RNN Architectures

Temporal Sequences

Mrinmoy Maity

Feb 19, 2018

SCHOOL OF INFORMATICS
AND COMPUTING

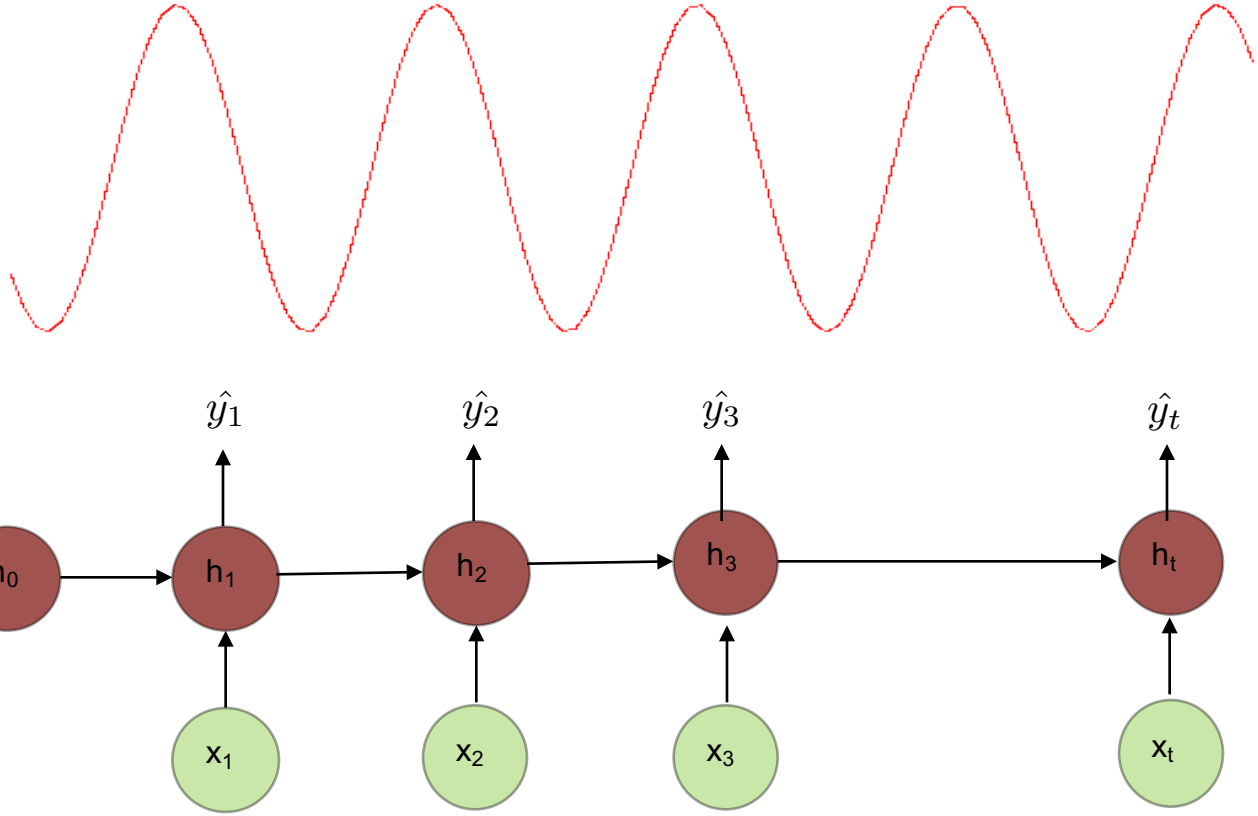
INDIANA UNIVERSITY

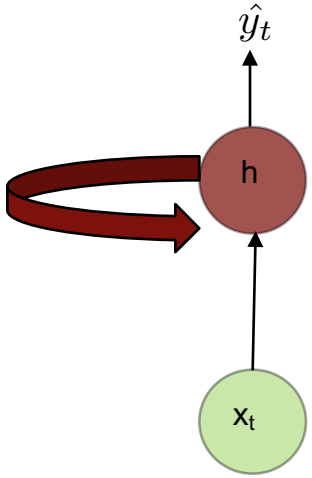
Department of Information and Library Science
Bloomington

Discussions – Part 1

- Basic RNN Architecture Unfolded
- Usefulness
- Vanilla RNN
- BPTT & Vanishing Gradients
- LSTM & Peephole Variants
- GRU
- Experiments

