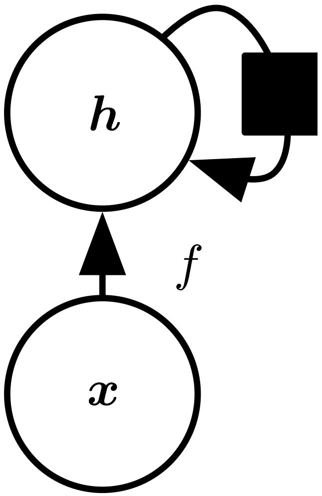
Deep Learning Day

wiederkehrend

Recurrent Neural Networks (RNNs)



Oliver Dürr

Institut für Datenanalyse und Prozessdesign
Zürcher Hochschule für Angewandte Wissenschaften

Winterthur, 22th Sep 2017

Example of RNNs: Image Captioning

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"boy is doing backflip on wakeboard."

"a young boy is holding a baseball bat."

"a cat is sitting on a couch with a remote control."

"a woman holding a teddy bear in front of a mirror."

"a horse is standing in the middle of a road."

Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

Examples of RNN: Sentiment Analysis

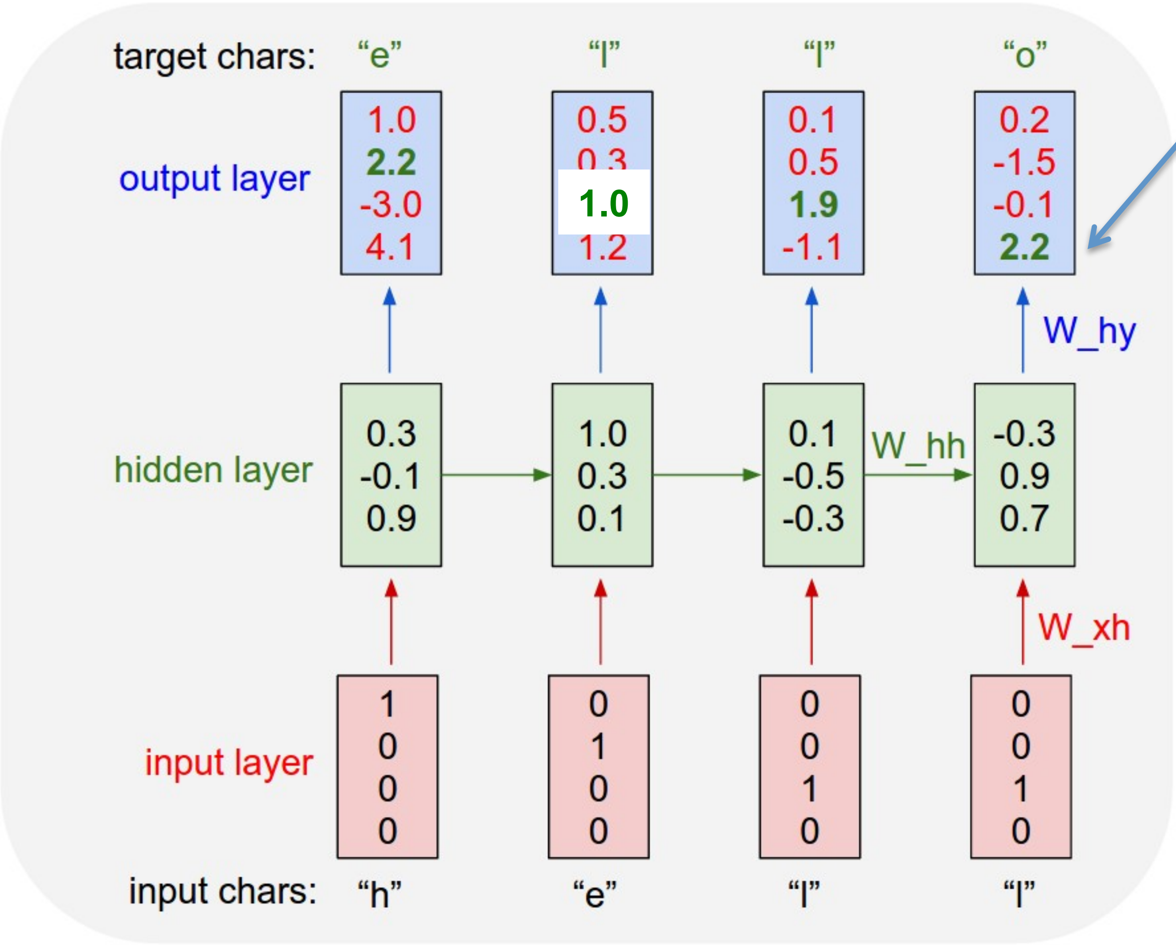
- "I'd rather would have lunch instead of going to the boring workshop"
  - Negative
- "The talk was not so bad after all."
  - Positive

Task:

Characters → Sentiment of Sentence

Word(embeddings) → Sentiment of Sentence

Example of RNN: char level language model

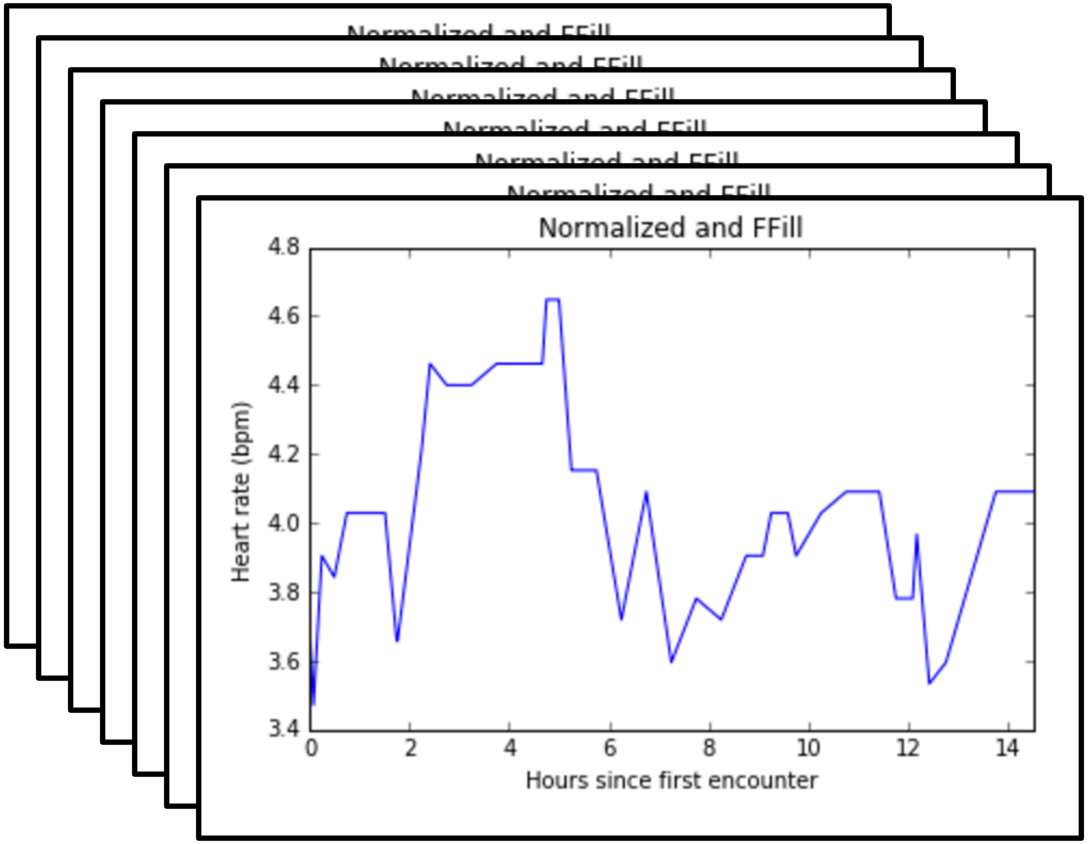


Take softmax to get probabilities

p(‘o’|’hell’)

