Neural Networks and Deep Learning Part II

Dan Roth & Ben Zhou
danroth@seas.upenn.edu|http://www.cis.upenn.edu/~danroth/|461C, 3401 Walnut

Slides were created by Dan Roth (for CIS519/419 at Penn or CS446 at UIUC), Daniel Khashabi, Nitish Gupta and Ben Zhou
(or by other authors who have made their ML slides available.)

Administration (11/18/20)

Are we recording? YES!

Available on the web site

- Remember that all the lectures are available on the website before the class
  - Go over it and be prepared
  - A new set of written notes will accompany most lectures, with some more details, examples and, (when relevant) some code.

- HW4 is out – NNs and Bayesian Learning
  - Due 12/3
  - Recitations will be devoted to introducing you to PyTorch

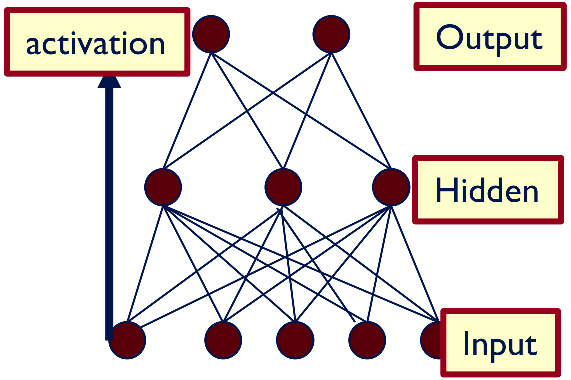
- Projects
  - Most of you have chosen a project and a team.

Projects

- CIS 519 students need to do a team project: Read the project descriptions and follow the updates on the Project webpage
  - Teams will be of size 2-4
  - We will help grouping if needed
- There will be 3 options for projects.
  - Natural Language Processing (Text)
  - Computer Vision (Images)
  - Speech (Audio)
- In all cases, we will give you datasets and initial ideas
  - The problem will be multiclass classification problems
  - You will get annotated data only for some of the labels, but will also have to predict other labels
  - 0-zero shot learning; few-shot learning; transfer learning
- A detailed note will come out today.
- Timeline:
  - 11/11 Choose a project and team up
  - 11/23 Initial proposal describing what your team plans to do
  - 12/2 Progress report
  - 12/15-20 (TBD) Final paper + short video
- Try to make it interesting!

Recap: Multi-Layer Perceptrons

- Multi-layer network
  - A global approximator
  - Different rules for training it
- The Back-propagation
  - Forward step
  - Back propagation of errors



- Congrats! Now you know the most important algorithm in neural networks!

- Today:
  - Convolutional Neural Networks
  - Recurrent Neural Networks
  - Attention and Transformers

Receptive Fields

- The receptive field of an individual sensory neuron is the particular region of the sensory space (e.g., the body surface, or the retina) in which a stimulus will trigger the firing of that neuron.
  - In the auditory system, receptive fields can correspond to wave amplitudes in auditory space
- Designing “proper” receptive fields for the input Neurons is a significant challenge.

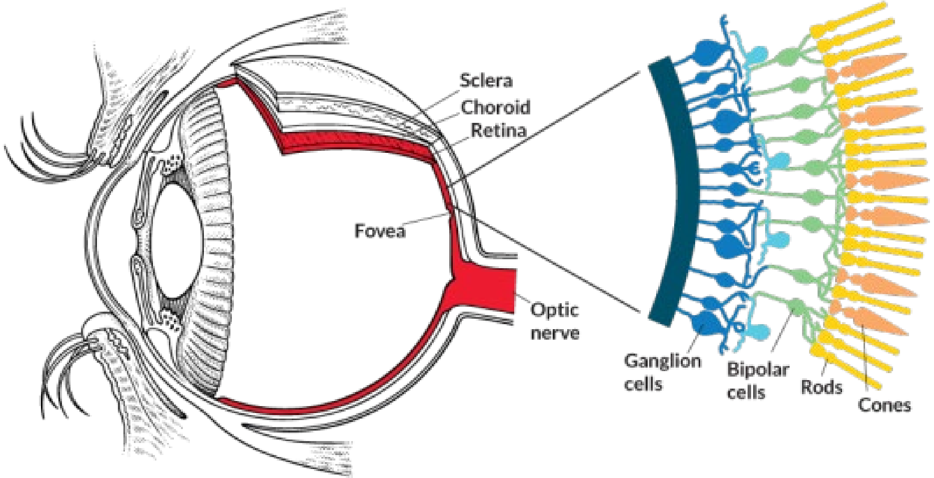
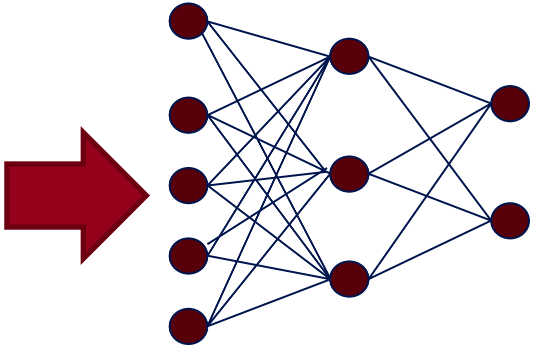