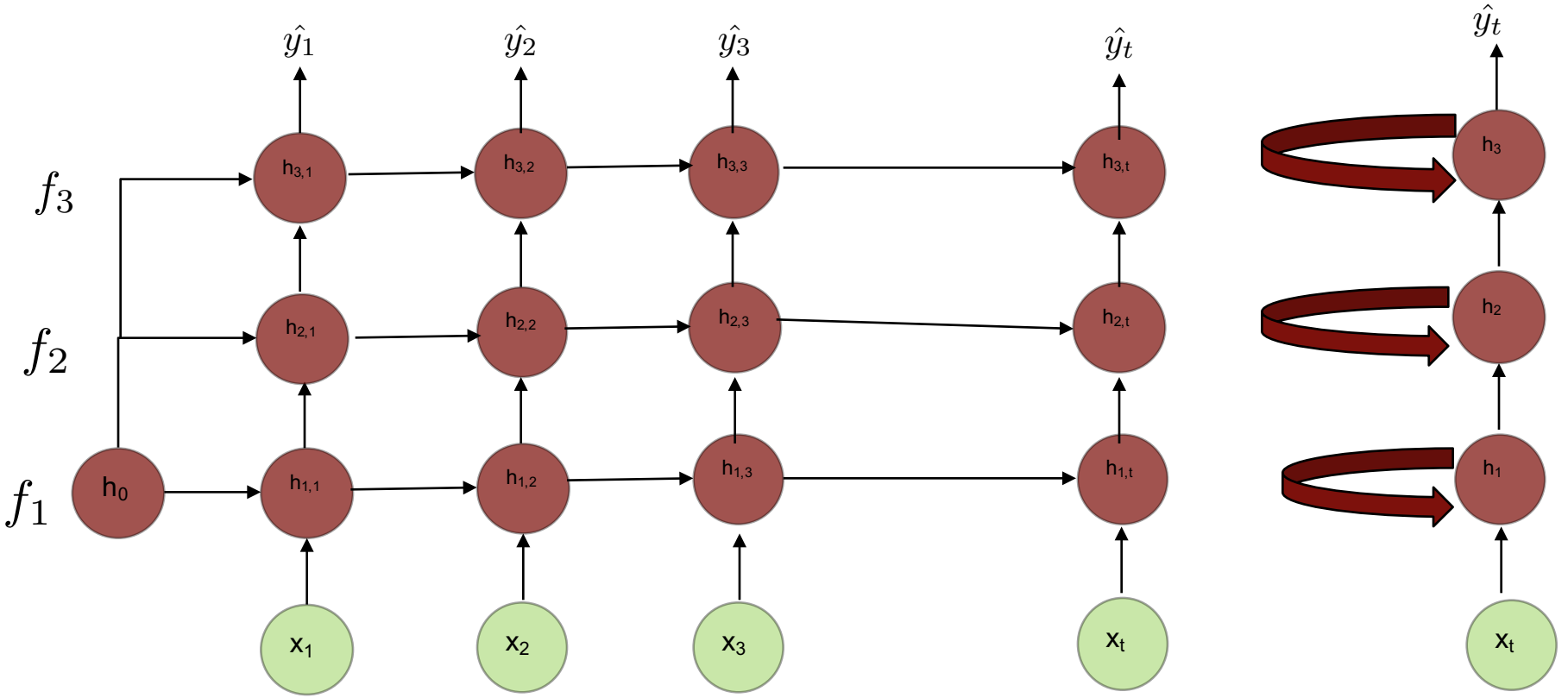
RNN Unfolded





$$< \hat{y}_t, h_t >= f(x_t, h_{t-1})$$

Deep Unidirectional RNNs



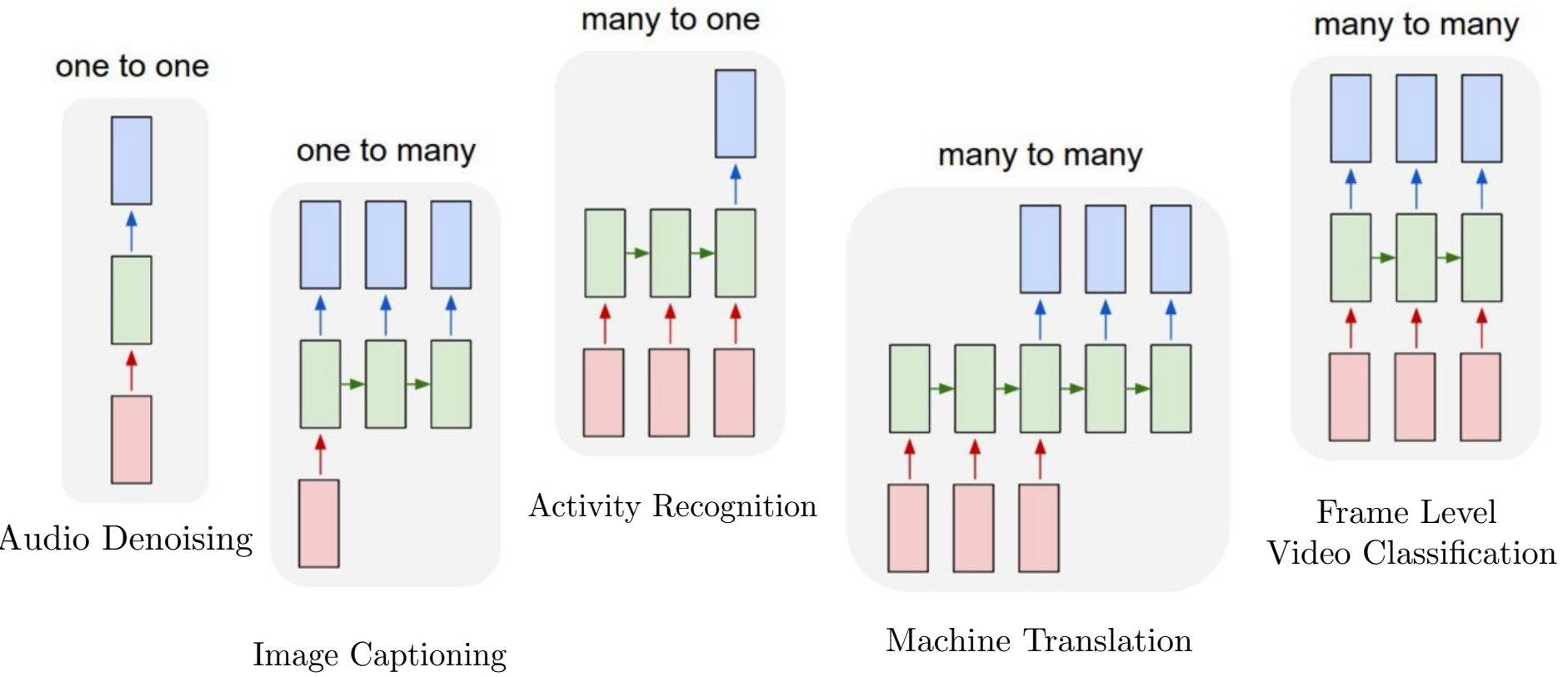
$$< \hat{y}_t, h_t^{(3)} >= f_3(f_2(f_1(x_t, h_{t-1}^{(1)})h_{t-1}^{(2)})h_{t-1}^{(3)})$$

Applications

Operate on data which is inherently temporal

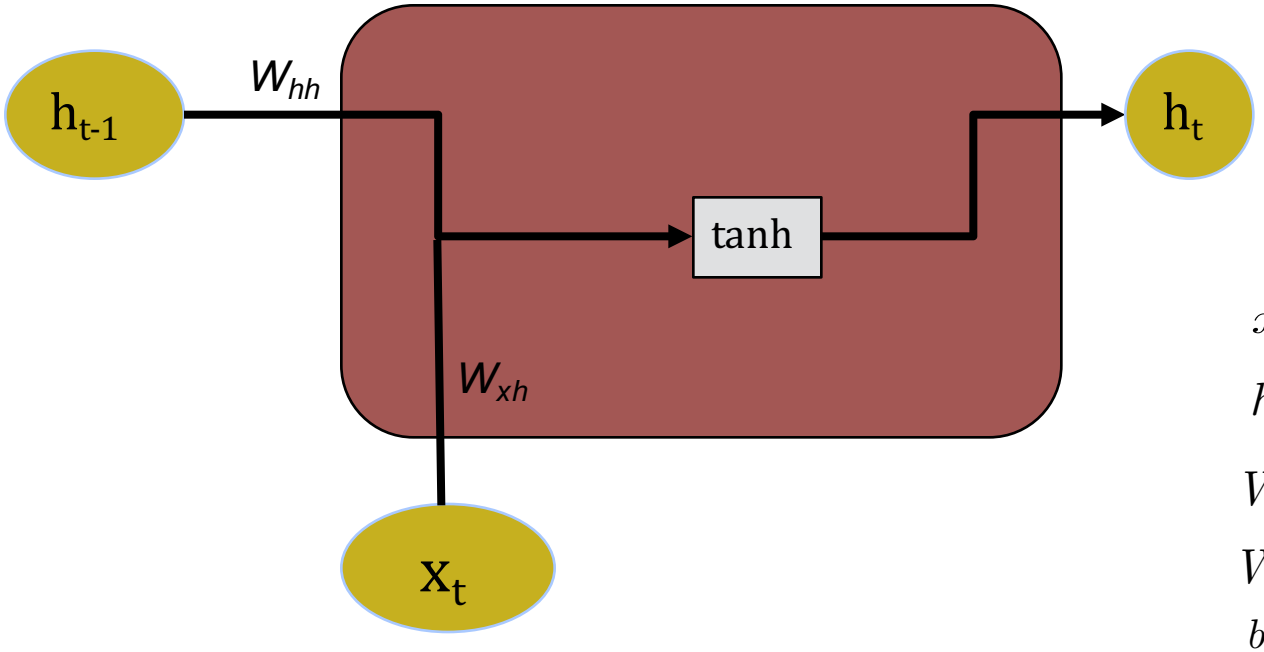
- Speech Recognition
- Activity Detection
- Machine Translation
- Natural Language Processing

RNN Architecture Variants



Vanilla RNN Cell – Simplest Form

- Input cell structure: $h_t = f(h_{t-1}, x_t) = \tanh(h_{t-1} W_{hh} + x_t W_{xh} + b_h)$
- Output layer: $\hat{y}_t = g(h_t)$



$x_t \in R^{n \times f}$

$h_t \in R^{n \times h}$

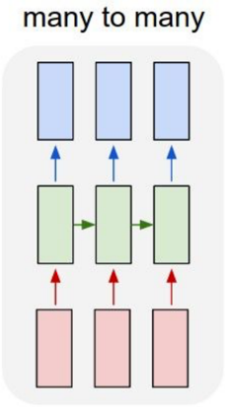
$W_{xh} \in R^{x \times h}$

$W_{hh} \in R^{h \times h}$

$b_h \in R^h$

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Backpropagation in RNNs



- Consider single layer RNN

$$h_t = \tanh(h_{t-1}W_{hh} + x_t W_{xh} + b_h)$$
$$\hat{y}_t = h_t W_{hy} + b_y$$

- Loss defined as squared error

$$L_t = (\hat{y}_t - y_t)^2$$
$$L = \sum_t L_t = \sum_t (\hat{y}_t - y_t)^2$$

- We want to calculate update rules for disentangled weights at any time step 't'

$$\frac{\partial E_t}{\partial W_{hy}} = 2(\hat{y}_t - y_t)\frac{\partial \hat{y}_t}{\partial W_{hy}} = 2(\hat{y}_t - y_t) \otimes h_t$$

$$[\frac{\partial E_t}{\partial W_{hy}}]_{h\times o} = 2[h_t]_{h\times 1}[(\hat{y}_t - y_t)^\top]_{o\times 1}$$

- Update of W_{hy} at time 't' can be computed from information available at time 't'

Backpropagation Through Time (BPTT)

- How about tangled weights W_{hh}?
  - We start with previous formulations

$$h_t = \tanh(h_{t-1}W_{hh} + x_t W_{xh} + b_h)$$
$$\hat{y}_t = h_t W_{hy} + b_y$$
$$L_t = (\hat{y}_t - y_t)^2$$

  - Updates contributed from individual time steps
    - h_1 only dependent on h_0 which is constant after initialization

$$\frac{\partial E_1}{\partial W_{hh}} = \frac{\partial E_1}{\partial \hat{y}_1}\frac{\partial \hat{y}_1}{\partial h_1}\frac{\partial h_1}{\partial W_{hh}}$$

    - h_2 depends on h_0 as well as h_1 and change in W_{hh} affects h_1

$$\frac{\partial E_2}{\partial W_{hh}} = \frac{\partial E_2}{\partial \hat{y}_2}\frac{\partial \hat{y}_2}{\partial h_2}\frac{\partial h_2}{\partial W_{hh}} + \frac{\partial E_2}{\partial \hat{y}_2}\frac{\partial \hat{y}_2}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial W_{hh}}$$

