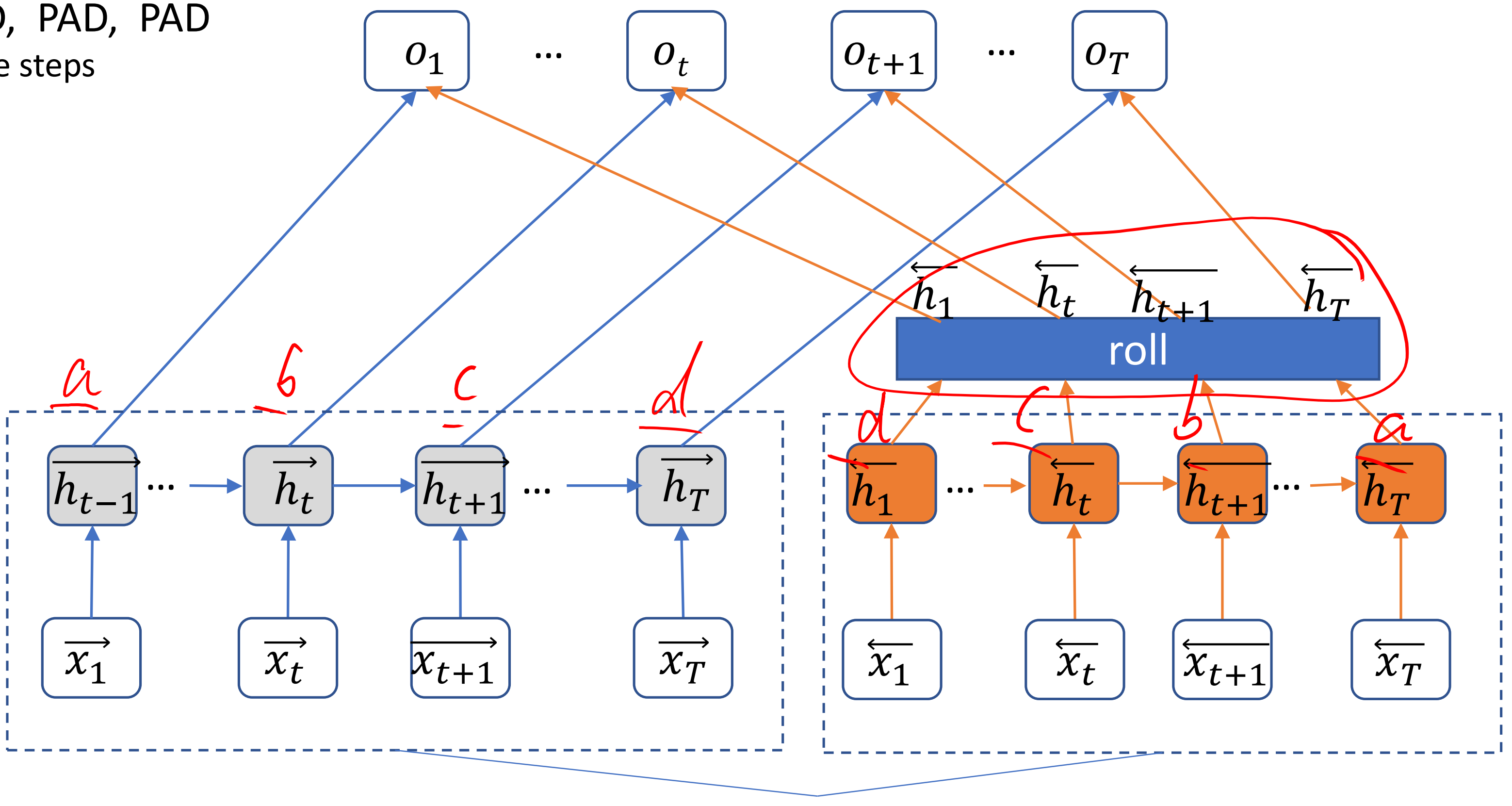
Machine translation



source from: http://colah.github.io/posts/2015-09-NN-Types-FP/

Bi-directional RNN (implementation)

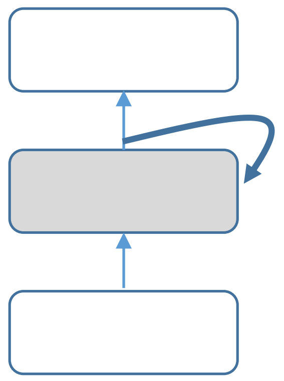
- Pre-process the inputs
  - $\overrightarrow{x}$: a, b, c, d, PAD, PAD, PAD
  - $\overleftarrow{x} = roll\,(\overrightarrow{x})$: d, c, b, a, PAD, PAD, PAD
    - roll(): reverse the data from valid time steps
- Using vanilla RNN
  - For t=1 to T
    - $\overrightarrow{h_t} = \tanh(\overrightarrow{U}\overrightarrow{x_t} + \overrightarrow{W}\overrightarrow{h_{t-1}} + \vec{b})$
    - $\overleftarrow{h_t} = \tanh(\overleftarrow{U}\overleftarrow{x_t} + \overleftarrow{W}\overleftarrow{h_{t+1}} + \overleftarrow{b})$
  - $\overleftarrow{h} = roll(\overleftarrow{h})$
  - For t =1 to T
    - $o_t = V[\overrightarrow{h_t}, \overleftarrow{h_t}] + c$
    - [,] concatenate
- Using LSTM/GRU
  - $\overrightarrow{h_t}$ =LSTM$_{lr}$$(\overrightarrow{h_{t-1}}, \overrightarrow{c_{t-1}}, \overrightarrow{x_t})$
  - $\overleftarrow{h_t}$ =LSTM$_{rl}$ $(\overleftarrow{h_{t+1}}, \overleftarrow{c_{t+1}}, \overleftarrow{x_t})$


o_1 … o_t o_{t+1} … o_T
$\overleftarrow{h_1}$ $\overleftarrow{h_t}$ $\overleftarrow{h_{t+1}}$ $\overleftarrow{h_T}$
roll
a b c d
d c b a
$\overrightarrow{h_{t-1}}$ … $\overrightarrow{h_t}$ $\overrightarrow{h_{t+1}}$ … $\overrightarrow{h_T}$
$\overleftarrow{h_1}$ … $\overleftarrow{h_t}$ $\overleftarrow{h_{t+1}}$ … $\overleftarrow{h_T}$
$\overrightarrow{x_1}$ $\overrightarrow{x_t}$ $\overrightarrow{x_{t+1}}$ $\overrightarrow{x_T}$
$\overleftarrow{x_1}$ $\overleftarrow{x_t}$ $\overleftarrow{x_{t+1}}$ $\overleftarrow{x_T}$

