Deep Learning Lessons

Outline Deep Learning Lessons

- Lesson 1: Linear models. Feed Forward NN. Simple Perceptron. Multilayer Perceptron (MLP). Neural language modeling and Word embeddings. Use of words embeddings.
- **Lesson 2: Recurrent NN (RNN): GRU, LSTM. Recursive. Embeddings of more complex units with RNNs.**
- Lesson 3: Convolutional NN for NLP. NLP applications. Libraries and languages for NN: PyTorch.

Outline Lesson 2

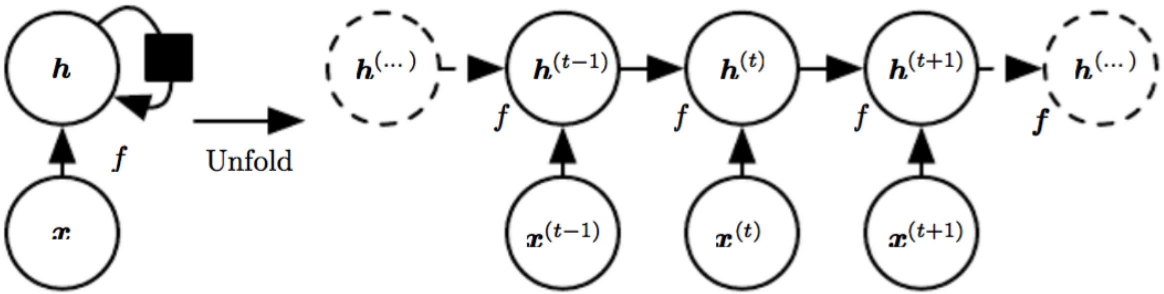
- Introduction
- Recurrent Neural Networks
- Gated Units
- Applications

Motivation

- In natural language processing we often deal with SEQUENCES (e.g. Language modeling, part-of-speech tagging, machine translation…)
- We need models that take information of SEQUENCE

- HOW?
- By recurrency

Recurrent computational graph

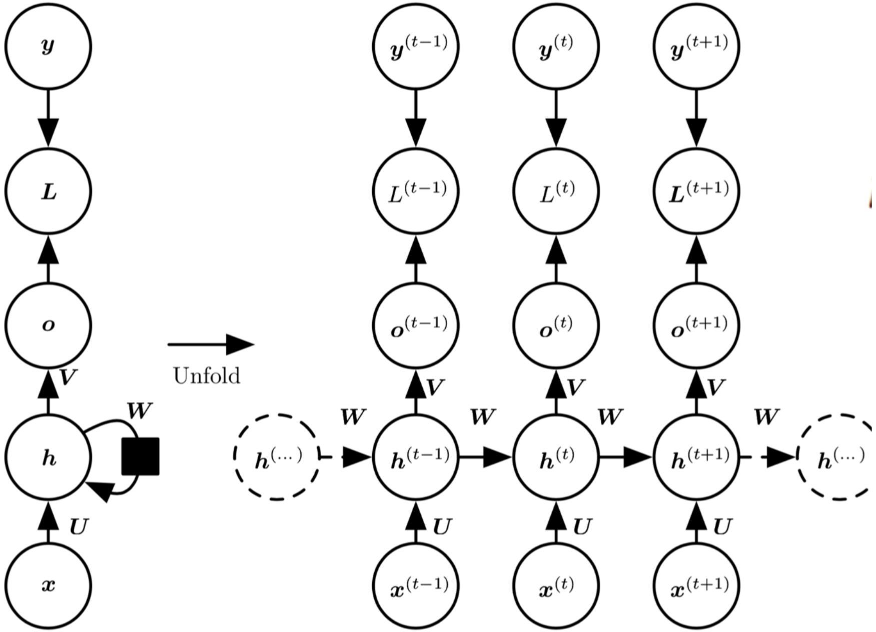
- Regardless of the sequence length, the learned model always has **the same input size**, because it is specified in terms of transition from one state to another state, rather than specified in terms of a variable-length history of states.
- 

Vanilla RNN

Recurrent Neural Network (RNN) adding the "temporal" evolution

Allow to build specific connections capturing "history"

$$
\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}, \\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}), \\
\boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}, \\
\hat{\boldsymbol{y}}^{(t)} &= \mathrm{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}
$$



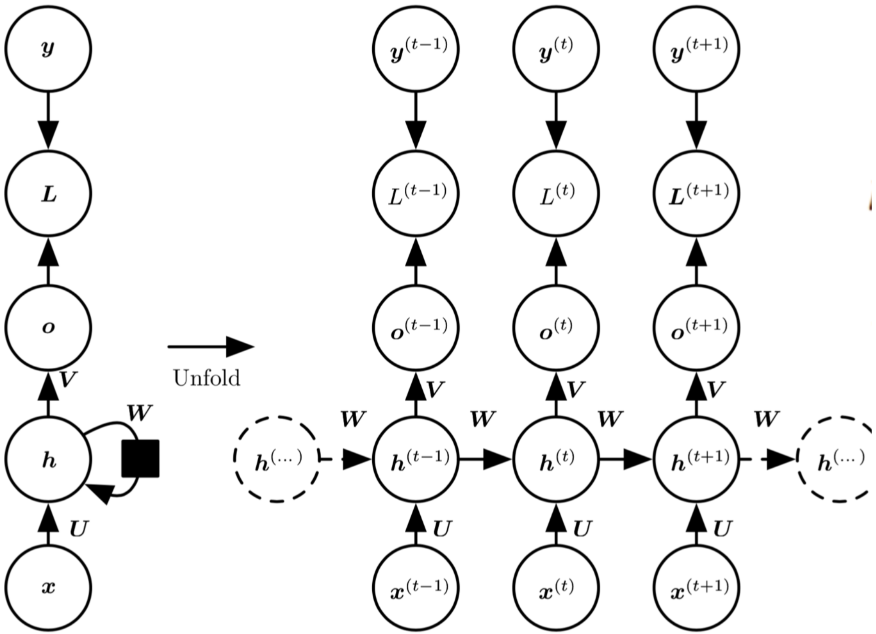
TIME-STEPS: the memory do you want to include in your network.
If you want your network to have memory of 60 characters, this number should be 60.

Recurrent Neural Network (RNN) : sharing Weights

Hence we have two data flows: **Forward in space + time** propagation: **2 projections pe r layer activation** Last time-step includes the context of our decisions recursively

W, U, V shared across all steps

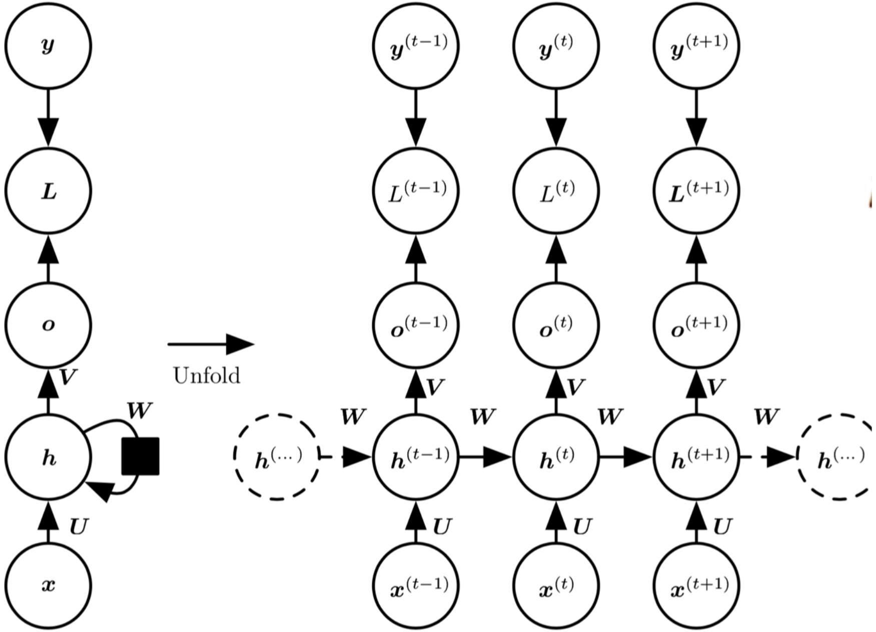
$$
\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}, \\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}), \\
\boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}, \\
\hat{\boldsymbol{y}}^{(t)} &= \mathrm{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}
$$



Recurrent Neural Network (RNN) : parameters

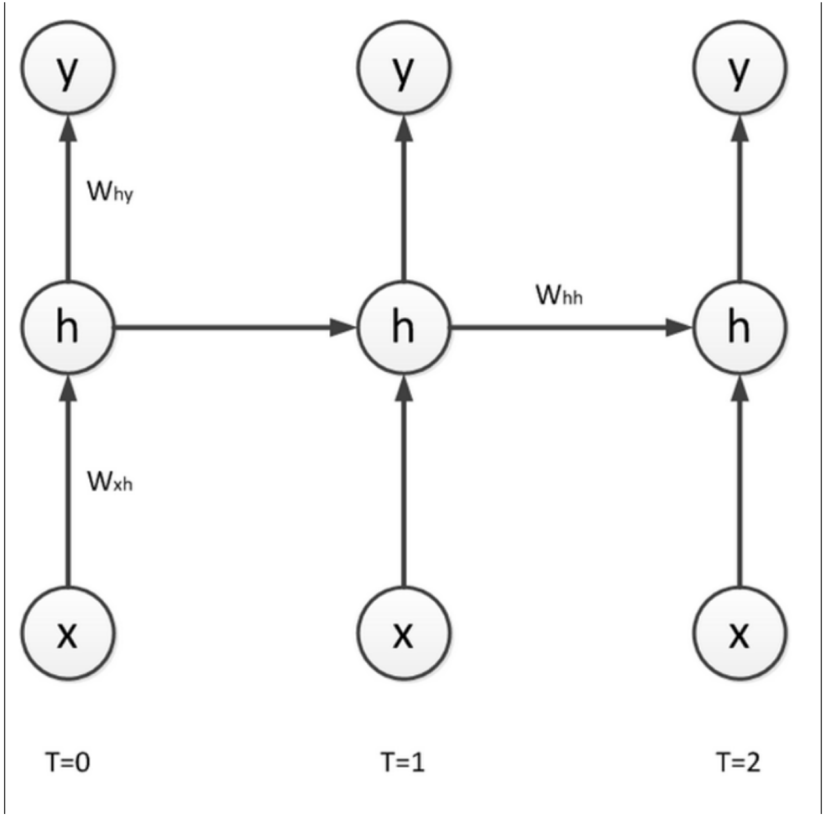
Allow to build specific connections capturing ”history”

$$
\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}, \\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}), \\
\boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}, \\
\hat{\boldsymbol{y}}^{(t)} &= \mathrm{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}
$$

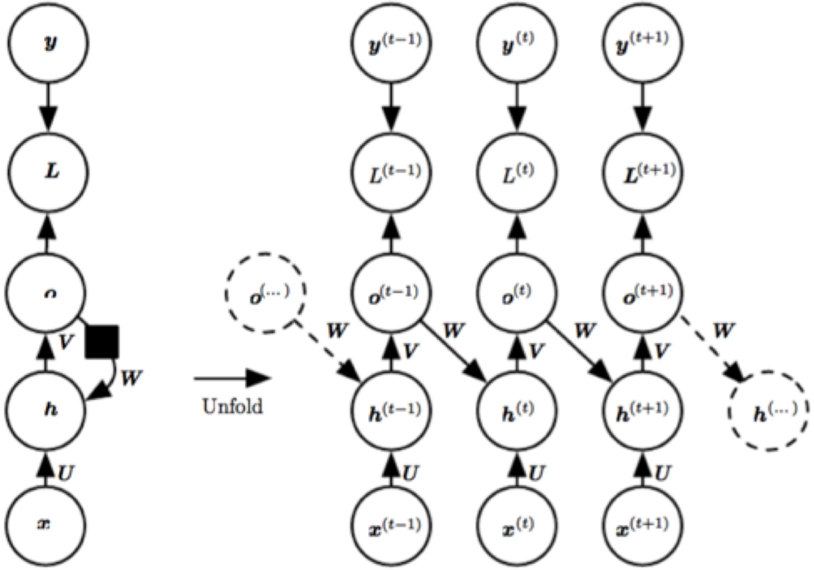


Quick Exercise

- The figure shows a RNN with one input unit x, one logistic hidden unit h, and one linear output unit y. The Network parameters are Wxh=-0,1, Whh=0.5 and Why=0.25, hbias=0.4 and ybias=0.0. The input takes the values 18, 9, -8 at time steps 0,1 and 2.
- 1-Compute the hidden value h0
- 2-Compute the output value y1
- 3-Compute the output value y2



Alternatives of recurrence



Training a RNN I: BPTT

- Backpropagation through time (BPTT): The training algorithm for updating network weights to minimize error including time

Remember BackPropagation

1. Present a training input pattern and propagate it through the network to get an output.
2. Compare the predicted outputs to the expected outputs and calculate the error.
3. Calculate the derivatives of the error with respect to the network weights.
4. Adjust the weights to minimize the error.
5. Repeat.

BPTT I: Loss

$$
\begin{aligned}
& L\left(\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\tau)}\}, \{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(\tau)}\}\right) \\
= & \sum_t L^{(t)} \\
= & -\sum_t \log p_{\text{model}}\left(y^{(t)} \mid \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t)}\}\right),
\end{aligned}
$$

The total loss for a given sequence x(t)

$$
\frac{\partial L}{\partial L^{(t)}} = 1.
$$

Our goal is to calculate the gradients of the error with respect to our parameters U, W and V and and then learn good parameters using Stochastic Gradient Descent.