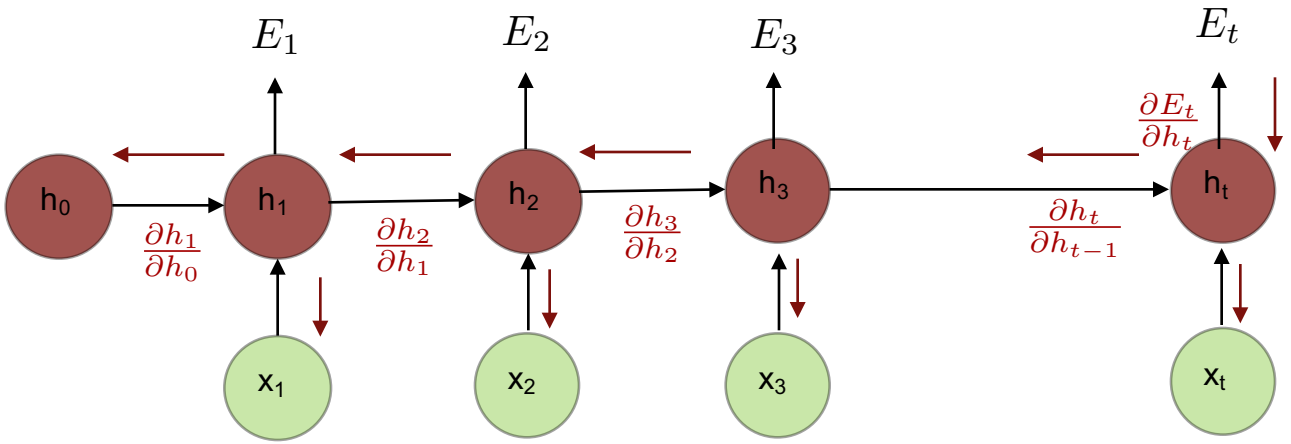
    - Similarly

$$\frac{\partial E_3}{\partial W_{hh}} = \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\frac{\partial h_3}{\partial W_{hh}} + \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial W_{hh}} + \frac{\partial E_3}{\partial \hat{y}_3}\frac{\partial \hat{y}_3}{\partial h_3}\frac{\partial h_3}{\partial h_2}\frac{\partial h_2}{\partial h_1}\frac{\partial h_1}{\partial W_{hh}}$$

Backpropagation Through Time (BPTT)

- Generically update on *t*-th time step

$$\frac{\partial E_t}{\partial W_{hh}} = \sum_{k=0}^{t} \frac{\partial E_t}{\partial \hat{y}_t} \frac{\partial \hat{y}_t}{\partial h_t} \frac{\partial h_t}{\partial h_k} \frac{\partial h_k}{\partial W_{hh}}$$



- What about W_{xh} ?

Vanishing Gradients

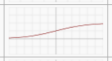
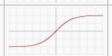
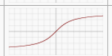
- We have already studied vanishing gradient problem in deep neural networks

- BPTT in RNN makes the problem worse
  - Lets rewrite the cell ops

$$h_t = \tanh(z_t)$$
$$z_t = h_{t-1}W_{hh} + x_t W_{xh} + b_h$$

- Partial derivatives can be rewritten as

$$\frac{\partial E_t}{\partial W_{hh}} = \sum_{k=0}^{t} \frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\frac{\partial h_t}{\partial h_k}\frac{\partial h_k}{\partial W_{hh}}$$
$$= \sum_{k=0}^{t} \frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\left(\prod_{j=k+1}^{t}\frac{\partial h_j}{\partial z_j}\frac{\partial z_j}{\partial h_{j-1}}\right)\frac{\partial h_k}{\partial W_{hh}}$$
$$= \sum_{k=0}^{t} \frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\left(\prod_{j=k+1}^{t}(1-h_j^2)W_{hh}\right)\frac{\partial h_k}{\partial W_{hh}}$$

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1+e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

- In presence of non-linear activations whose derivatives are < 1, $\prod_{j=k+1}^{t}\frac{\partial h_j}{\partial h_{j-1}}$ becomes smaller even for a shallow RNN making updates difficult
- Similarly multiplying W_{hh} multiple times during BPTT makes updates prone to be vanished

$$||\frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\left(\prod_{j=k+1}^{t}\frac{\partial h_j}{\partial z_j}\frac{\partial z_j}{\partial h_{j-1}}\right)||_{l_2} \leq \rho^{t-k}||\frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}||_{l_2}, \rho < 1$$
$$\lim_{t-k\to\infty}||\frac{\partial E_t}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\left(\prod_{j=k+1}^{t}\frac{\partial h_j}{\partial z_j}\frac{\partial z_j}{\partial h_{j-1}}\right)||_{l_2} = 0$$

Solutions : Vanishing Gradients

- Use of activation function like ReLU whose derivative is either 0 or 1
- Proper initialization of weight matrices W_{xh} and W_{hh}
- Use smaller number of time steps
- Truncated backpropagation
- Gating mechanism
  - Change cell architectures
  - Next slides
- **Can there be exploding gradient problem with RNN?**

Long Short Term Memory (LSTM)

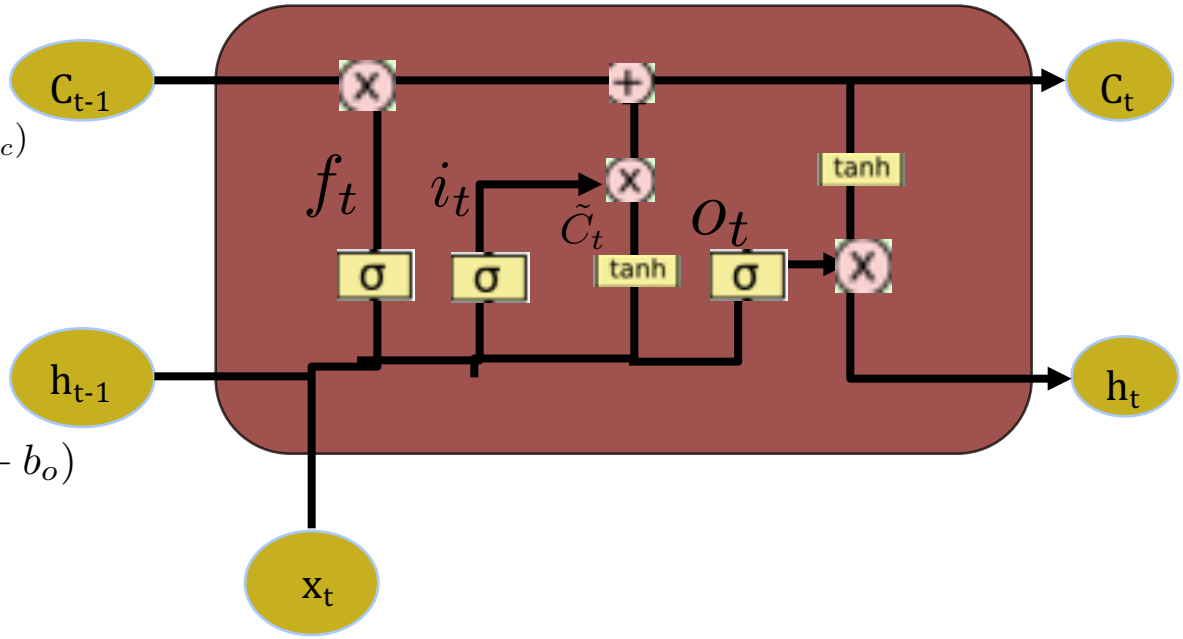
- Input cell structure:
  - Let information flow C_{t-1} to C_t
  - What to forget ? $f_t = \sigma(h_{t-1}W_{hf} + x_tW_{xf} + b_f)$
  - What to write ? $i_t = \sigma(h_{t-1}W_{hi} + x_tW_{xi} + b_i)$
  - To what write ? Candidates