Normal RNN (vanilla RNN, LSTM, GRU)

RNN Applications and Architectures

Overview



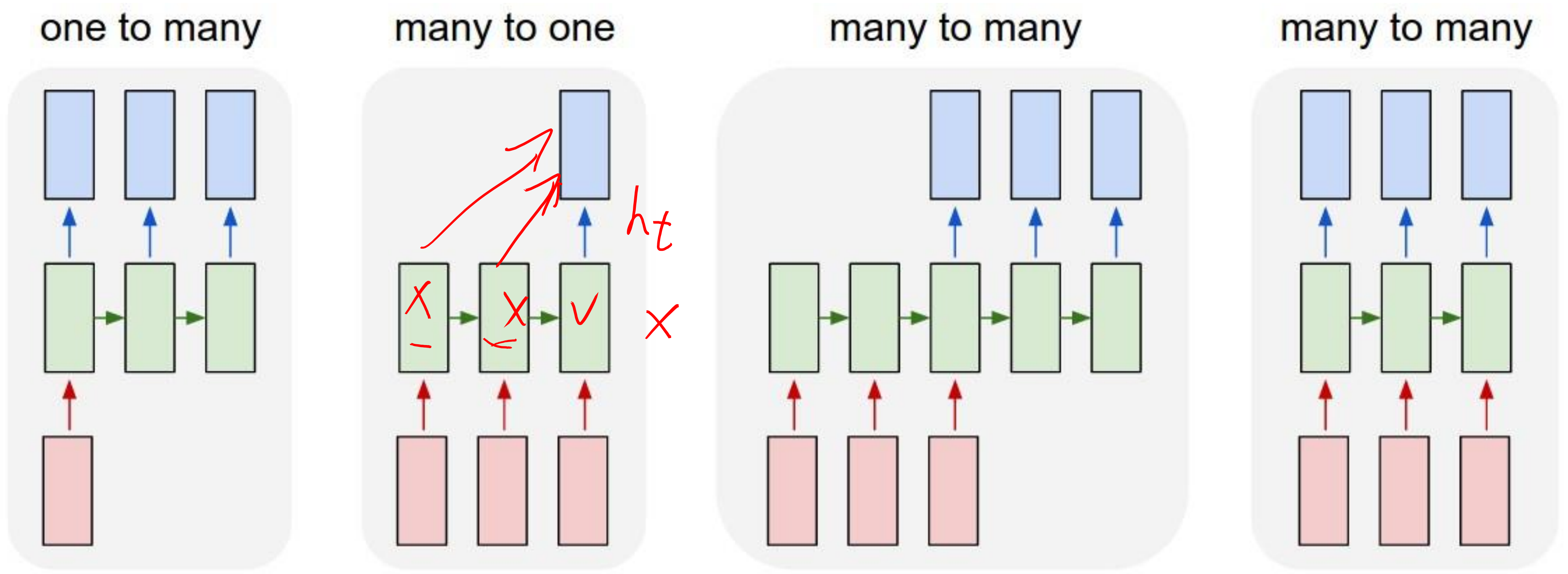


Image caption

Sentiment analysis

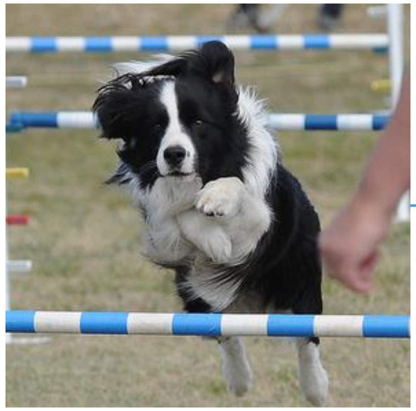
Machine translation, Question answering

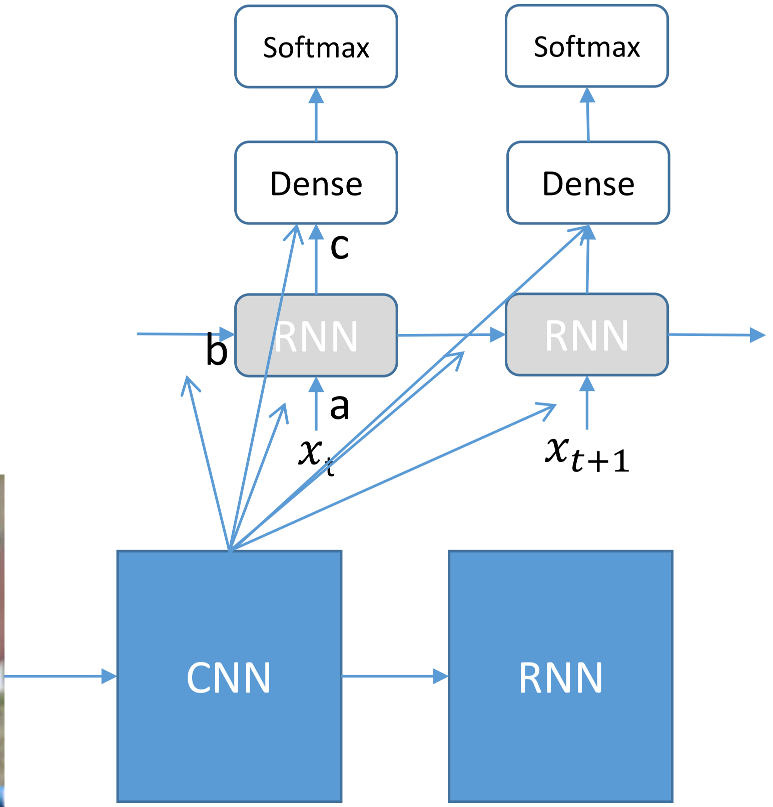
Language modelling

Image source: https://karpathy.github.io/2015/05/21/rnn-effectiveness/

Image caption generation [14]