BPTT II: backward in time and in space

$$(\nabla_{\boldsymbol{o}^{(t)}} L)_i = \frac{\partial L}{\partial o_i^{(t)}} = \frac{\partial L}{\partial L^{(t)}} \frac{\partial L^{(t)}}{\partial o_i^{(t)}} = \hat{y}_i^{(t)} - \mathbf{1}_{i=y^{(t)}}.$$

Gradient on the outputs at time step t

Backward in time

Backward in space

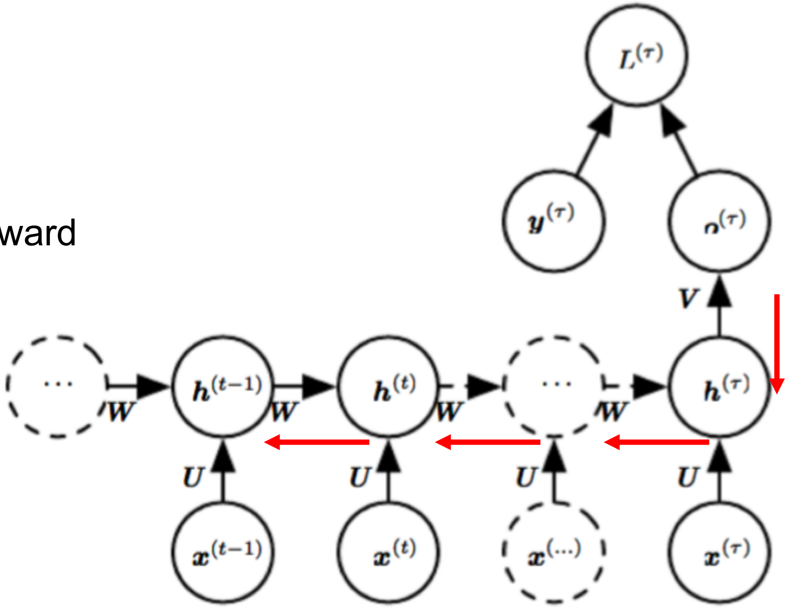
$$\nabla_{\boldsymbol{h}^{(t)}} L = \left(\frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}}\right)^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L)$$

$$= \boldsymbol{W}^{\top} \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t+1)}\right)^2\right) (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \boldsymbol{V}^{\top} (\nabla_{\boldsymbol{o}^{(t)}} L),$$

BPTT III: gradient of the Loss

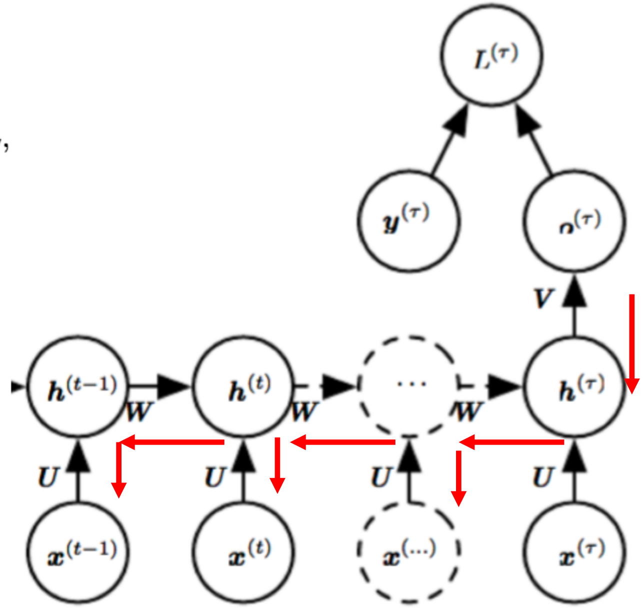
$$\nabla_{\boldsymbol{h}^{(\tau)}} L = \boldsymbol{V}^\top \nabla_{\boldsymbol{o}^{(\tau)}} L.$$

Start at the end of the sequence, going backward

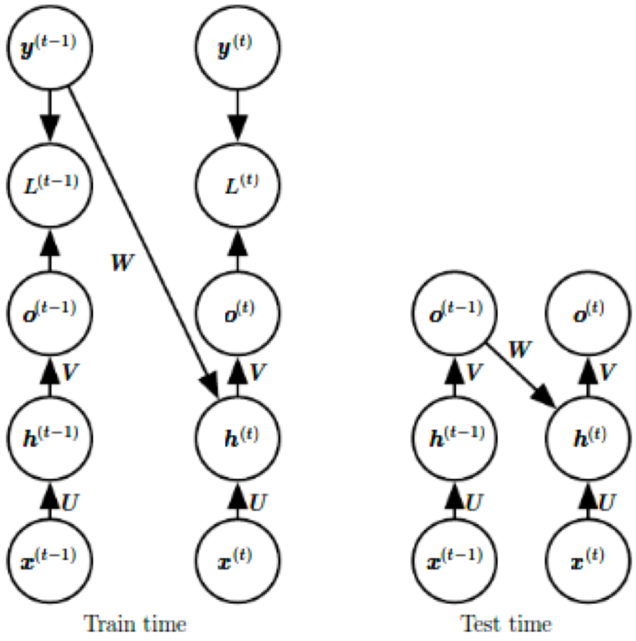


BPTT IV: gradient on the remaining parameters

$$
\begin{aligned}
\nabla_{\boldsymbol{c}} L &= \sum_t \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{c}}\right)^\top \nabla_{\boldsymbol{o}^{(t)}} L = \sum_t \nabla_{\boldsymbol{o}^{(t)}} L, \\
\nabla_{\boldsymbol{b}} L &= \sum_t \left(\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{b}^{(t)}}\right)^\top \nabla_{\boldsymbol{h}^{(t)}} L = \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) \nabla_{\boldsymbol{h}^{(t)}} L, \\
\nabla_{\boldsymbol{V}} L &= \sum_t \sum_i \left(\frac{\partial L}{\partial o_i^{(t)}}\right) \nabla_{\boldsymbol{V}^{(t)}} o_i^{(t)} = \sum_t \left(\nabla_{\boldsymbol{o}^{(t)}} L\right) \boldsymbol{h}^{(t)\top}, \\
\nabla_{\boldsymbol{W}} L &= \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}}\right) \nabla_{\boldsymbol{W}^{(t)}} h_i^{(t)} \\
&= \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) \left(\nabla_{\boldsymbol{h}^{(t)}} L\right) \boldsymbol{h}^{(t-1)\top}, \\
\nabla_{\boldsymbol{U}} L &= \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}}\right) \nabla_{\boldsymbol{U}^{(t)}} h_i^{(t)} \\
&= \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) \left(\nabla_{\boldsymbol{h}^{(t)}} L\right) \boldsymbol{x}^{(t)\top},
\end{aligned}
$$



Note on teacher forcing



Pro's

No need BPTT
Parallel Training

Con's

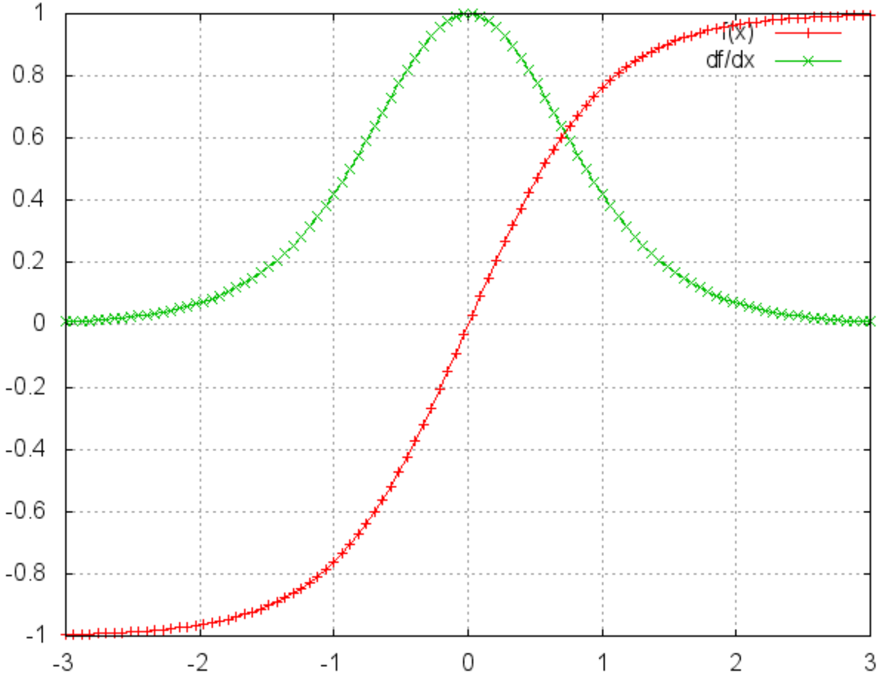
Exposure bias (but in practice is not so relevant)
Recurrence from the output, not internal state

Vanishing gradient

Vanishing gradient I

- **During training gradients explode/vanish easily because of depth-in-time** → Exploding/Vanishing gradients!

Vanishing gradient II



tanh and derivative.

- Traditional activation functions such as hyperbolic tangent have gradients in the range (-1.1) and backpropagation computes gradients by the chain rule
    - The vanishing (and exploding) gradient problem is caused by the repeated use of the recurrent weight matrix in RNN
        - This has the effect of multiplying n of these small numbers to compute gradients
        - This means that the gradient (error signal) decreases exponentially with n

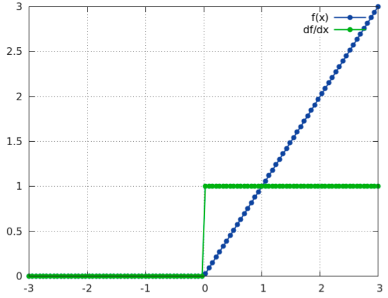
$$
\begin{aligned}
\nabla_{\boldsymbol{W}} L &= \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}} \right) \nabla_{\boldsymbol{W}^{(t)}} h_i^{(t)} \\
&= \sum_t \operatorname{diag}\left(1 - \left(\boldsymbol{h}^{(t)} \right)^2 \right) \left(\nabla_{\boldsymbol{h}^{(t)}} L \right) \boldsymbol{h}^{(t-1)^\top},
\end{aligned}
$$

Question

- Do FNNs with several hidden layers suffer from vanishing gradient problem?

Standard Solutions

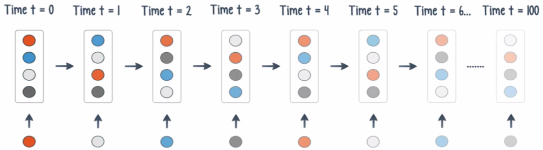
- Proper initialization of Weight Matrix
- Regularization of outputs or Dropout
- Use of ReLU Activations as it’s derivative is either 0 or 1



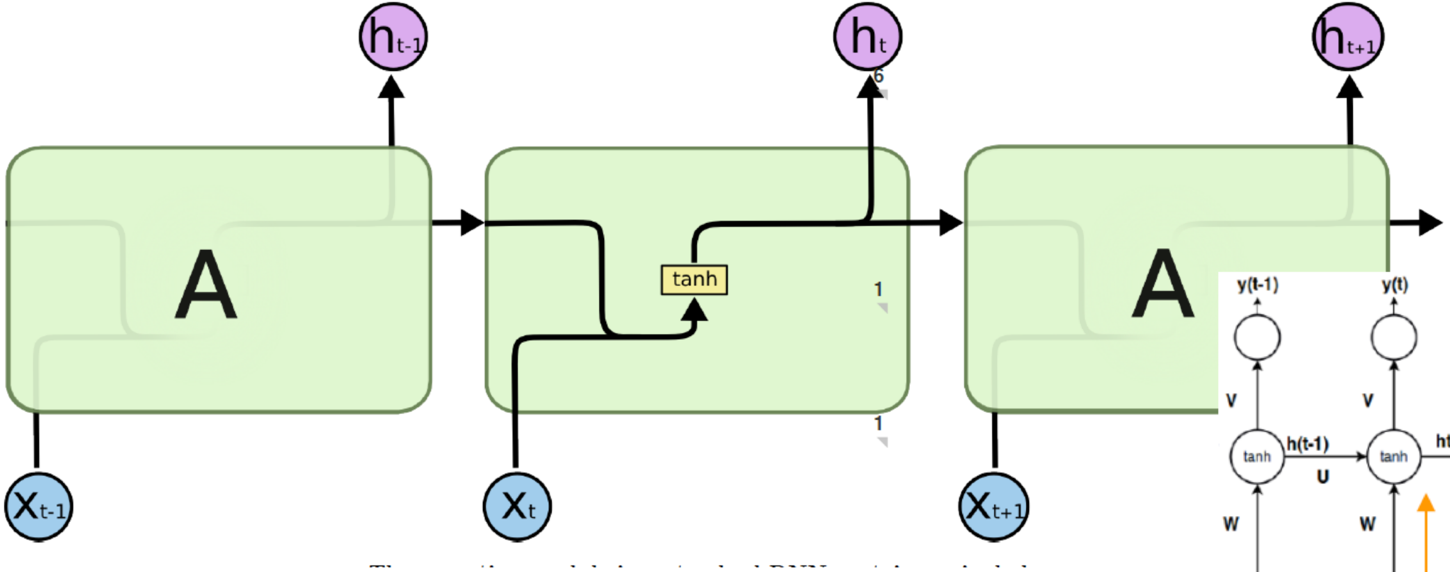
SLIDES FROM: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

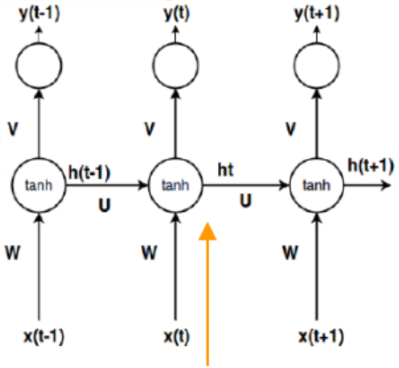
Standard RNN

Decay of information through time

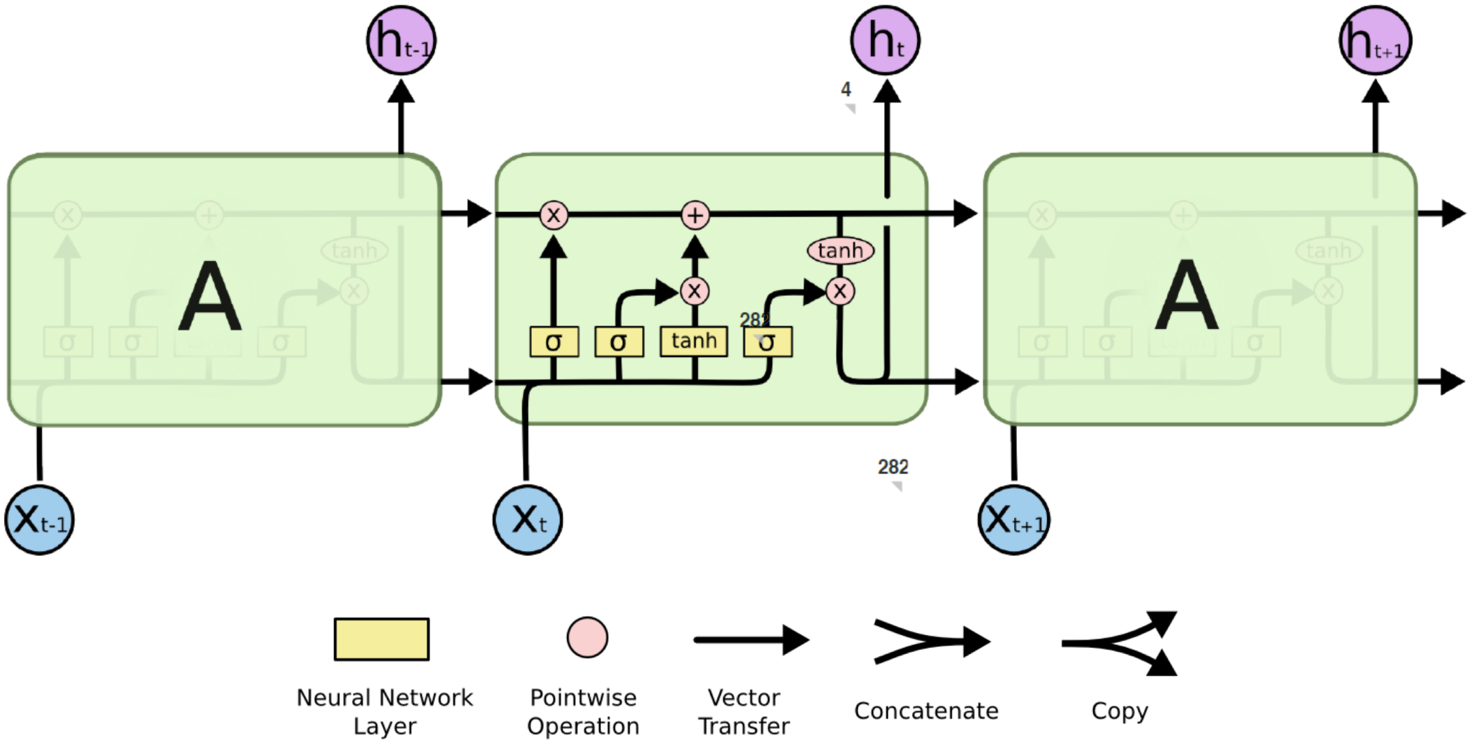


https://www.nextbigfuture.com/2016/03/recurrent-neural-nets.html





Long-Short Term Memory (LSTM)