    $\tilde{C}_t = \tanh(h_{t-1}W_{hc} + x_tW_{xc} + b_c)$
  - Update cell state

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

    $o_t = \sigma(h_{t-1}W_{ho} + x_tW_{xo} + b_o)$

    $h_t = o_t * \tanh(C_t)$



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Peephole LSTM

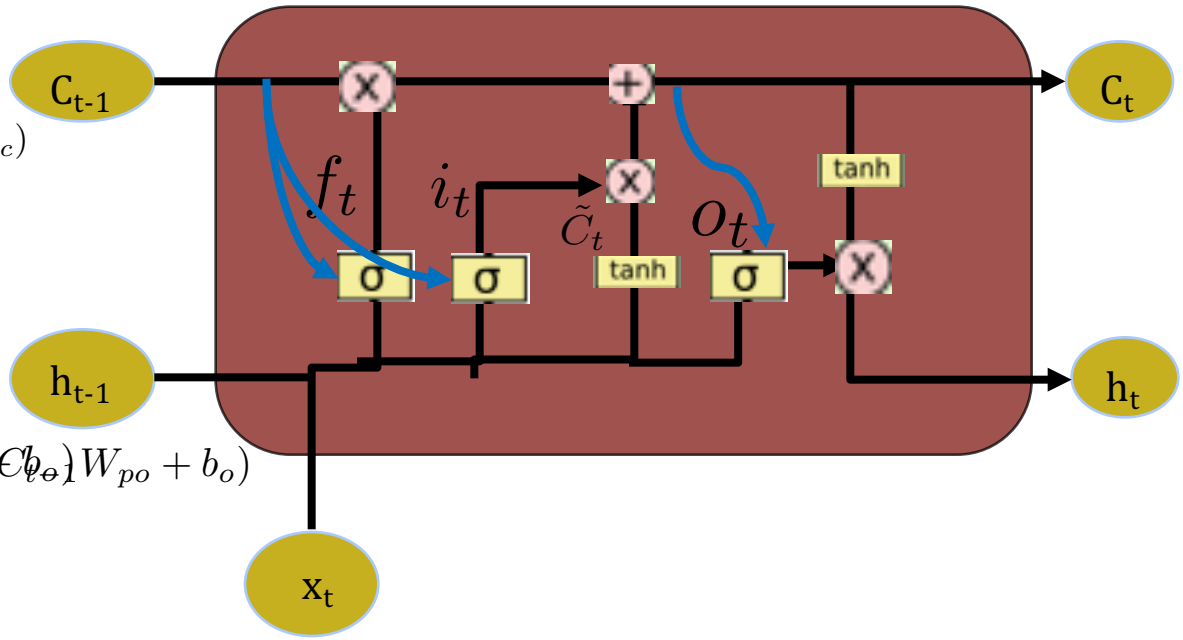
- Input cell structure:
  - Let information flow C_{t-1} to C_t
  - What to forget ? $f_t = \sigma(h_{t-1}W_{hf} + x_tW_{xf} + C_{t-1}W_{pf} + b_f)$
  - What to write ? $i_t = \sigma(h_{t-1}W_{hi} + x_tW_{xi} + C_{t-1}W_{pi} + b_i)$
  - To what write ? Candidates

    $\tilde{C}_t = \tanh(h_{t-1}W_{hc} + x_tW_{xc} + b_c)$
  - Update cell state

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

    $o_t = \sigma(h_{t-1}W_{ho} + x_tW_{xo} + C_{t-1}W_{po} + b_o)$

    $h_t = o_t * \tanh(C_t)$



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Gated Recurrent Units (LSTM)

- Two major differences with LSTM
  - Combine 'Input' and 'forget' gate into one z_t
  - Maintain only one state h_t instead of two
  - Read gate

$$r_t = \sigma(h_{t-1}W_{hr} + x_tW_{xr} + b_r)$$

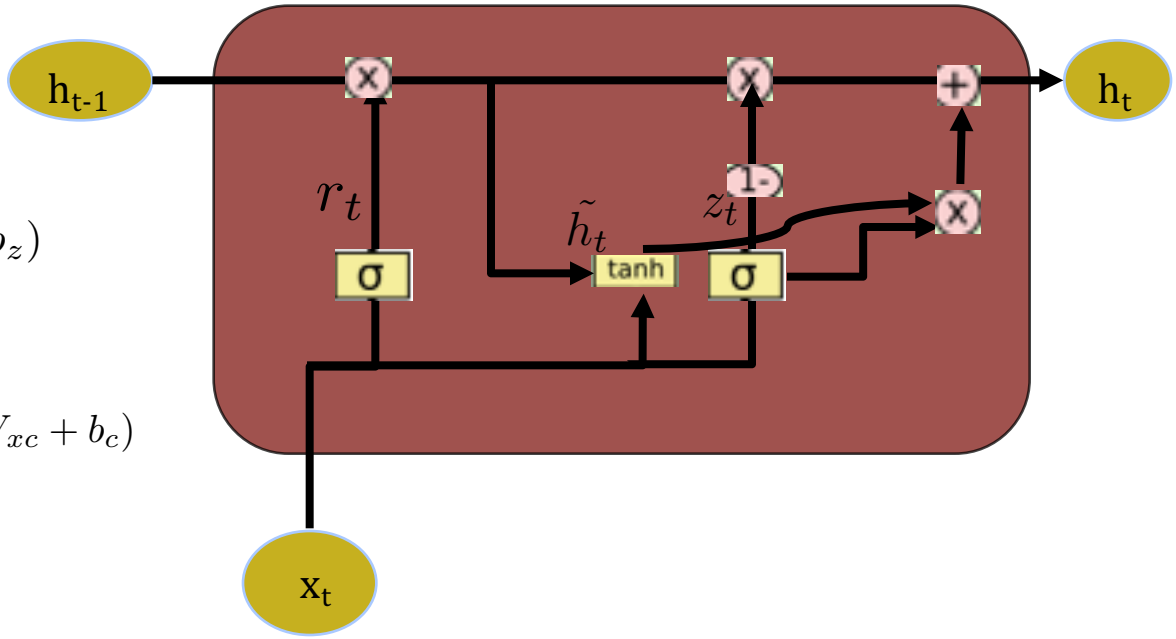
  - Update gate:

$$z_t = \sigma(h_{t-1}W_{hz} + x_tW_{xz} + b_z)$$

  - Cell update and output

$$\tilde{h_t} = \tanh((h_{t-1} * r_t)W_{hc} + x_tW_{xc} + b_c)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h_t}$$



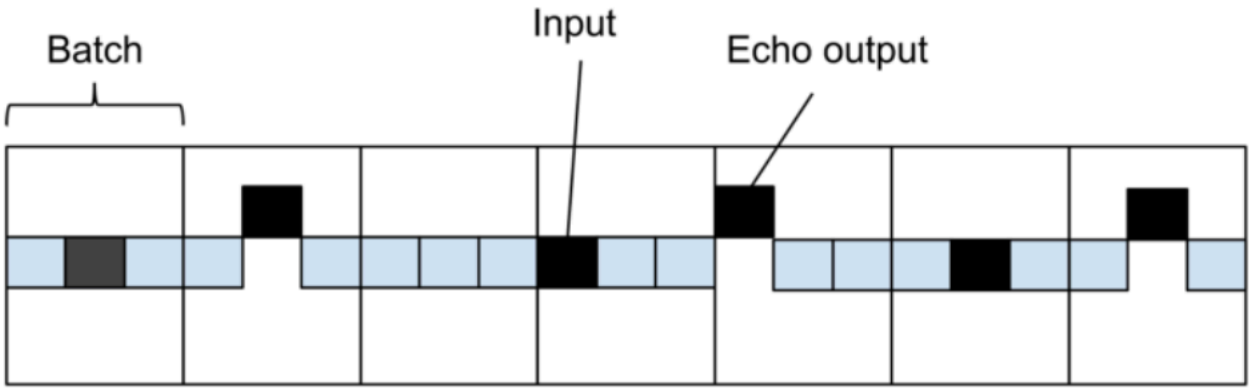
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Tensorflow RNN

- Basic Architecture of RNN models in Tensorflow
- Unfolded RNNs and sharing weights across time
- Compare vanilla RNN, LSTM & GRU
- Writing custom cells and multilayered RNNs

Experiments: Echo State Network

- Generate X s.t.
  - Sample of each sequence randomly generated from classes {0,1}
  - $P(x_t=0) = P(x_t=1) = 0.5$
- Generate Y of same shape as X
  - Roll over X for k time steps



https://r2rt.com/recurrent-neural-networks-in-tensorflow-i.html

Experiments: Double Dependency

- Generate X s.t.
  - Sample of each sequence randomly generated from classes {0,1}
  - $P(x_t=0) = P(x_t=1) = 0.5$
- Generate Y of same shape as X s.t.
  - $P(y_t=0) = P(y_t=1) = 0.5$ in unbiased cases
  - Increase $P(y_t=1)$ by 0.50 when $X_{t-3}=1$
  - Decrease $P(y_t=1)$ by 0.25 when $X_{t-8}=1$
- How to measure performance of models ?
  - 3 situations
    - Learns no dependeny
    - Learn single dependency (X_{t-3})
    - Learn double dependency (X_{t-8})

https://r2rt.com/recurrent-neural-networks-in-tensorflow-i.html

Experiments: Double Dependency

- Case 1: Learn no dependencies
  - $P(y_t=1) = P(y_t=1 \mid x_{t-3} = 1, x_{t-8} = 1) * P(x_{t-3} = 1) * P(x_{t-8} = 1)$
    $+ P(y_t=1 \mid x_{t-3} = 1, x_{t-8} = 0) * P(x_{t-3} = 1) * P(x_{t-8} = 0)$
    $+ P(y_t=1 \mid x_{t-3} = 0, x_{t-8} = 1) * P(x_{t-3} = 0) * P(x_{t-8} = 1)$
    $+ P(y_t=1 \mid x_{t-3} = 0, x_{t-8} = 0) * P(x_{t-3} = 0) * P(x_{t-8} = 0)$

