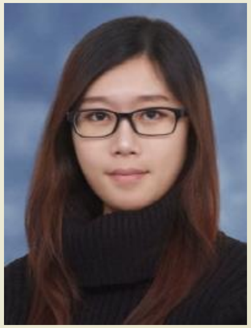
Recurrent Neural Network

Il-Chul Moon, Wonsung Lee, Sungrae Park, Su-Jin Shin, Kyungwoo Song, Weonyoung Joo, JunKeon Park, YoonYeong Kim, Joonho Jang

Dept. of Industrial and Systems Engineering
KAIST
icmoon@kaist.ac.kr

Wonsung Lee

Sungrae Park


Su-Jin Shin

Kyungwoo Song

Weonyoung Joo

JunKeon Park

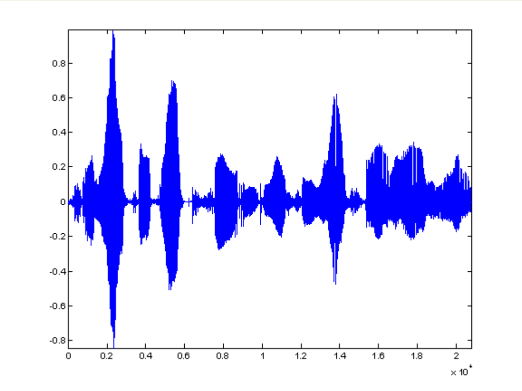
YoonYeong Kim

Joonho Jang

ACKNOWLEDGEMENTS

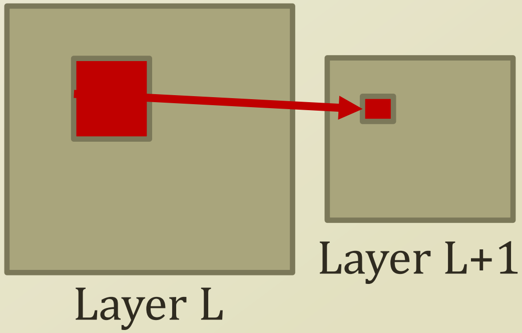
Neural Networks for Various Domains

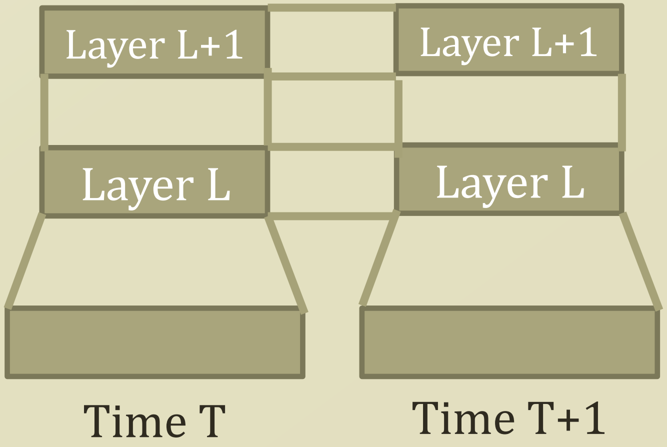
- Previous structure : Fully connected networks
- Recent advances in neural networks
  - Computer vision
  - Language models
- Application domain influences network structures
  - Convolutional structure utilizing localized connections
  - Recurrent structure utilizing chain connections



Speech Recognition

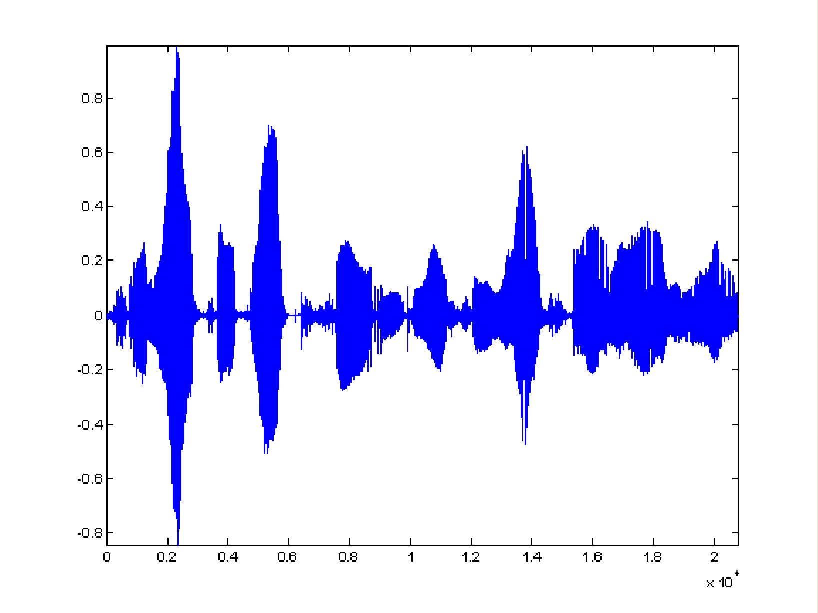
Image Classification
Object Recognition

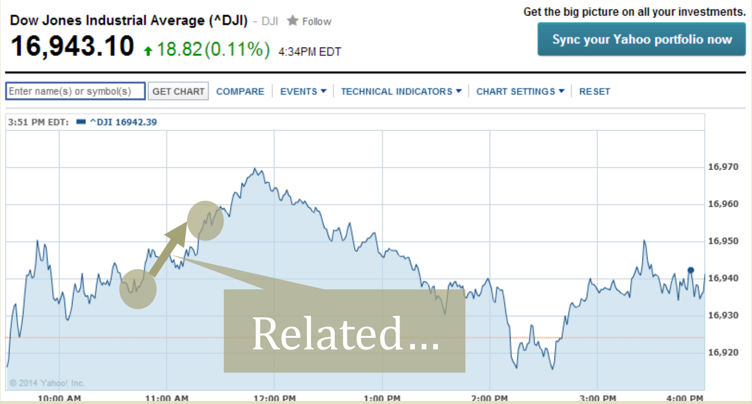




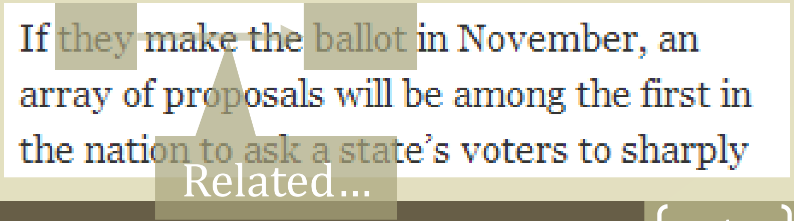
Example: Time Series Data

- Imagine the following case
  - Data points on the plane
  - Have a temporal trace of data points
  - Now, any broken assumption in the analysis?
- Any real world applications
  - Many, many, many…
  - Stock market analysis, text mining…





Related…



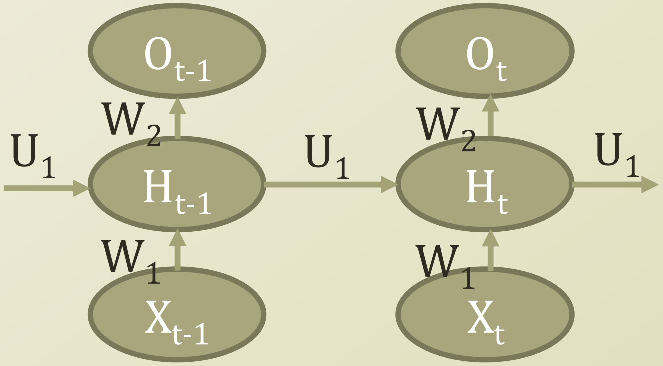
If they make the ballot in November, an array of proposals will be among the first in the nation to ask a state's voters to sharply

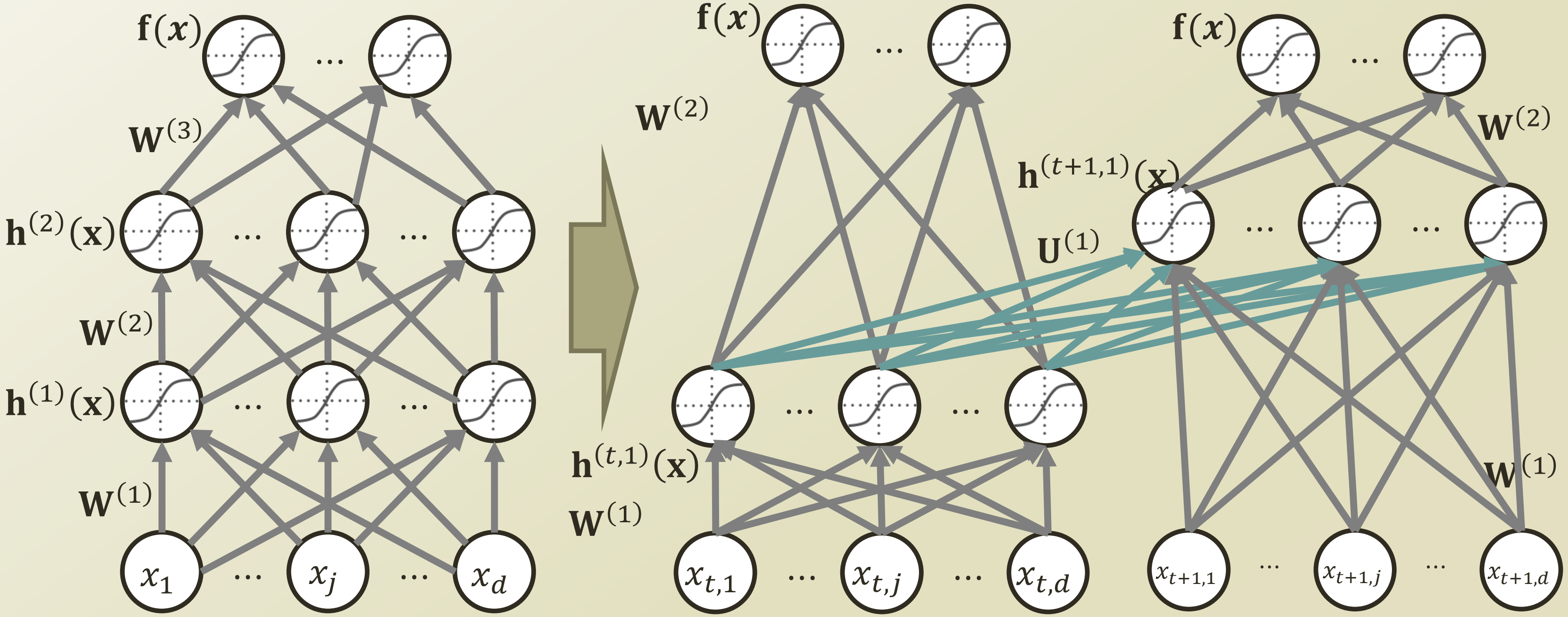
Related…

RECURRENT NEURAL NETWORK

Modeling Temporal Data with NN

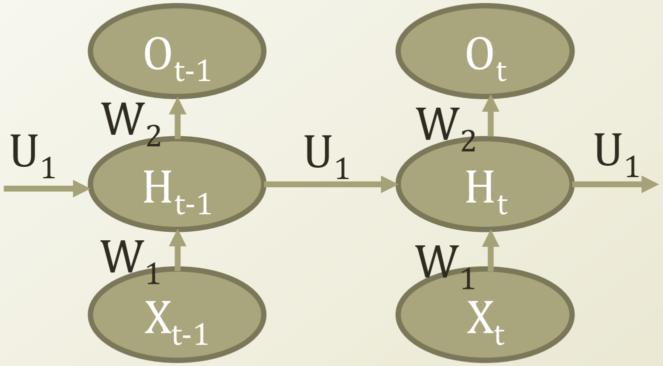
- Limitation of convolutional neural network
  - No temporal modeling in the input and the hidden information
  - Need to model the information flow from the previous hidden layer
- The definition of Recurrent Neural Network
  - $h_t = \sigma(Uh_{t-1} + Wx_t + b)$