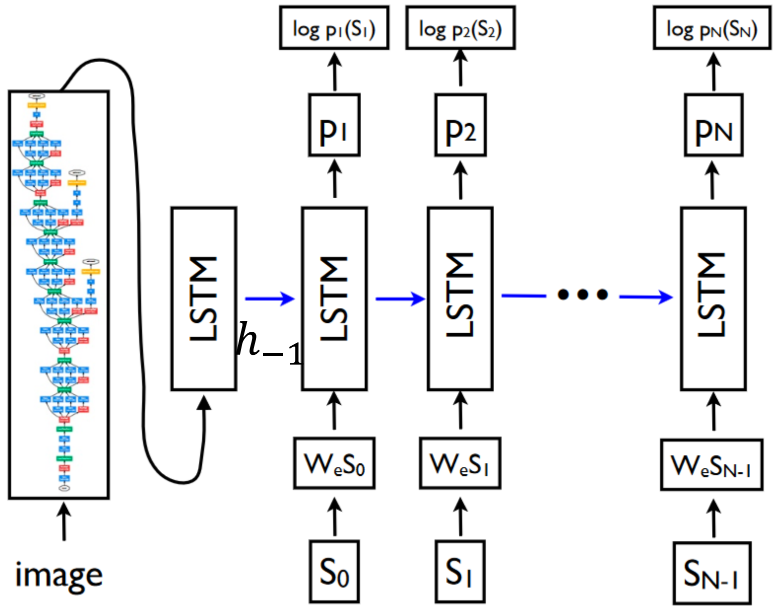
- Given an image
  - Generate a description for it
- Datasets
  - Flickr 8K, Flickr 30K and MS COCO
- Metric
  - BLEU [16]
- Solution
  - CNN for visual feature extraction
  - RNN for word generation
  - How to combine the two models?
    - $P(x_1, x_2, \dots, x_n | I) = \prod_t P(x_t | I, x_{t-1}, x_{t-2}, \dots, x_1)$





Show and Tell: A Neural Image Caption Generator [15]

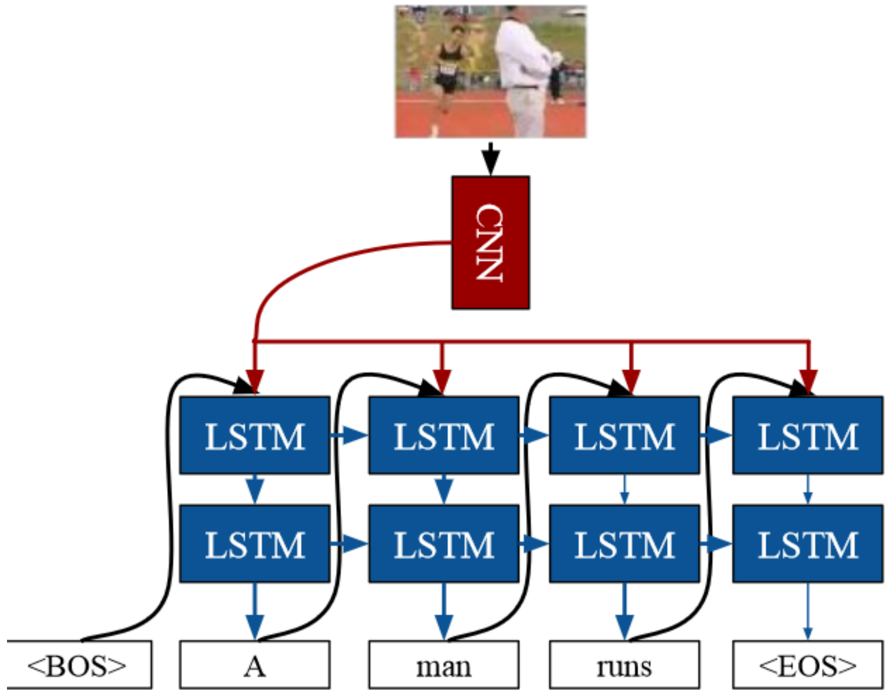
- $P(x_t|I, x_{t-1}, x_{t-2}, \dots, x_1, x_0 = S_0)$
  - S_0 a special word, 'START'
  - $P(x_1|I,\ x_0 = S_0)$
    - $= P(x_1|x_0 = S_0, h_{-1})$
    - h_{-1} is generated from image I
  - $P(x_t|I, x_{t-1}, x_{t-2}, \dots, x_1, x_0 = S_0)$
    - $P(x_t|x_{t-1}, x_{t-2}, \dots, x_1, x_0 = S_0, h_{-1})$
- CNN feature is used only once
  - Inferior results if feed it into every position
- Fine-tune the top layer of CNN



S_k is one-hot vector for a word, $R^{|V|}$
W_e is the embedding matrix, $R^{|V|*m}$
m is the output feature size of CNN

Long-term Recurrent Convolutional Networks for Visual Recognition and Description [18]

- CNN feature is concatenated with input
  - Detailed experiments on
    - beam width (3-5)
    - Integration of CNN feature (right figure)
    - Fine-tuning of CNN parameters (yes)
    - LSTM VS Vanilla RNN (LSTM)
    - Multi-stack LSTM (2)