Gating method

1. Change the way in which past information is kept → create the notion of **cell state, a memory unit that keeps long-term information in a safer way by protecting it from recursive operations**

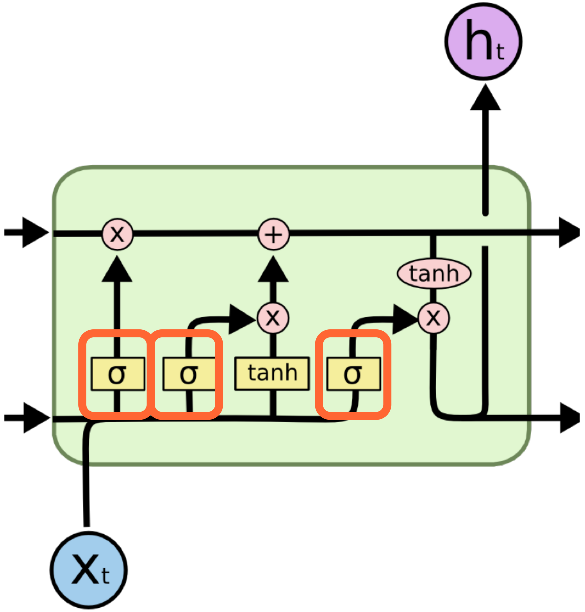
2. **Make every RNN unit able to decide whether the current time-step information matters or not**, to accept or discard (optimized reading mechanism)

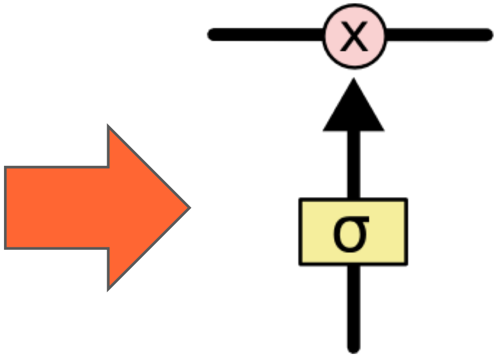
3. **Make every RNN unit able to forget whatever may not be useful anymore** by clearing that info from the cell state (optimized clearing mechanism)

4. **Make every RNN unit able to output the decisions whenever it is ready to do so** (optimized output mechanism)

Long Short-Term Memory (LSTM)

Three **gates** are governed by *sigmoid* units (btw [0,1]) define the control of in & out information..





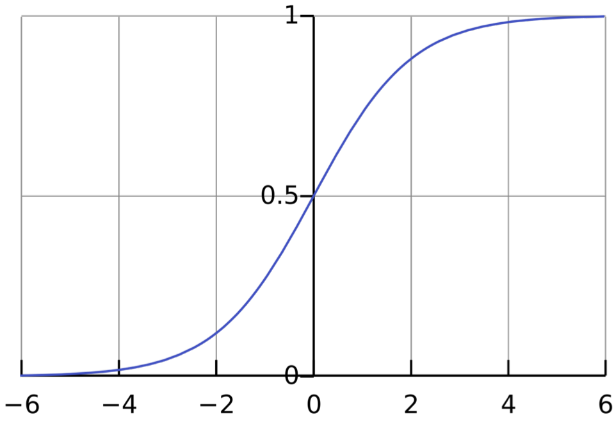
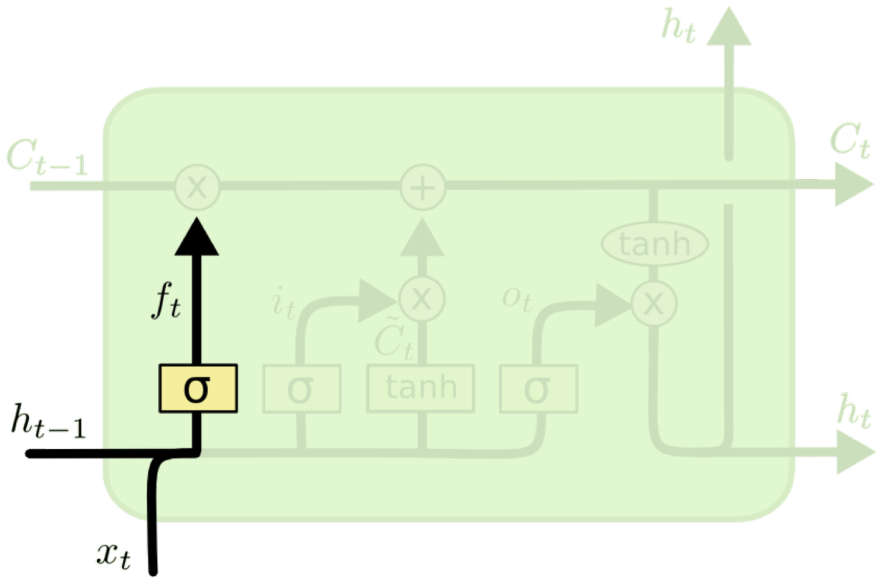


Figure: Cristopher Olah, [“Understanding LSTM Networks”]() (2015)

slide credit: Xavi Giro

Forget Gate



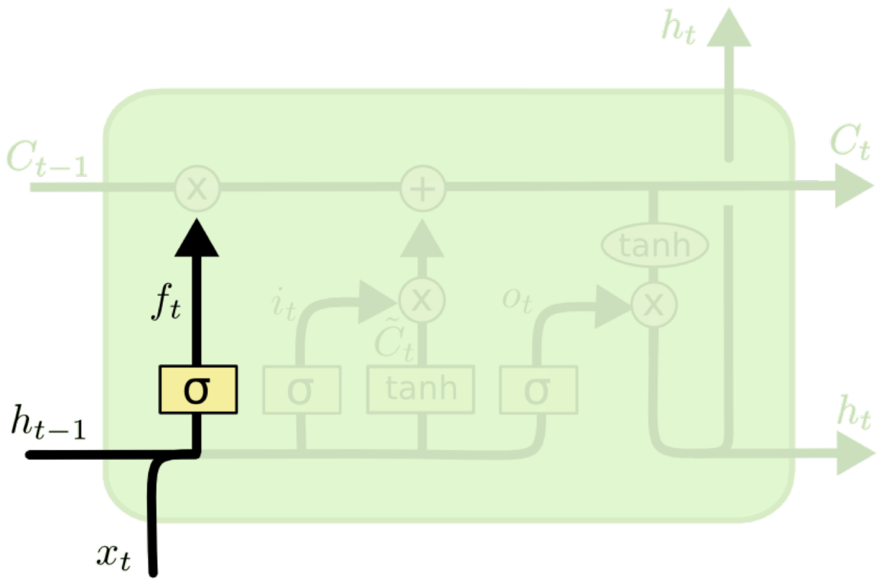
What part of memory to "forget" – zero means forget this bit

Forget Gate:

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: Alberto Montes

Forget Gate: Example



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

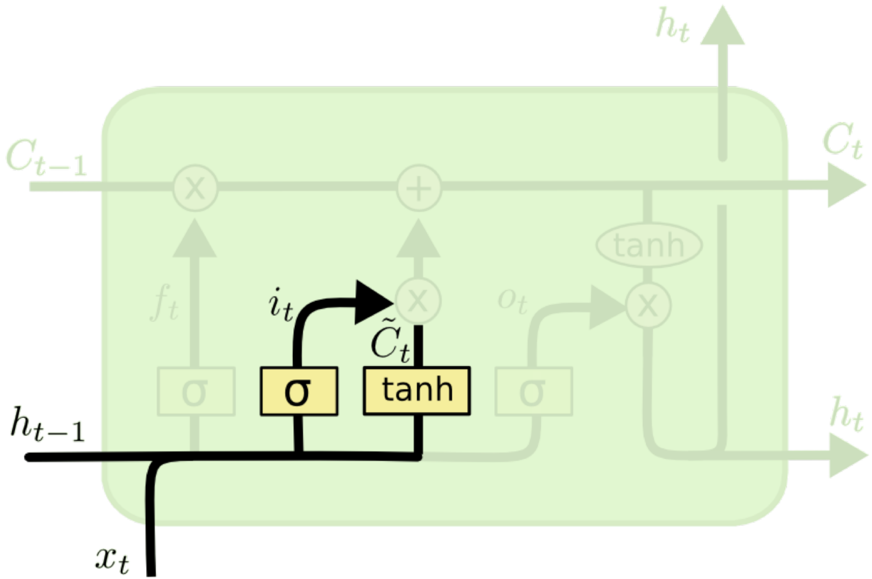
LANGUAGE MODELING

Joan es un chico activo y Anna es una chica calmada

Forget about ”male” gender

Figure: Cristopher Olah, “Understanding LSTM Networks” (2015) / Slide: Alberto Montes

Input Gate



What bits to insert into the next states

Input Gate Layer

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

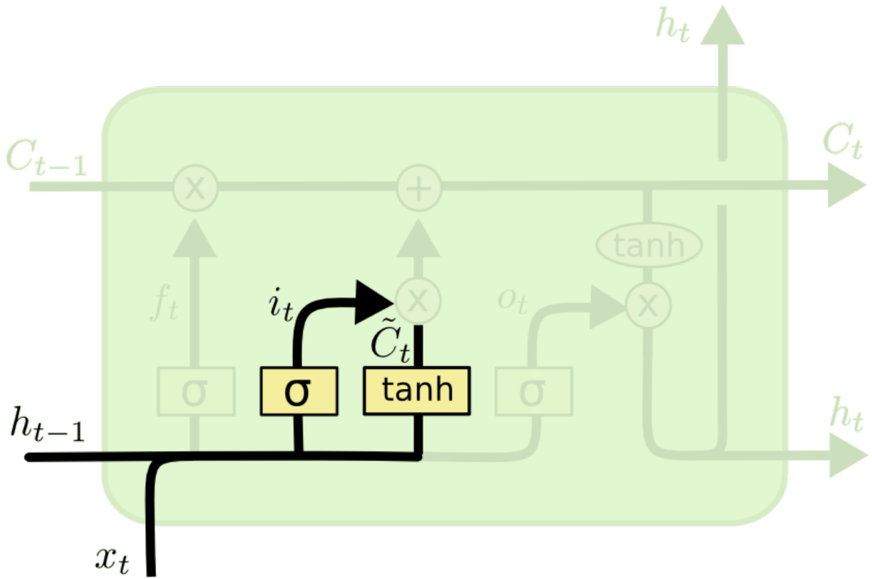
New contribution to cell state

$$\tilde{C}_t = \underbrace{\tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)}_{\text{Classic neuron}}$$

What content to store into the next state

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) /

Input Gate: Example



Input Gate Layer

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

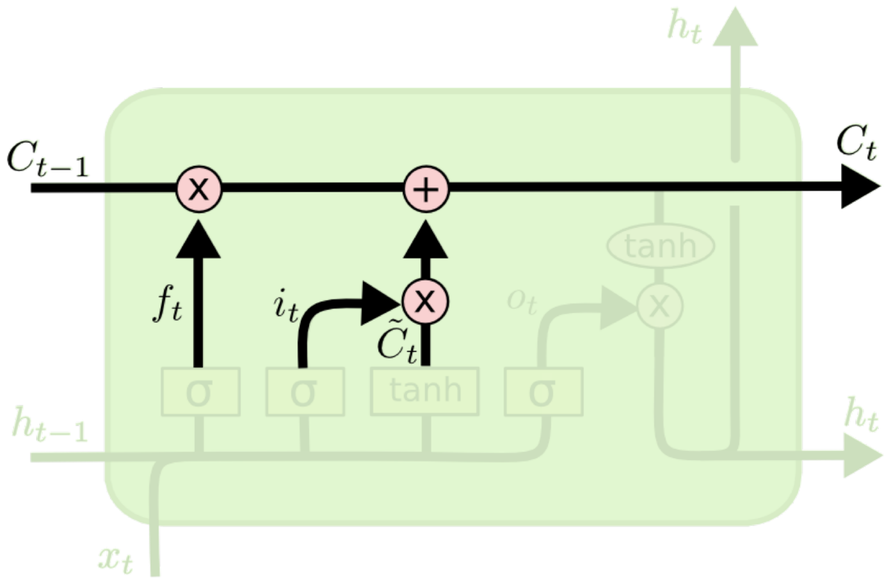
LANGUAGE MODELING

Joan es un chico activo y Anna es una chica calmada

Input about ”female” gender

Figure: Cristopher Olah, “Understanding LSTM Networks” (2015) / Slide: Alberto Montes

Update Cell State



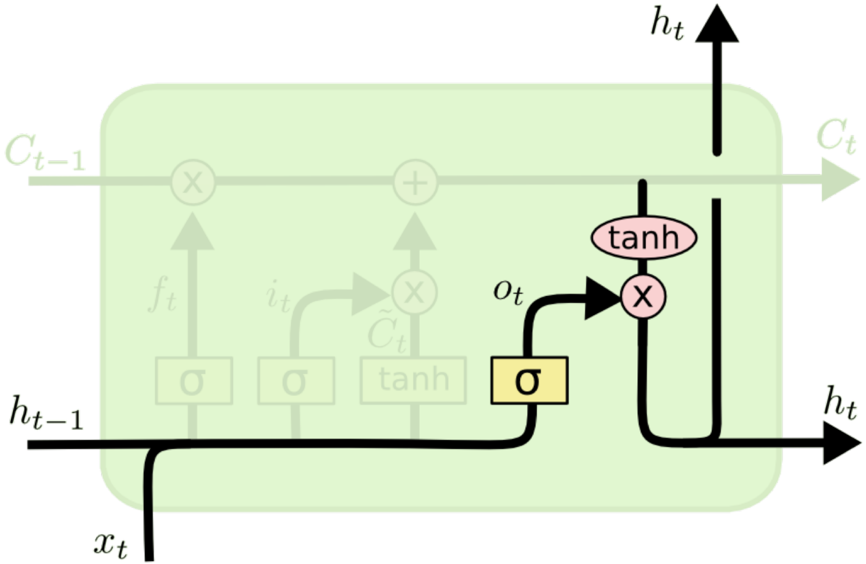
Next memory cell content – mixture of not-forgotten part of previous cell and insertion

Update Cell State (memory):