Image Classification

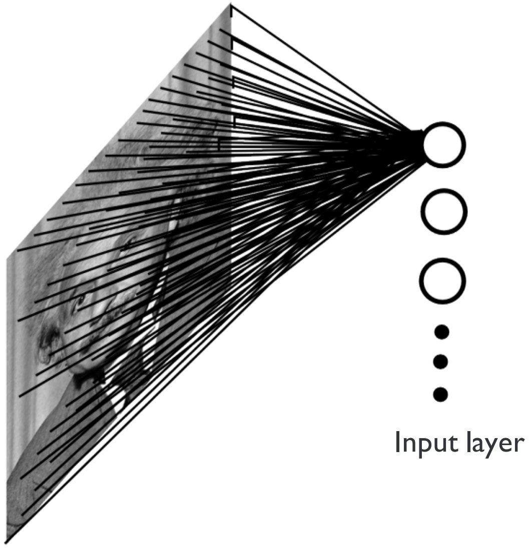
Consider a task with image inputs

- Receptive fields should give expressive features from the raw input to the system
- How would you design the receptive fields for this problem?



Human face
or not?

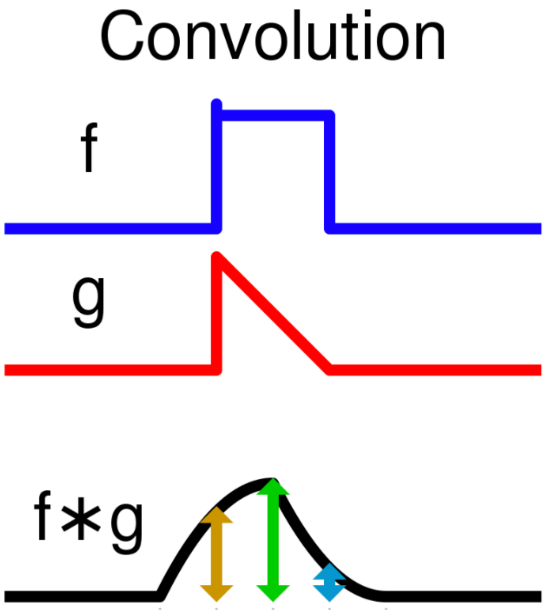
- A <u>fully connected layer</u>:
  - Example:
    - 100×100 sized image
    - 1000 units in the hidden layer
  - Problems:
    - 10^7 edges!
    - Spatial correlations lost!
    - Variables sized inputs.
    - Potential overfitting



Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer

- A solution:
  - Filters to capture different patterns in the input space.



ters across different locations (assuming input is stationary)

with learned filters

arned during trainin

able-sized inputs w

pooling layer.

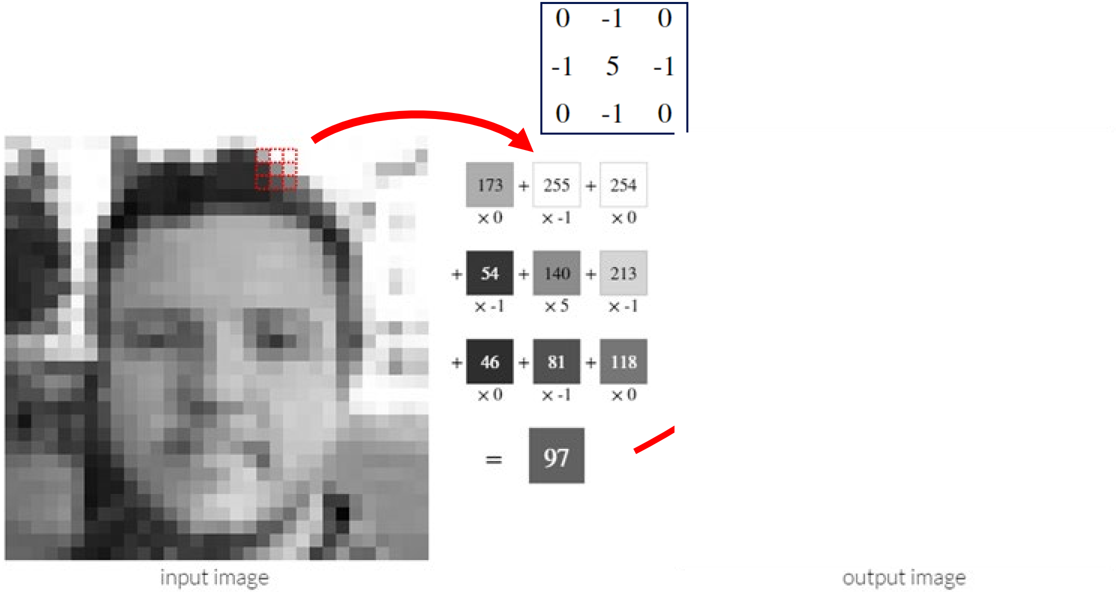
So what is a convolution?