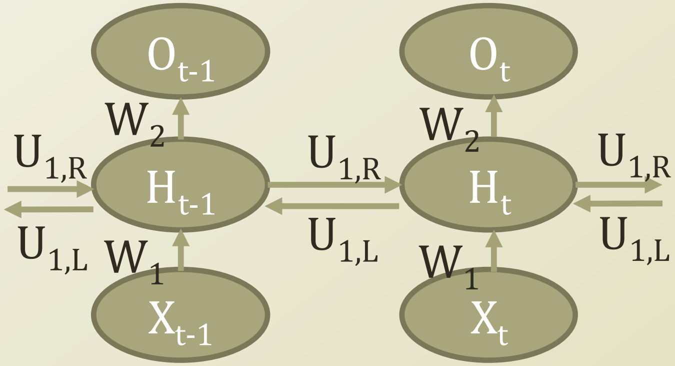


Variant of Recurrent Neural Network

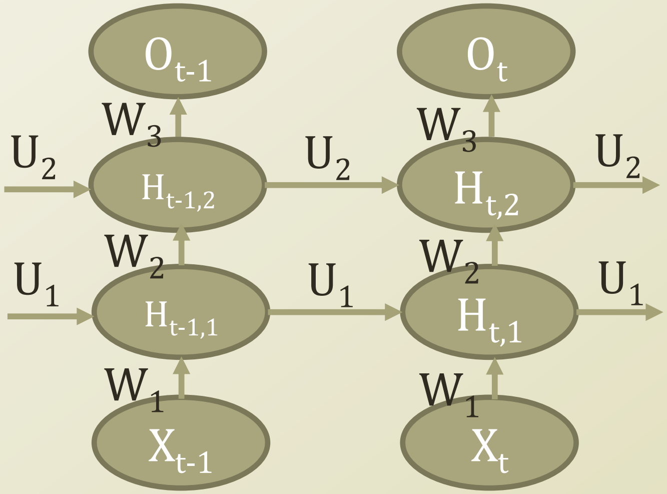
- Bidirectional RNN for influence from X_{t+1} to X_t
- Deep RNN for further complex modeling on X and H
- Sequence-to-Sequence RNN when generate an output after the whole sequence input



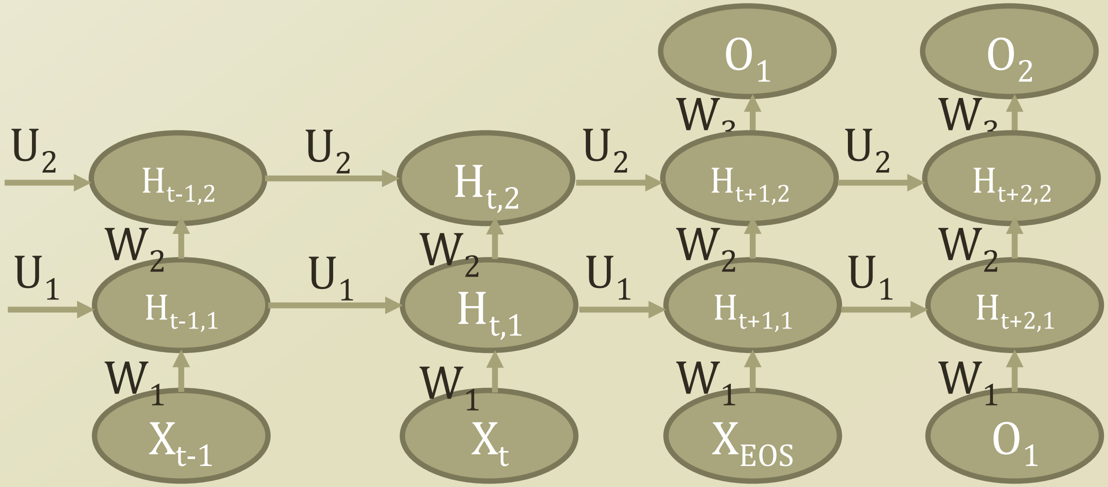
Recurrent Neural Network



Bidirectional Recurrent Neural Network



Deep Recurrent Neural Network



Deep Sequence-to-Sequence
Recurrent Neural Network

Backpropagation

- Backpropagation
  - Do
    - For training examples, x

      // forward pass of the neural net.

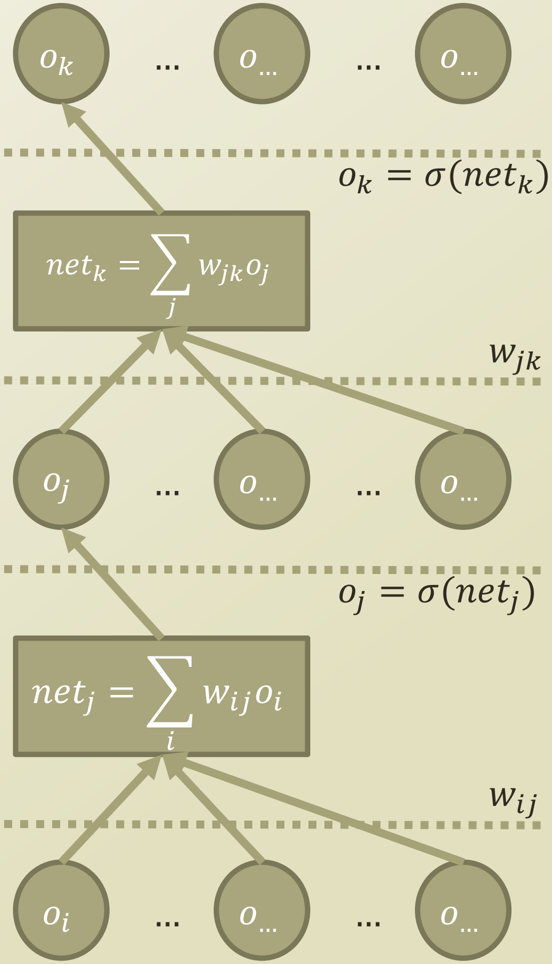
      - $o_k = f(x; w)$
      - $E = \frac{1}{2}(\sum_k (t_k - o_k)^2)$

      // backward pass of the neural net.

      - Calculate $\delta_k = (t_k - o_k) o_k (1 - o_k)$
      - For the backward-pass from the top to the bottom
        - Calculate $\delta_j = o_j (1 - o_j) \sum_k \delta_k w_{jk}$

      // weight update

      - Update w_{jk} with $w_{jk}^{t+1} \leftarrow w_{jk}^{t} + \eta \delta_k o_j$
      - For the backward-pass from the top to the bottom
        - Update $w_{ij}^{t+1} \leftarrow w_{ij}^{t} + \eta o_i \delta_j$
  - Until converges

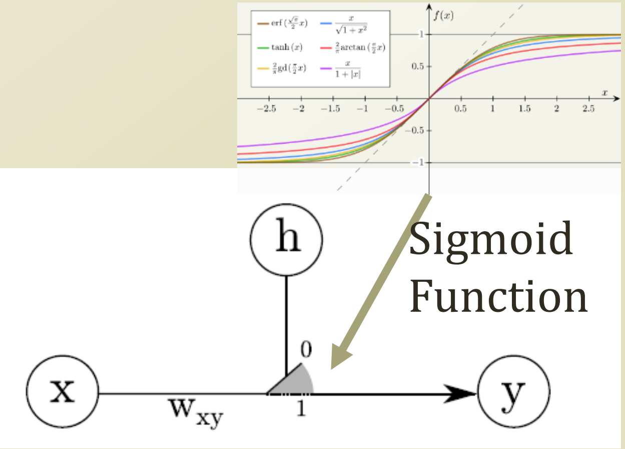

o_k … $o_{…}$ … $o_{…}$
$o_k = \sigma(net_k)$
$net_k = \sum_j w_{jk} o_j$
w_{jk}
o_j … $o_{…}$ … $o_{…}$
$o_j = \sigma(net_j)$
$net_j = \sum_i w_{ij} o_i$
w_{ij}
o_i … $o_{…}$ … $o_{…}$

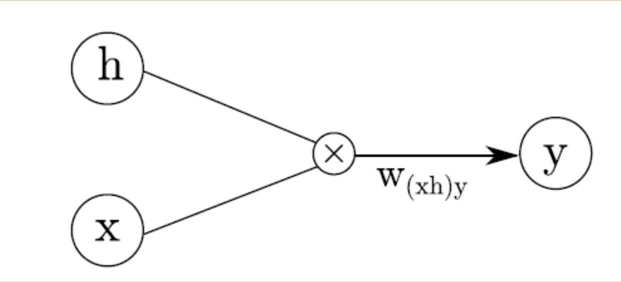
Vanishing Gradient Problem

- Logistic function : $f(x) = \frac{1}{1+e^{-x}}$
  - $\frac{d}{dx} f(x) = \frac{d}{dx}(1 + e^{-x})^{-1} = e^{-x}(1 + e^{-x})^{-2} = f(x)(1 - f(x))$
  - $0 < f(x) < 1$
  - $max \frac{d}{dx} f(x) = \frac{1}{4}$
- Multi-layered delta signal
  - $\delta_j = o_j(1 - o_j) \sum_k \delta_k w_{jk}$
- Back-propagation algorithm
  - $w_{ij}^{t+1} \leftarrow w_{ij}^{t} + \eta o_i \delta_j$
- If a delta signal is weak, the weight update is impossible
  - Deep layered neural network
  - Single layered recurrent neural network