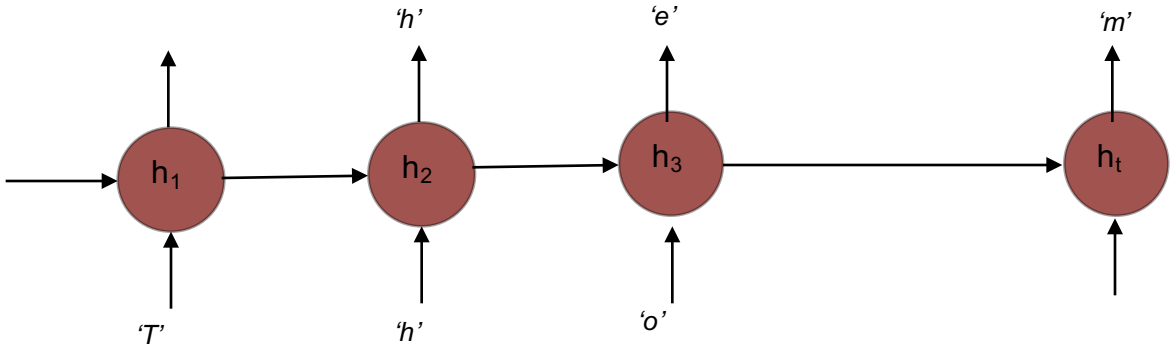
    $= 0.75 * 0.25 + 1 * 0.25 + 0.25 * 0.25 + 0.5 * 0.25$
    $= 0.625$
  - $P(y_t=0) = 0.375$
  - $L = -(0.625 * np.log(0.625) + 0.375 * np.log(0.375)) = 0.66$

- Case 2: Learn single dependency
  - $P(y_t=1 \mid x_{t-3} = 1) = P(y_t=1 \mid x_{t-3} = 1, x_{t-8} = 1) * P(x_{t-8} = 1)$
    $+P(y_t=1 \mid x_{t-3} = 1, x_{t-8} = 0) * P(x_{t-8} = 0)$
    $= 0.75 * 0.5 + 1 * 0.5 = 0.875$
  - $P(y_t=1 \mid x_{t-3} = 0) = 0.375$
  - $P(y_t=0 \mid x_{t-3} = 1) = 0.125$
  - $P(y_t=1 \mid x_{t-3} = 0) = 0.625$
  - $L = 0.51$

- Case 3: Learn both dependencies
  - $L = 0.45$

Experiments: Character Level Language Modeling

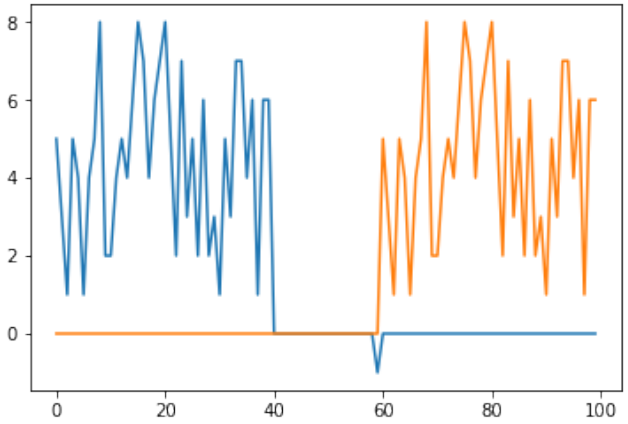
- Objective
  - Learn to predict one character at a time
- Trained on Shakespeare corpus data
  - All characters in our data as vocabulary
  - Case sensitive
  - Restrict special characters using unidentified symbol to prevent producing gibberish
  - Characters represented as one-hot vector because of limited vocabulary size



https://r2rt.com/recurrent-neural-networks-in-tensorflow-ii.html

Experiments: Copy Memory Problem

- Pick number of classes used for data generation, lets say 10
- Generate a sequence of length 100
  - First 40 numbers are uniformly sampled from all classes
  - Next 60 numbers are zero (no relevant information)
  - At 60th step, insert a blip to indicate display what it memorizes
- For constructing targets, shift data 60 steps

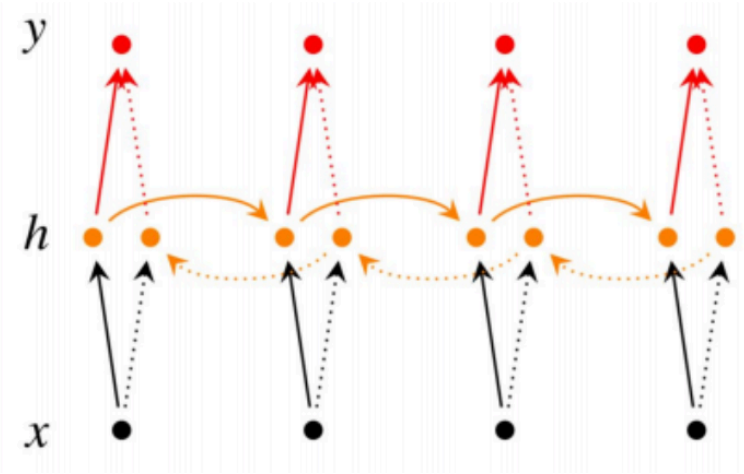


Discussions- Part 2

- Bidirectional RNNs
- Sequence Tagging
- Custom Cell Architectures
- Generative RNNs
- Statistical Machine Translation
- Combining CNNs and RNNs

Bidirectional RNN

- Machine Translation
  - Objective : Translate English to Mandarin
  - Translate “Homework is due on 9th March” -> “*家庭作业将于3月9日到期*”
  - To translate it properly, if we want to understand context of the word ‘due’, we need to look words before and after it
- Video Classification
  - To identify action at any point in video, we might need to look couple of frames before and after the current frame
- Bidirectional RNN is concatenation of two RNNs where input is reversed to second set of parameters



Sequence Tagging

- Class of Natural Language Processing tasks on which seeks to locate and classify named entities within text into pre-defined categories
- Entities
  - ORG – Organization
  - LOC – Location
  - MISC – Miscellaneous
  - PER – Person
- Multiword vs Single Word
  - Differentiate using (B)eginning and (I)ntermediate

```
Deep  Learning Systems is offered for the first time at Indiana University
B-ORG I-ORG    I-ORG   O  O       O   O   O     O    O  B-ORG   I-ORG
```

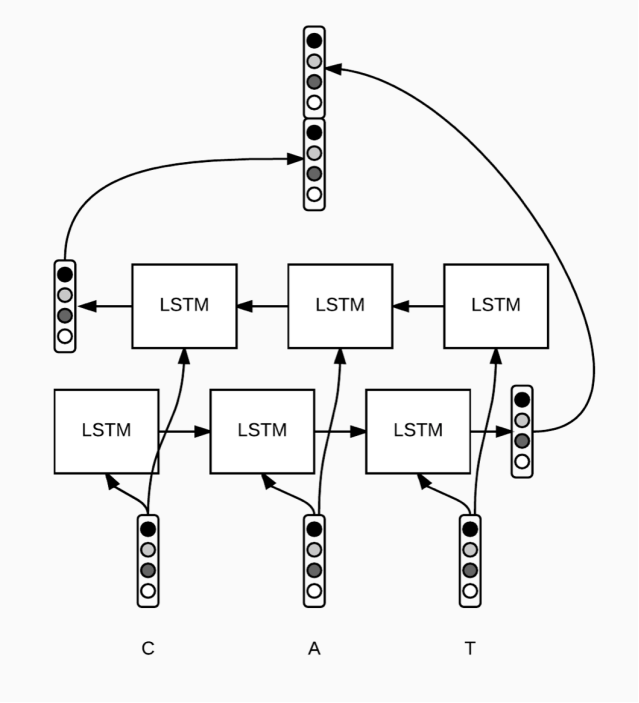
```
Are you flying to Japan during Spring Break
O   O   O      O  B-LOC O      O      I-MISC
```

Why RNNs are suitable for Sequence Tagging ?