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

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: Alberto Montes

Output Gate



What part of cell to output

Output Gate Layer

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

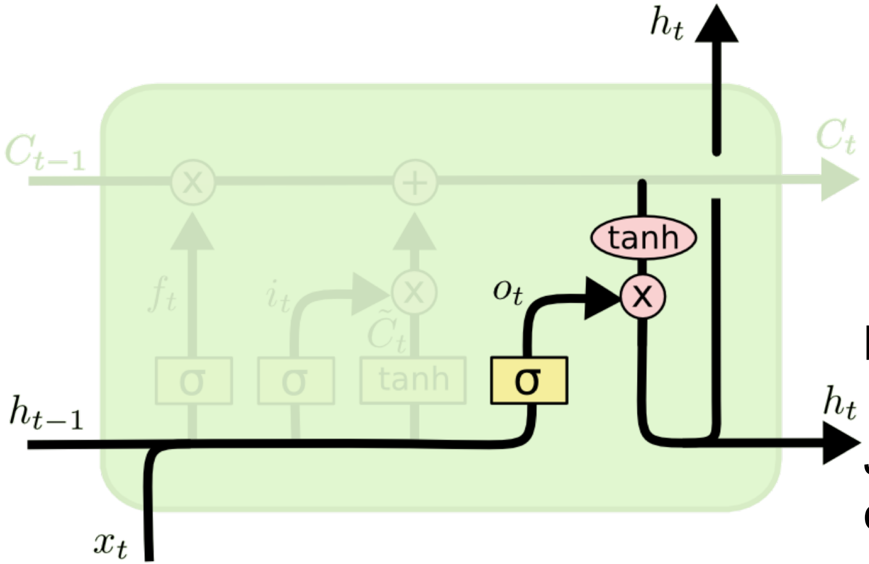
Output to next layer

$$h_t = o_t * \tanh\left(C_t\right)$$

tanh maps bits to [-1,+1] range

Figure: Cristopher Olah, "Understanding LSTM Networks" (2015) / Slide: A

Output Gate: Example



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

LANGUAGE MODELING

Joan es un chico activo y Anna es una chica calmada

Info relevant for a verb? : ”female”, “singular”, “3rd person”

Figure: Cristopher Olah, “Understanding LSTM Networks” (2015) / Slide: Alberto Montes

Implementing an LSTM

For $t = 1,\ldots,T$:

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

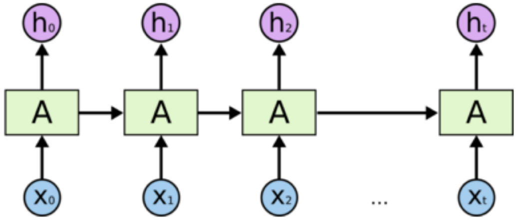
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

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

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

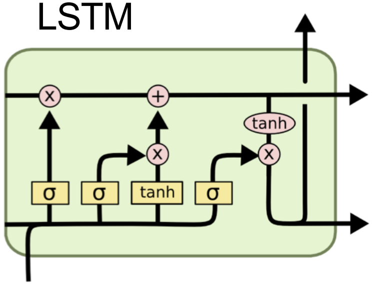
$$h_t = o_t * \tanh\left(C_t\right)$$

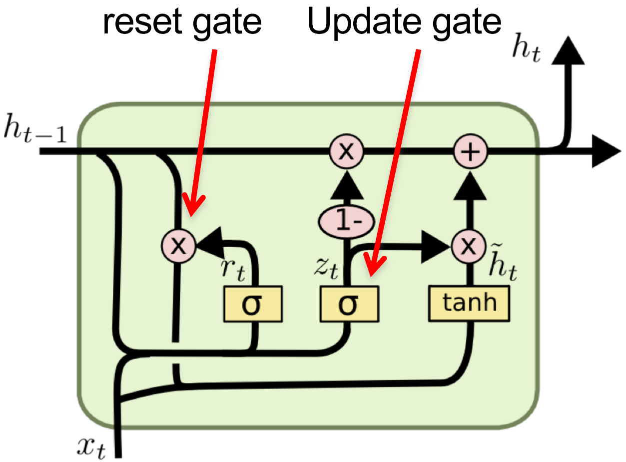


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

GRU – gated recurrent unit

(more compression)





$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

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

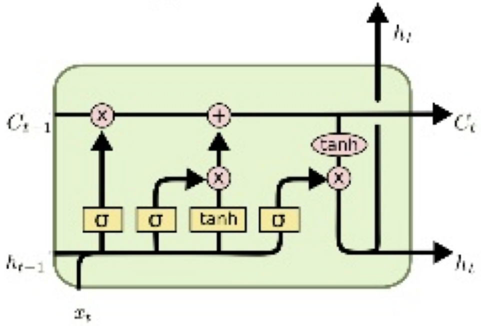
X,*: element-wise multiply

It combines the forget and input into a single update gate.
It also merges the cell state and hidden state. This is simpler than LSTM. There are many other variants too.

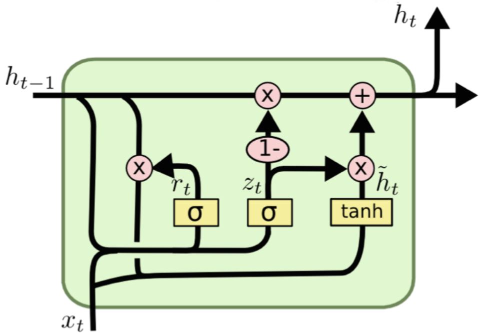
Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." AMNLP 2014.

LSTM and GRU

- LSTM [Hochreiter&Schmidhuber97]



- GRU [Cho+14]



GRUs also takes x_t and h_{t-1} as inputs. They perform some calculations and then pass along h_t. What makes them different from LSTMs is that GRUs don't need the cell layer to pass values along. The calculations within each iteration insure that the h_t values being passed along either retain a high amount of old information or are jump-started with a high amount of new information.

Visual Comparison FNN, Vanilla RNNs and LSTMs

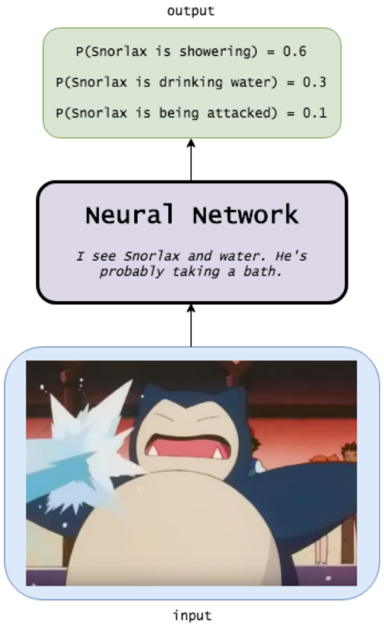


Image src http://blog.echen.me/2017/05/30/exploring-lstms/

Visual Comparison FNN, Vanilla RNNs and LSTMs

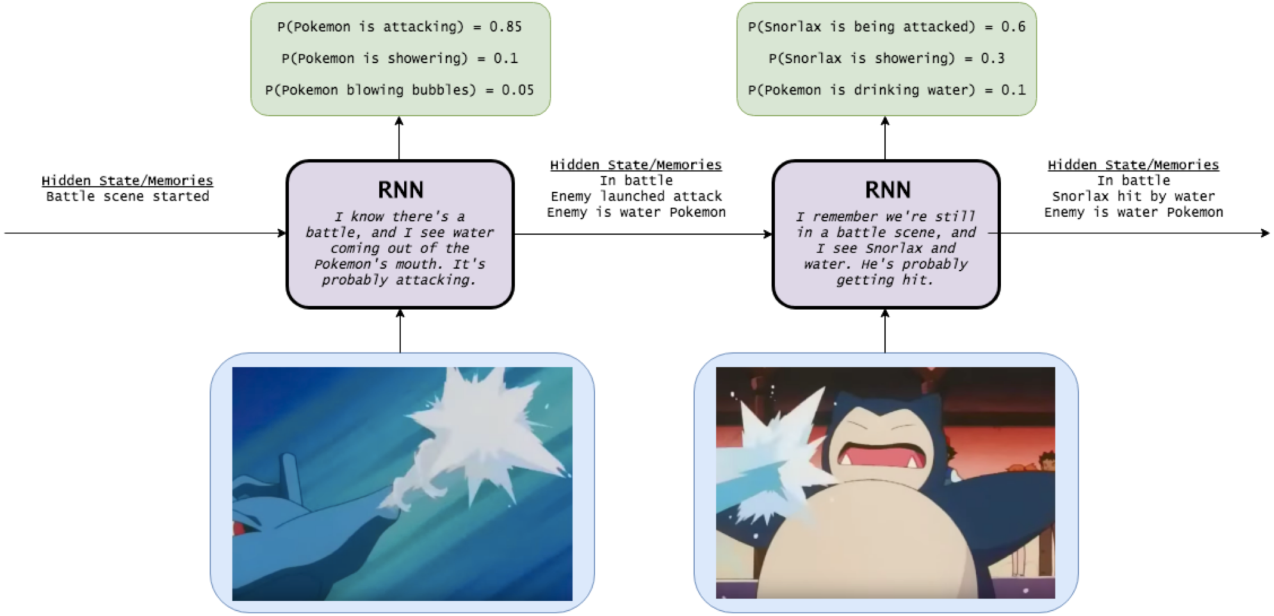
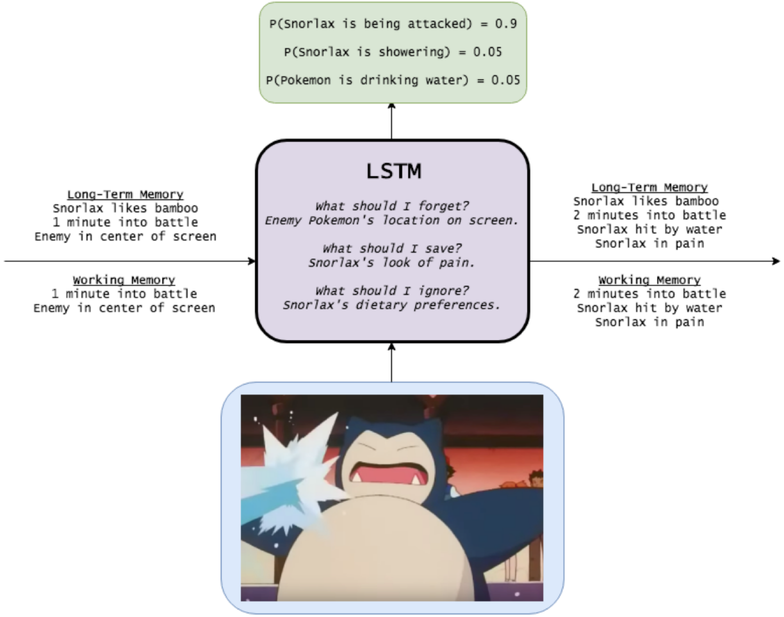


Image src http://blog.echen.me/2017/05/30/exploring-lstms/

Visual Comparison FNN, Vanilla RNNs and LSTMs



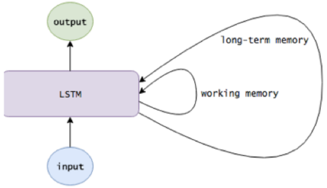
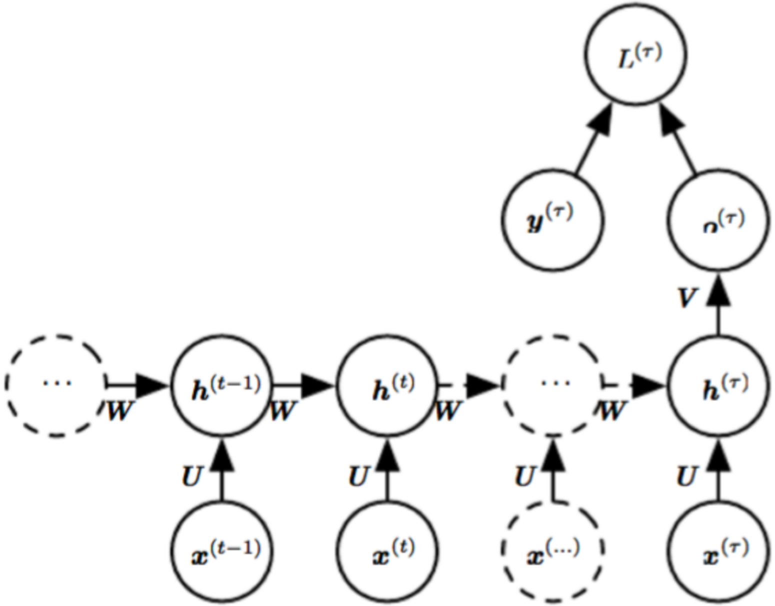


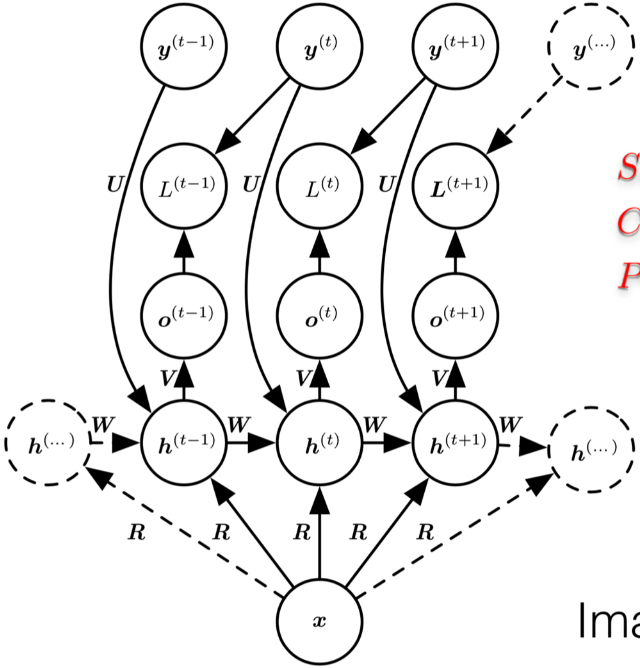
Image src http://blog.echen.me/2017/05/30/exploring-lstms/

Variants of RNNs

Sequence to vector



Vector to sequence



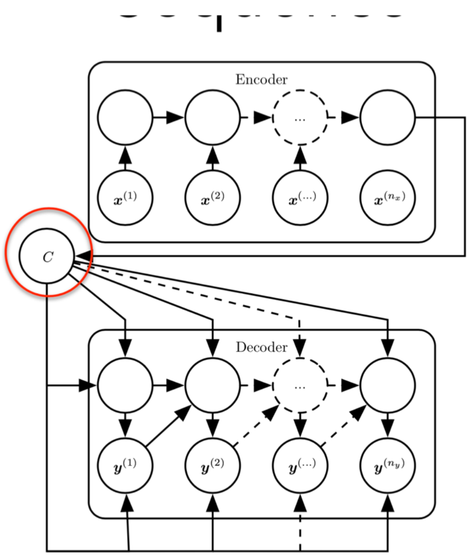
Sequence $y^{(t)}$:
Current time step INPUT, and
Previous time step TARGET

Image as extra input

Exercise

- Draw the computational graph of a sequence to sequence model

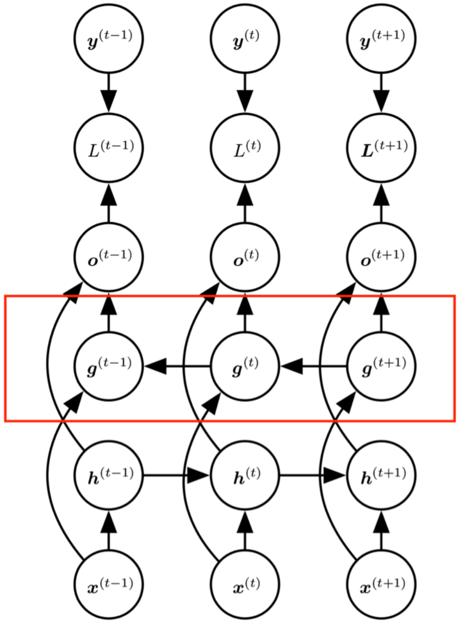
Sequence to sequence



HOT!!!