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

Example: (see tutorial)

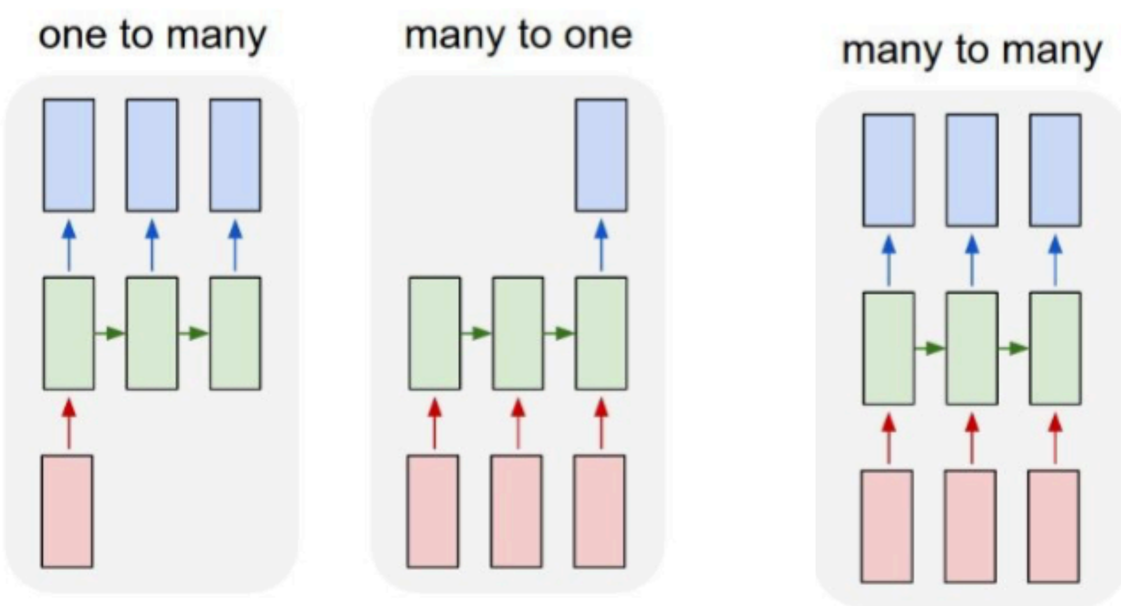


Alive in ER

Clinical Measurements and other data

Use cases of RNNs

Recurrent neural networks (RNN) are used to model sequences or time steps



An **inner state** h_t produces some output [triggered by input]

We focus on the time evolution of the inner state, first.

E.g. Image Captioning. Image -> Seq of words

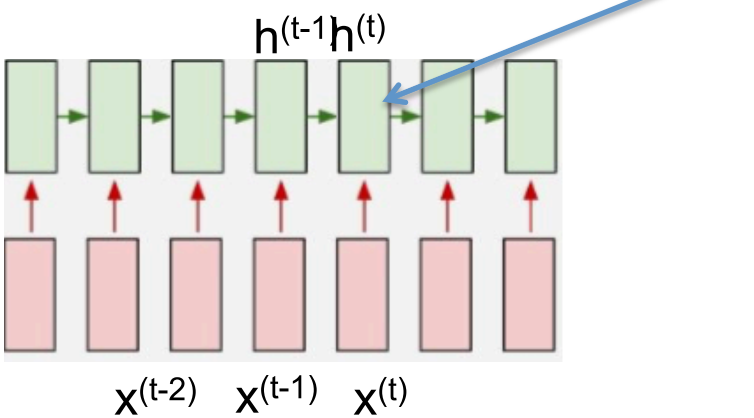
E.g. Sentiment Classification. Seq of words →Sentiment

E.g. Language models.
seq of letters →seq of letters
Predicting the next letter

Here (patient data) →dead or alive

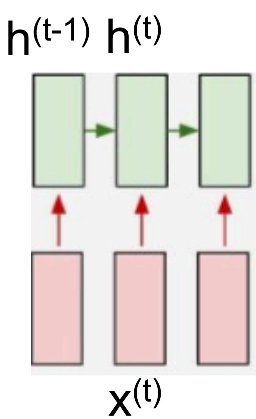
Illustration: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Properties of the inner state



This state $h^{(t)}$ contains all the relevant information from the past.

We can forget about the past



$x^{(t)}$ and $h^{(t)}$ are sufficient.

- We don't need older times.

$h^{(t)}$ summarizes / abstracts $(x^{(1)},\ldots,x^{(t-1)})$

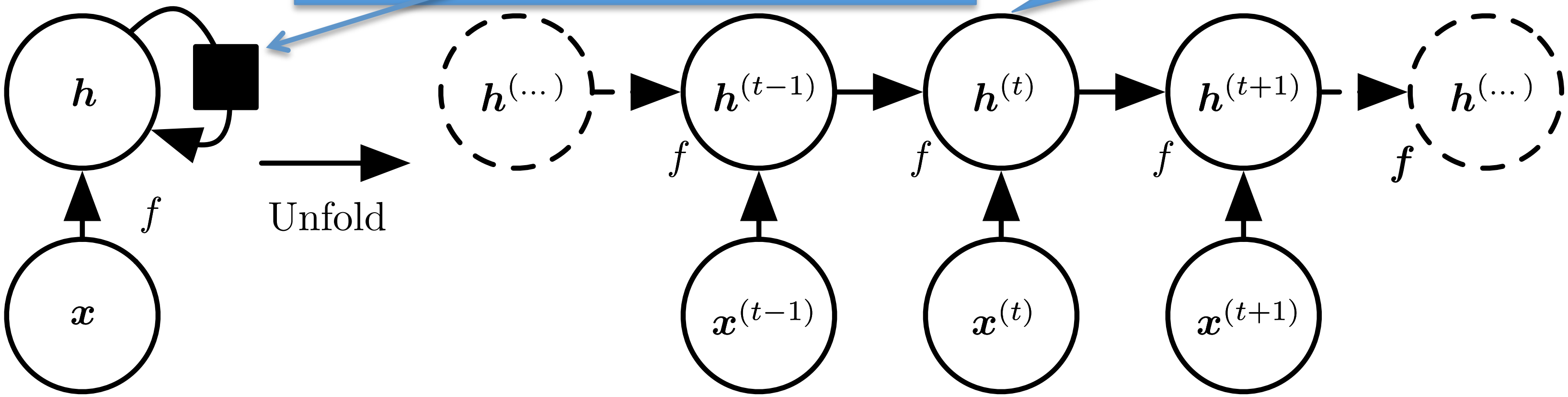
2 ways to draw

Network is driven by sequence x(t) (a vector)

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}, W)$$

$h^{(t)}$ summarizes / abstracts $(x^{(1)},\ldots,x^{(t-1)})$

Same for all times (*recurrent*)



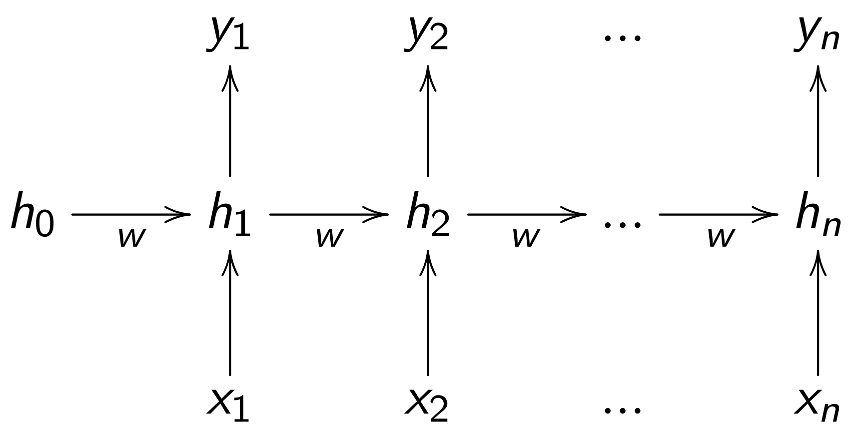
Left: Circuit Diagram (black square delay of one time step)
Right: Unrolled / unfolded