- Record based approach
  - Maintain a database of all named entities
  - For each word, search entire database and classify/tag
- Parameterized approach
  - Why ?
    - Record based approach often not scalable and sometimes not feasible
  - Exact match often not needed e.g. made-up names, unknown places
  - Could sentence structures indicate the class of word ?
  - To understand the way sentence has been structured, one has to rely on temporal dependencies
- Here we deal with a subtask of Sequence Tagging, Named Entity Recognition

Character Level Word Representation

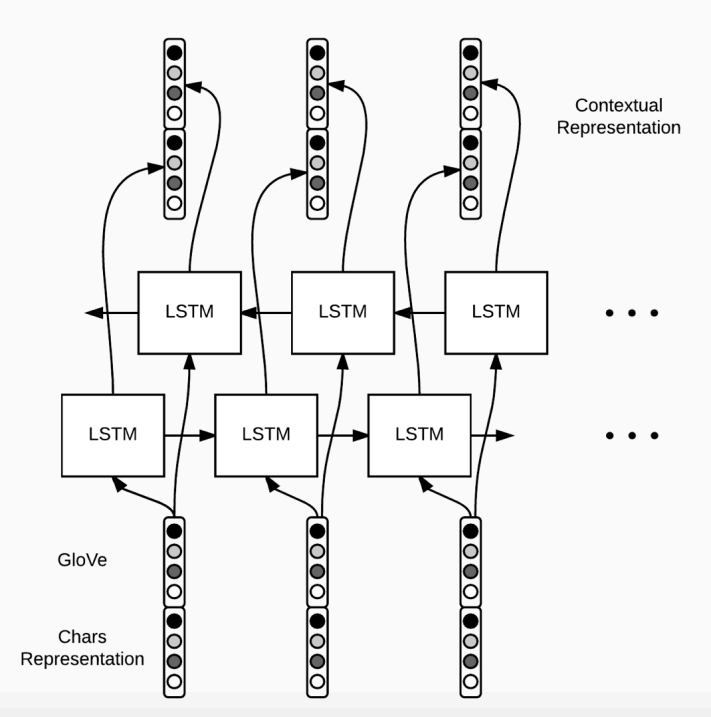
- Word Representation
  - Lets assume word $w_i = [c_1, c_2, \ldots..c_n]$ where $c_i \in R^{num-alphabets}$
- Apply bi-LSTM over sequence of characters to obtain fixed sized vector $v_i \in R^f$
- (Optional) Augment character level vector by concatenating with other word embeddings



"Bidirectional LSTM-CRF Model for Sequence Tagging." Z. Huang, W. Xu, K. Yu
Diagram: https://guillaumegenthial.github.io/sequence-tagging-with-tensorflow.html

Contextual Word Representation

- Significance of a single word in context to a sentence
- Word level representations $v_i \in R^f$ provided as inputs to uni-directional or bidirectional LSTM
- Depending on choice of RNN architecture, decide to concatenate contextual representation $h_i \in R^k$



"Bidirectional LSTM-CRF Model for Sequence Tagging." Z. Huang, W. Xu, K. Yu
Diagram: https://guillaumegenthial.github.io/sequence-tagging-with-tensorflow.html

Design Final Layer for Classification

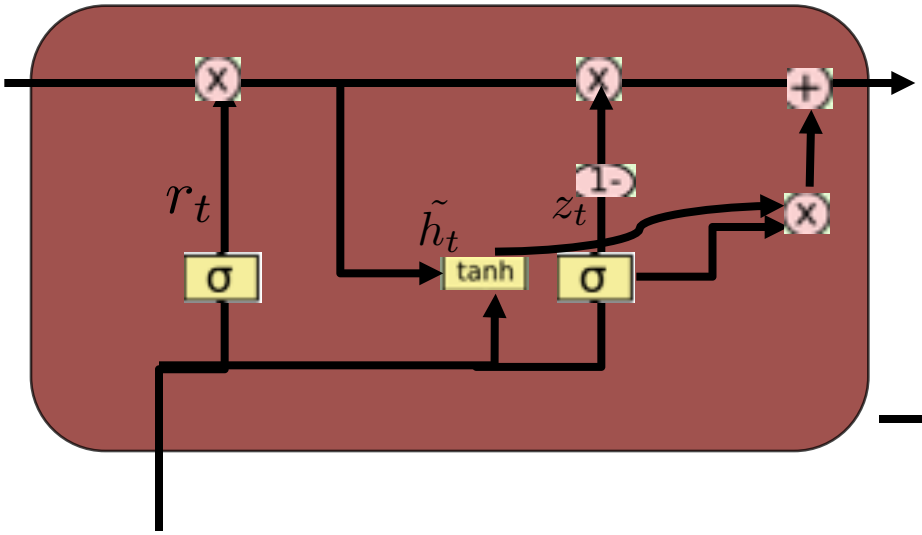
- Again, design choice for final layer output
- Many to many architecture of RNN
  - Each word(input) has a corresponding classification tag
- Softmax
  - Use fully connected output layer with output size for each word as number of classes 'm'
  - Use Softmax layer to normalize $p(\hat{y}_i = f(h_i) = k|w_i) = \frac{e^{h_i}}{\sum_{j=1}^{m} e^{h_j}}$

$$p(\hat{y}) = \prod_{i=1}^{n} p(\hat{y}_i)$$

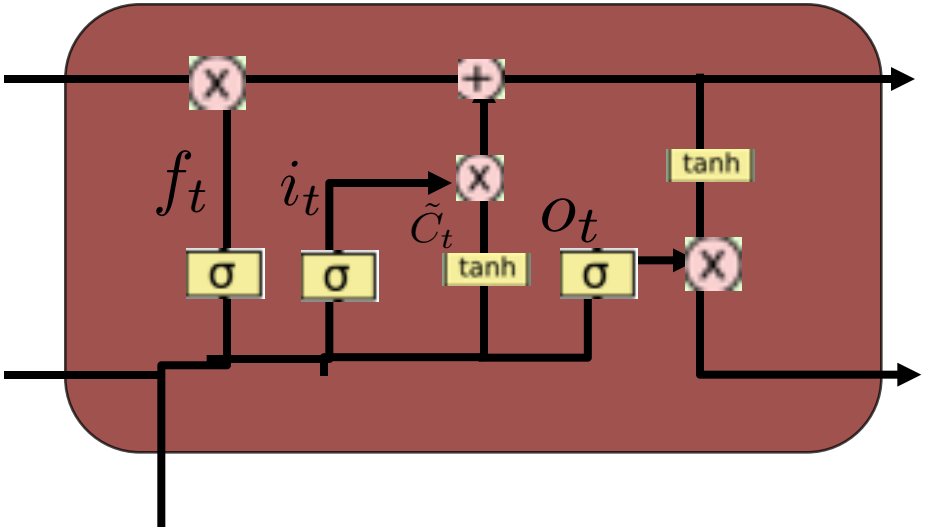
- Joint distribution of tags $p(\hat{y}) = \frac{e^{C(h_1, h_2, ..., h_n)}}{Z}$
  - Linear chain CRF: $C(h_1, h_2, ..., h_n) = \sum_{j=1}^{n} h_j + \sum_{j=1}^{n-1} T[h_j, h_{j+1}]$
  - Apply Softmax using $Z = \sum_{y_1} \sum_{y_2} ..... \sum_{y_n} e^{C(h_1, h_2, ..., h_n)}$

"Bidirectional LSTM-CRF Model for Sequence Tagging". Z. Huang, W. Xu, K. Yu

Cell Architectures: Code Walkthrough



GRU



LSTM

Customized Cell Architectures: Peephole LSTM

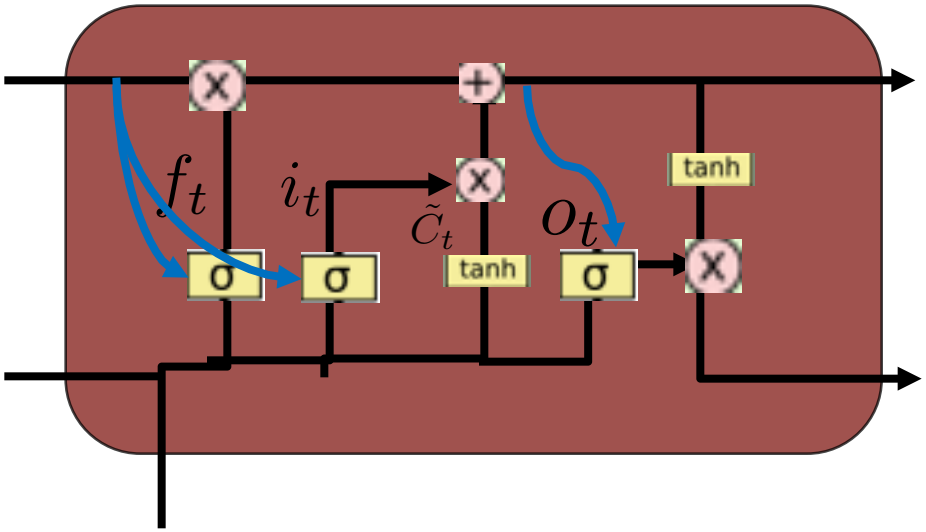
- Add extra blue arrows to existing implementations\
  - Define the variables corresponding to three arrows
  - Update equation of three gates