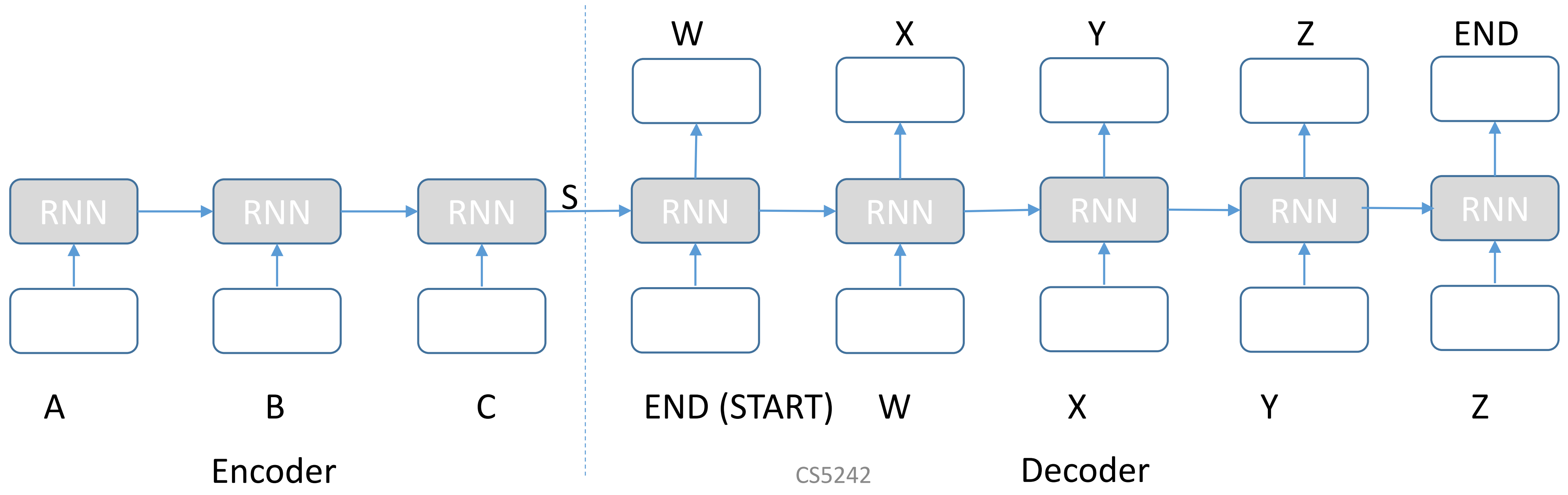


Machine translation [19]

- Given a sentence in one language, e.g. English
  - Singapore MRT is not stable
- Return a sentence in another language, e.g. Chinese
  - 新加坡地铁不稳定
- Maximize $\sum_{<x,y>} logP(y_1, y_2, \ldots, y_m | x_1, x_2, \ldots, x_n)$
  - RNN is good at processing sequence of words

Sequence to Sequence [19]

- Seq2Seq
  - $P(y_1, y_2, \ldots, y_m | x_1, x_2, \ldots, x_n) = P(y_1, y_2, \ldots, y_m | S)$
  - S is a summary of input
  - $logP(y_1, y_2, \ldots, y_m | S) = \sum_i logP(y_t | y_1, \ldots, y_{t-1}, S)$

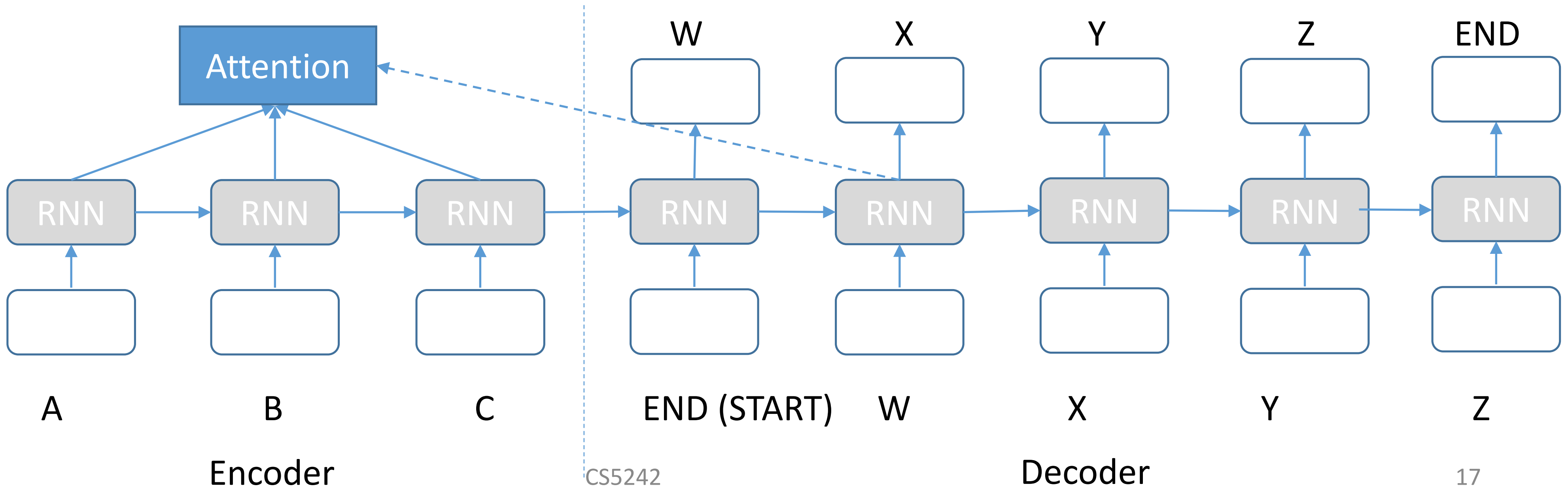


Sequence to Sequence[19]

- Seq2Seq
  - $P(y_1, y_2, \ldots, y_m | x_1, x_2, \ldots, x_n) = P(y_1, y_2, \ldots, y_m | S)$
  - S is a summary of input
  - $logP(y_1, y_2, \ldots, y_m | S) = \sum_i logP(y_t | y_1, \ldots, y_{t-1}, S)$
  - Encoder and Decoder are two RNN networks
    - They have their own parameters
    - End-to-end training
  - Reverse the input sequence
    - $x_1, x_2, \ldots, x_n \rightarrow x_n, x_{n-1}, \ldots, x_1$
    - Generate correct $y_1, y_2, ..$ → overall better output sequence ?
      - x_1 is near y_1, x_2 is near y_2, …
    - Multiple stacks of RNN → better output sequence