State hidden unit in network

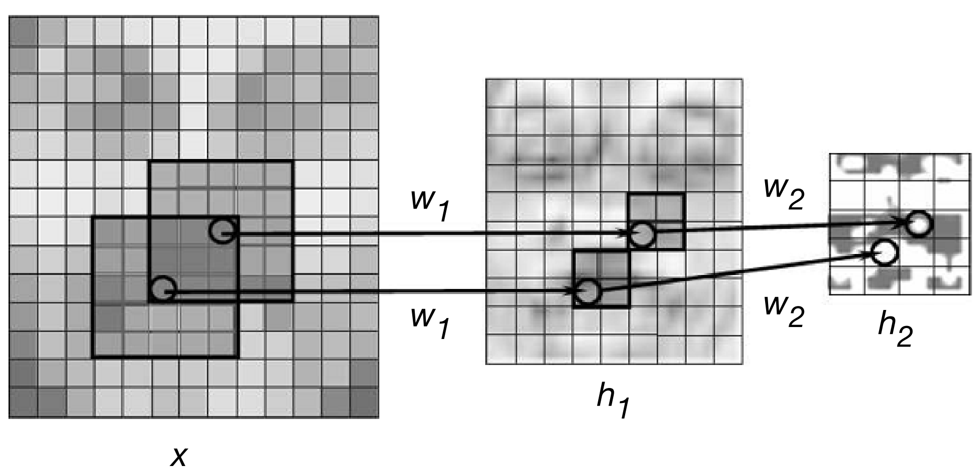
Illustration: http://www.deeplearningbook.org/

Weight Sharing: Key to success

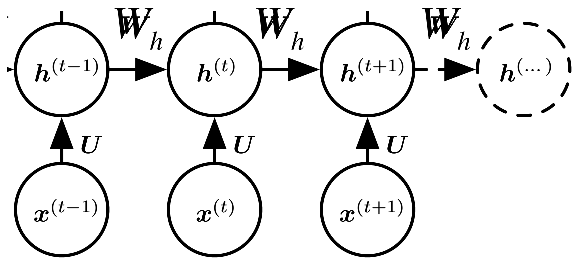
Recurrent neural network shares weights between time-steps



Convolutional neural network shares weights between local regions



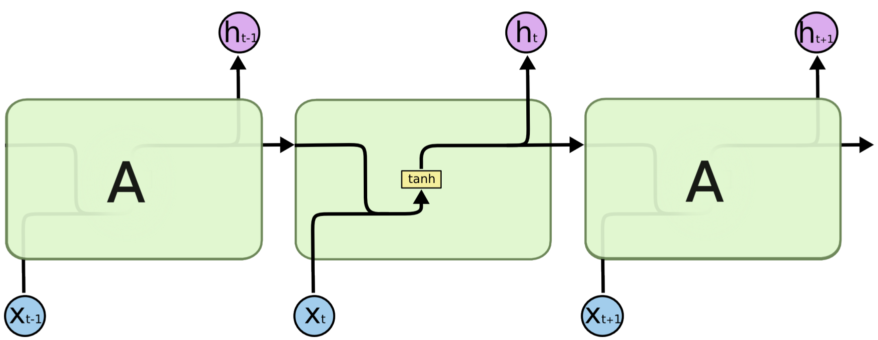
RNN with Matrix Multiplication and Non-linearity



$$h^{(t)} = \tanh(h^{(t-1)}W_h + x^{(t)}U + b)$$

$h^{(t)}$ is vector, size controls complexity.

Alternative view (Colah's Drawing)



Network is defined completely by W.

W depend on size of hidden state and input

$$h^{(t)} = \tanh([h^{(t-1)}, x^{(t)}]W + b) = \tanh(h^{(t-1)}W_h + x^{(t)}U + b)$$

Appending columns at vector

$$W = \begin{pmatrix} W_h \\ U \end{pmatrix}$$

Illustration: http://www.deeplearningbook.org/ http://colah.github.io/posts/2015-08-Understanding-LSTMs/

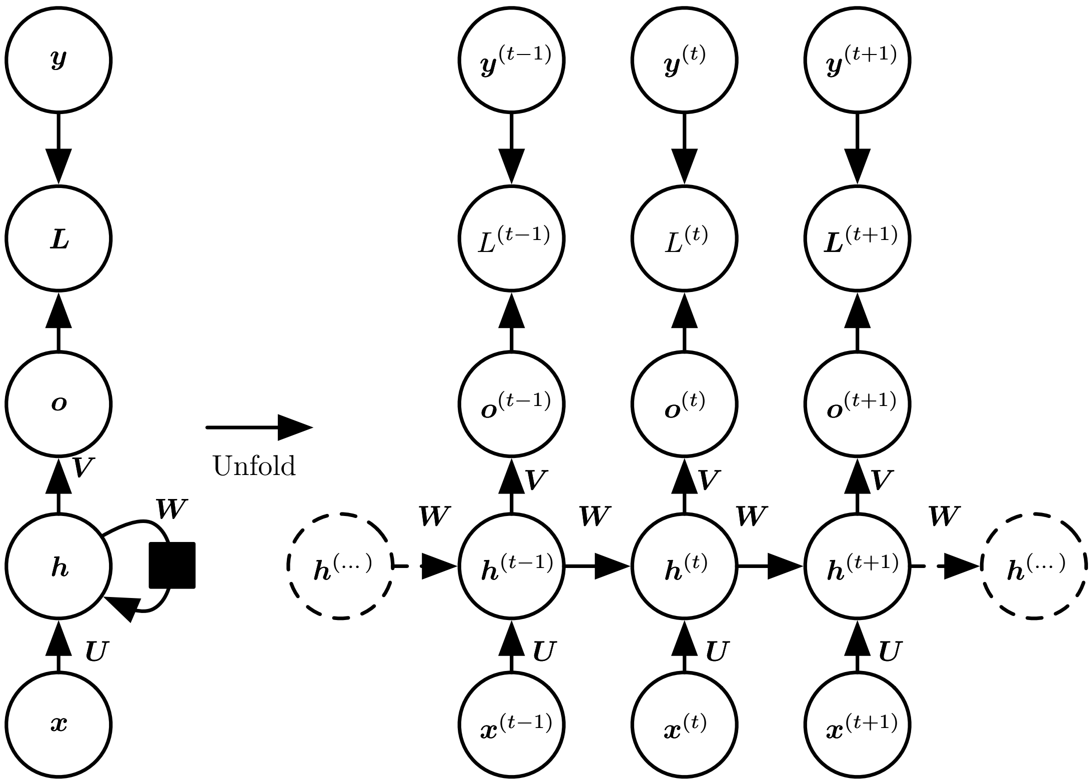
Training the weights

- We add an output
  - Depend on task
- We add a loss function
  - Depend on task
- We train using standard back propagation

Example 2

A sequence $x^{(t)}$ corresponds to an outcome at each time step outcomes y

- x letter in a string of letters
- y next letter



For categorical and one hot

$$L^{(t)} = y^{(t)} \cdot \log(\hat{y}^{(t)})$$

$$L = \sum_t L^{(t)}$$

$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}(\boldsymbol{o}^{(t)})$$

$$\boldsymbol{o}^{(t)} = \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}$$