$$f_t = \sigma(h_{t-1}W_{hf} + x_tW_{xf} + C_{t-1}W_{pf} + b_f)$$

$$i_t = \sigma(h_{t-1}W_{hi} + x_tW_{xi} + C_{t-1}W_{pi} + b_i)$$

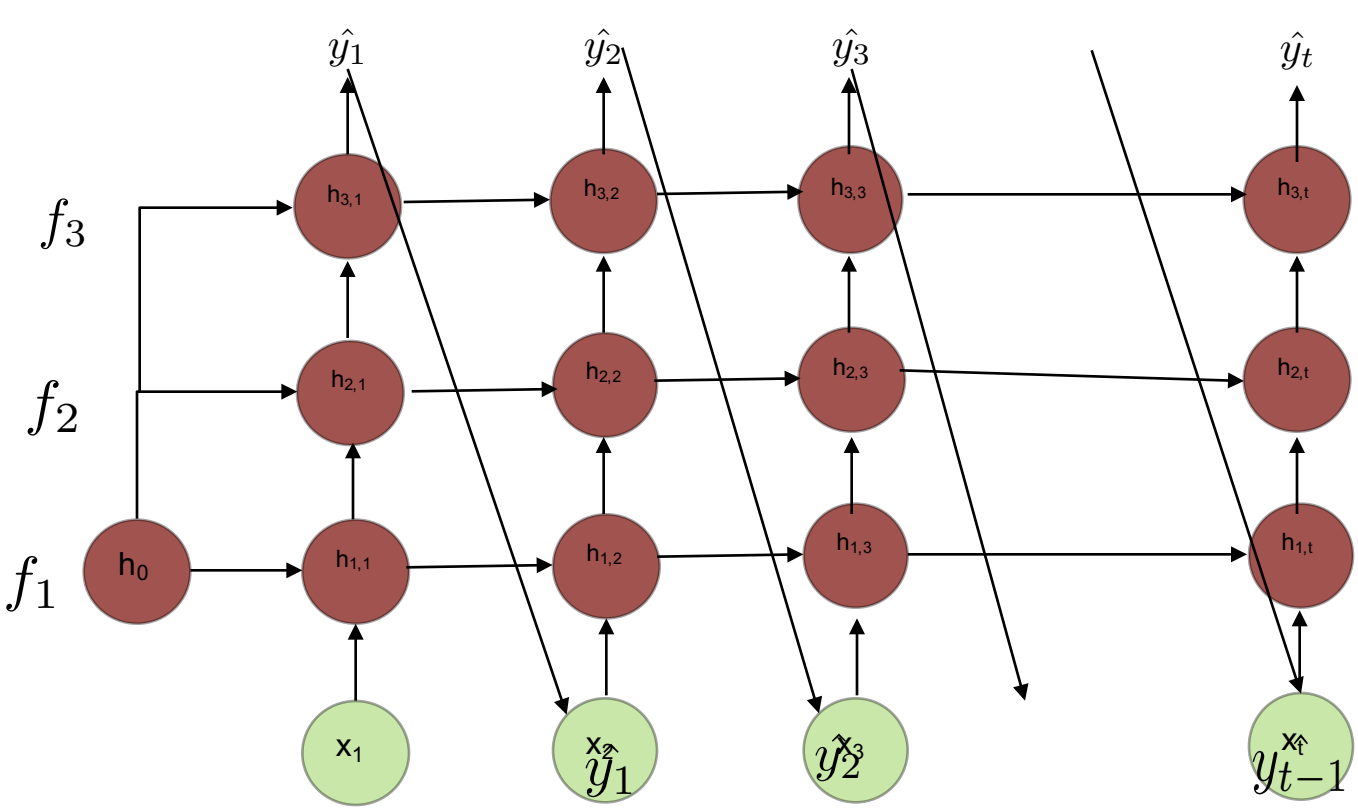
$$o_t = \sigma(h_{t-1}W_{ho} + x_tW_{xo} + C_{t-1}W_{po} + b_o)$$

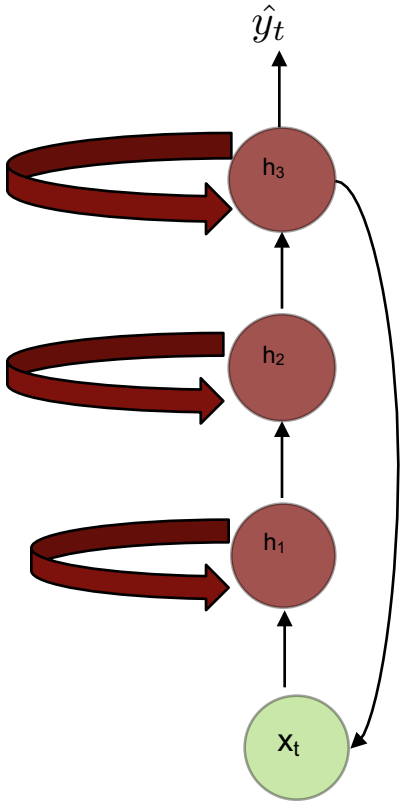
  - Rest remains unchanged



Model Architecture

Discriminative vs Generative RNN





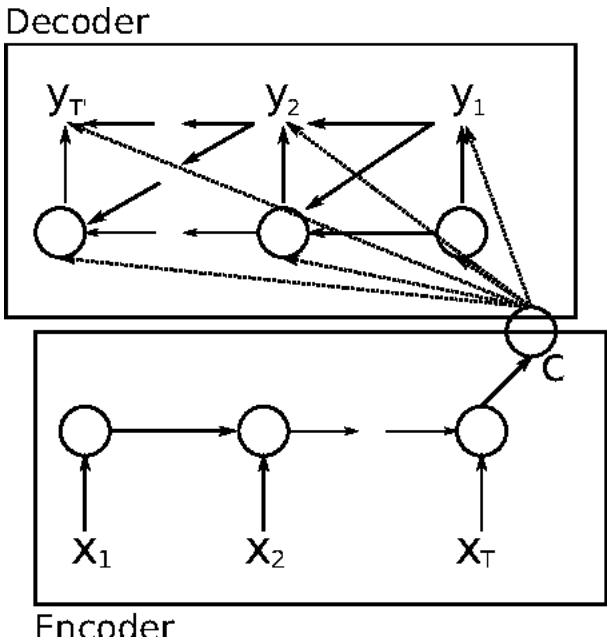
$$< \hat{y}_t, h_t^{(3)} >= f_3(f_2(f_1(\hat{y_{t-1}}, h_{t-1}^{(1)})h_{t-1}^{(2)})h_{t-1}^{(3)})$$

Applications ?

Statistical Machine Translation

- Objective: Translate any sentence in a language to another sentence in different language

- Or more specifically
  - Propose a neural network that consists of encoder and decoder
  - Encoder encodes variable length sequence to a fixed length vector
  - Decoder decodes fixed length vector to variable length sequence(in different language)
  - Formally we are trying to predict $p(y_1, ...y_{T'}|x_1, ...x_T)$
- Encoder
  - $(h_t)_{enc} = f(x_t, (h_{t-1})_{enc})$
- Decoder
  - $(h_t)_{dec} = g(y_{t-1}, (h_{t-1})_{dec}, c)$
  - $(y_t)_{dec} = g_1((h_t)_{dec})$
- Each RNN cell used in experiment is GRU
- For input/output pair (x_n, y_n), maximize joint conditional log-likelihood
  - $max_\theta \frac{1}{N} \sum_{n=1}^{N} \log p_\theta(y_n|x_n)$



"Learning Phase Representations using RNN Encoder-Decoder for Statistical Machine Translation". K. Cho, D. Bahdanau, F.Bougares, H. Schwenk and Y. Bengio

Statistical Machine Translation

Source	Translation Model	RNN Encoder–Decoder
at the end of the	[a la fin de la] [f la fin des années] [être supprimés à la fin de la]	[à la fin du] [à la fin des] [à la fin de la]
for the first time	[r © pour la premirëre fois] [été donnés pour la première fois] [été commémorée pour la première fois]	[pour la première fois] [pour la première fois ,] [pour la première fois que]
in the United States and	[? aux ?tats-Unis et] [été ouvertes aux États-Unis et] [été constatées aux États-Unis et]	[aux Etats-Unis et] [des Etats-Unis et] [des États-Unis et]
, as well as	[?s , qu'] [?s , ainsi que] [?re aussi bien que]	[, ainsi qu'] [, ainsi que] [, ainsi que les]
one of the most	[?t ?l' un des plus] [?l' un des plus] [être retenue comme un de ses plus]	[l' un des] [le] [un des]