Gating Mechanism in Deep Neural Network

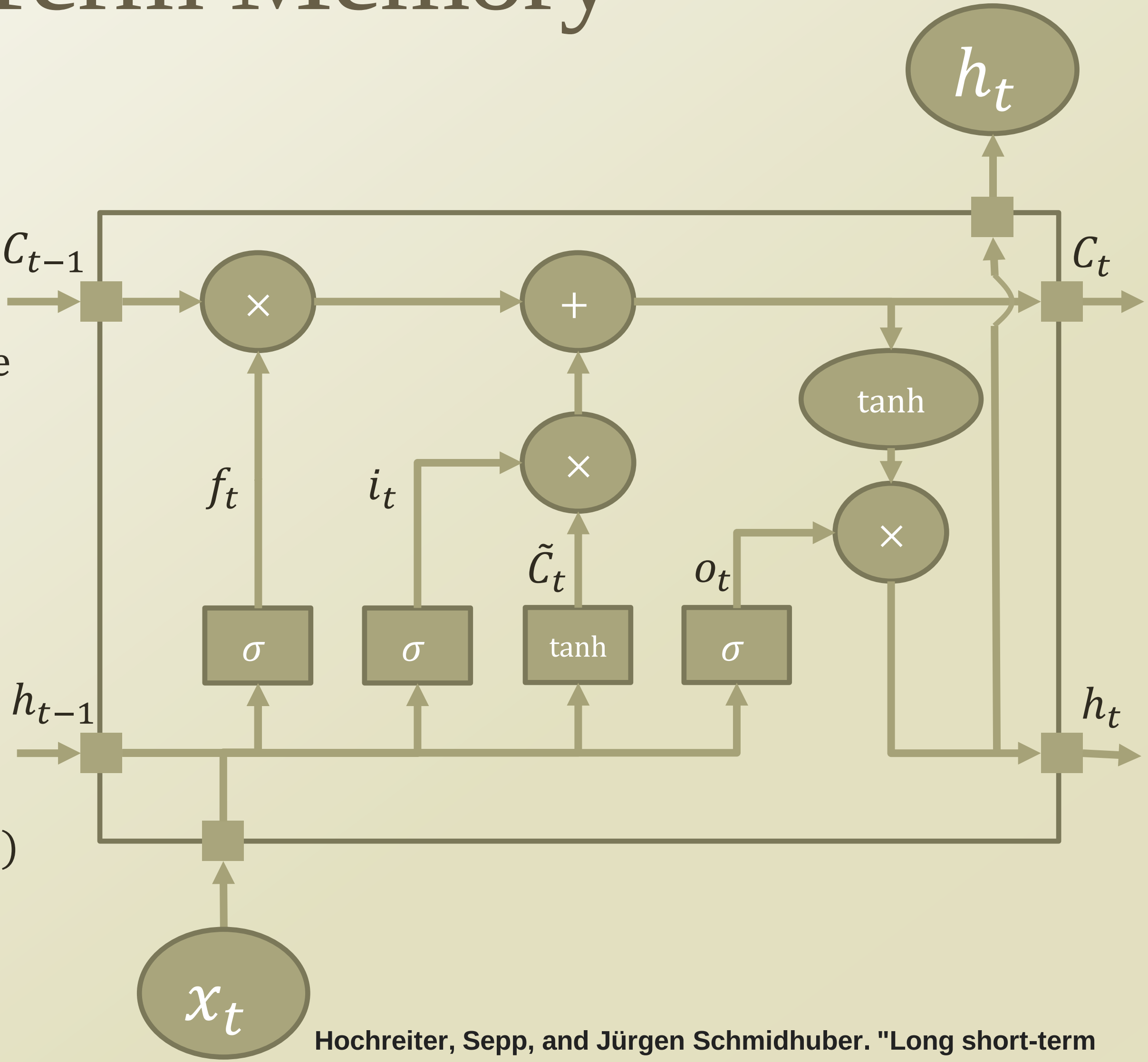
- Gating mechanism in neural network
  - Making a neural network have paths along which information can flow across several layers without attenuation
  - Inserting a modelers' assumption about information flows into deep architecture
- Gating mechanism
  - A neuron h is used as a switch that stops or not the flow of information between two other neurons x and y.
  - Switch gates (layer-level)
    - $\boldsymbol{y} = \begin{cases} \boldsymbol{x} & if\ h = 1 \\ \boldsymbol{0}, otherwise \end{cases}$ or $\boldsymbol{y} = h * \boldsymbol{x}$ (h is scalar)
  - Multiplicative gates (element-level)
    - $\boldsymbol{y} = \boldsymbol{x} \odot \boldsymbol{h}$ ($\boldsymbol{h}$ is same dimensional vector)


$\mathrm{erf}(\frac{\sqrt{\pi}}{2}x)$
$\frac{x}{\sqrt{1+x^2}}$
$\tanh(x)$
$\frac{2}{\pi}\arctan(\frac{\pi}{2}x)$
$\frac{2}{\pi}\mathrm{gd}(\frac{\pi}{2}x)$
$\frac{x}{1+|x|}$
$f(x)$
x
1
0.5
−0.5
−1
−2.5
−2
−1.5
−1
−0.5
0.5
1
1.5
2
2.5
h
Sigmoid Function
0
1
x
w_{xy}
y



Long Short Term Memory

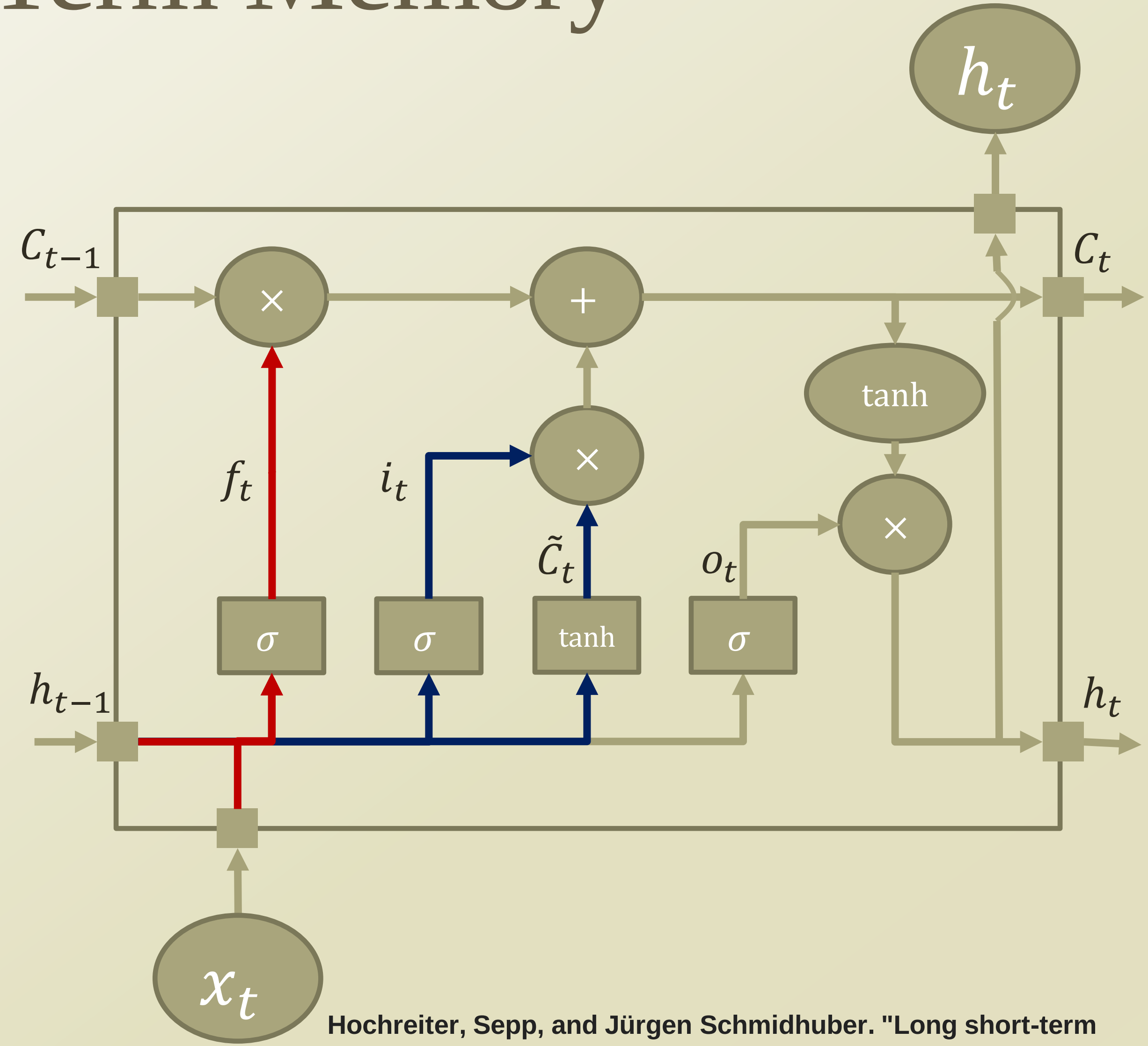
- LSTM cell
  - Introducing a state to the LSTM cell
  - Gate mechanism to add or remove information from the cell state
- Gate mechanism
  - Sigmoid activation
  - Pointwise multiplication
- Mathematical formulation
  - $f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right)$
  - $i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$
  - $\tilde{C}_t = tanh(W_C[h_{t-1}, x_t] + b_C)$
  - $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$
  - $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$
  - $h_t = o_t * \tanh(C_t)$



Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

Long Short Term Memory

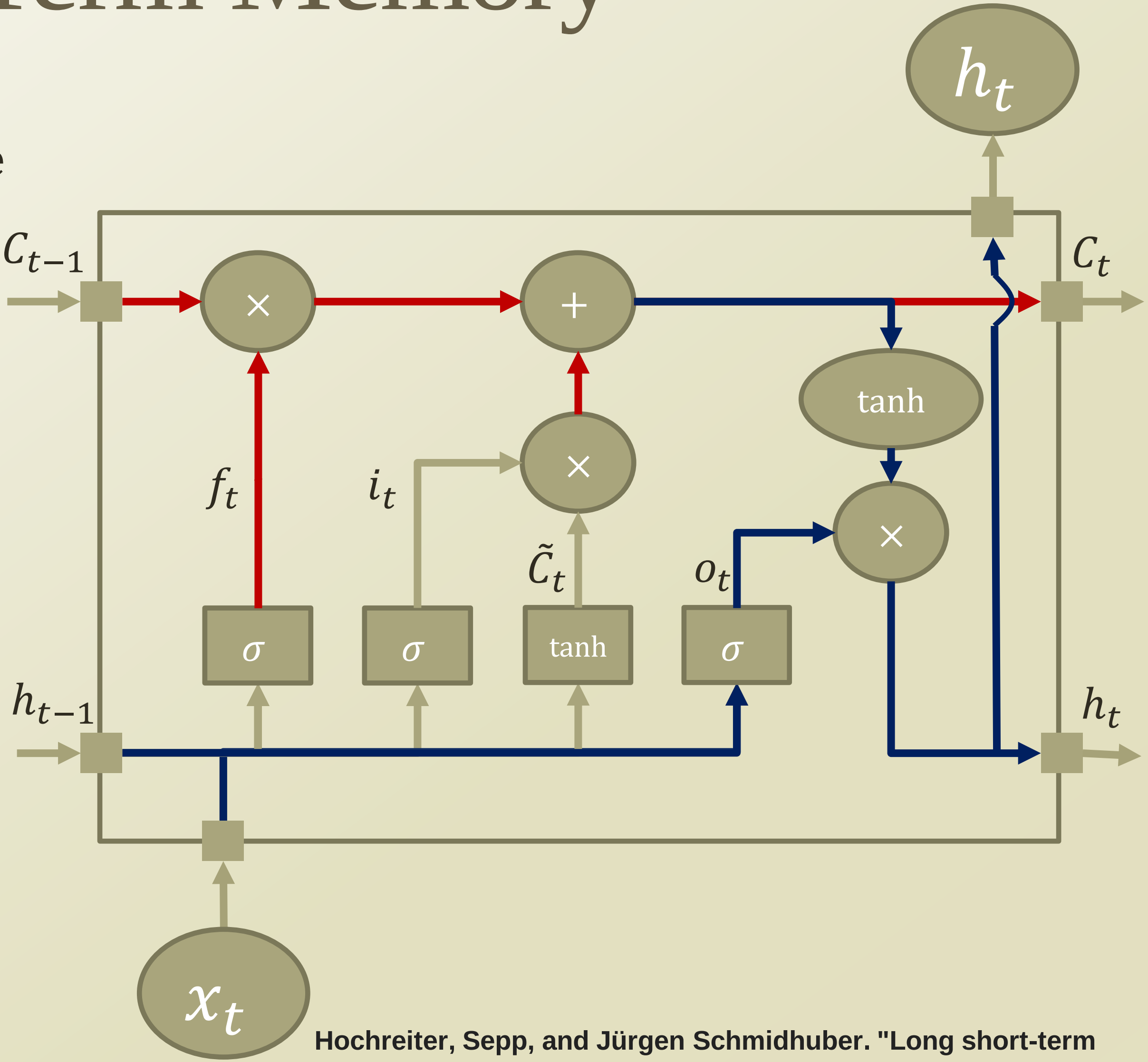
- Forgetting process
  - f_t $= \sigma(W_f[h_{t-1}, x_t] + b_f)$
    - f_t : Forget gate layer
    - Forgetting from the state of the last cell, C_{t-1}
    - Number of outputs equal to the cell state dimension
- Remembering process
  - i_t $= \sigma(W_i[h_{t-1}, x_t] + b_i)$
    - i_t : Remember gate layer
    - Output dim.==Cell state dim.
  - $\tilde{C}_t$ $= tanh(W_C[h_{t-1}, x_t] + b_C)$
    - $\tilde{C}_t$: Information to be remembered
    - Output dim.==Cell state dim.



Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

Long Short Term Memory

- Application to the cell state
  - $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$
    - Forgetting process
    - Remembering process
- Output process
  - $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$
    - Output depends upon the state
    - Should be filtered by the past output and the current input
  - $h_t = o_t * \tanh(C_t)$
    - Squash the cell state to fit the range between [-1,1]

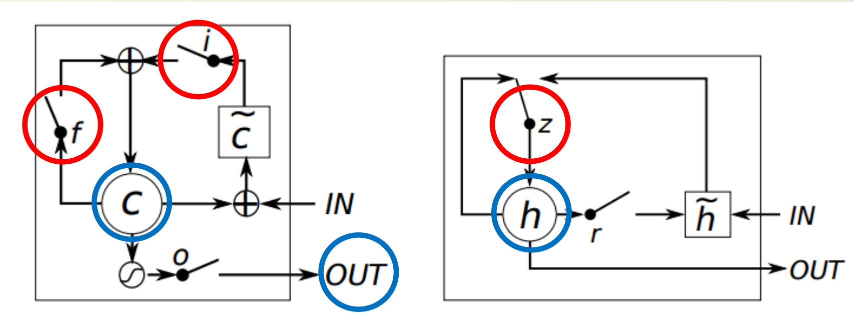


Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

Pros and Cons of LSTM

- LSTM enables
  - The long range information delivery through the cell state
  - Cell state does not require a layer propagation
- Problem of LSTM
  - Too many parameters to learn
    - W_f, W_i, W_c, W_o
    - Many new hazards to make the state propagation work
- Variants of LSTM
  - Gated recurrent unit
    - $z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$
    - $r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$
    - $h_t = z_t * h_{t-1} + (1 - z_t) * \tanh(W_h x_t + U_h(r_t * h_{t-1}) + b_h)$
    - No separate storage of information through the cell state
    - No concatenation of input and hidden information

Gated Recurrent Unit

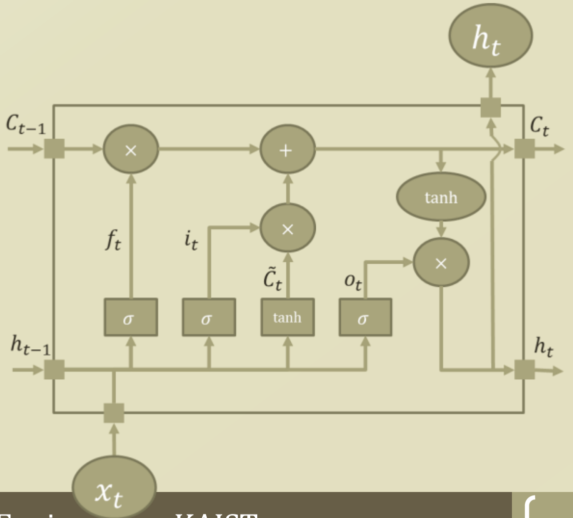


(a) Long Short-Term Memory (b) Gated Recurrent Unit

- Alternative of LSTM
- GRU model
  - Output : $h_t^j = (1 - z_t^j)h_{t-1}^j + z_t^j \tilde{h}_t^j$
    - z_t^j : update gate computed by $z_t^j = \sigma(W_z \mathrm{x}_t + U_z \mathrm{h}_{t-1})\ j$

Multiplicative gates

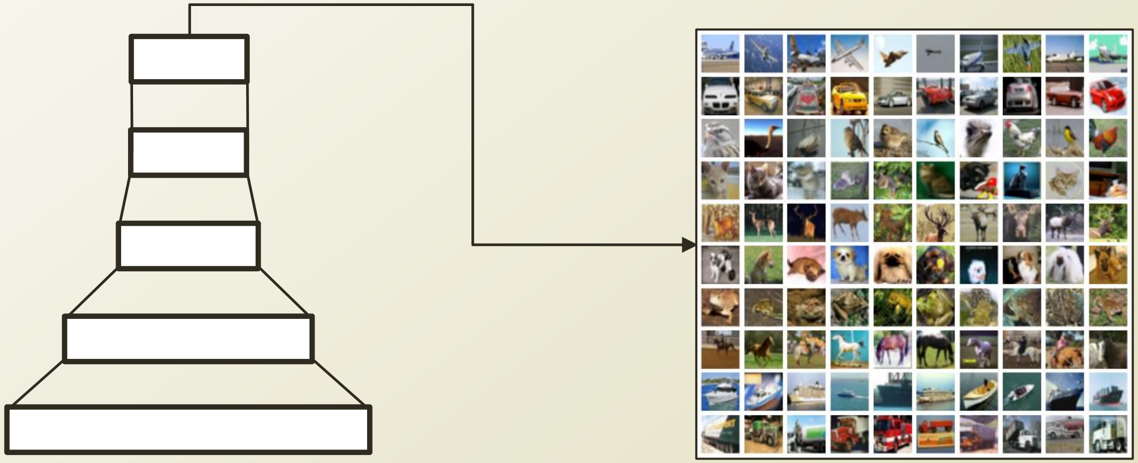
  - candidate cell : $\tilde{h}_t^j = \tanh\left(W\mathrm{x}_t + U(\boldsymbol{r}_t \odot \mathbf{h}_{t-1})\right)^j$
    - r_t^j : reset gates computed by $r_t^j = \sigma(W_r \mathrm{x}_t + U_r \mathrm{h}_{t-1})^j$ **Multiplicative gates**
    - $\odot$: an element-wise multiplication
- Difference from LSTM
  - Existence of the memory content C
  - The location of the gates
    - LSTM : independent gates f and I
    - GRU : using gate z



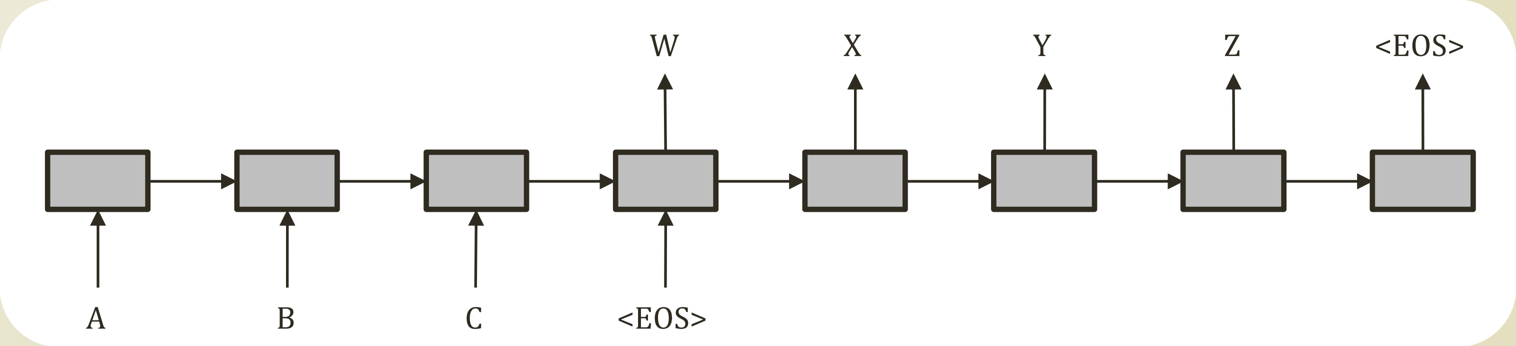
TEXT ANALYSIS TRENDS

Deep Neural Networks for Seq. Mapping

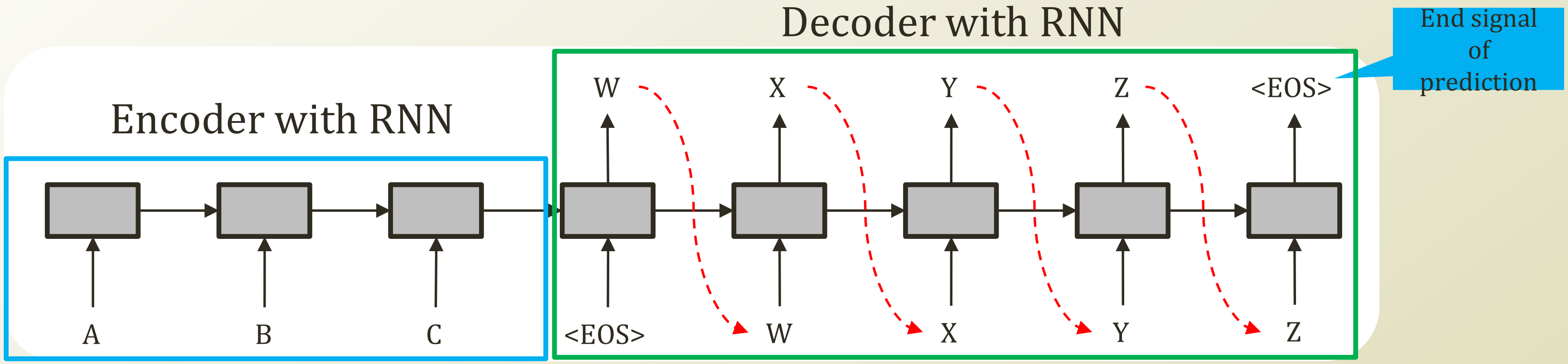
- The Limitation of conventional deep neural networks (DNNs)



  - It can be used for image classification: image recognition by CNN
  - However, they cannot be used to map sequences to sequences (with variable length)
    - Machine translation and speech recognition: it usually requires RNN
    - Example problem that maps "ABC" to "WXYZ"



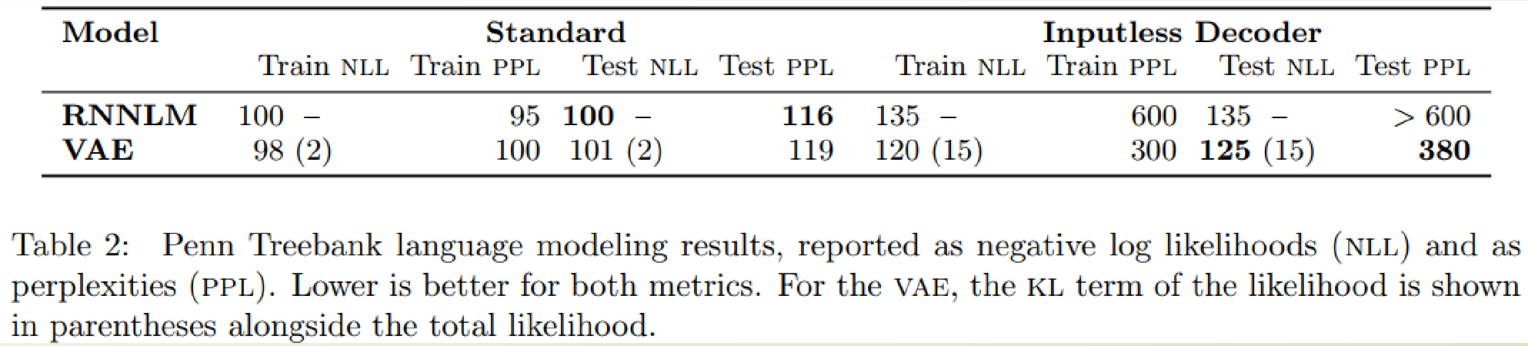
Structural Properties of Seq2Seq



- Prev. DNNs requires the dimensionality of the inputs and outputs is known and fixed.
- Seq2Seq architectures overcome the limitation.
  - Encoder with RNN (usually LSTM)
    - Input seq. vector with variable length → fixed dim. vector (bottleneck)
    - Reversing the order of input seq. is useful for long sentences (by introducing many short-term dependencies)
  - Decoder with RNN (usually LSTM)
    - Fixed dim. Vector with variable length → Target seq. vector
    - Conditioned on a previously predicted token or a ground-truth token