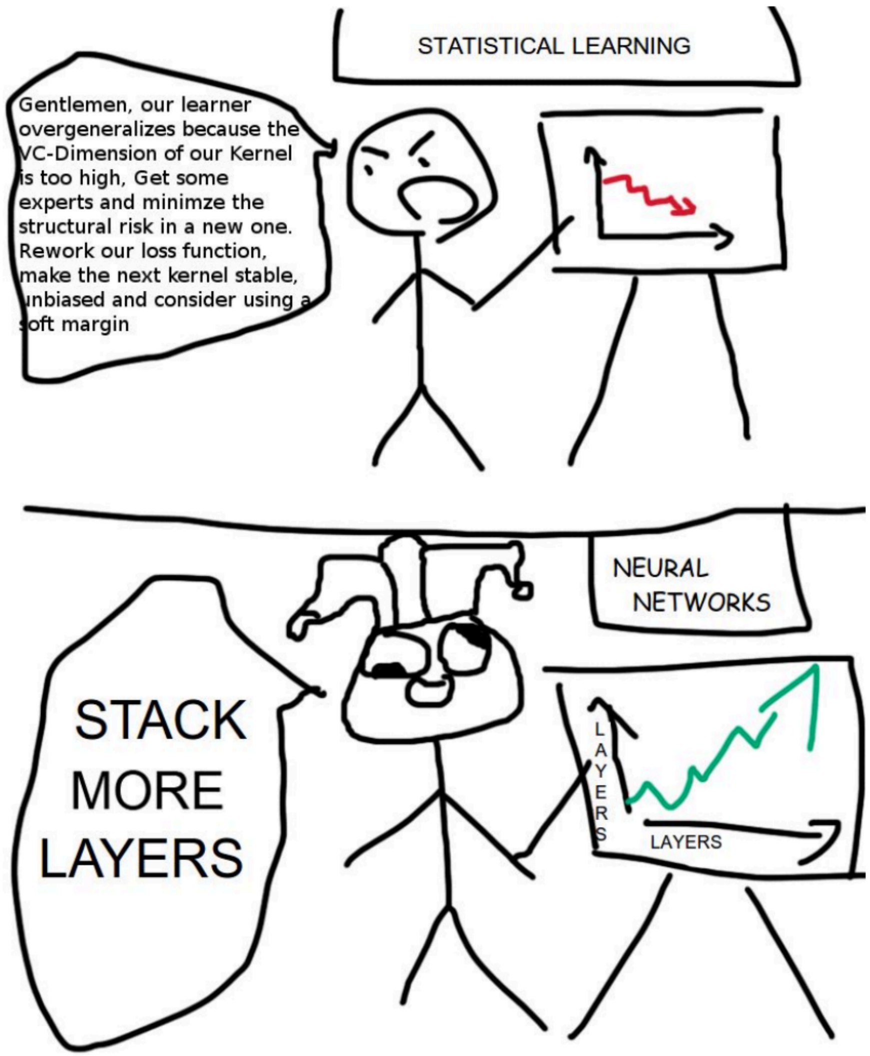
$$\boldsymbol{a}^{(t)} = \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}$$

$$\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{a}^{(t)})$$

W,V,U,b,c are learnt

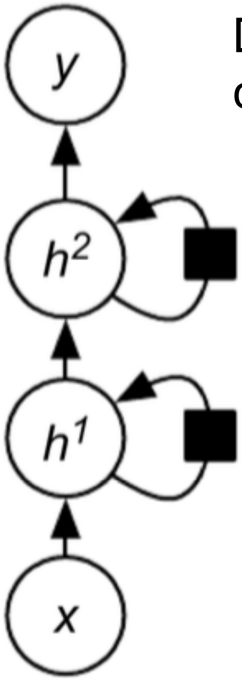
Illustration: http://www.deeplearningbook.org/

The Art of Deep Learning

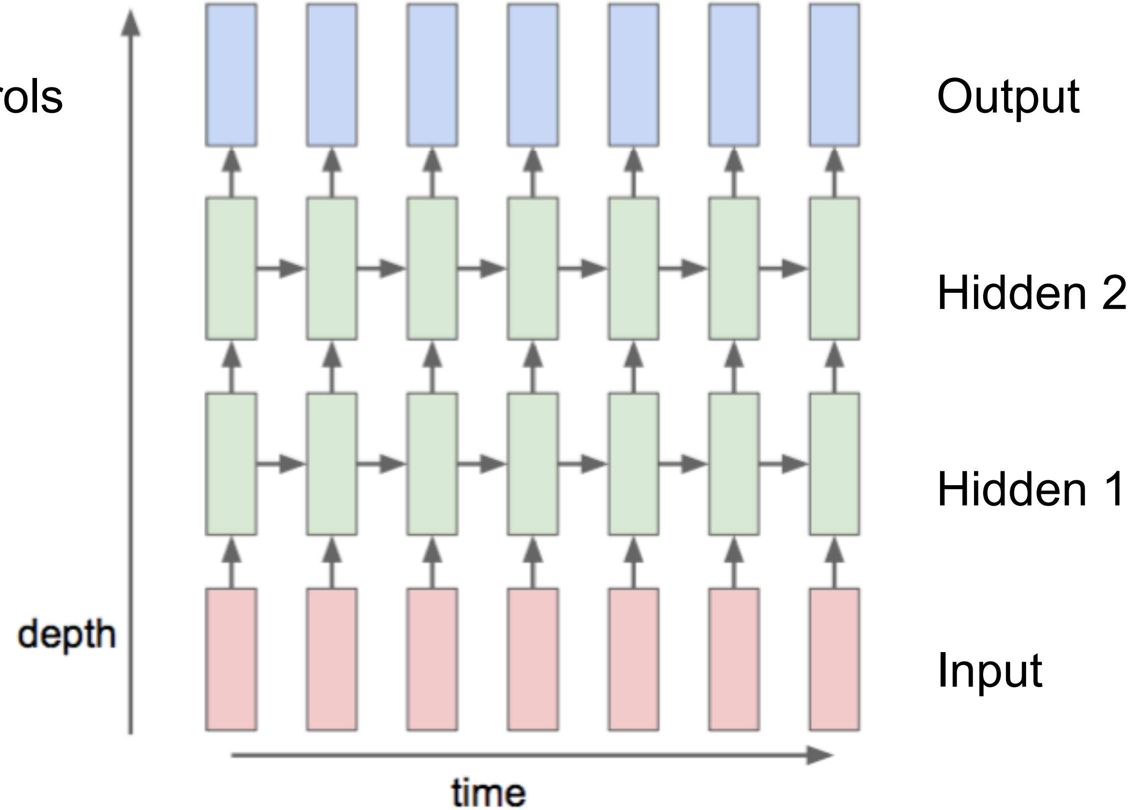


Thanks to Lukas Tuggner for pointing me to:
Taken from: http://futureai.media.mit.edu/wp-content/uploads/sites/40/2015/09/GRID-LSTM.pptx_.pdf

Other architectures: Deep RNNs



Deepness controls complexity



Simply use the output h as a new input. Other approaches are possible, see e.g. DL-book

Illustration: http://www.deeplearningbook.org/ and : http://karpathy.github.io/2015/05/21/rnn-effectiveness/

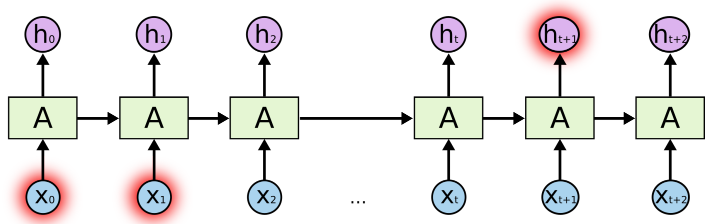
Vanishing Gradient

Vanishing Gradient

- Long range dependencies can be found for many systems and are important to model.
  - Example in text understanding:

  Lisa was born in Springfield (***USA***) …she can speak fluently ***English***.