To learn the context Variable C which represents a semantic summary of the input sequence, and for later decoder RNNN

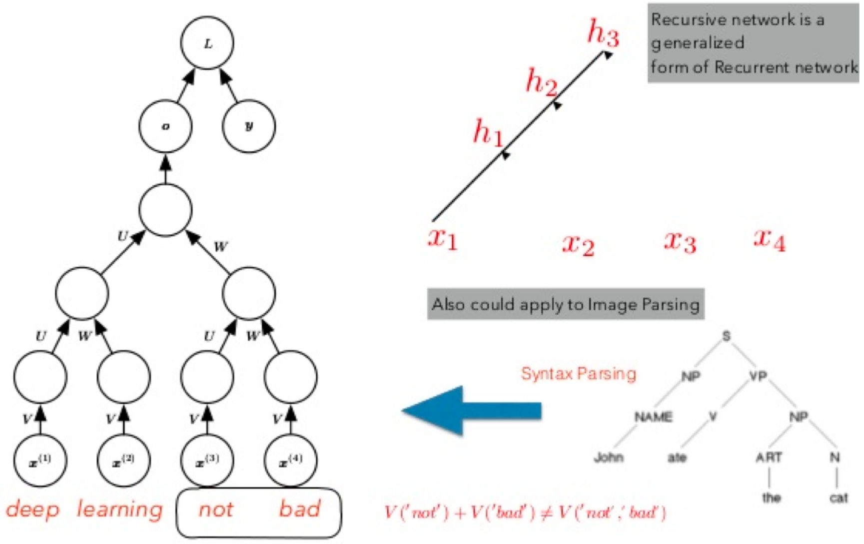
Bidirectional RNNs



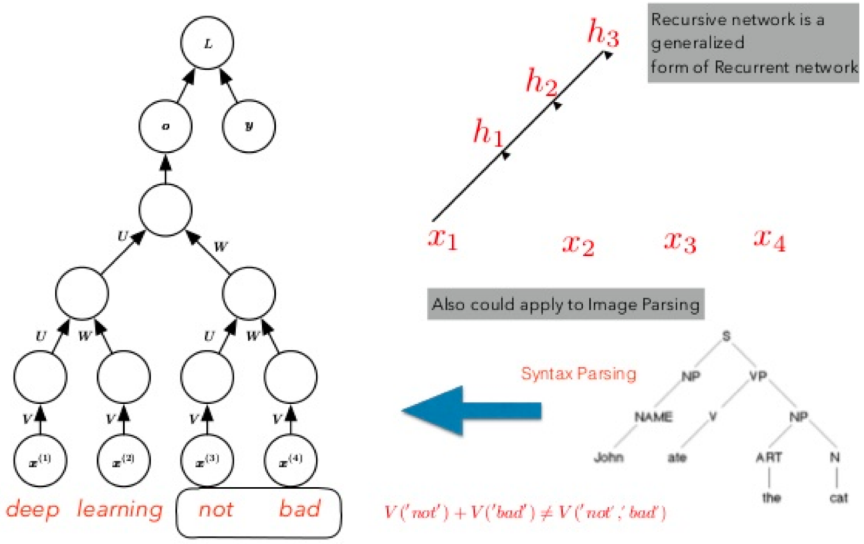
Question

- You have a pet dog whose mood is heavily dependent on the current and past few days' weather. You've collected data for the past 365 days on the weather, which you represent as a sequence as x<1>,...,x<365>. You've also collected data on your dog's mood, which you represent as y<1>,...,y<365>. You'd like to build a model to map from x→y. Should you use a Unidirectional RNN or Bidirectional RNN for this problem?

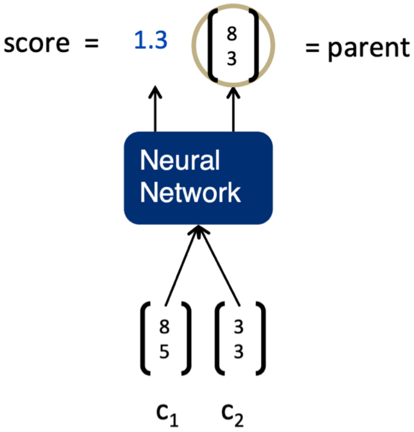
Recursive networks



Recursive networks



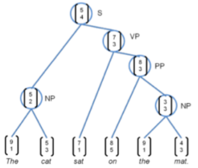
Recursive Neural Network Definition



$$\text{score} = U^{\mathsf{T}} p$$

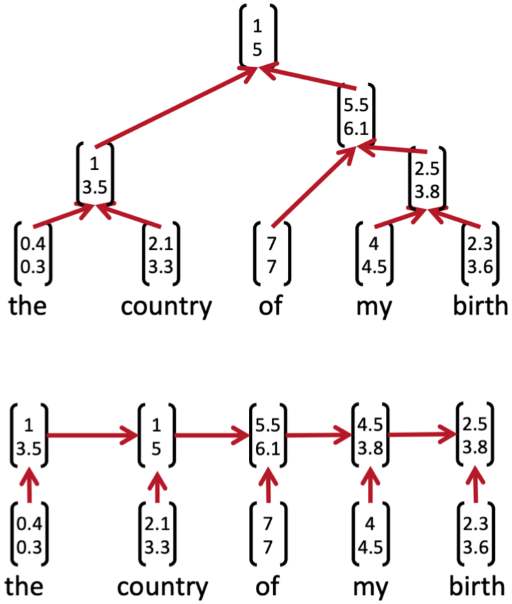
$$\boxed{p = \tanh\left(W\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right),}$$

Same *W* parameters at all nodes of the tree



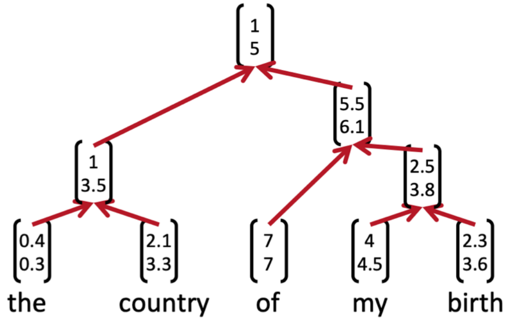
Images from Stanford CS224N

Recursive vs recurrent

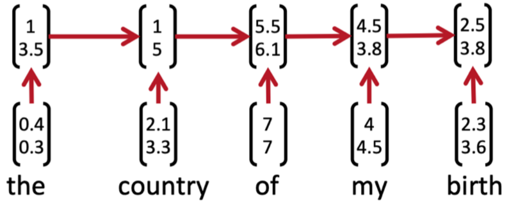


Recursive vs Recurrent

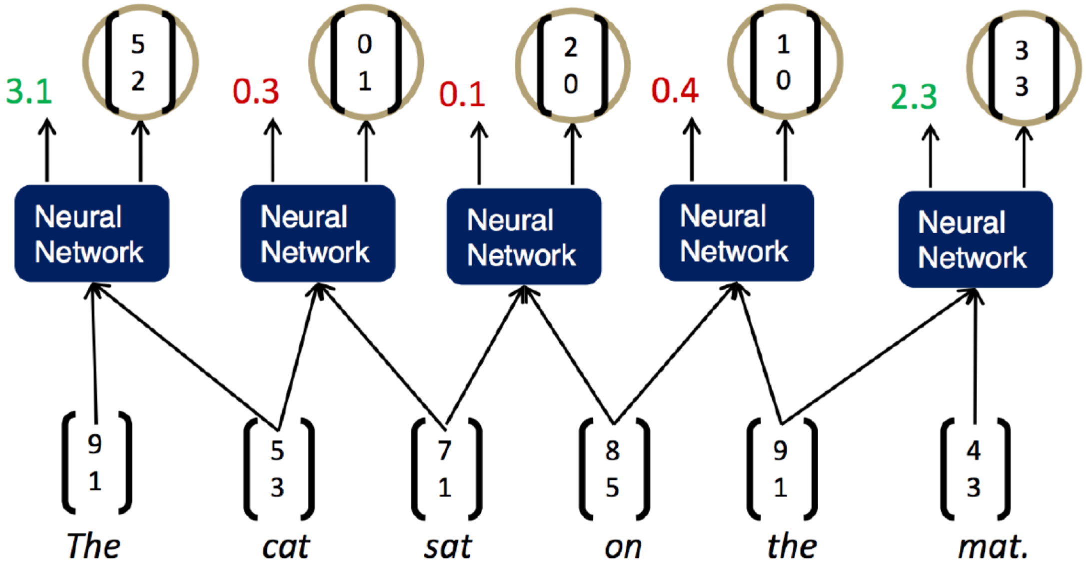
- Recursive neural nets require a parser to get tree structure



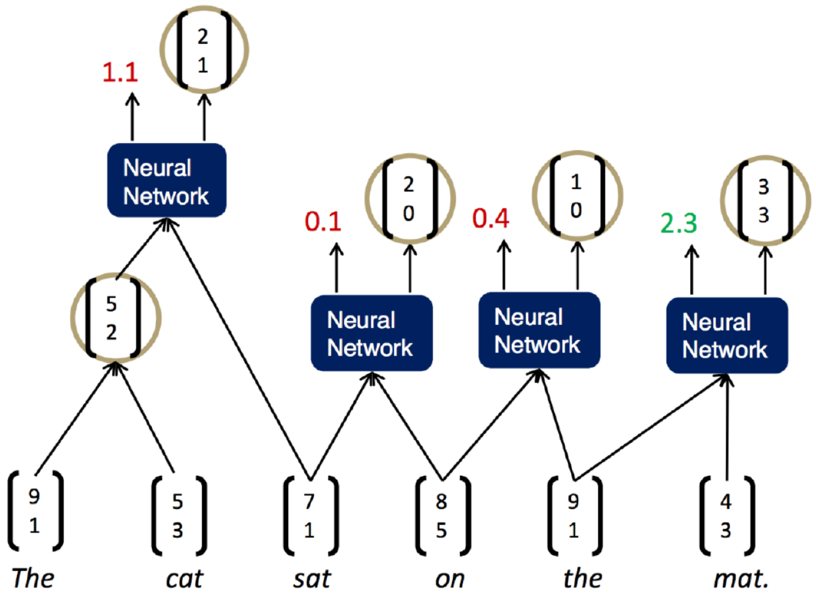
- Recurrent neural nets cannot capture phrases without prefix context and often capture too much of last words in final vector



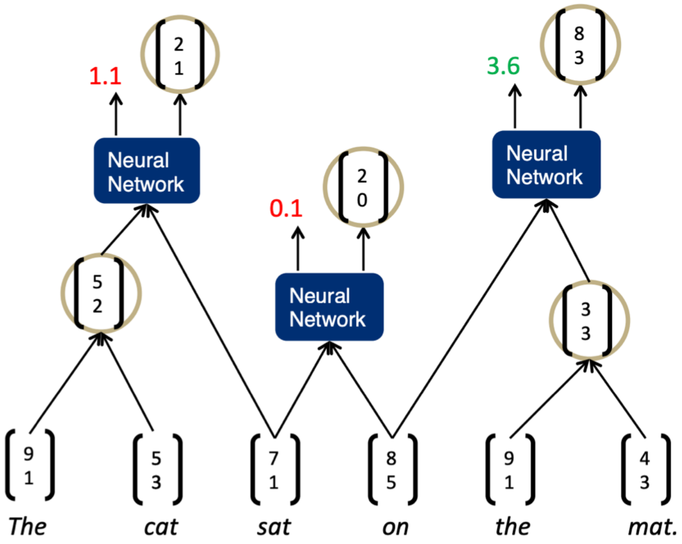
Each time step merges two adjacent nodes I



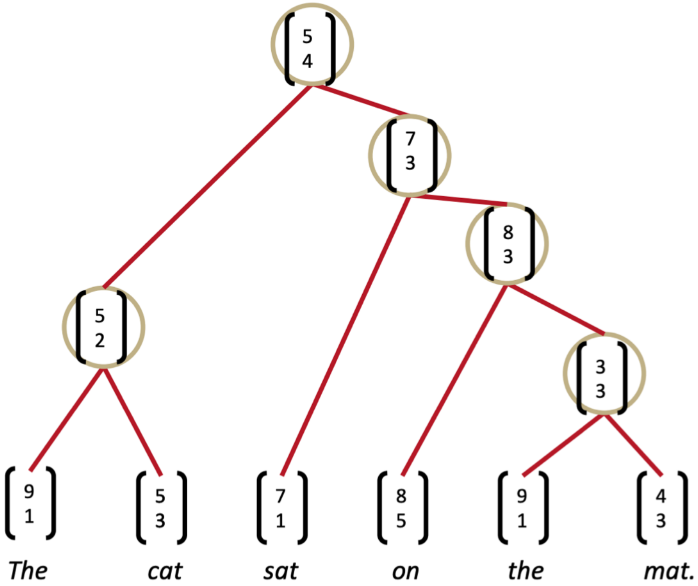
Each time step merges two adjacent nodes II



Each time step merges two adjacent nodes III



And so on...