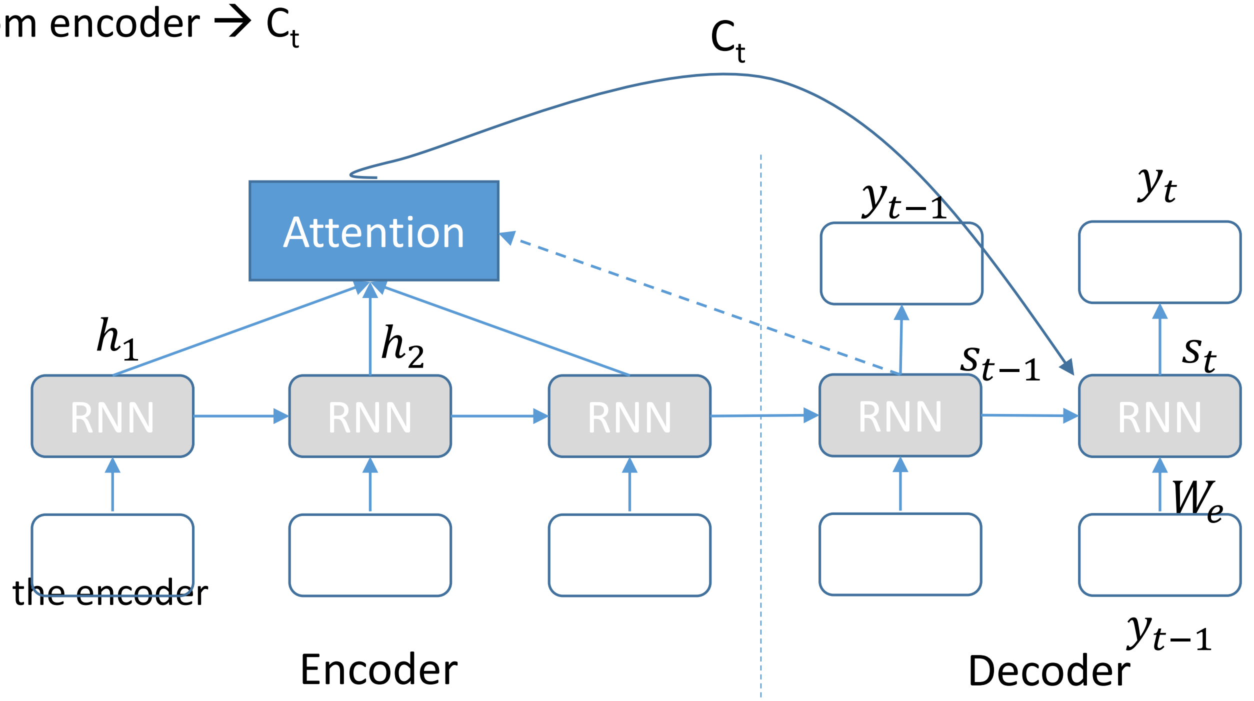
Attention modelling [20]

- Each output word depends on
  - All input words (hidden states), each with different contribution
    - Singapore MRT is not stable
    - 新加坡 地铁   不   稳定



Attention modelling [20]

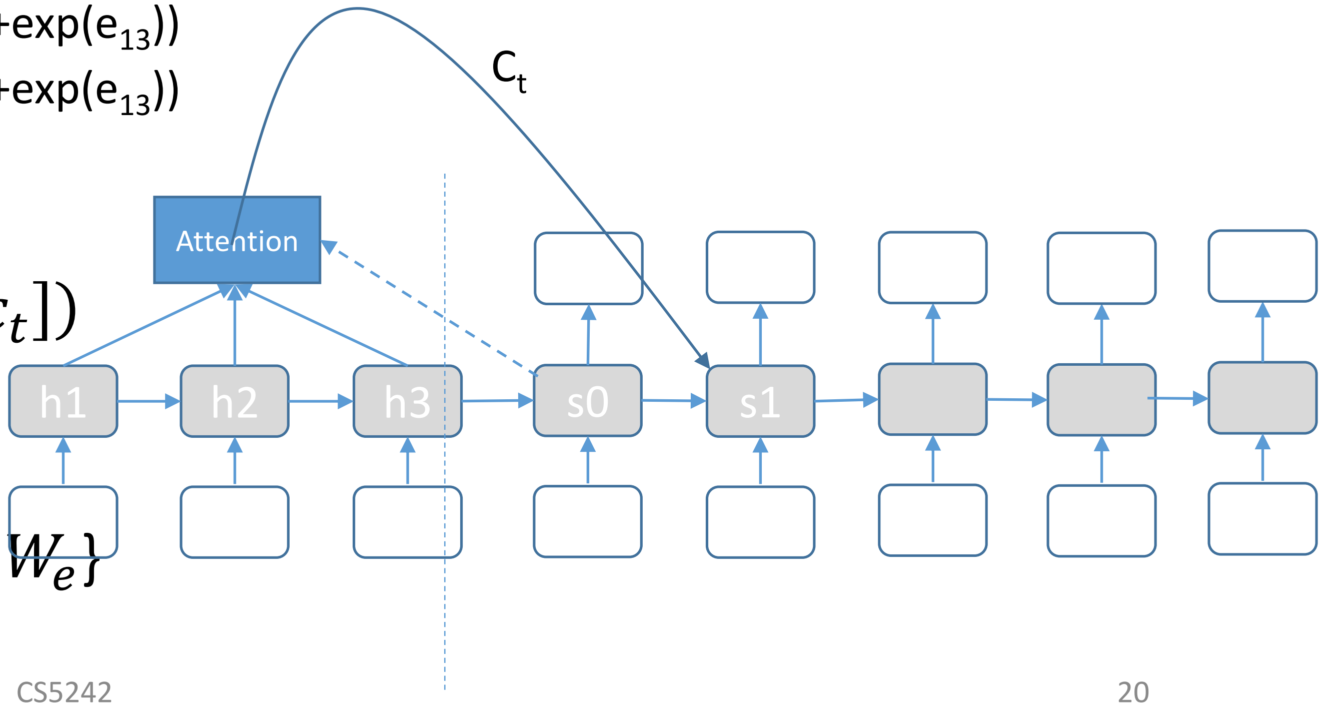
- $logP(y_1, y_2, \ldots, y_m | x_1, x_2, \ldots, x_n) = \sum_t logP(y_t | y_1, \ldots, y_{t-1}, x_1, x_2, \ldots, x_n) = \sum_t logP(y_t | y_1, \ldots, y_{t-1}, h_1, h_2, \ldots h_n) = \sum_t logP(y_t | s_t)\, P(s_t | y_1, \ldots, y_{t-1}, h_1, h_2, \ldots h_n)$
- Extended GRU for the decoder
  - consider the related words from the encoder during decoding
    - Weighted combination of hidden states from encoder → C_t
  - $s_t = (1 - z_t) \circ s_{t-1} + z_t \circ \tilde{s}_t$
  - $\tilde{s}_t = \tanh(W([r_t \circ s_{t-1}, W_e y_{t-1}, c_t])$
  - $r_t = \sigma(W_r[s_{t-1}, W_e y_{t-1}, c_t])$
  - $z_t = \sigma(W_z[s_{t-1}, W_e y_{t-1}, c_t])$
  - $c_t = \sum_{j=1}^{n} \alpha_{tj} h_j$
    - α_{tj} attention weight
      - $\alpha_{tj=softmax(e_{tj})}$ j=1…n
      - $e_{tj} = v_a^T \tanh(W_a s_{t-1} + U_a h_j)$
      - Larger weight for more related words from the encoder