  Long range dependency (USA and English)

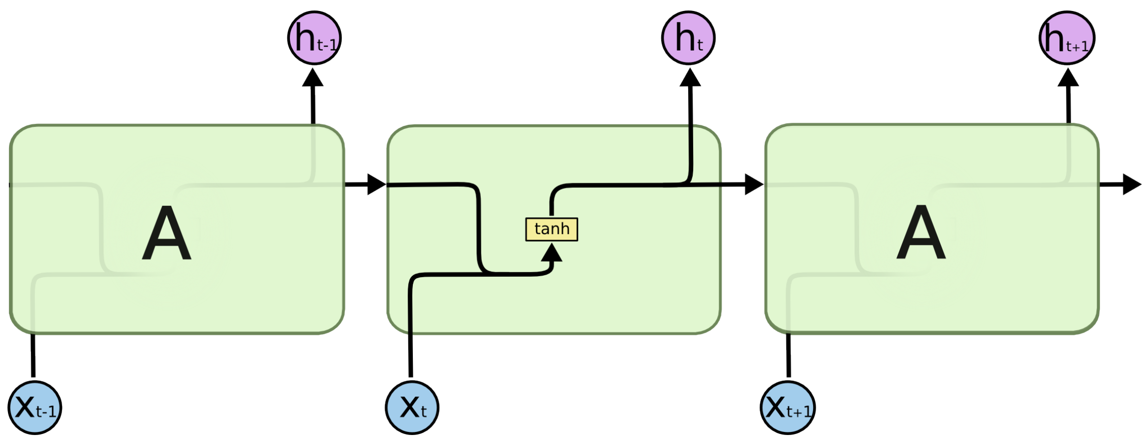


- Long range interactions cannot be trained with standard RNN
  - Vanishing Gradient (Hochreiter 1991, Diplomarbeit “Untersuchungen zu dynamischen neuronalen Netzen”)

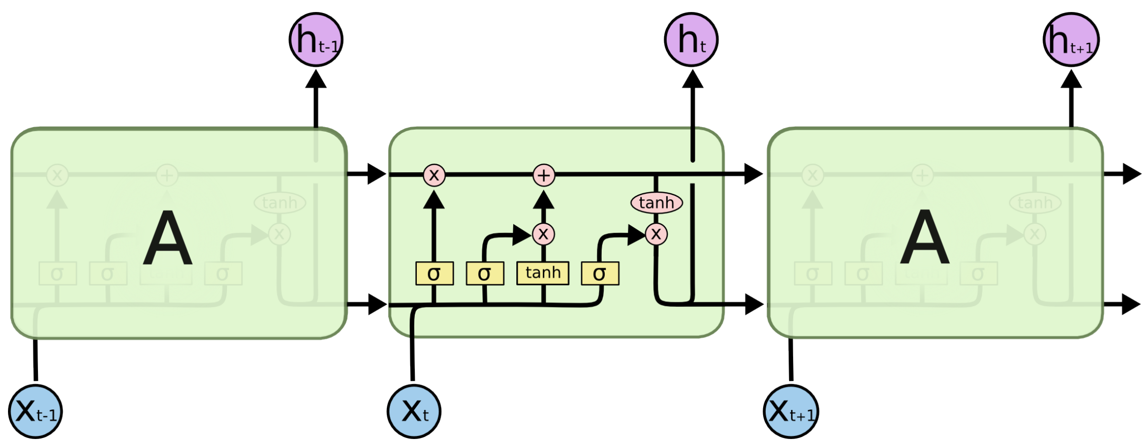
- We don’t fix the training, we change the model
  - RNN-cell →LTSM-cell

Slide credit: Colah’s blog

Replacing RNN Cells with LSTM Cells



Standard RNN-Cell



LSTM-Cell
Complicated inner state

In TensorFlow:

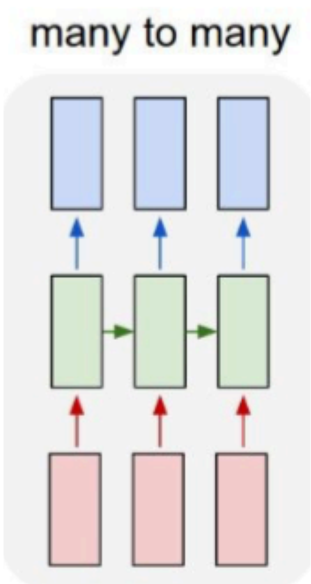
```
#cell = tf.nn.rnn_cell.BasicRNNCell(state_size)
cell = tf.nn.rnn_cell.BasicLSTMCell(state_size)
```