Convolution:
A mathematical operation on two functions that produces a third function expressing how the shape of one is modified by the other.

Slide Credit: Marc'Aurelio Ranzato

Convolution Operator (2)

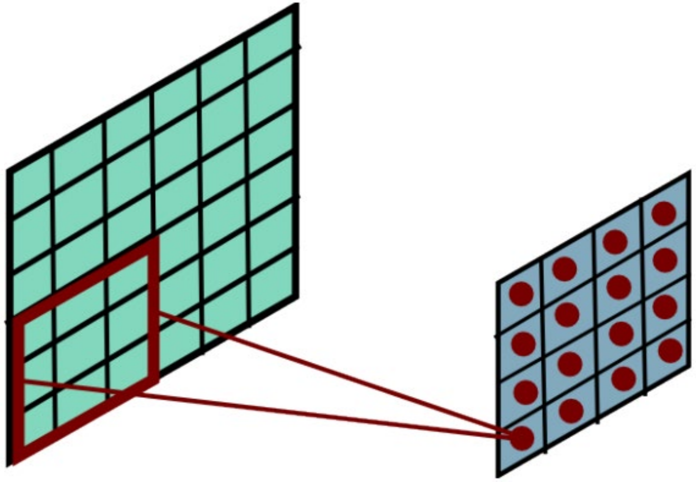
- Convolution in two dimension:
  - Example: Sharpen kernel:



Try other kernels: http://setosa.io/ev/image-kernels/

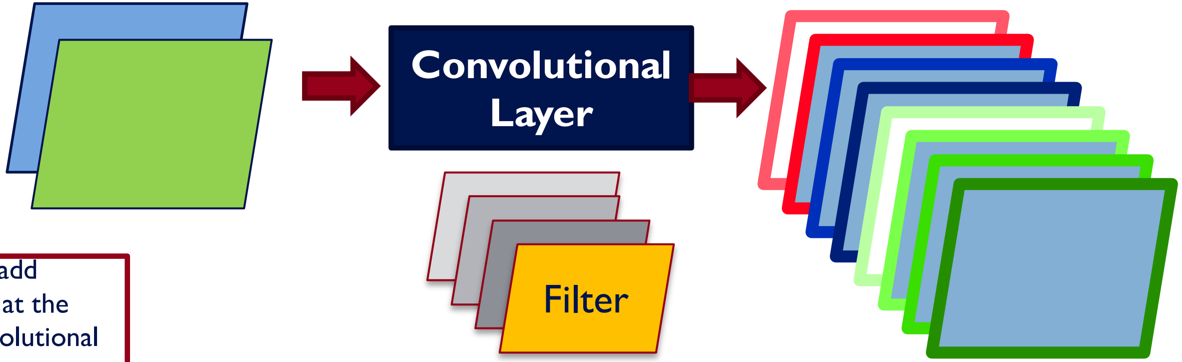
Convolution Operator (3)

- Convolution in two dimension:
  - Convolve a filter matrix across the image matrix



Convolutional Layer

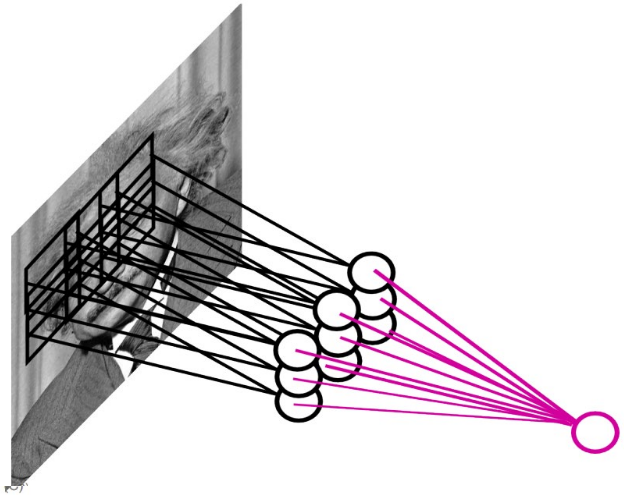
- The convolution of the input (vector/matrix) with weights (vector/matrix) results in a response vector/matrix.
- We can have multiple filters in each convolutional layer, each producing an output.
- If it is an intermediate layer, it can have multiple inputs!