Attention modelling

- Encoder
  - Input word vectors: a=[0.1,0,1], b=[1,0.1,0], c=[0.2,0.3,1]
  - Hidden representation (vector)
    - h1, h2, h3
    - h1=tanh(Ua+Wh0)
    - h2=tanh(Ub+Wh1)
    - h3=tanh(Uc+Wh2)

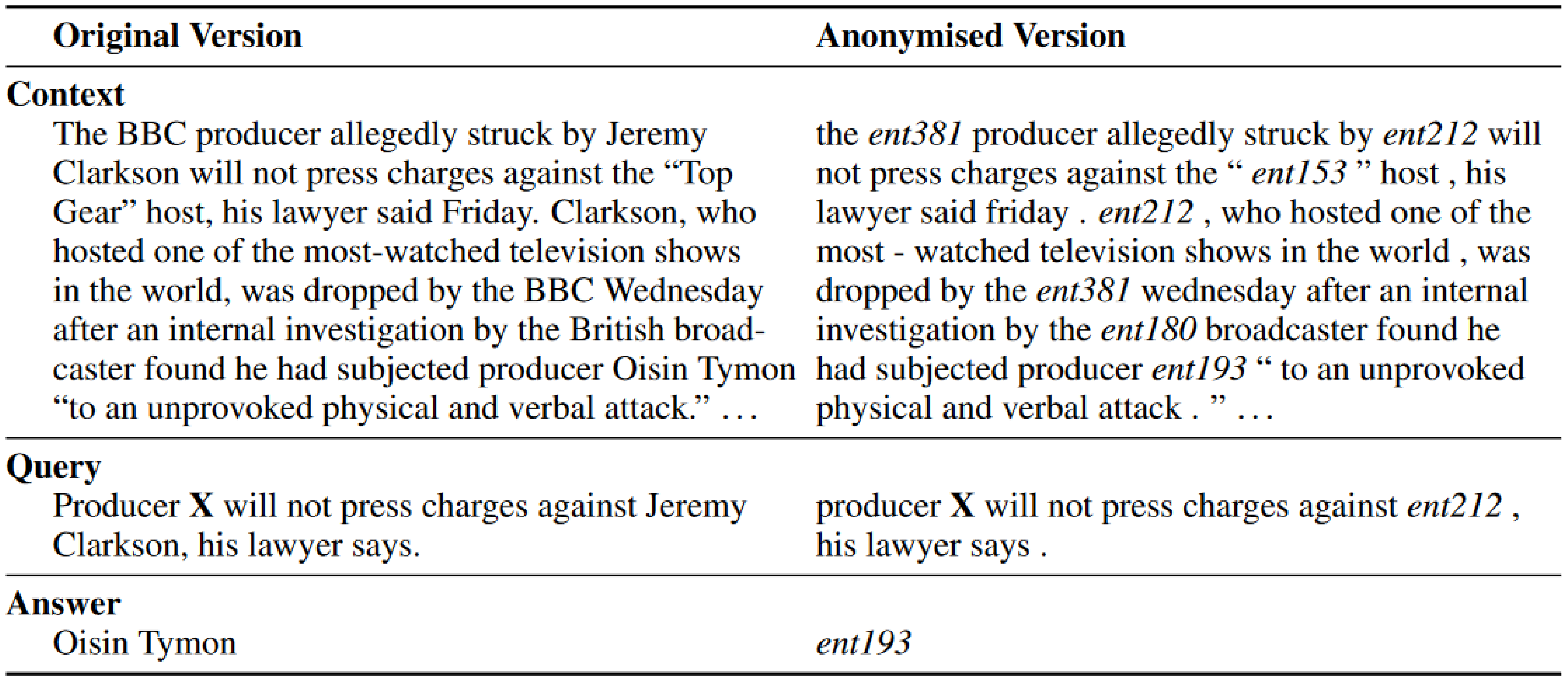
- Decoder
  - Given hidden state vector s_0
    - To compute the weights of h1, h2, h3 for computing s1
      - $e_{11} = a(s_0, h_1)$, $e_{12}=a(s_0, h_2)$, $e_{13}=a(s_0, h_3)$
      - $a(s_0, h_1) = v^T \tanh(W_a s_0 + U_a h_1)$
      - $a(s_0, h_2) = v^T \tanh(W_a s_0 + U_a h_2)$
      - $a(s_0, h_3) = v^T \tanh(W_a s_0 + U_a h_3)$
      - $\alpha_{11} = \exp(e_{11}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - $\alpha_{12} = \exp(e_{12}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - $\alpha_{13} = \exp(e_{13}) / (\exp(e_{11})+\exp(e_{12})+\exp(e_{13}))$
      - $c_1=\alpha_{11}h_1+\alpha_{12}h_2+\alpha_{13}h_3$
  - $s_t = (1 - z_t) \circ s_{t-1} + z_t \circ \tilde{s}_t$
  - $\tilde{s}_t = \tanh(W([r_t \circ s_{t-1}, W_e y_{t-1}, c_t])$
  - $r_t = \sigma(W_r[s_{t-1}, W_e y_{t-1}, c_t])$
  - $z_t = \sigma(W_z[s_{t-1}, W_e y_{t-1}, c_t])$
  - Parameters: {v, W_a, U_a, W, W_r, W_z, W_e}



Question answering

- Given a context, a question
- Outputs the answer
  - A word
  - A substring of the context
  - A new sentence