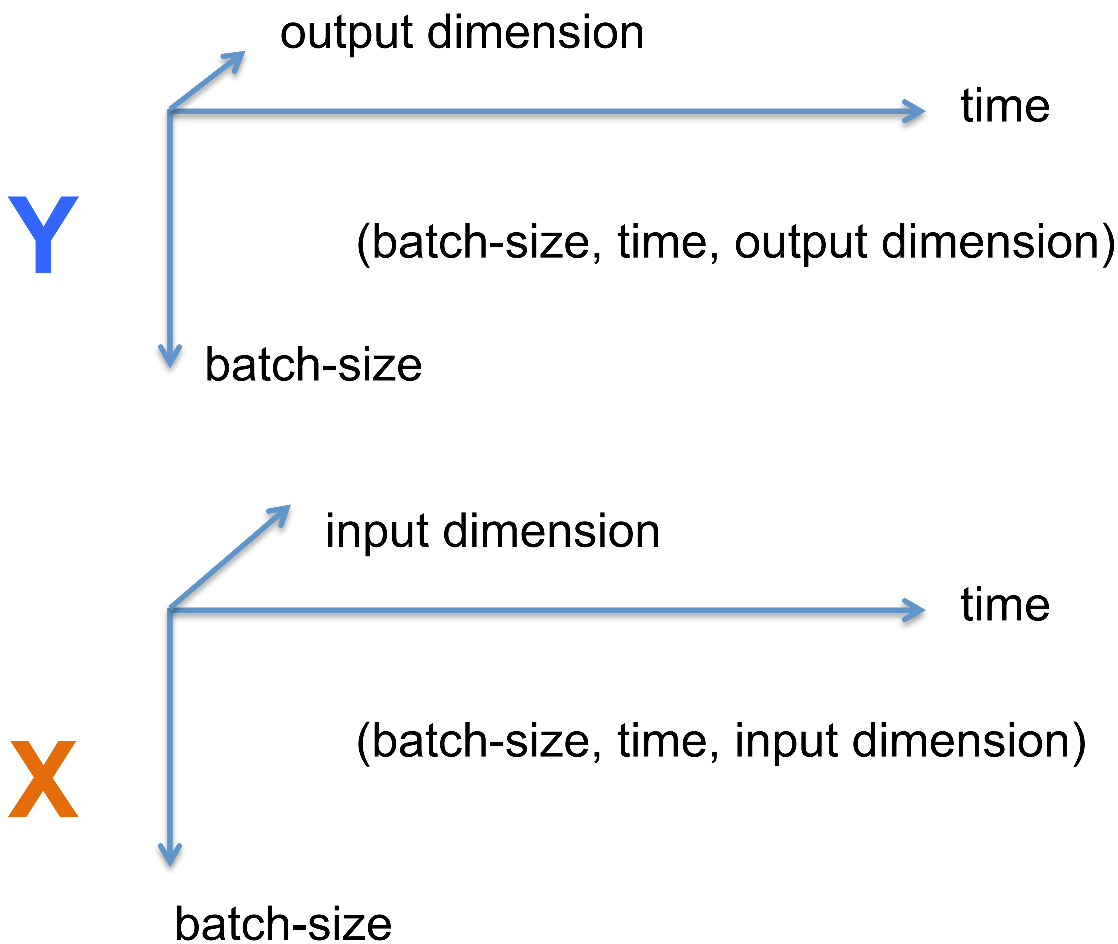
Slide credit: Colah’s blog

Training of LSTMs

Training: shape to the tensors

- Done in mini-batches to benefit from parallel power on GPU





Note that tensorshapes need to be fixed

Training: (Last hint for Tutorial) Technical Detail Masking

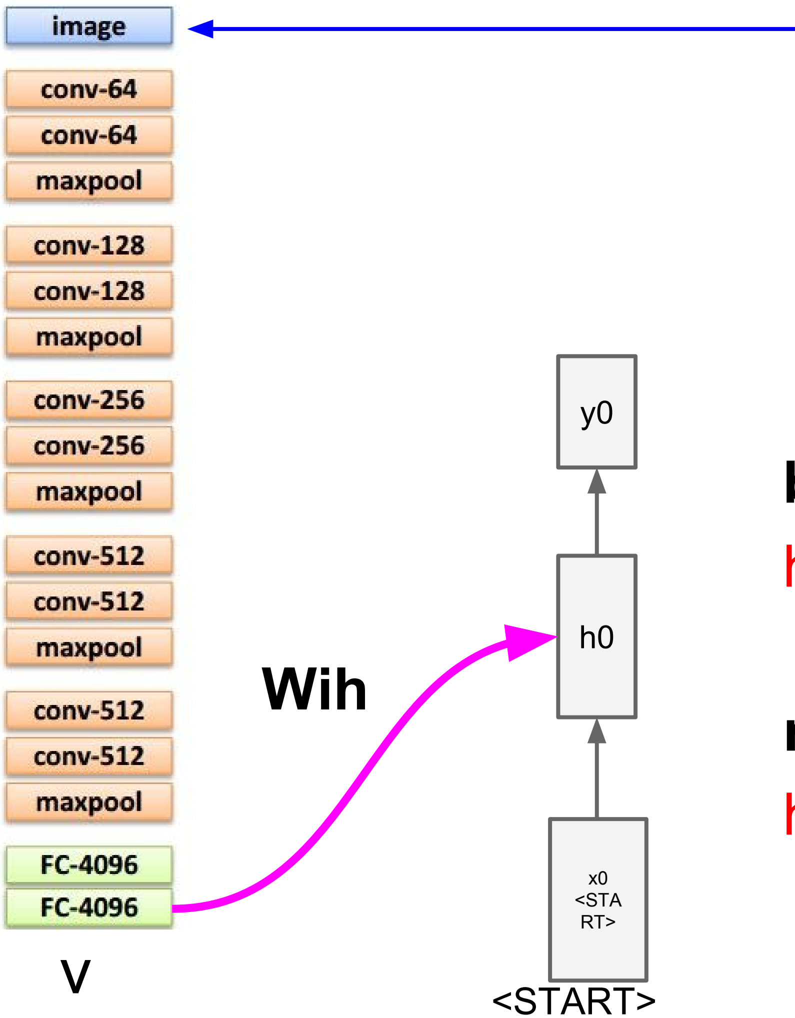
- Sometimes sequences have different length
- Solution
  - Clamp all to fixed size e.g. 500
  - If too short
    - Use masking to indicate if cell ends earlier

Resources

- Many figures are taken from the following resources:
  - **Deep Learning Book** chap10
    - http://www.deeplearningbook.org/contents/rnn.html
  - CS231n
    - Lecture on RNN: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf
    - Video to CS231n https://www.youtube.com/watch?v=iX5V1WpxxkY
  - CAS Machine Intelligence
    - https://tensorchiefs.github.io/dl_course/
  - Blog Posts
    - Karpathy, May 2015: The unreasonable effectiveness of Recurrent Neural Networks http://karpathy.github.io/2015/05/21/rnn-effectiveness/
    - Colah, August 2015: Understanding LSTM Networks http://colah.github.io/posts/2015-08-Understanding-LSTMs/
    - R2RT, July 2016: http://r2rt.com/recurrent-neural-networks-in-tensorflow-i.html
    - WildML, August 2016: Praktical consideration e.g. how to use sequences with different length. http://www.wildml.com/2016/08/rnns-in-tensorflow-a-practical-guide-and-undocumented-features/
- Further ipython notebooks:
  - https://github.com/oduerr/dl_tutorial/blob/master/tensorflow/RNN

Backup

Image Captioning



test image

before:

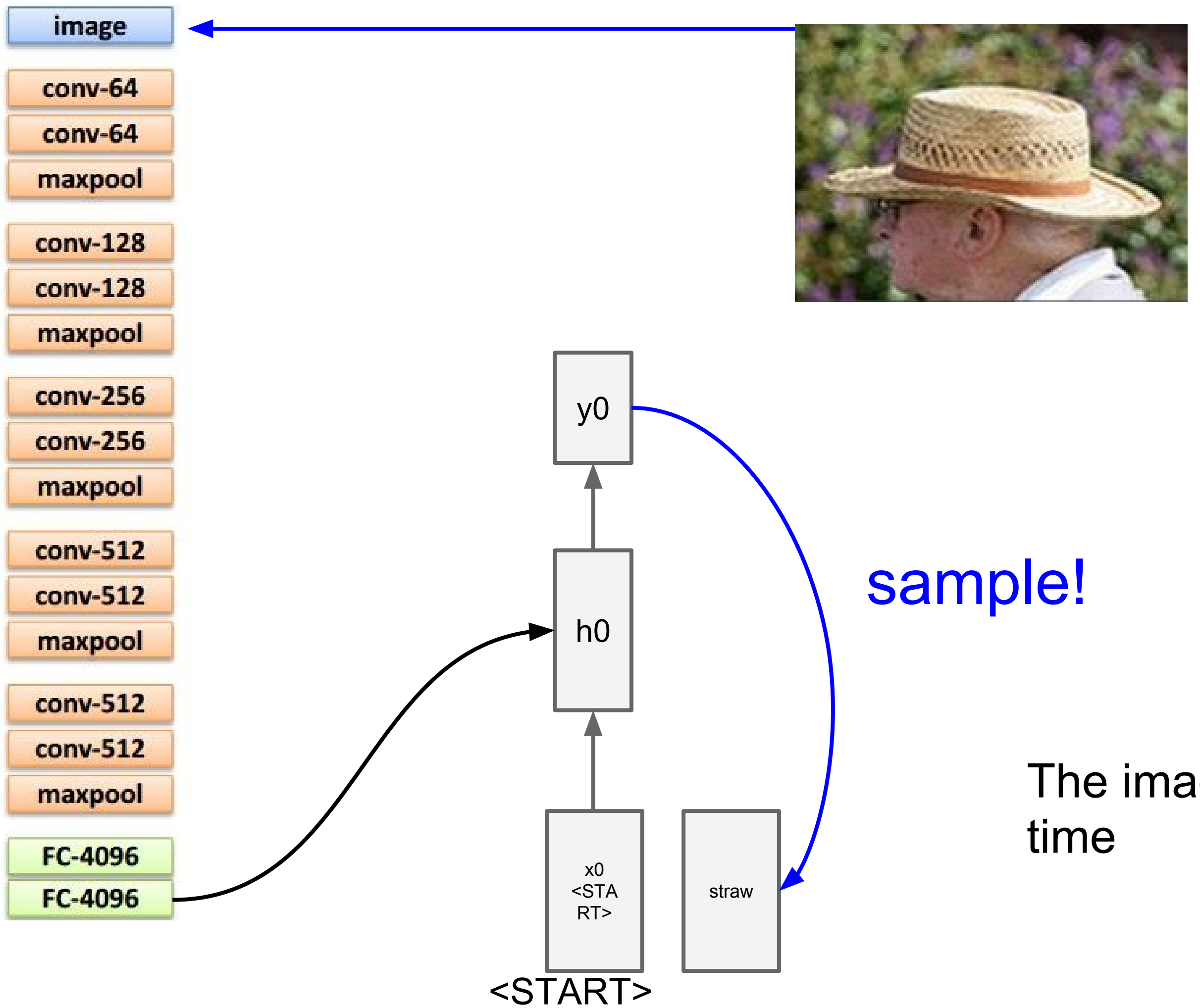
$$h = \tanh(Wxh * x + Whh * h)$$

now:

$$h = \tanh(Wxh * x + Whh * h + \mathbf{Wih * v})$$

Slides taken from: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

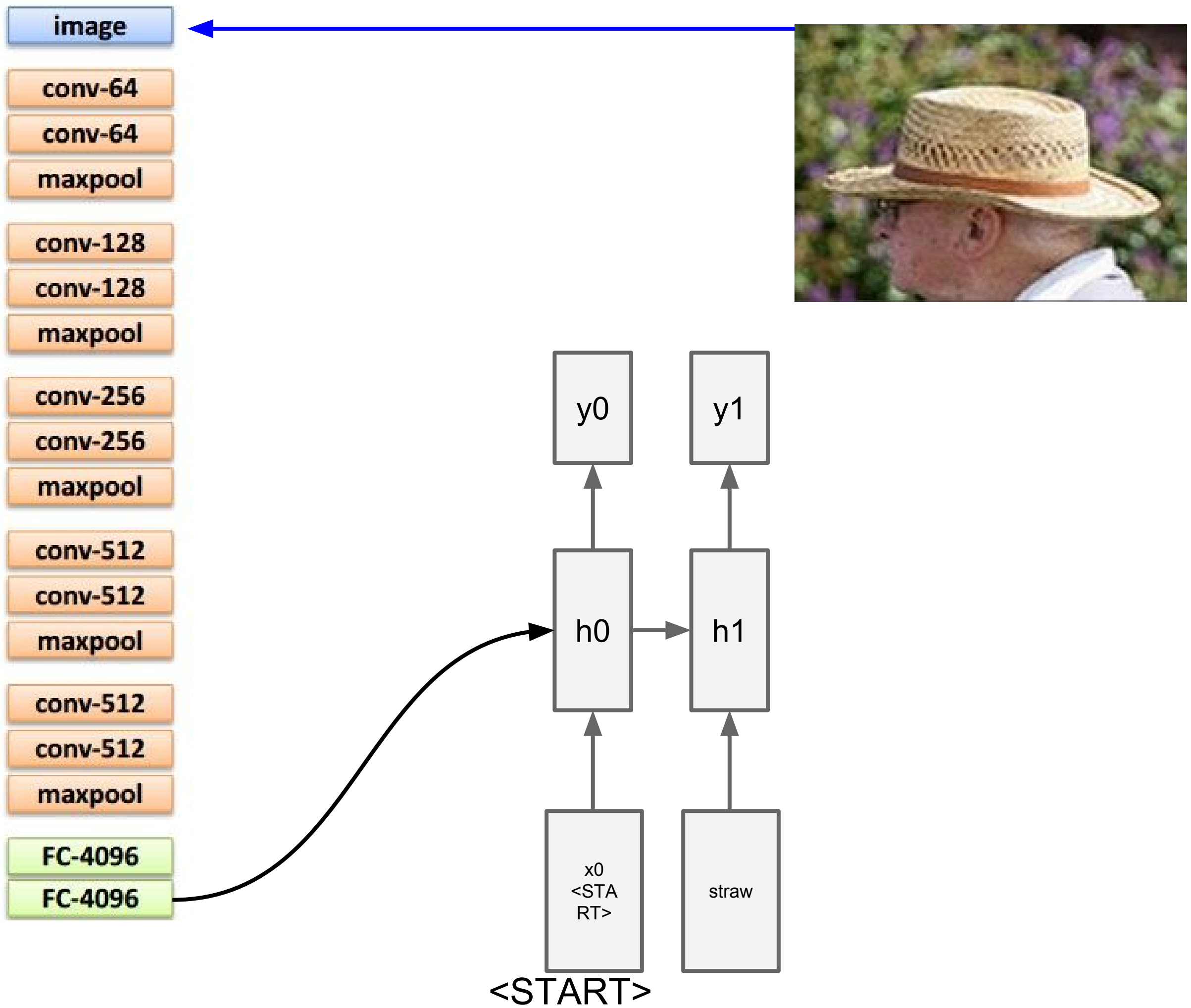
Image Captioning



The image is only feed in the first time

Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

Image Captioning



test image

Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