Final tree score

- The score of a tree is computed by the sum of the parsing decisión scores at each node

  - s(x,y)= $\sum_{n \in nodes(y)} s_n$;

  - x is sentence; y is parse tree

Training objective

- $J = \sum_i s(x_i, y_i) - \max_{y \in A(x_i)} \left(s(x_i, y) - \Delta(y, y_i) \right);$

- The loss $\Delta(y, y_i)$ penalizes all incorrect decisions

- Structure search for A(x) was greedy (join best nodes each time)
  (alternative: beam search)

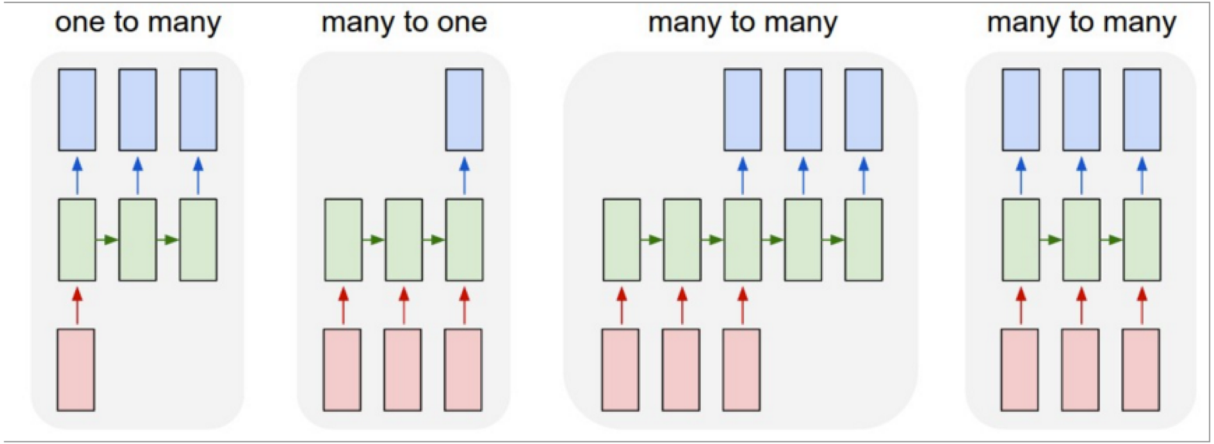
Backpropagation Through Structure

- Principally the same as general backpropagation
- Three differences resultinng from the recursion and tree structure:
- 1. Sum derivatives of W from all nodes (like RNN)
- 2. Split derivatives at each node (for tree)
  - During forward prop, the parent is computed using 2 children
  - Hence, the errors need to be computed wrt each of them
- 3. Add error messages from parent + node itself
  - What came up (fprop) must come down (bprop)
  - Total error messages = error messages from parent + error message from own score

Some Fun LSTM Examples

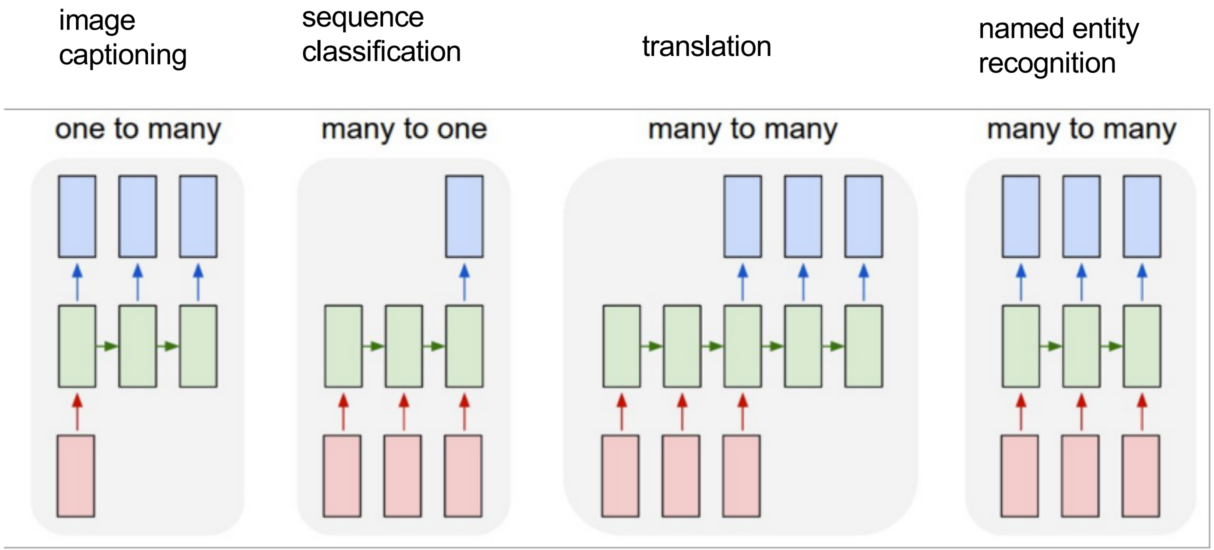
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Question: Can LSTMs be used for other sequence tasks: which ones?



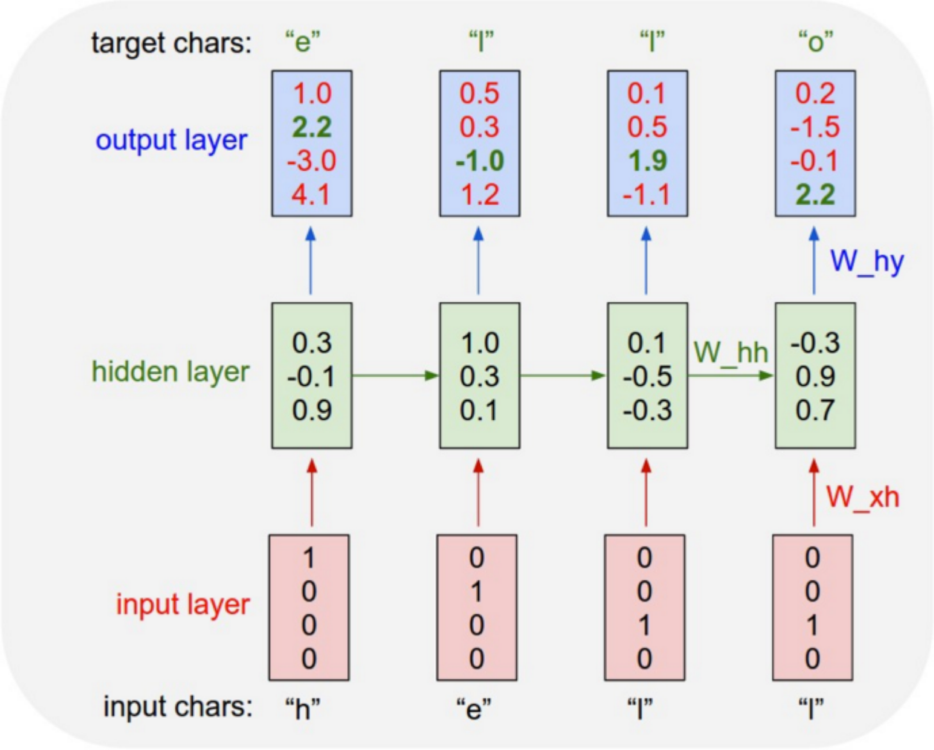
http://karpathy.github.io/2015/05/21/rnn-effectiveness/

LSTMs can be used for other sequence tasks: which ones?



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character-level language model



Test time:

- pick a seed character sequence
- generate the next character
- then the next
- then the next …

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character-level language model

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character-level language model

```
First Citizen:
Nay, then, that was hers,
It speaks against your other service:
But since the
youth of the circumstance be spoken:
Your uncle and one Baptista's daughter.

SEBASTIAN:
Do I stand till the break off.

BIRON:
Hide thy head.

VENTIDIUS:
He purposeth to Athens: whither, with the vow
I made to handle you.

FALSTAFF:
My good knave.
```

Yoav Goldberg:
order-10 unsmoothed
character n-grams

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character-level language model

```
MMMMM----- Recipe via Meal-Master (tm) v8.05

      Title: BARBECUE RIBS
 Categories: Chinese, Appetizers
      Yield: 4 Servings

      1 pk Seasoned rice
      1    Beer -- cut into
           -cubes
      1 ts Sugar
    3/4 c  Water
           Chopped finels,
           -up to 4 tblsp of chopped
      2 pk Yeast Bread/over

MMMMM--------------------------FILLING-
      2 c  Pineapple, chopped
    1/3 c  Milk
    1/2 c  Pecans
           Cream of each
      2 tb Balsamic cocoa
      2 tb Flour
      2 ts Lemon juice
           Granulated sugar
      2 tb Orange juice
```

```
  1 c  Sherry wheated curdup
  1    Onion; sliced
  1 ts Salt
  2 c  Sugar
1/4 ts Salt
1/2 ts White pepper, freshly ground
       Sesame seeds
  1 c  Sugar
1/4 c  Shredded coconut
1/4 ts Cumin seeds

Preheat oven to 350.  In a medium bowl, combine milk,
flour and water and then cornstarch. add tomatoes, or
nutmeg; serve.
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Character-level language model

For $\bigoplus_{n=1,\dots,m}$ where $\mathcal{L}_{m_\bullet} = 0$, hence we can find a closed subset $\mathcal{H}$ in $\mathcal{H}$ and any sets $\mathcal{F}$ on X, U is a closed immersion of S, then $U \to T$ is a separated algebraic space.

Proof. Proof of (1). It also start we get

$$S = \mathrm{Spec}(R) = U \times_X U \times_X U$$

and the comparicoly in the fibre product covering we have to prove the lemma generated by $\coprod Z \times_U U \to V$. Consider the maps M along the set of points Sch_{fppf} and $U \to U$ is the fibre category of S in U in Section, ?? and the fact that any U affine, see Morphisms, Lemma ??. Hence we obtain a scheme S and any open subset $W \subset U$ in $Sh(G)$ such that $\mathrm{Spec}(R') \to S$ is smooth or an

$$U = \bigcup U_i \times_{S_i} U_i$$

which has a nonzero morphism we may assume that f_i is of finite presentation over S. We claim that $\mathcal{O}_{X,x}$ is a scheme where $x, x', s'' \in S'$ such that $\mathcal{O}_{X,x'} \to \mathcal{O}'_{X',x'}$ is separated. By Algebra, Lemma ?? we can define a map of complexes $\mathrm{GL}_{S'}(x'/S'')$ and we win. $\square$

To prove study we see that $\mathcal{F}|_U$ is a covering of $\mathcal{X}'$, and $\mathcal{T}_i$ is an object of $\mathcal{F}_{X/S}$ for $i > 0$ and $\mathcal{F}_p$ exists and let $\mathcal{F}_i$ be a presheaf of $\mathcal{O}_X$-modules on $\mathcal{C}$ as a $\mathcal{F}$-module. In particular $\mathcal{F} = U/\mathcal{F}$ we have to show that

$$\widetilde{M}^\bullet = \mathcal{I}^\bullet \otimes_{\mathrm{Spec}(k)} \mathcal{O}_{S,s} - i_X^{-1}\mathcal{F})$$

is a unique morphism of algebraic stacks. Note that

$$\mathrm{Arrows} = (Sch/S)^{opp}_{fppf}, (Sch/S)_{fppf}$$

and

$$V = \Gamma(S, \mathcal{O}) \longmapsto (U, \mathrm{Spec}(A))$$

LaTeX "almost compiles"

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

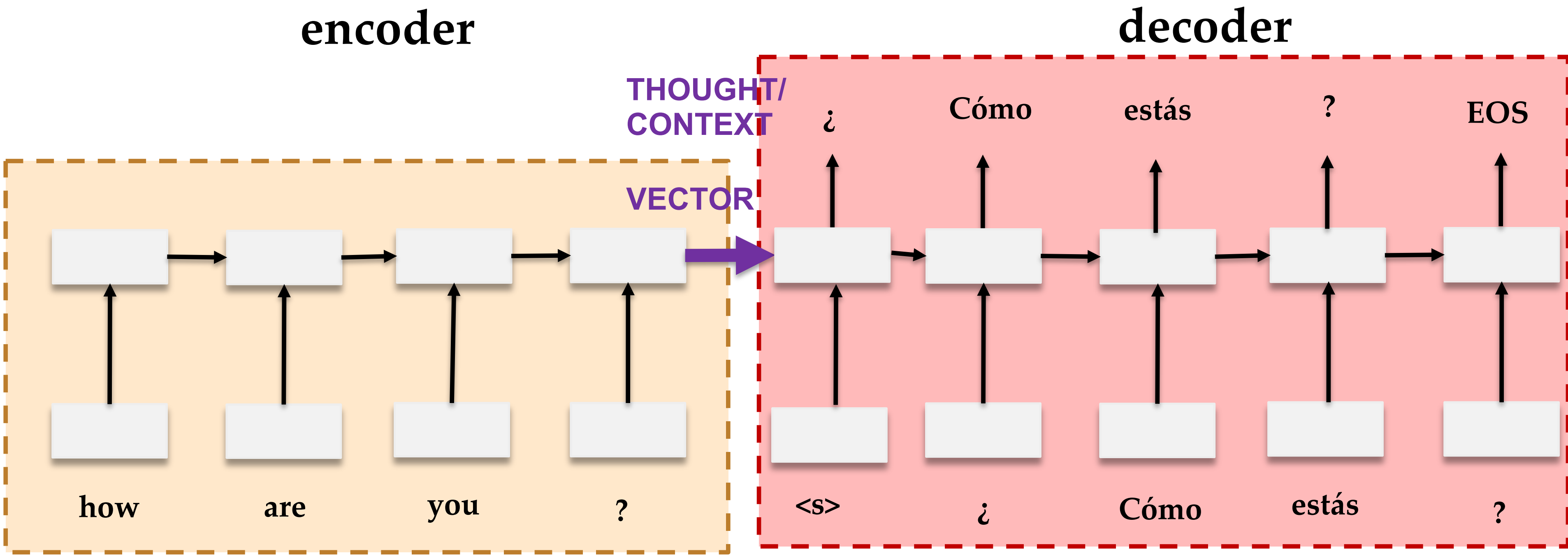
Character-level language model

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
  for (i = 0; i < blocks; i++) {
    seq = buf[i++];
    bpf = bd->bd.next + i * search;
    if (fd) {
      current = blocked;
    }
  }
```

http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Attention

Sequence-to-sequence models



Any problem with these models?