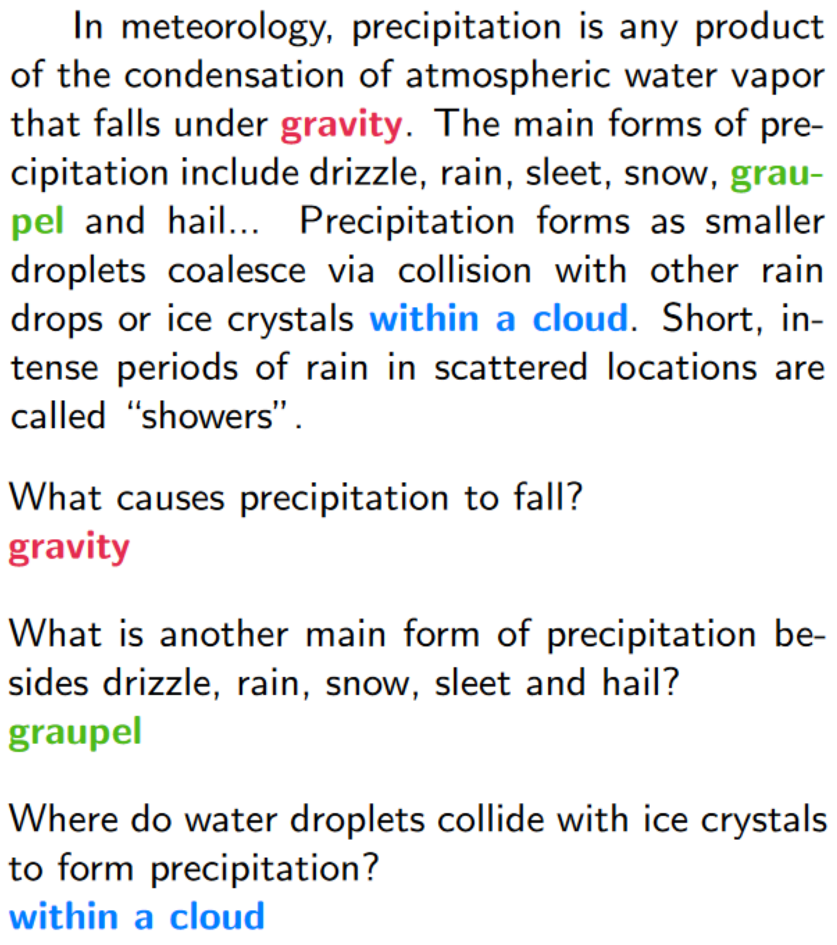
Example



	Original Version	Anonymised Version
Context	The BBC producer allegedly struck by Jeremy Clarkson will not press charges against the "Top Gear" host, his lawyer said Friday. Clarkson, who hosted one of the most-watched television shows in the world, was dropped by the BBC Wednesday after an internal investigation by the British broadcaster found he had subjected producer Oisin Tymon "to an unprovoked physical and verbal attack." . . .	the *ent381* producer allegedly struck by *ent212* will not press charges against the " *ent153* " host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* " to an unprovoked physical and verbal attack . " . . .
Query	Producer **X** will not press charges against Jeremy Clarkson, his lawyer says.	producer **X** will not press charges against *ent212* , his lawyer says .
Answer	Oisin Tymon	*ent193*

Source from [21]

Example



In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

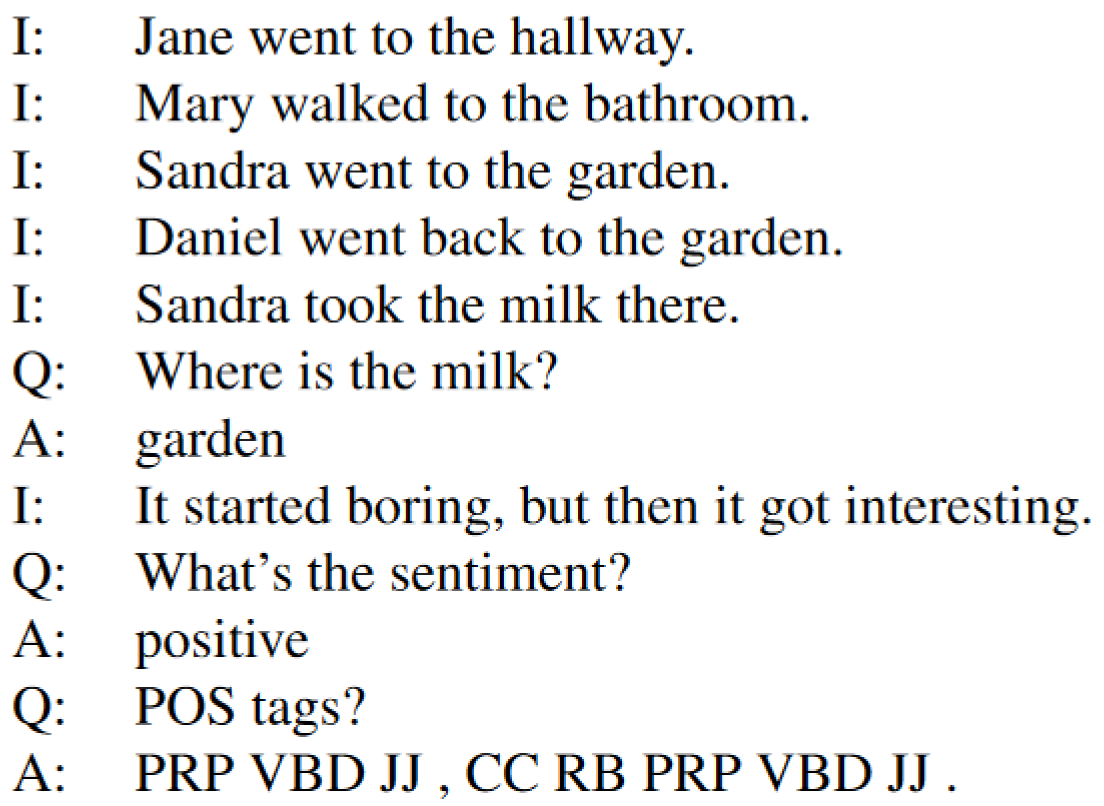
What causes precipitation to fall?
gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

Source from [22]