Experimental Results of Seq2Seq

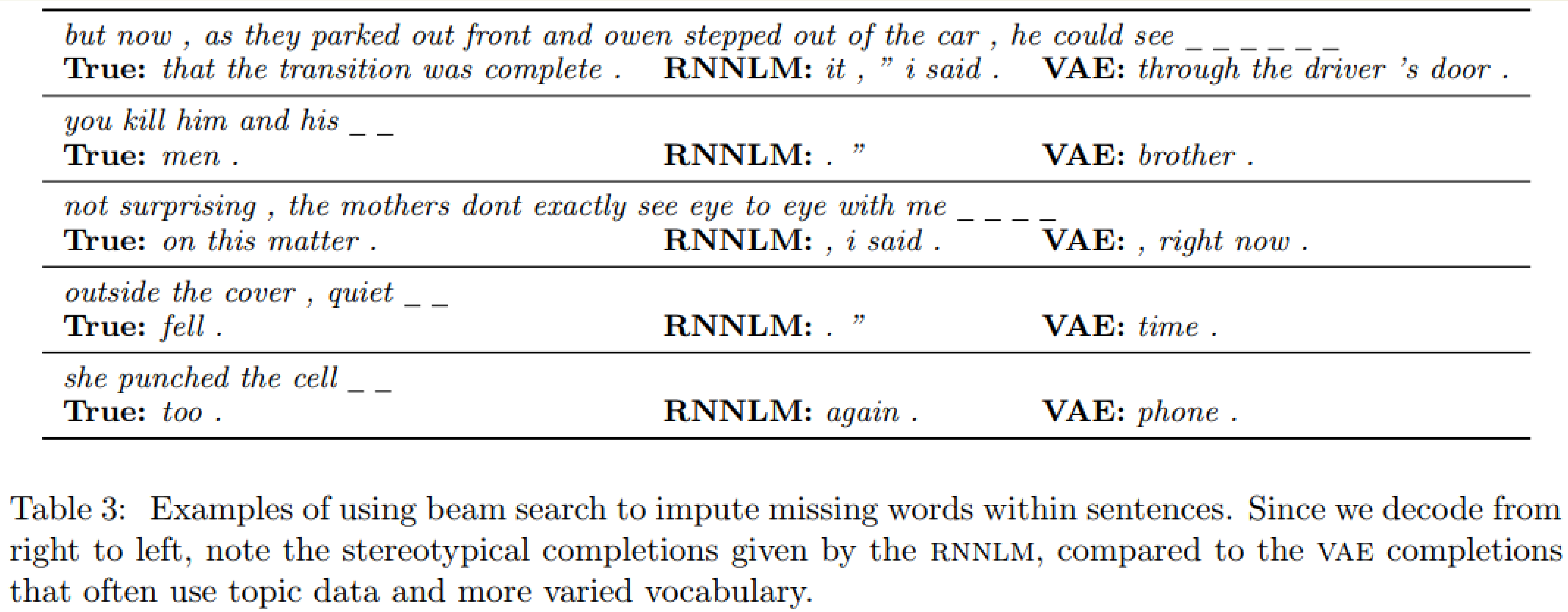
- Language modeling performance of VAE with LSTMs
  - It performs roughly the same, or actually a little bit worse than RNN-LM.



Model	**Standard**				**Inputless Decoder**			
	Train NLL	Train PPL	Test NLL	Test PPL	Train NLL	Train PPL	Test NLL	Test PPL
RNNLM	100 –	95	**100** –	**116**	135 –	600	135 –	> 600
VAE	98 (2)	100	101 (2)	119	120 (15)	300	**125** (15)	**380**

Table 2: Penn Treebank language modeling results, reported as negative log likelihoods (NLL) and as perplexities (PPL). Lower is better for both metrics. For the VAE, the KL term of the likelihood is shown in parentheses alongside the total likelihood.

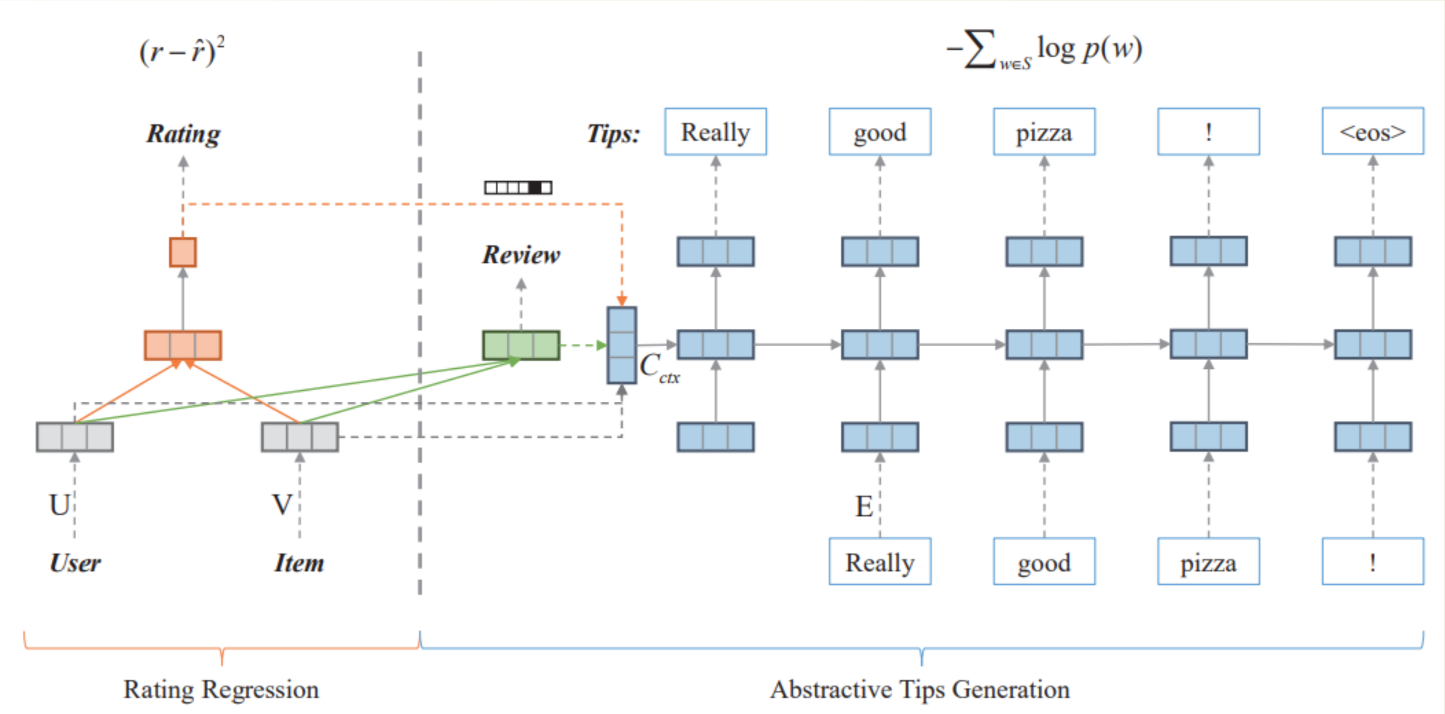
- Imputing missing words



but now , as they parked out front and owen stepped out of the car , he could see _ _ _ _ _ _ _		
True: *that the transition was complete .*	**RNNLM:** *it , " i said .*	**VAE:** *through the driver 's door .*
you kill him and his _ _		
True: *men .*	**RNNLM:** *. "*	**VAE:** *brother .*
not surprising , the mothers dont exactly see eye to eye with me _ _ _ _		
True: *on this matter .*	**RNNLM:** *, i said .*	**VAE:** *, right now .*
outside the cover , quiet _ _		
True: *fell .*	**RNNLM:** *. "*	**VAE:** *time .*
she punched the cell _ _		
True: *too .*	**RNNLM:** *again .*	**VAE:** *phone .*

Table 3: Examples of using beam search to impute missing words within sentences. Since we decode from right to left, note the stereotypical completions given by the RNNLM, compared to the VAE completions that often use topic data and more varied vocabulary.

Recommendation with Automatically Generated Tips



- The key contributions
  - Deep learning based framework named NRT (Neural Rating and Tips generation) which can simultaneously predict ratings and generate abstractive tips given in the form of a natural language sentence.
  - The first attempt utilizing tips information to improve the recommendation quality. (rather than item's reviews or specifications)
  - SOTA performance on both tasks of rating prediction and abstractive tips generation.

Results of Generated Recommendation Tips

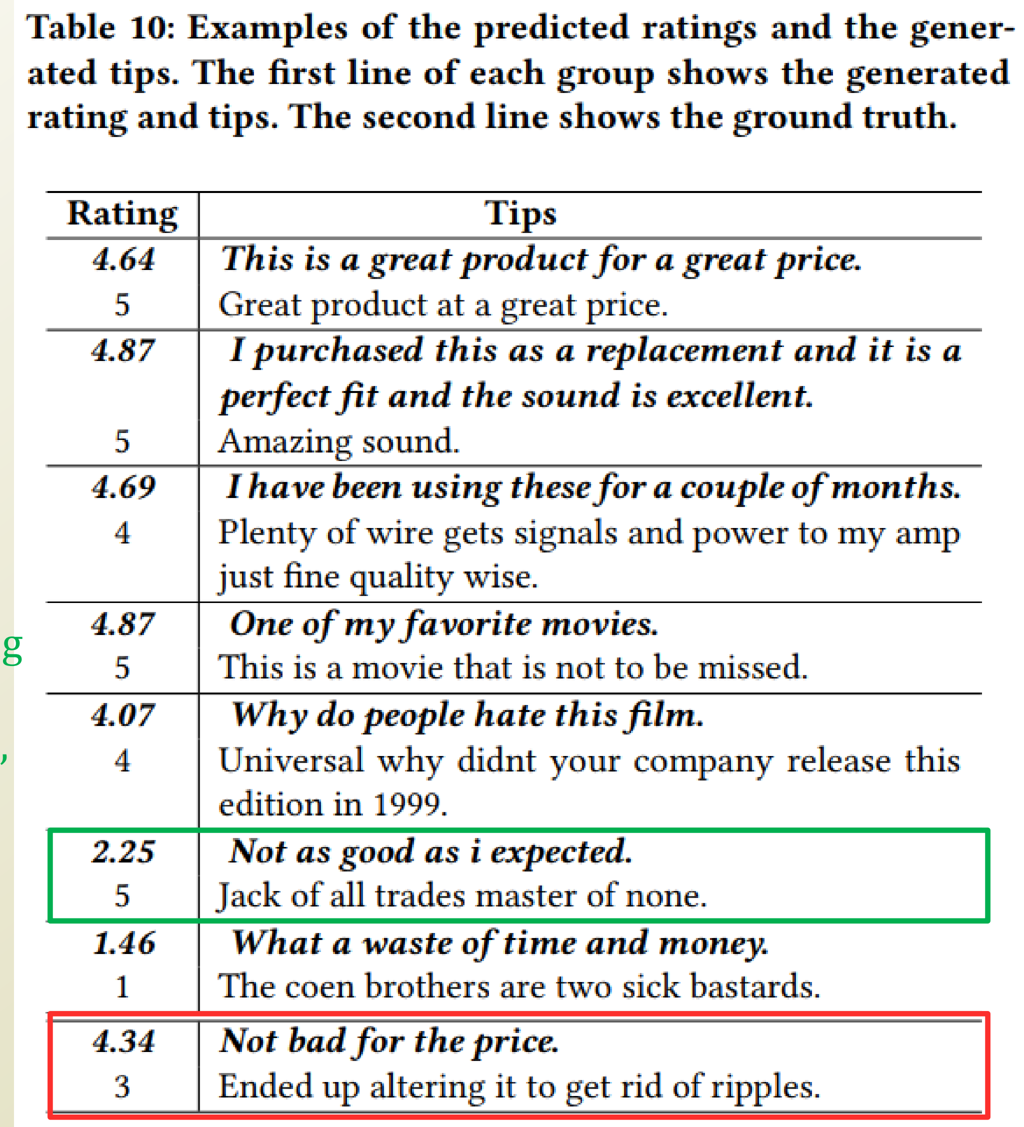


Table 10: Examples of the predicted ratings and the generated tips. The first line of each group shows the generated rating and tips. The second line shows the ground truth.