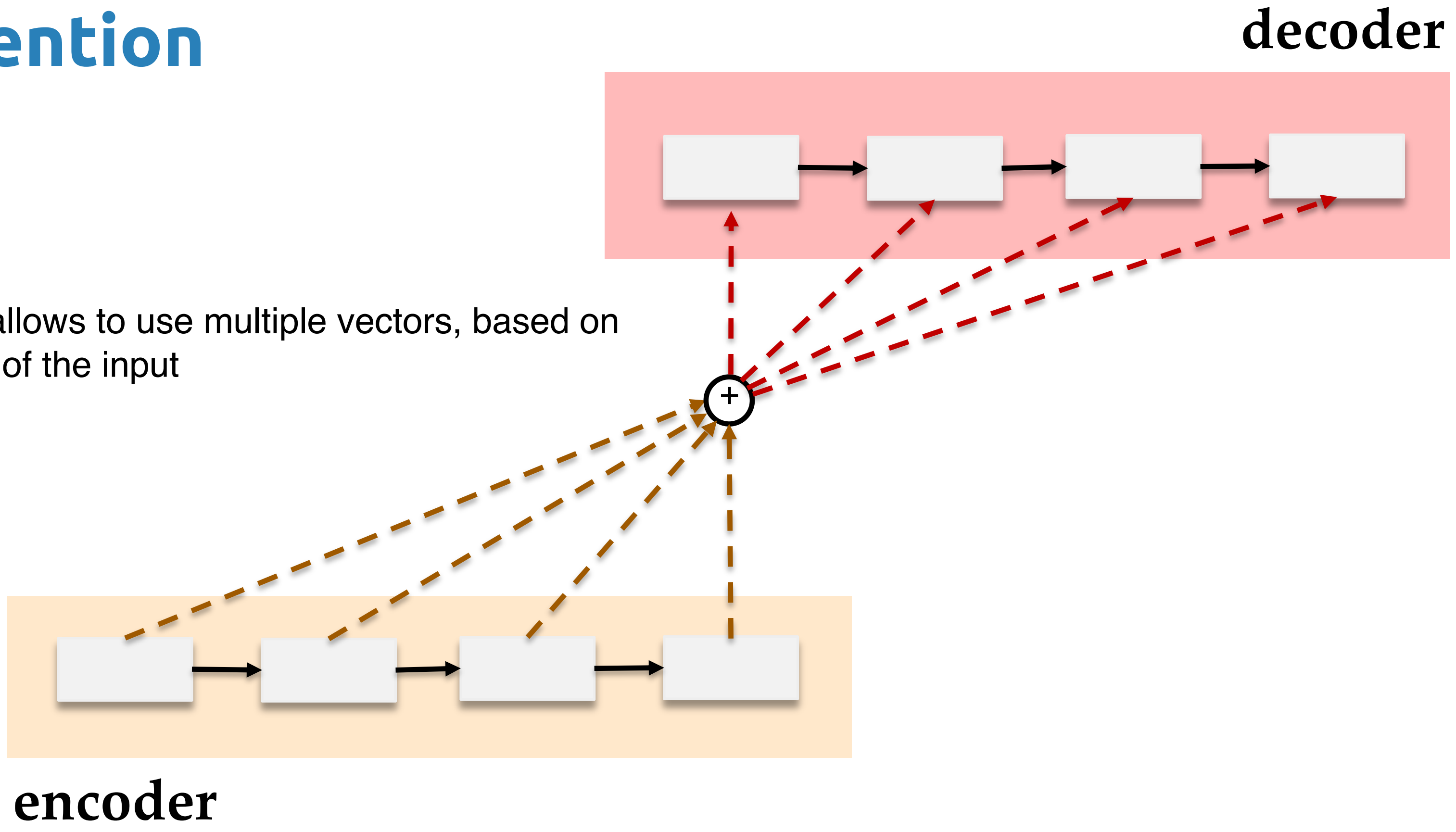
“You can’t cram the meaning of a whole %&!$ing sentence into a single $&!*ing vector!”
— Ray Mooney

- (BaHdanau et al, 2015)

Additive attention

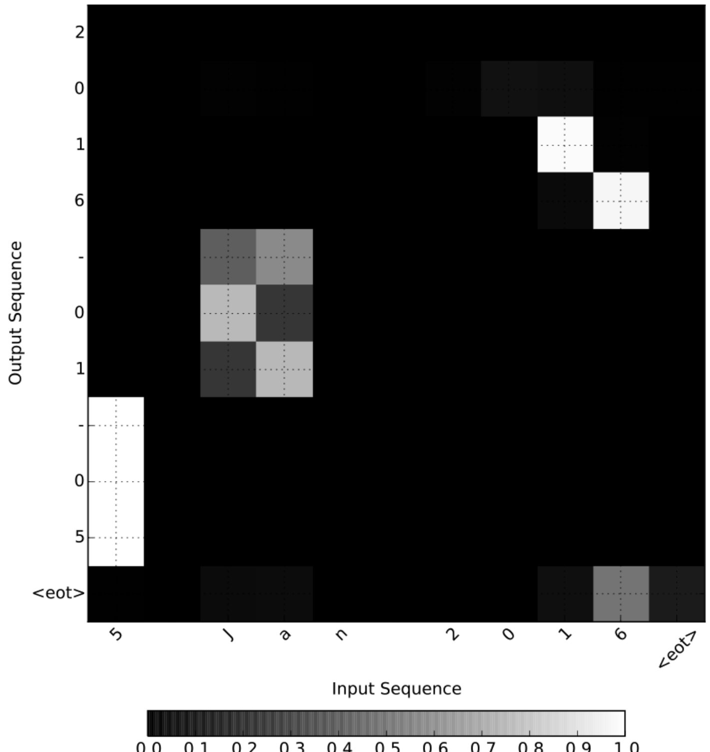
Attention

Attention allows to use multiple vectors, based on the length of the input



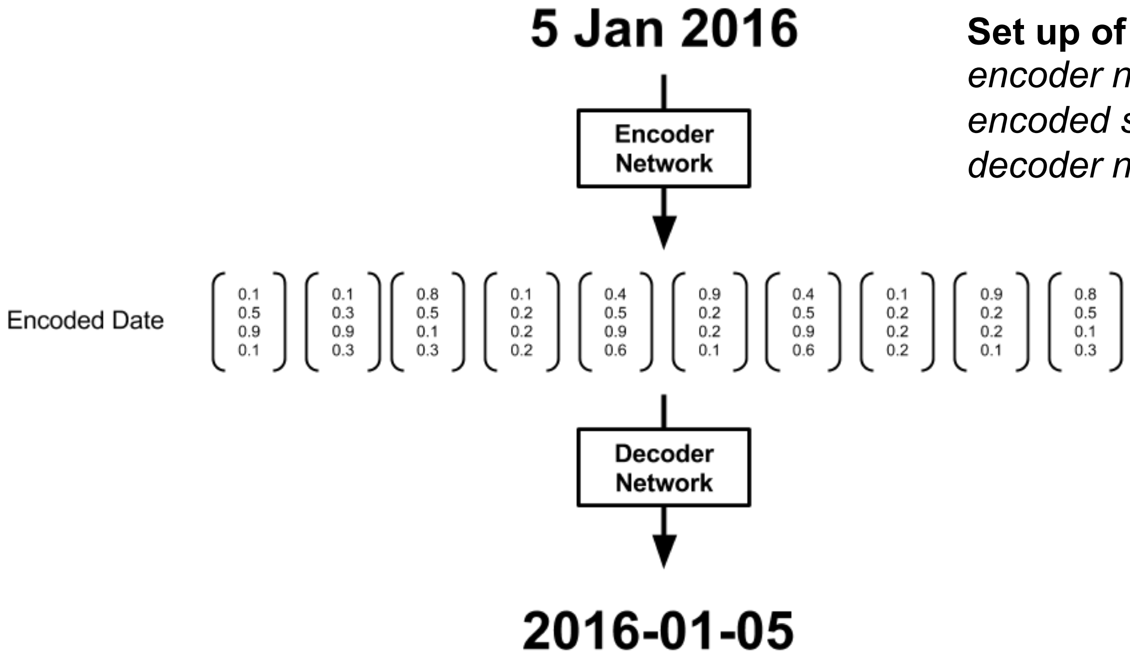
Attention Integration & Visualization Step-by-step

[from Medium]



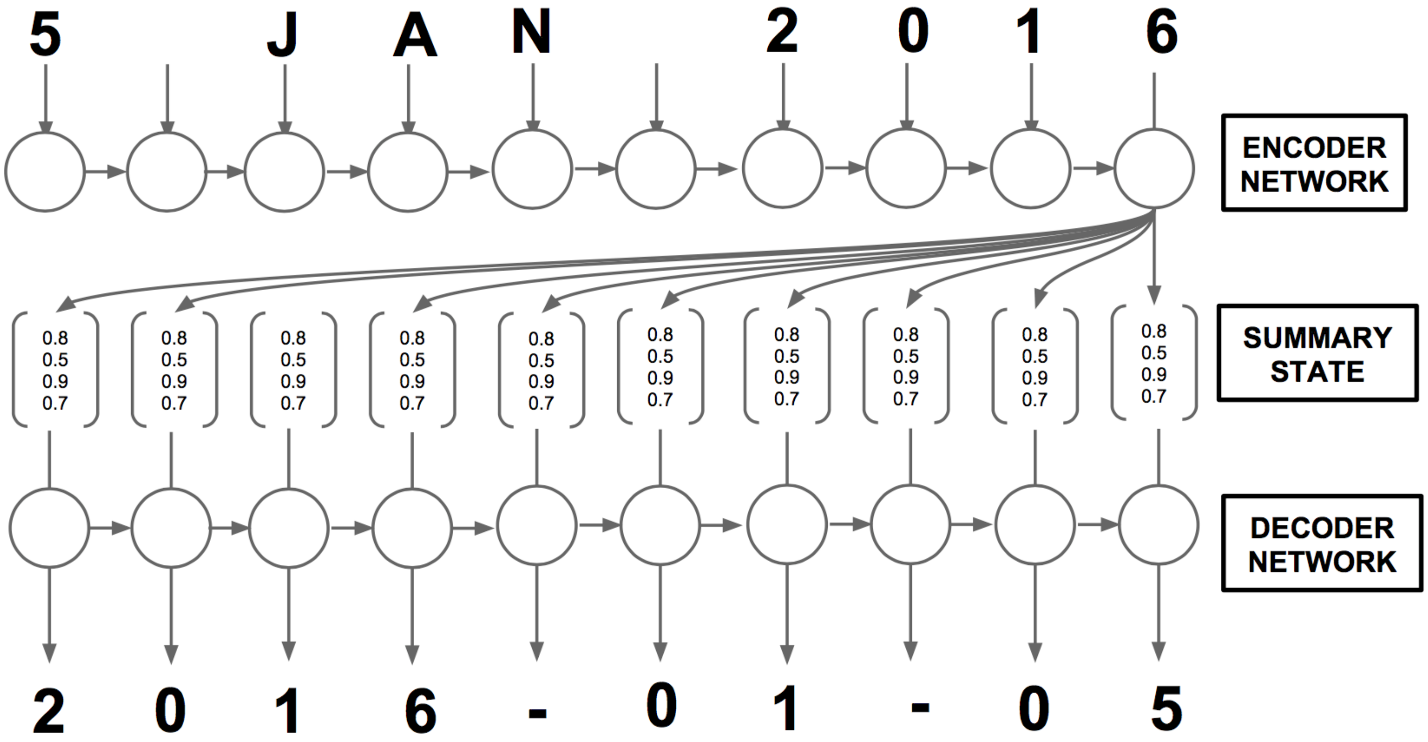
Attention map for the freeform date "5 Jan 2016". We can see that the neural network used "16" to decide that the year was 2016, "Ja" to decide that the month was 01 and the first bit of the date to decide the day of the month.

Encoder-decoder architecture set up

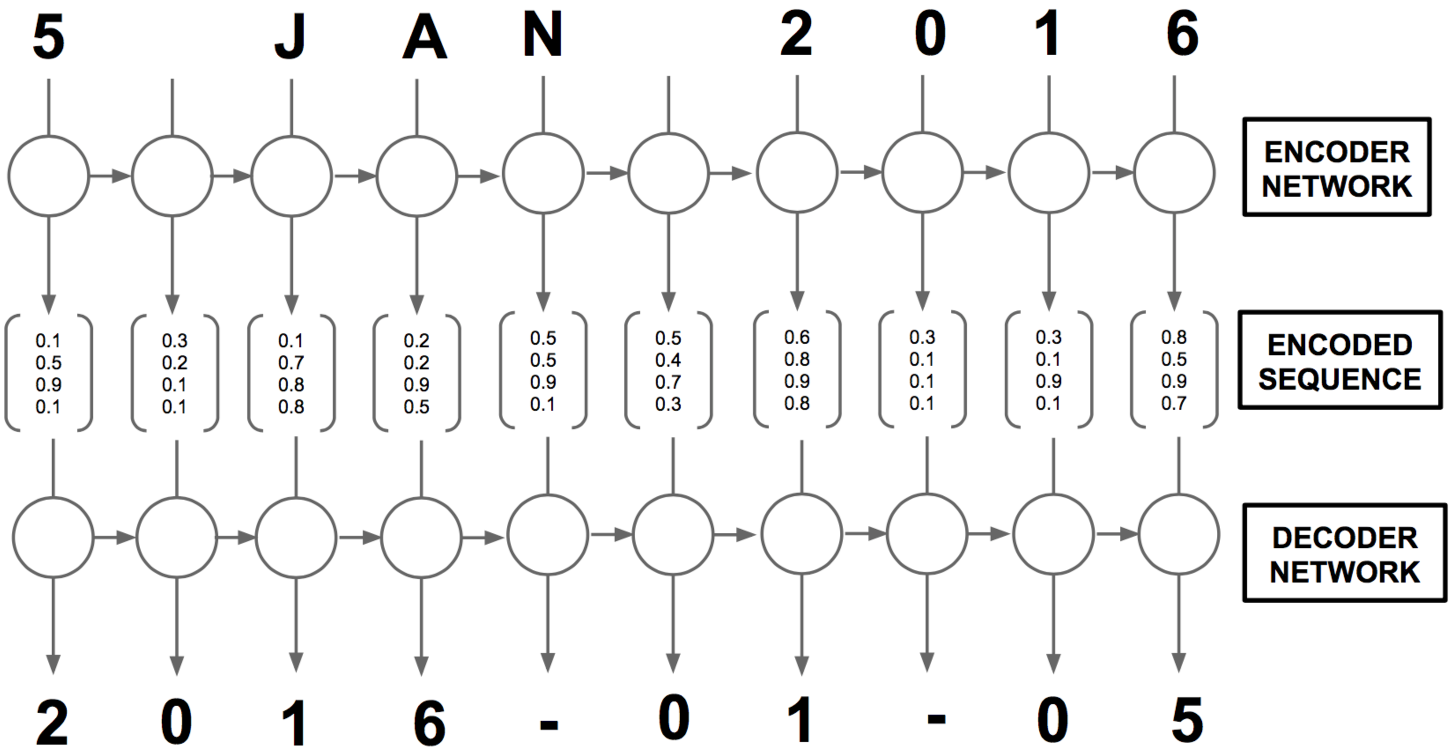


Set up of the encoder-decoder architecture. *The encoder network processes the input sequence into an encoded sequence which is subsequently used by the decoder network to produce the output.*

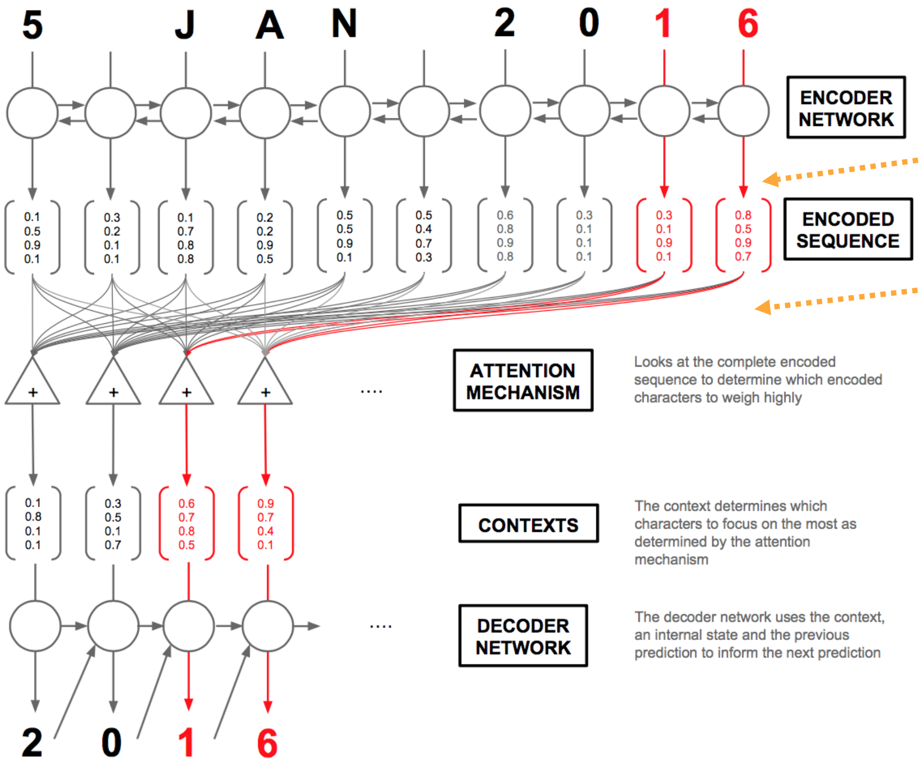
Use of a summary state in the encoder-decoder