(a) Long, frequent source phrases

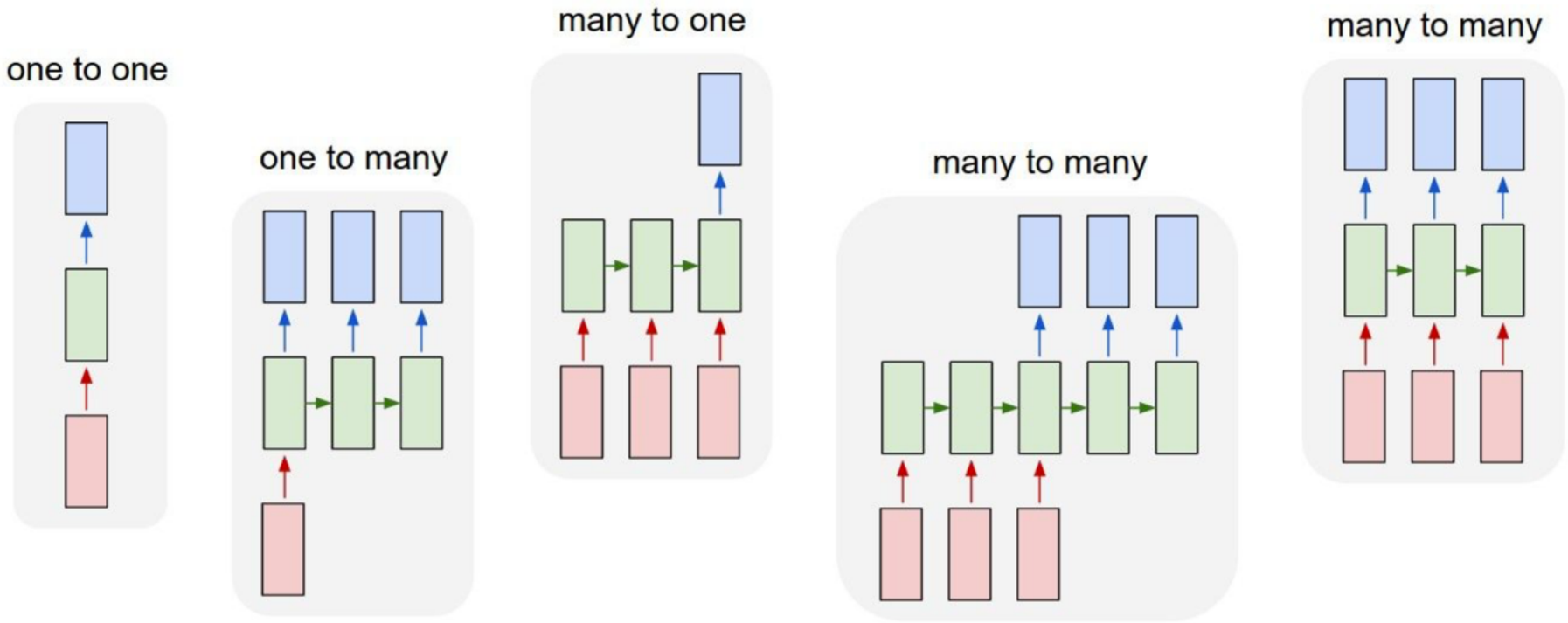
Source	Translation Model	RNN Encoder–Decoder
, Minister of Communications and Transport	[Secrétaire aux communications et aux transports :] [Secrétaire aux communications et aux transports]	[Secrétaire aux communications et aux transports] [Secrétaire aux communications et aux transports :]
did not comply with the	[vestimentaire , ne correspondaient pas à des] [susmentionnée n' était pas conforme aux] [présentées n' étaient pas conformes à la]	[n' ont pas respecté les] [n' était pas conforme aux] [n' ont pas respecté la]
parts of the world .	[© gions du monde .] [régions du monde considérées .] [région du monde considérée .]	[parties du monde .] [les parties du monde .] [des parties du monde .]
the past few days .	[le petit texte .] [cours des tout derniers jours .] [les tout derniers jours .]	[ces derniers jours .] [les derniers jours .] [cours des derniers jours .]
on Friday and Saturday	[vendredi et samedi à la] [vendredi et samedi à] [se déroulera vendredi et samedi ,]	[le vendredi et le samedi] [le vendredi et samedi] [vendredi et samedi]

(b) Long, rare source phrases

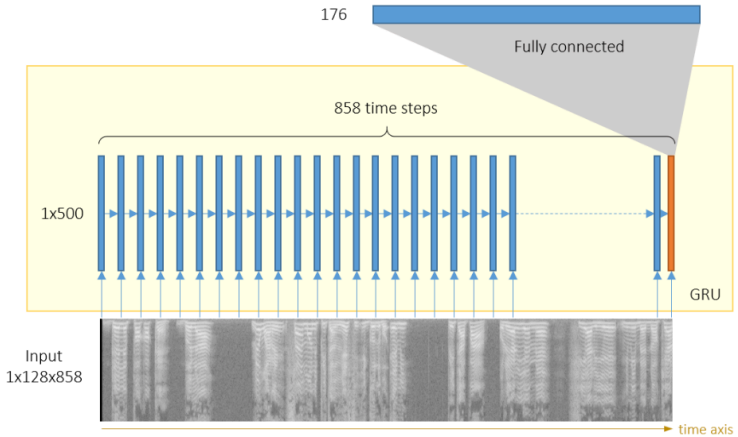
English to French translation

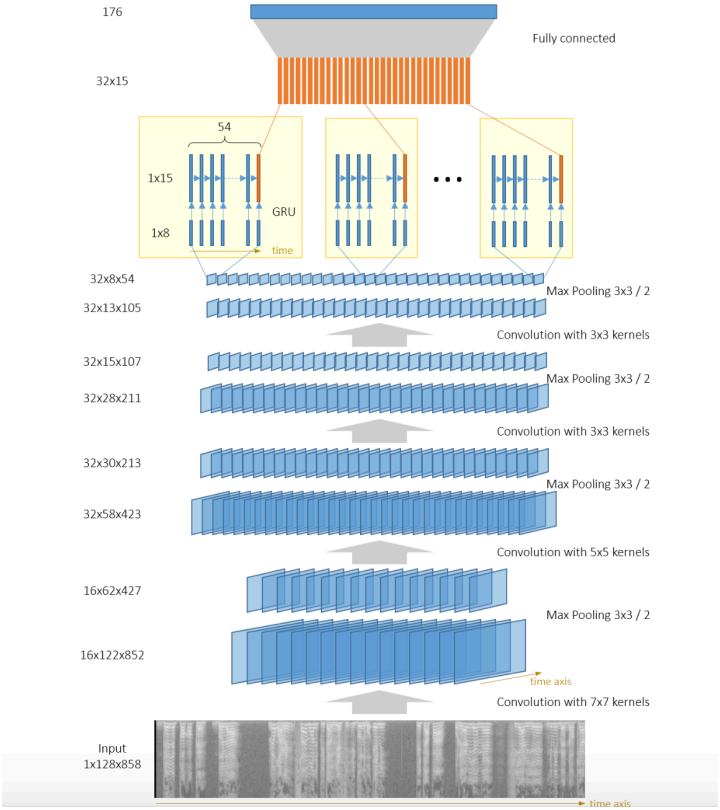
"Learning Phase Representations using RNN Encoder-Decoder for Statistical Machine Translation" . K. Cho, D. Bahdanau, F.Bougares, H. Schwenk and Y. Bengio

Which architecture does RNN Encoder-Decoder belongs?



CNN + RNN





https://yerevann.github.io/2016/06/26/combining-cnn-and-rnn-for-spoken-language-identification/

CNN + RNN: Experiments

- Voice Classification
  - Given ‘mp3’ or ‘wav’ file, classify the voice of speaker
  - Produce 2D spectrogram from audio
  - Consider each spectrogram as an image and use convolution as feature extractor
  - Last layer of convolution given as input to RNN

- Activity Recognition
  - Given a short video, categorize the activity
  - Extract frames in regular interval from video and give it as input to CNN
  - Output of last layer of CNN given to RNN as input

- CNN handover to RNN
  - Separate RNN acting over each channel
  - Consider CNN as 3D tensor as use RNN constructed using 2D cells

https://yerevann.github.io/2016/06/26/combining-cnn-and-rnn-for-spoken-language-identification/

Thanks