Rating	Tips
4.64	***This is a great product for a great price.***
5	Great product at a great price.
4.87	***I purchased this as a replacement and it is a perfect fit and the sound is excellent.***
5	Amazing sound.
4.69	***I have been using these for a couple of months.***
4	Plenty of wire gets signals and power to my amp just fine quality wise.
4.87	***One of my favorite movies.***
5	This is a movie that is not to be missed.
4.07	***Why do people hate this film.***
4	Universal why didnt your company release this edition in 1999.
2.25	***Not as good as i expected.***
5	Jack of all trades master of none.
1.46	***What a waste of time and money.***
1	The coen brothers are two sick bastards.
4.34	***Not bad for the price.***
3	Ended up altering it to get rid of ripples.

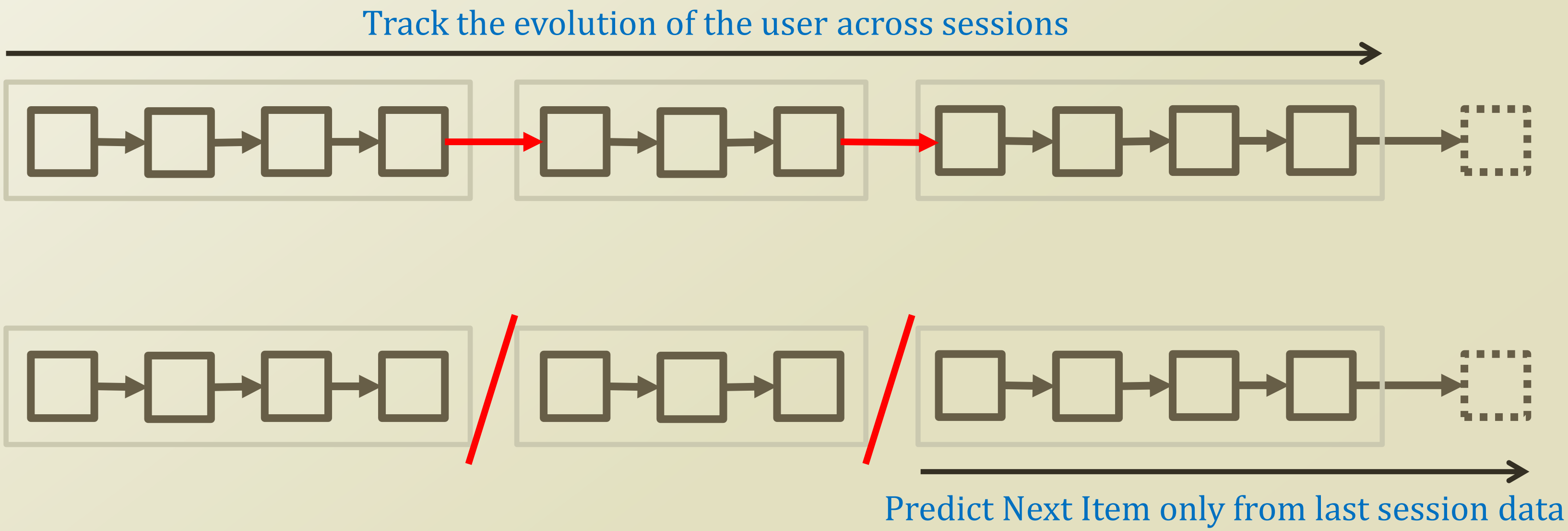
The ground-truth rating 5 might be clicked by a fat finger. Nevertheless, NRT generates a consistent sentiment between this case's predicted rating and tips

The predicted rating 4.34 is positive, but the sentiment of the generated tips is neutral, consistent with the ground-truth.

SESSION-BASED RECOMMENDATION SYSTEMS

Session-Based Recommendations

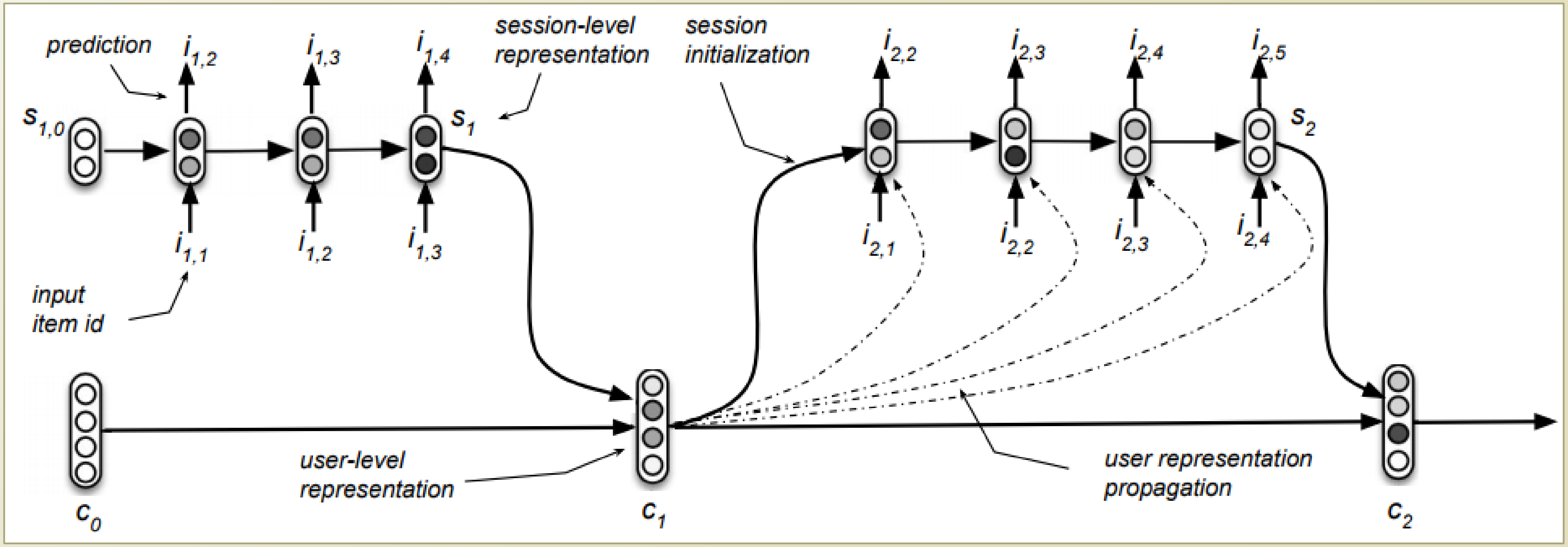
- Motivation
  - Session-***aware*** recommender system
    - There are also domains in which user profiles are readily **available**.
    - In these cases it is reasonable to assume that the user behavior in past sessions might provide valuable information for providing recommendations in the next session.
      - *When user identifiers propagate information from the previous user session to the next, thus improving the recommendation accuracy.*


Track the evolution of the user across sessions
Predict Next Item only from last session data

Quadrana, Massimo, et al. "Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks." *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017.

RNN Based Sample Structure

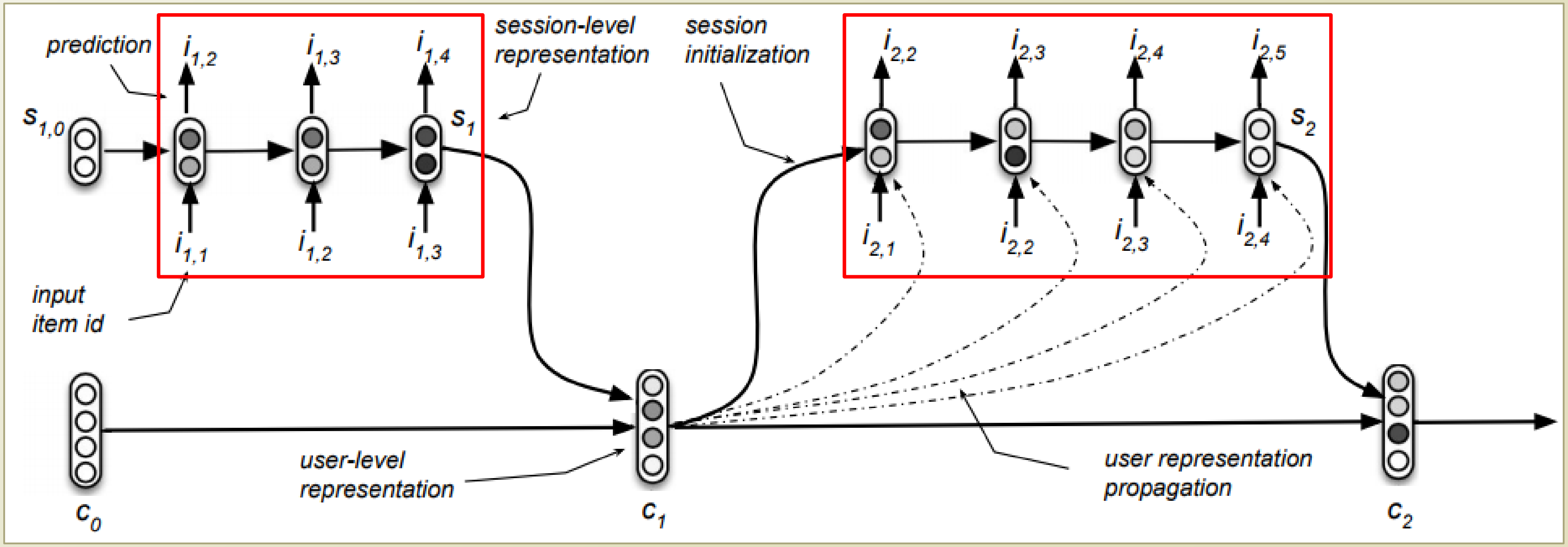
- Architecture
  - Core Network : Recurrent Neural Network (Especially GRU)
    - Input : The current item ID in the session.
    - Output : A score(probability) for each item representing the likelihood of being the next item in the session.



Quadrana, Massimo, et al. "Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks." *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017.