Example

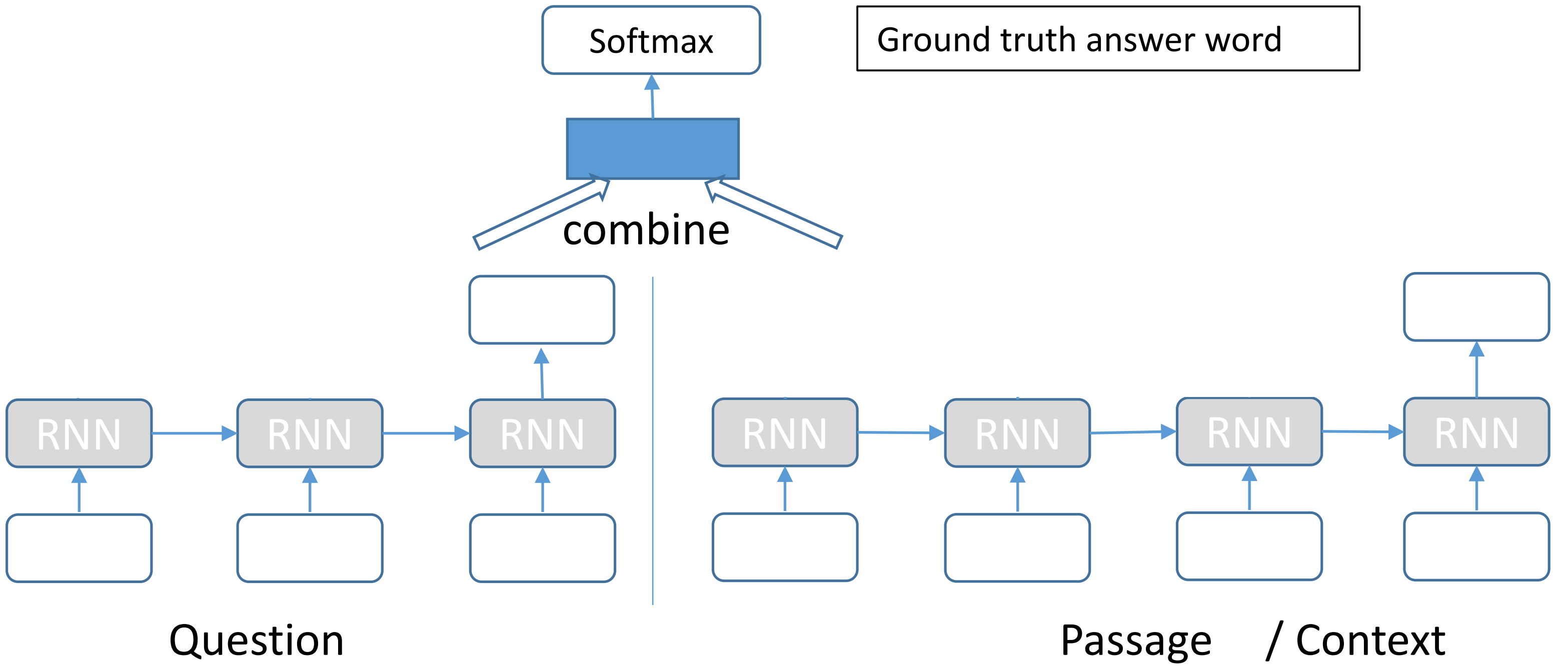


I:	Jane went to the hallway.
I:	Mary walked to the bathroom.
I:	Sandra went to the garden.
I:	Daniel went back to the garden.
I:	Sandra took the milk there.
Q:	Where is the milk?
A:	garden
I:	It started boring, but then it got interesting.
Q:	What's the sentiment?
A:	positive
Q:	POS tags?
A:	PRP VBD JJ , CC RB PRP VBD JJ .

Source from [23]

Solution [24]

- P(a=w|c, q) = P(a=w|cq)~similarity(w, cq)
- Steps
  1. Extract representation of question and passage (context)
  2. Combine question and context
     - Concatenation
     - Addition
  3. Generate the prediction
     - Matching the combined feature with each candidate word feature
       - E.g. inner-product
       - Use the similarity as input to softmax


Softmax
Ground truth answer word
combine
RNN
RNN
RNN
RNN
RNN
RNN
RNN
Question
Passage / Context

Summary

- Conditional RNN for image caption generation
- Seq2seq (attention modelling) for machine translation
- Complex combination for question answering

Reference

- [1] https://www.quora.com/What-are-differences-between-recurrent-neural-network-language-model-hidden-markov-model-and-n-gram-language-model
- [2] https://code.google.com/archive/p/word2vec/
- [3] https://nlp.stanford.edu/projects/glove/
- [4] Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber. LSTM: A Search Space Odyssey. https://arxiv.org/abs/1503.04069
- [5] http://web.stanford.edu/class/cs224n/lectures/cs224n-2017-lecture8.pdf
- [6] http://www.deeplearningbook.org/contents/applications.html (12.4.3)
- [7] Quoc V. Le, Navdeep Jaitly, Geoffrey E. Hinton. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. 2015. arxiv.org/abs/1504.00941v2
- [8] "Layer Normalization" Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton. https://arxiv.org/abs/1607.06450.
- [9] "Recurrent Dropout without Memory Loss" Stanislau Semeniuta, Aliaksei Severyn, Erhardt Barth. https://arxiv.org/abs/1603.05118
- [10] https://www.tensorflow.org/api_docs/python/tf/contrib/rnn/LayerNormBasicLSTMCell
- [11] https://r2rt.com/non-zero-initial-states-for-recurrent-neural-networks.html
- [12] LSTM: A Search Space Odyssey. Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, Jürgen Schmidhuber. https://arxiv.org/abs/1503.04069
- [13] https://github.com/karpathy/char-rnn/issues/138#issuecomment-162763435
- https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html
- https://danijar.com/tips-for-training-recurrent-neural-networks/

- [14] https://github.com/kjw0612/awesome-rnn#image-captioning
- [15] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan, Show and Tell: A Neural Image Caption Generator, arXiv:1411.4555 / CVPR 2015
- [16] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. BLEU: A method for automatic evaluation of machine translation. In ACL, 2002.
- [17] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan L. Yuille, Deep Captioning with Multimodal Recurrent Neural Networks (m-RNN), arXiv:1412.6632 / ICLR 2015
- [18] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell, Long-term Recurrent Convolutional Networks for Visual Recognition and Description, arXiv:1411.4389 / CVPR 2015
- [19] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le, Sequence to Sequence Learning with Neural Networks, arXiv:1409.3215 / NIPS 2014
- [20] Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio, Neural Machine Translation by Jointly Learning to Align and Translate, arXiv:1409.0473 / ICLR 2015
- [21] Karl M. Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom, Teaching Machines to Read and Comprehend, arXiv:1506.03340 / NIPS 2015
- [22] SQuAD: 100,000+ Questions for Machine Comprehension of Text. Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang. https://arxiv.org/abs/1606.05250
- [23] Ankit Kumar, Ozan Irsoy, Jonathan Su, James Bradbury, Robert English, Brian Pierce, Peter Ondruska, Mohit Iyyer, Ishaan Gulrajani, and Richard Socher, Ask Me Anything: Dynamic Memory Networks for Natural Language Processing, arXiv:1506.07285
- [24] Question Answering Using Deep Learning. https://cs224d.stanford.edu/reports/StrohMathur.pdf
- [25] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. arXiv preprint arXiv:1608.07905, 2016
- https://machinelearningmastery.com/how-does-attention-work-in-encoder-decoder-recurrent-neural-networks
- https://sites.google.com/site/deeplearningdialogue/references
- https://blog.acolyer.org/2016/04/21/the-amazing-power-of-word-vectors/