Image Captioning

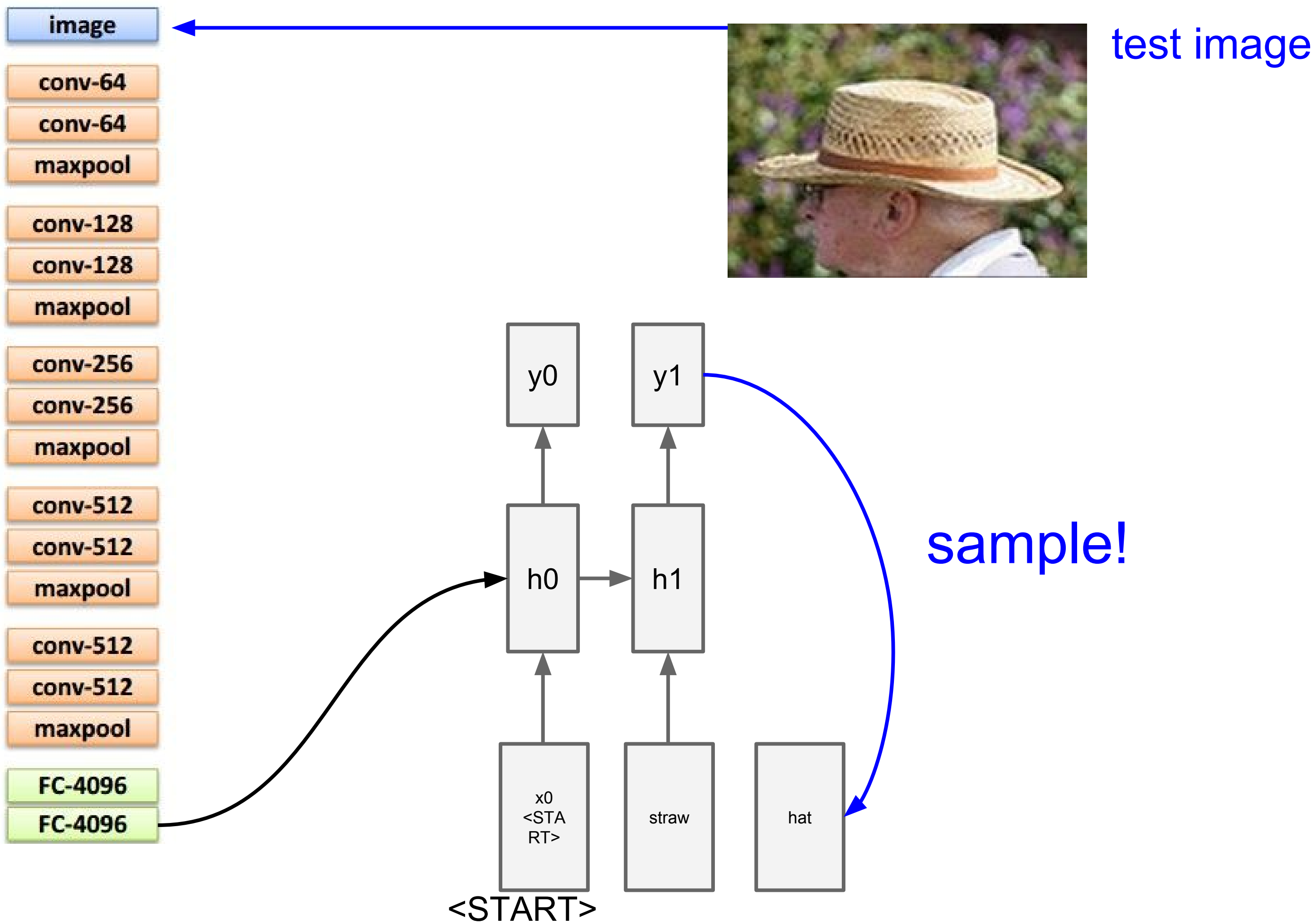


Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

Image Captioning

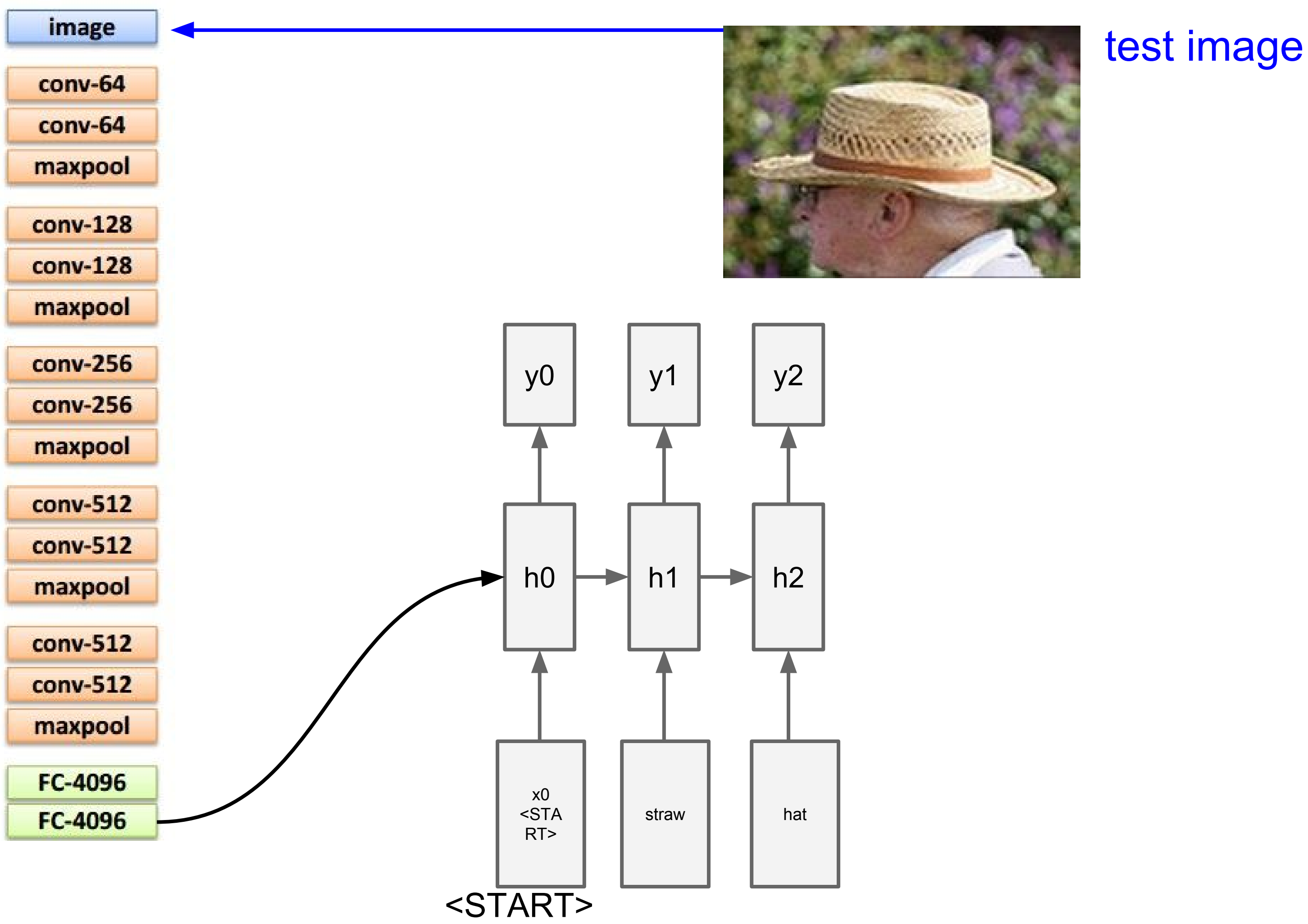


Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf

Image Captioning

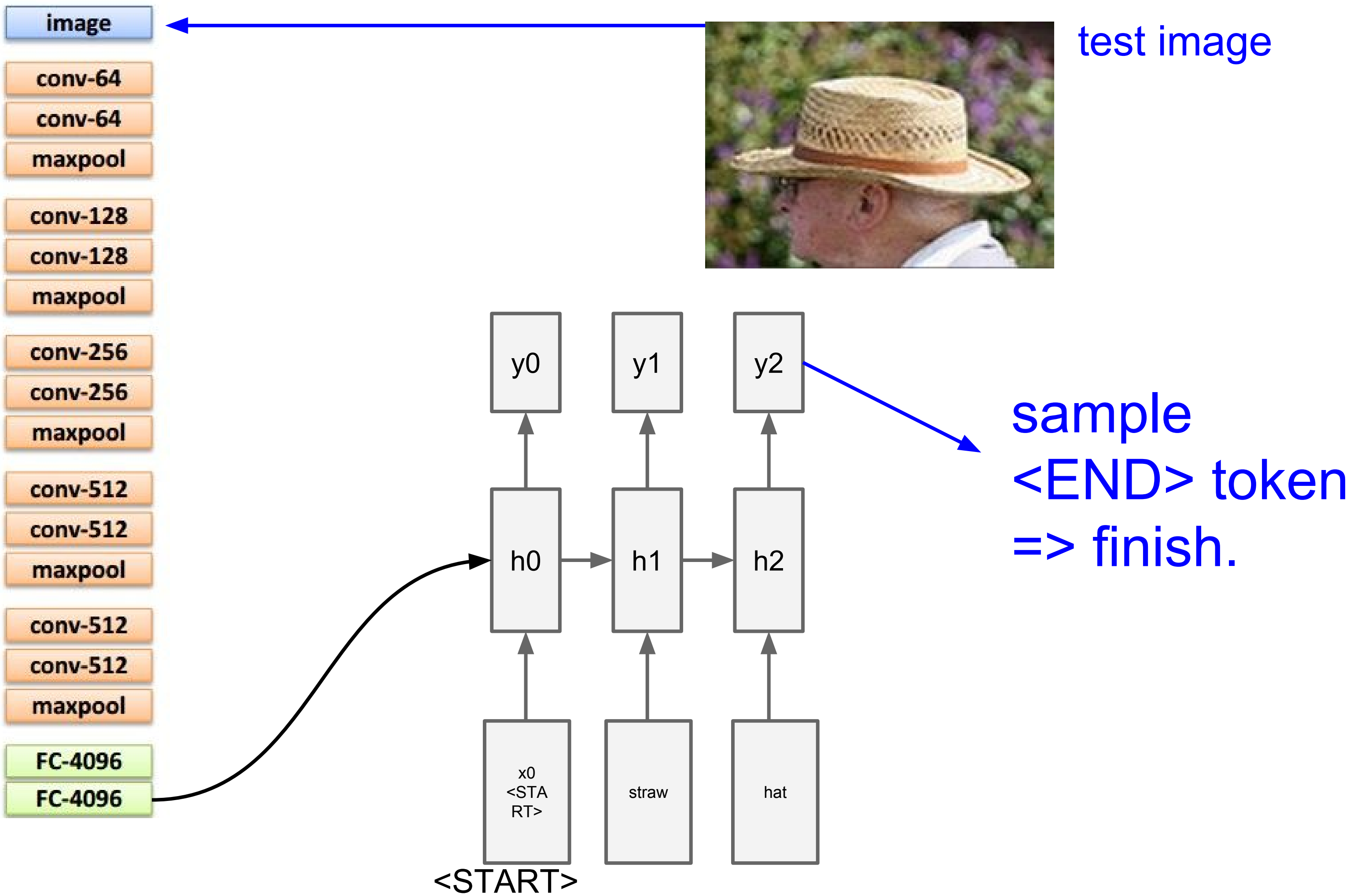


Illustration: http://cs231n.stanford.edu/slides/winter1516_lecture10.pdf