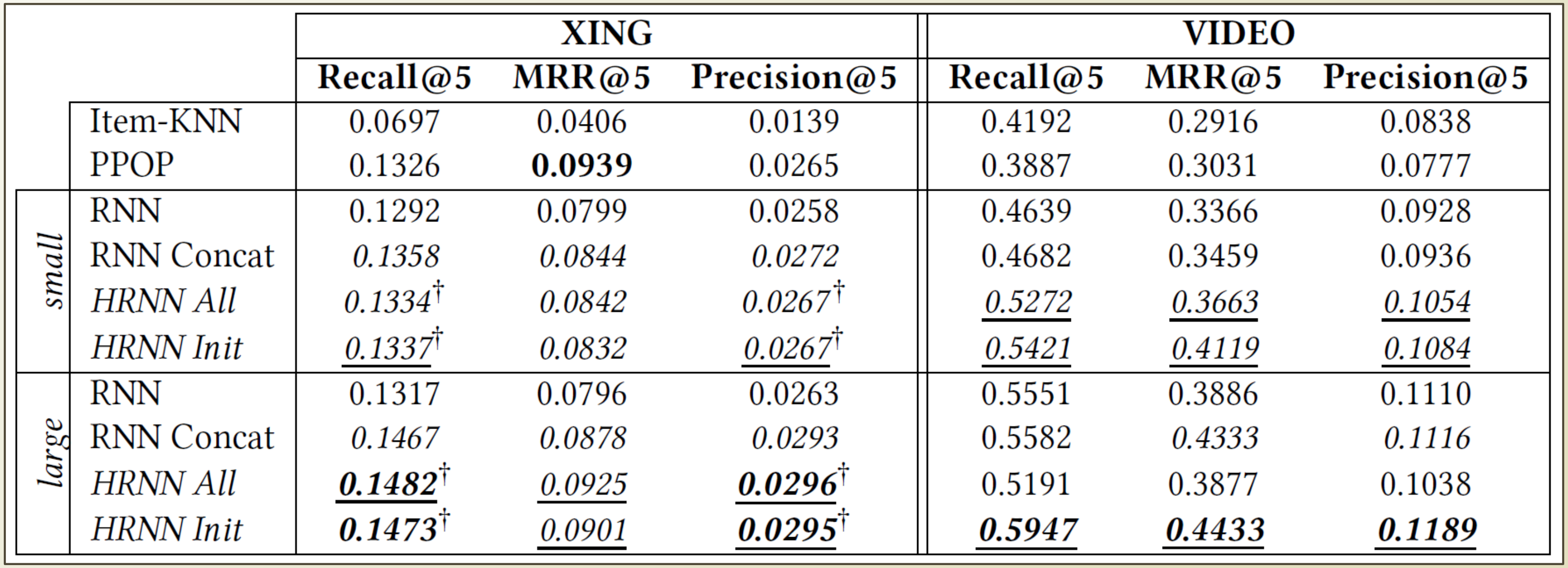
RNN Based Sample Structure

- Architecture
  - The session-level GRU(GRU_{ses}) with adding an ***additional GRU*** layer to model infor-mation across user sessions and to track the evolution of the ***user*** interests over time.
  - HRNN model adds one $user - level\ GRU(GRU_{urs})$ to model the user activity across sessions.



Quadrana, Massimo, et al. "Personalizing Session-based Recommendations with Hierarchical Recurrent Neural Networks." *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017.

Experimental Results



		XING			VIDEO		
		Recall@5	**MRR@5**	**Precision@5**	**Recall@5**	**MRR@5**	**Precision@5**
	Item-KNN	0.0697	0.0406	0.0139	0.4192	0.2916	0.0838
	PPOP	0.1326	**0.0939**	0.0265	0.3887	0.3031	0.0777
small	RNN	0.1292	0.0799	0.0258	0.4639	0.3366	0.0928
	RNN Concat	*0.1358*	*0.0844*	*0.0272*	0.4682	0.3459	0.0936
	HRNN All	*0.1334*†	*0.0842*	*0.0267*†	*0.5272*	*0.3663*	*0.1054*
	HRNN Init	*0.1337*†	*0.0832*	*0.0267*†	*0.5421*	*0.4119*	*0.1084*
large	RNN	0.1317	0.0796	0.0263	0.5551	0.3886	0.1110
	RNN Concat	*0.1467*	*0.0878*	*0.0293*	0.5582	*0.4333*	*0.1116*
	HRNN All	***0.1482***†	*0.0925*	***0.0296***†	0.5191	0.3877	0.1038
	HRNN Init	***0.1473***†	*0.0901*	***0.0295***†	***0.5947***	***0.4433***	***0.1189***

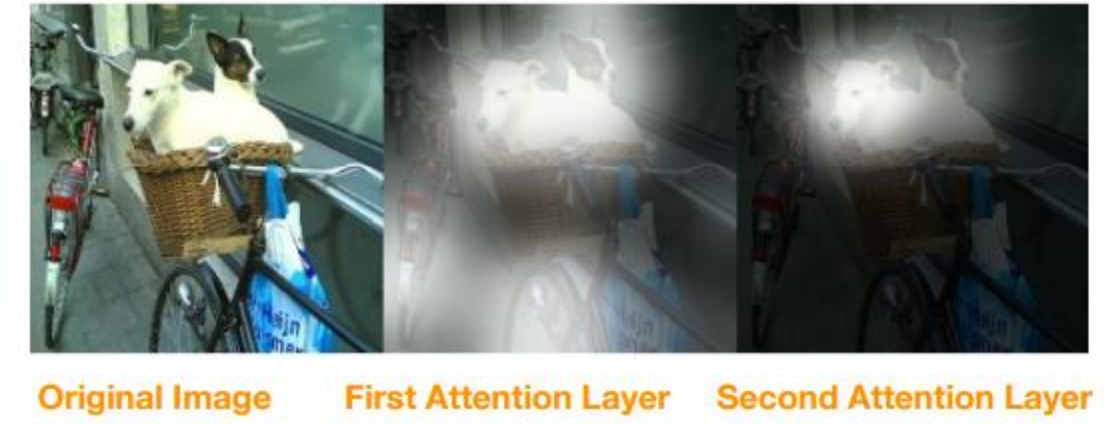
- HRNN Init has significantly better performance than all baselines.
- HRNN Init outperforms HRNN All in many cases.
  - The user activity within a session can be totally disconnected from her more recent sessions, and even from her general interests
  - HRNN Init models the user taste dynamics and lets the session-level GRU free to exploit them according to the actual evolution of the user interests within session.
  - Its greater flexibility leads to superior recommendation quality.

QUESTION AND ANSWERING CHALLENGES

Visual Question and Answering

What are sitting in the basket on a bicycle? ⇒ Dog

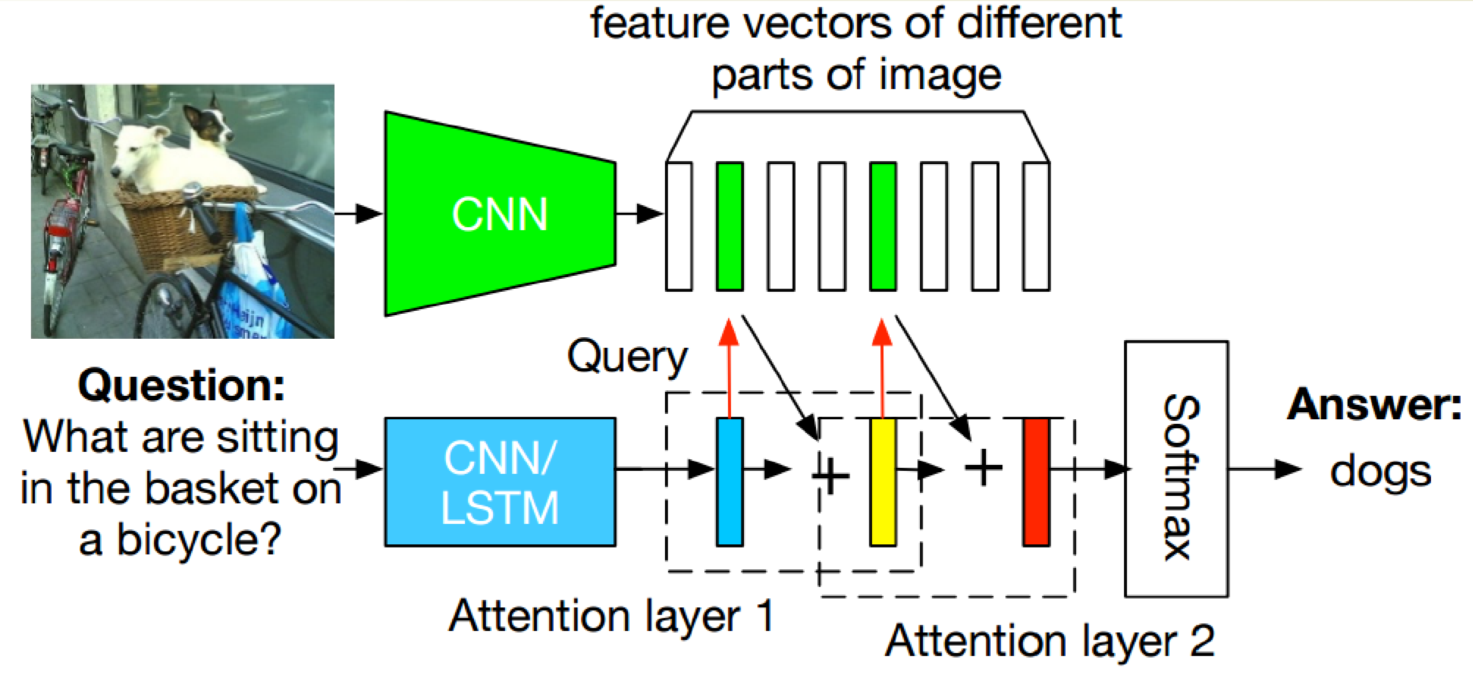
(a) Stacked Attention Network for Image QA



- In many cases, an answer only related to a small region of an image
- In the above image, there are many objects; bicycles, baskets, window, street and dogs.
- The answer to the question only relates to 'dogs'
- ⇒ Using the one global image feature vector to predict the answer could lead to suboptimal results due to the noises
  - There are many objective which is irrelevant to the potential answer

- Gradually Filter out noises and pinpoint the regions that are highly relevant to the answer
  - Original Image
  - ⇒ Dog, Bicycle, Basket
  - ⇒ Dog
- **Attention**

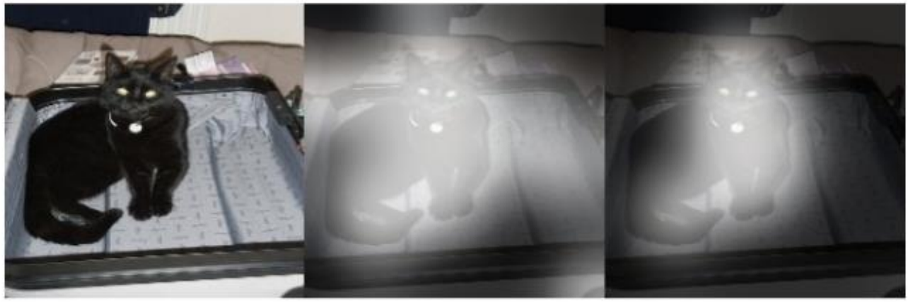
Visual Q/A Model Structures


feature vectors of different parts of image
CNN
Query
Question:
What are sitting in the basket on a bicycle?
CNN/ LSTM
Softmax
Answer:
dogs
Attention layer 1
Attention layer 2

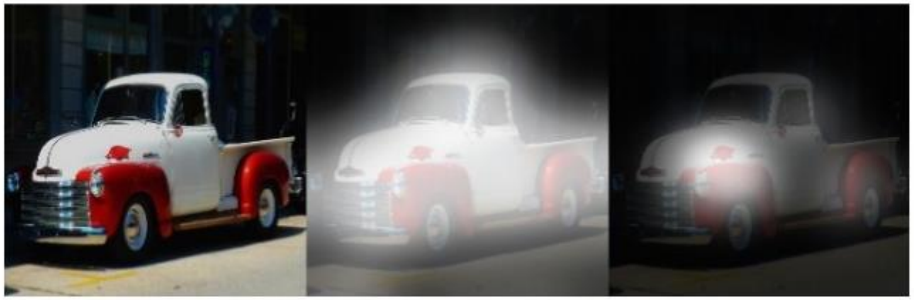
- Image Part
  - CNN (VGG-net)
- Text Part
  - CNN or LSTM
- Attention Part
  - Two Attention Layer
  - To capture important features of the image that corresponds to the question

Q/A Attention on Images

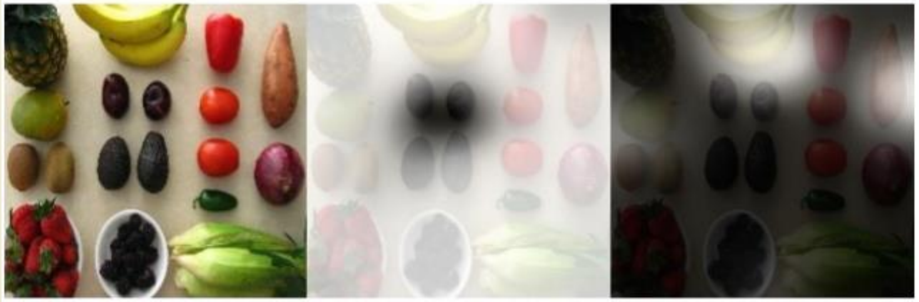
What is sitting in the luggage bag?
Answer: cat Prediction: cat