Use of the complete encoded sequence in the decoder



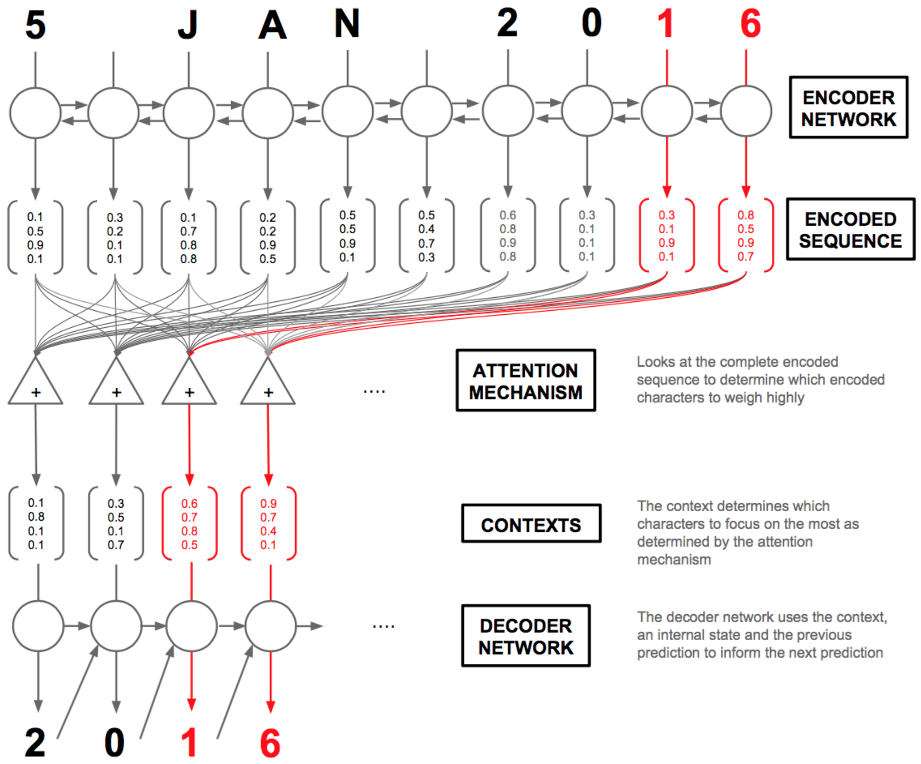
Overview of the attention mechanism



$$e_{j,t} = V_a \cdot \tanh(W_a s_{t-1} + U_a h_j)$$

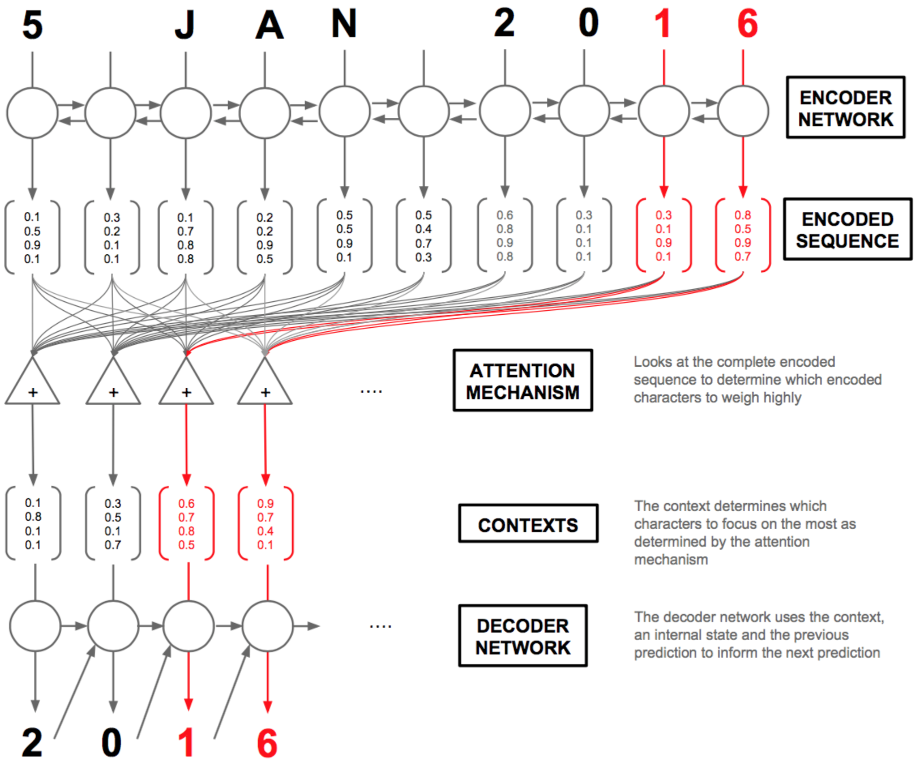
$$\alpha_{j,t} = \frac{\exp(e_j)}{\sum_{k=1}^{T} \exp(e_k)}$$

Overview of the attention mechanism



$$c_t = \sum_{k=1}^{T} \alpha_{k,t} h_k$$

Overview of the attention mechanism



$$r_t = \sigma(W_r y_{t-1} + U_r s_{t-1} + C_r c_t)$$

$$z_t = \sigma(W_z y_{t-1} + U_z s_{t-1} + C_z c_t)$$

$$\hat{s}_t = \tanh(W_p y_{t-1} + U_p[r_t \circ s_{t-1}] + C_p c_t)$$

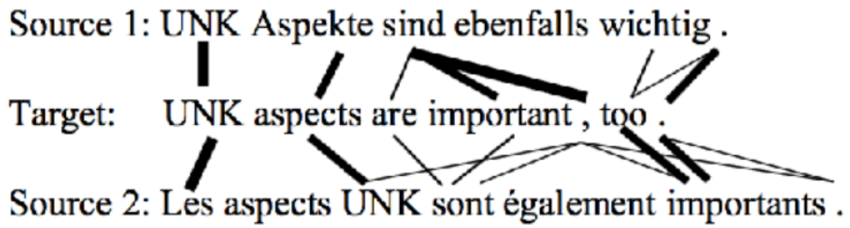
$$s_t = (1 - z_t) \circ s_{t-1} + z_t \circ \hat{s}_t$$

$$y_t = \sigma(W_o y_{t-1} + U_o s_t + C_o c_t)$$

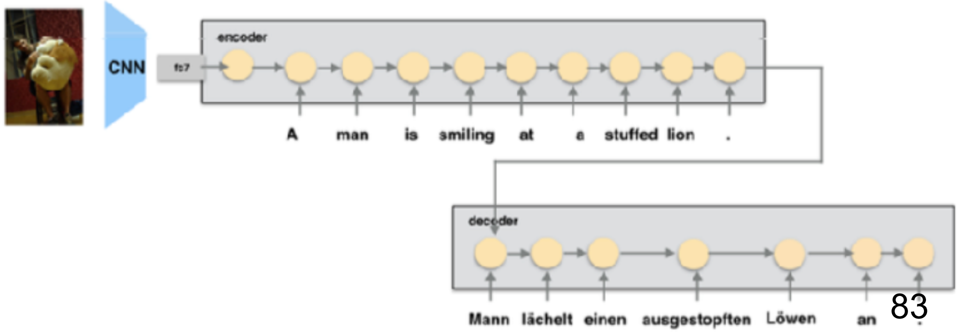
Improvements

Multiple Sources

- Attend to multiple sentences (Zoph et al., 2015)



- Attend to a sentence and an image (Huang et al. 2016)



Coverage

- **Problem:** Neural models tends to drop or repeat content
- In MT,
- 1.Over-translation: some words are unnecessarily translated for multiple times;
- 2. Under-translation: some words are mistakenly untranslated.
- SRC: Señor Presidente, abre la sesión.
- TRG: Mr President Mr President Mr President.
- **Solution:** Model how many times words have been covered e.g. maintaining a coverage vector to keep track of the attention history (Tu et al., 2016)

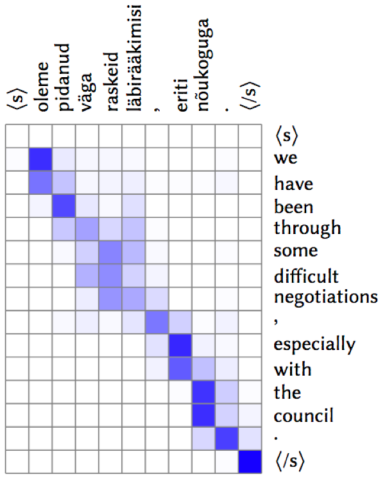
Modeling Coverage for Neural Machine Translation

Zhaopeng Tu[†] Zhengdong Lu[†] Yang Liu[‡] Xiaohua Liu[†] Hang Li[†]

Incorporating Markov Properties

- Intuition: Attention from last time tends to be correlated with attention this time

- Approach: Add information about the last attention when making the next decision



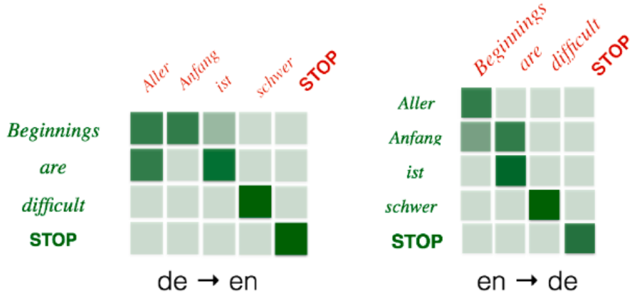
Incorporating Structural Alignment Biases into an Attentional Neural Translation Model

Trevor Cohn and **Cong Duy Vu Hoang** and **Ekaterina Vymolova**

Bidirectional Training

- -Background: Established that for latent variable translation models the alignments improve if both directional models are combined (koehn et al, 2005)

- -Approach: joint training of two directional models



Incorporating Structural Alignment Biases into an Attentional Neural Translation Model

Trevor Cohn and **Cong Duy Vu Hoang** and **Ekaterina Vymolova**

Supervised Training

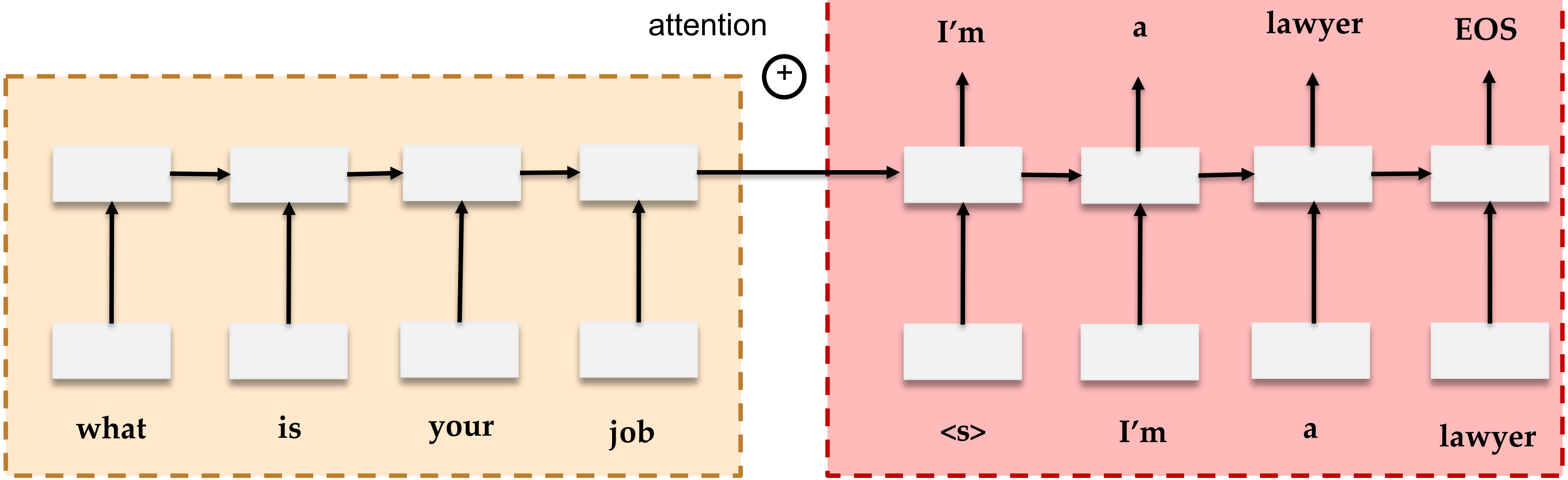
- Sometimes we can get “gold standard” alignments a –priori
  - Manual alignments
  - Pre-trained with strong alignment model
- Train the model to match these strong alignments

Applications

Chatbots

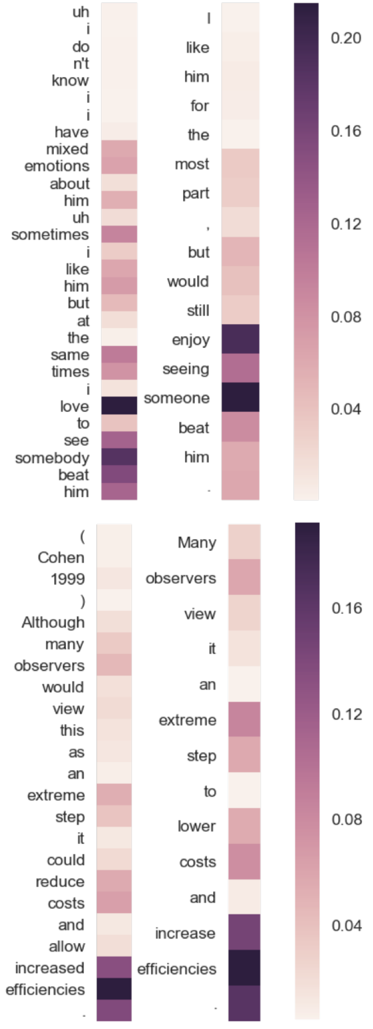
Human: what is your job
Enc-dec: i’m a lawyer
Human: what do you do ?
Enc-dec: i’m a doctor .

- a computer program that conducts a conversation



Natural Language Inference

Caption	A person in a black wetsuit is surfing a small wave.
Entailment	A person is surfing a wave.
Contradiction	A woman is trying to sleep on her bed.
Neutral	A person surfing a wave in Hawaii.



Character-level Intra Attention Network for Natural Language Inference

Han Yang and **Marta R. Costa-jussà** and **José A. R. Fonollosa**

Other NLP Tasks

- ***Text summarization***: process of shortening a text document with software to create a summary with the major points of the original document.
- ***Question Answering:*** automatically producing an answer to a question given a corresponding document.
- ***Semantic Parsing:*** mapping natural language into a logical form that can be executed on a knowledge base and return an answer
- ***Syntactic Parsing:*** process of analysing a string of symbols, either in natural language or in computer languages, conforming to the rules of a formal grammar

Let’s play with visualizing Attention

- https://github.com/abisee/attn_vis

Nice Links

- ATTENTION