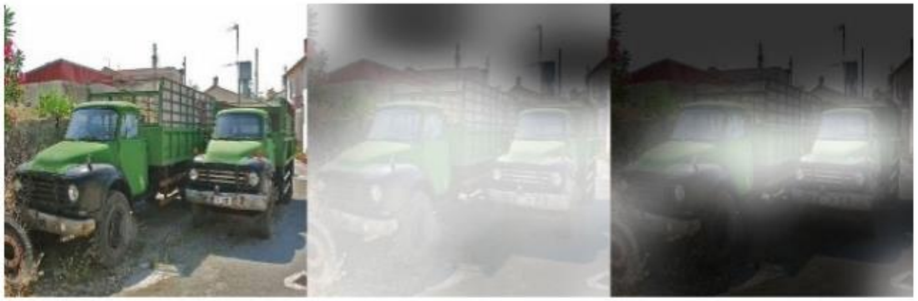
What is the color of the design?
Answer: red Prediction: red



What is the color of the surface?
Answer: white Prediction: white



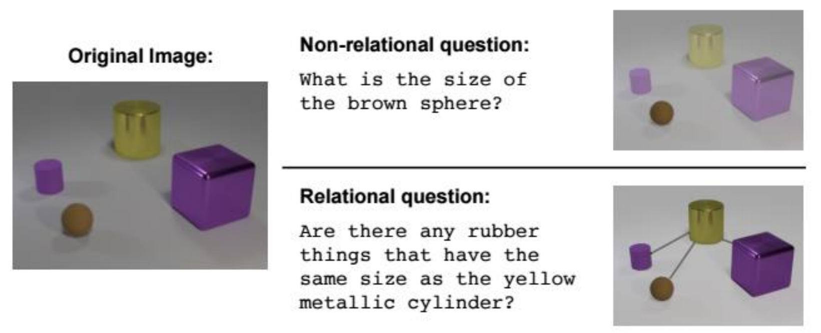
What is the color of the trucks?
Answer: green Prediction: green



Relative Questions on Visual Q/A

- Relational Reasoning
  - Solve relational question.
  - Discover and learn to reason about entities and their relations.

<u>Image</u>



Original Image:

Non-relational question:

What is the size of the brown sphere?

Relational question:

Are there any rubber things that have the same size as the yellow metallic cylinder?

Visual QA 에서, 특히 Object 들간의 relation을 묻는 Question

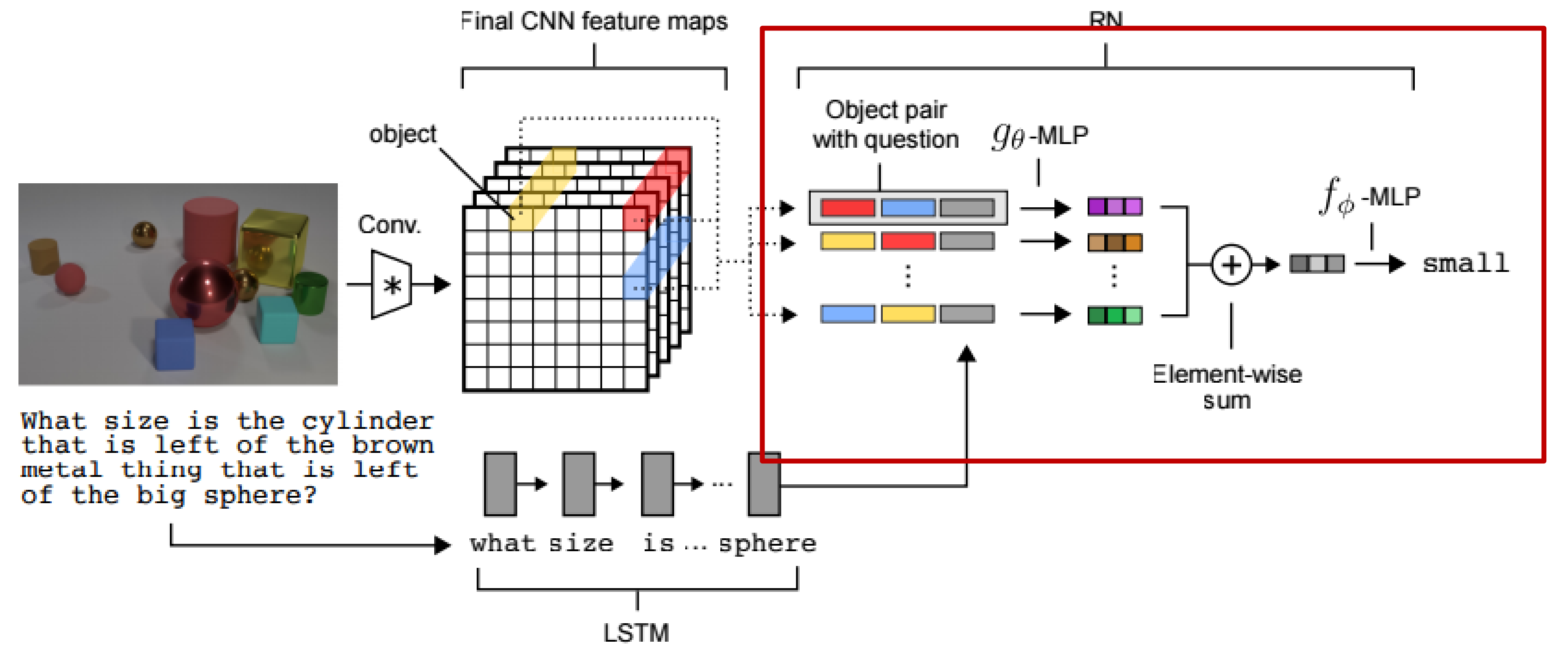
<u>Text</u>

“Sandra picked up the football”
“Sandra went to the office”

Question : “Where is the football?”
(answer: “office”)

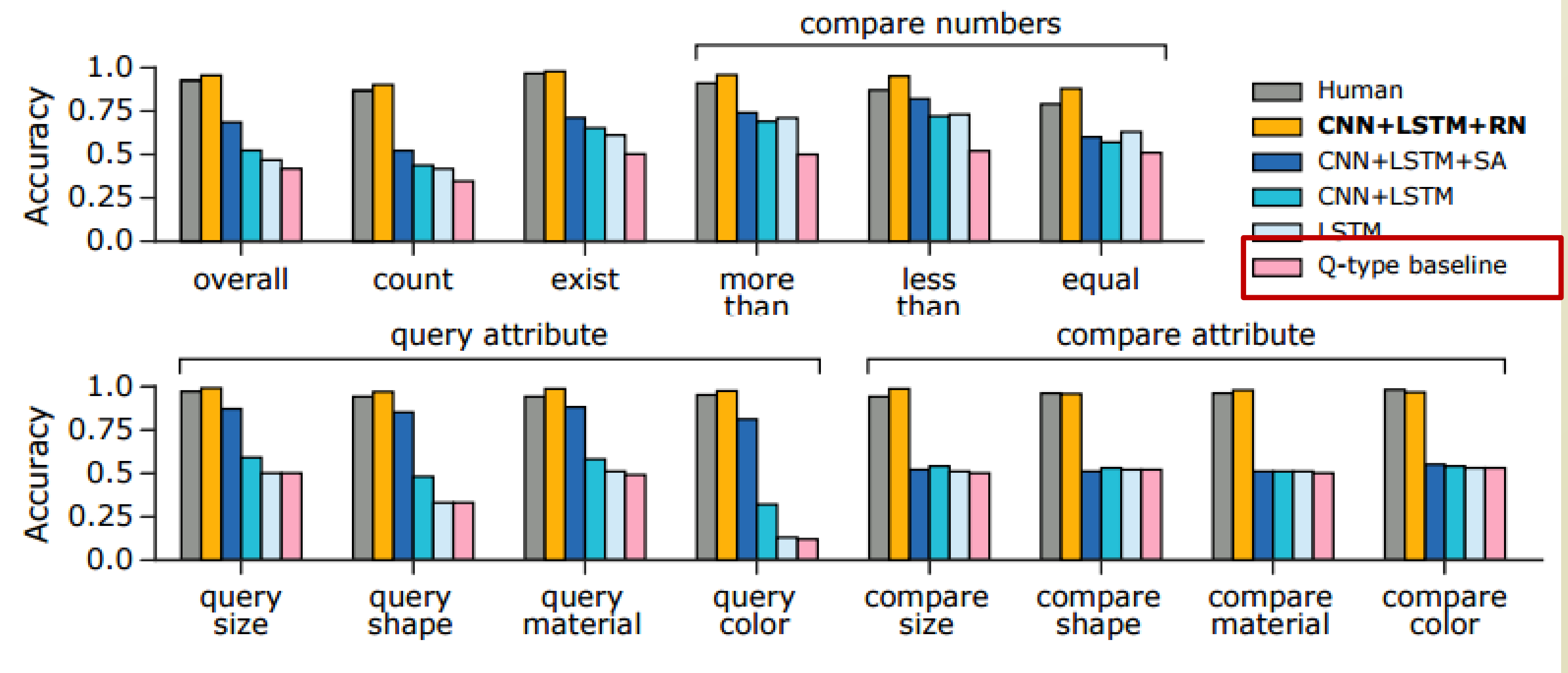
Text based QA(bAbi dataset)
https://research.fb.com/downloads/babi/

Relative Question Model Structure



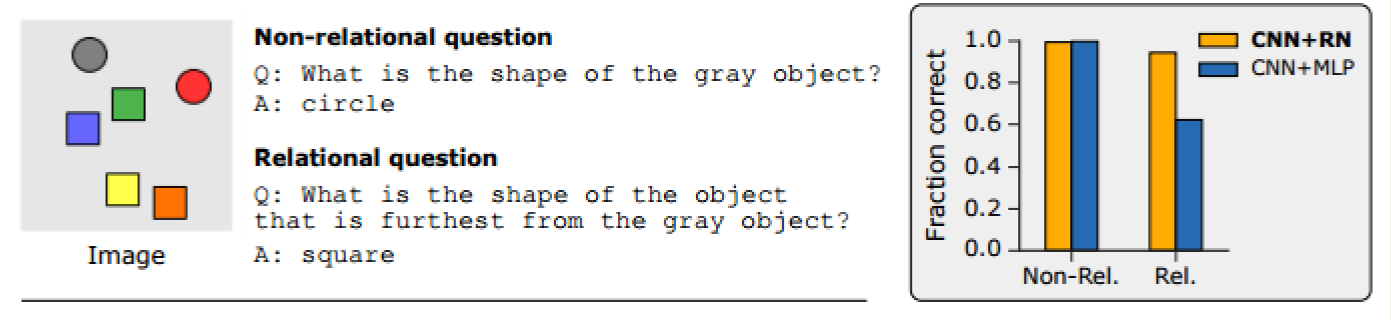
- 앞서 리뷰한 모델과 많은 부분이 유사.
  - Image feature extraction (CNN)
  - Question Modeling (LSTM)
    - LSTM의 마지막 output 만을 question 의 feature 로 활용
- 차별화 되는 부분은 Relation Network(RN) 제안 및 추가

Results of Relative Questions



: 가장 빈도수가 많은 답을 그대로

Sort-of-CLEVR Dataset



Non-relational question

```
Q: What is the shape of the gray object?
A: circle
```

Relational question

```
Q: What is the shape of the object
that is furthest from the gray object?
A: square
```