One can add nonlinearity at the output of convolutional layer

Pooling Layer

- How to handle variable sized inputs?
  - A layer which reduces inputs of different size, to a fixed size.
  - Pooling



Slide Credit: Marc'Aurelio Ranzato

Pooling Layer

- How to handle variable sized inputs?
  - A layer which reduces inputs of different size, to a fixed size.
  - **Pooling**
  - Different variations
    - Max pooling

$$h_i[n] = \max_{i \in N(n)} \tilde{h}[i]$$

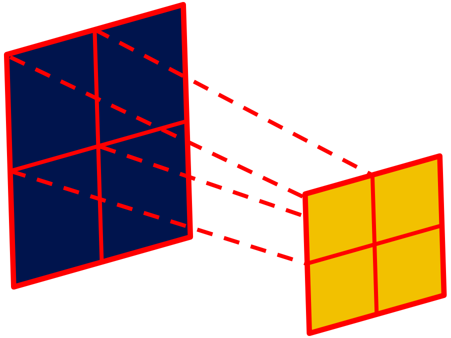
    - Average pooling

$$h_i[n] = \frac{1}{n} \sum_{i \in N(n)} \tilde{h}[i]$$

    - L2-pooling

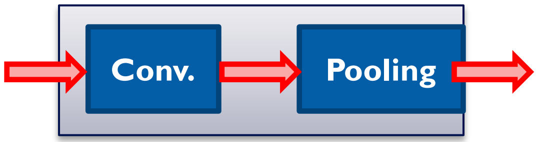
$$h_i[n] = \frac{1}{n} \sqrt{\sum_{i \in N(n)} \tilde{h}^2[i]}$$

    - etc

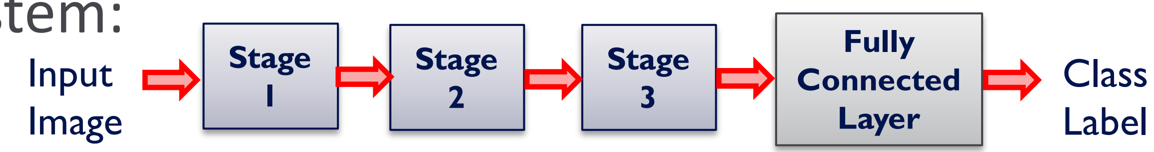


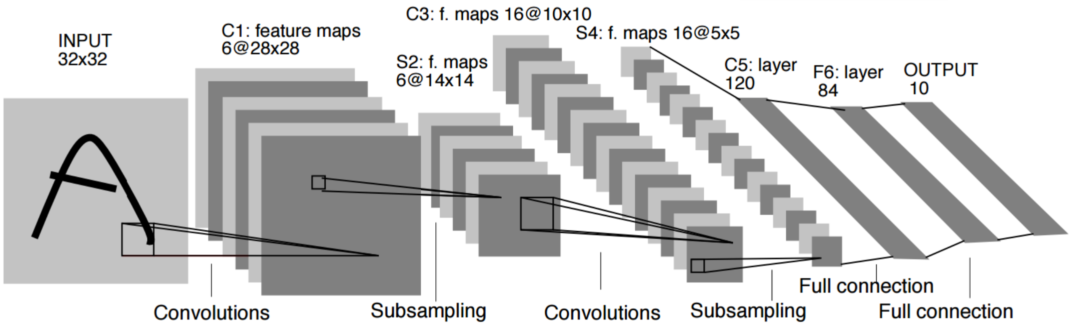
Convolutional Nets

- One stage structure:



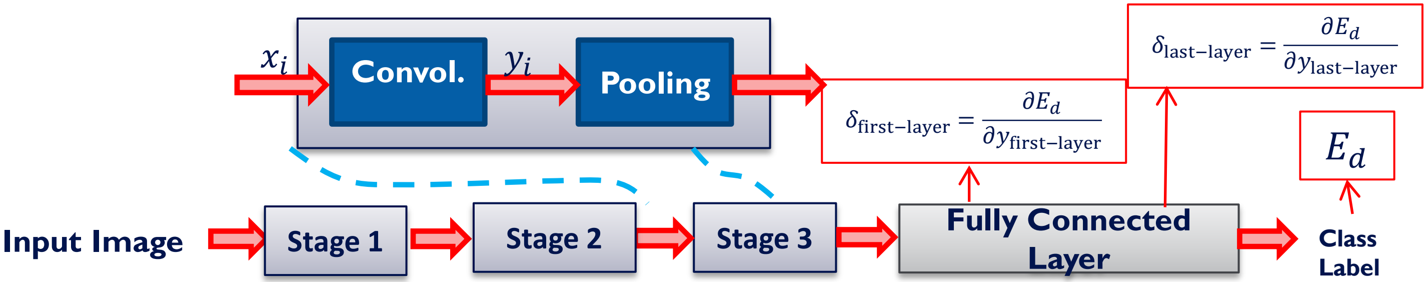
- Whole system:



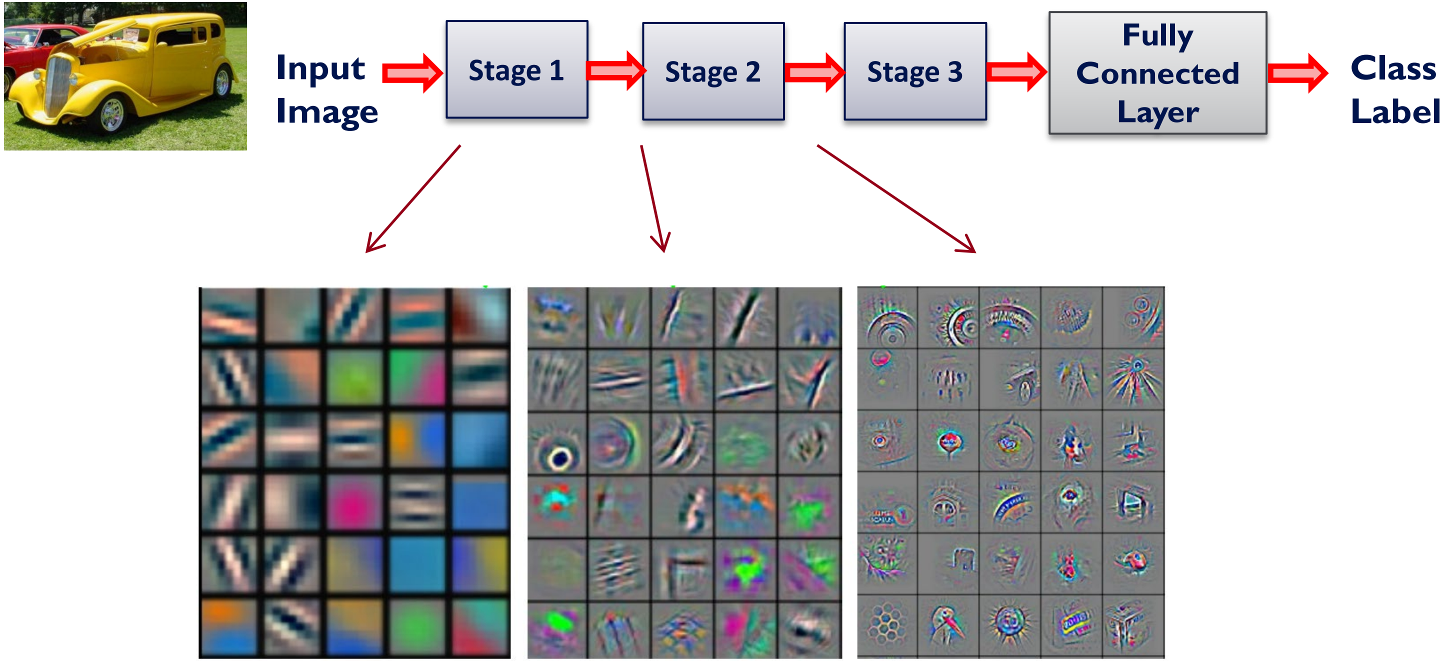


Training a ConvNet

- The same procedure from Back-propagation applies here.
  - Remember in backprop we started from the error terms in the last stage, and passed them back to the previous layers, one by one.
- Back-prop for the pooling layer:
  - Consider, for example, the case of "max" pooling.
  - This layer only routes the gradient to the input that has the highest value in the forward pass.
  - Hence, during the forward pass of a pooling layer it is common to keep track of the index of the max activation (sometimes also called *the switches*) so that gradient routing is efficient during backpropagation.



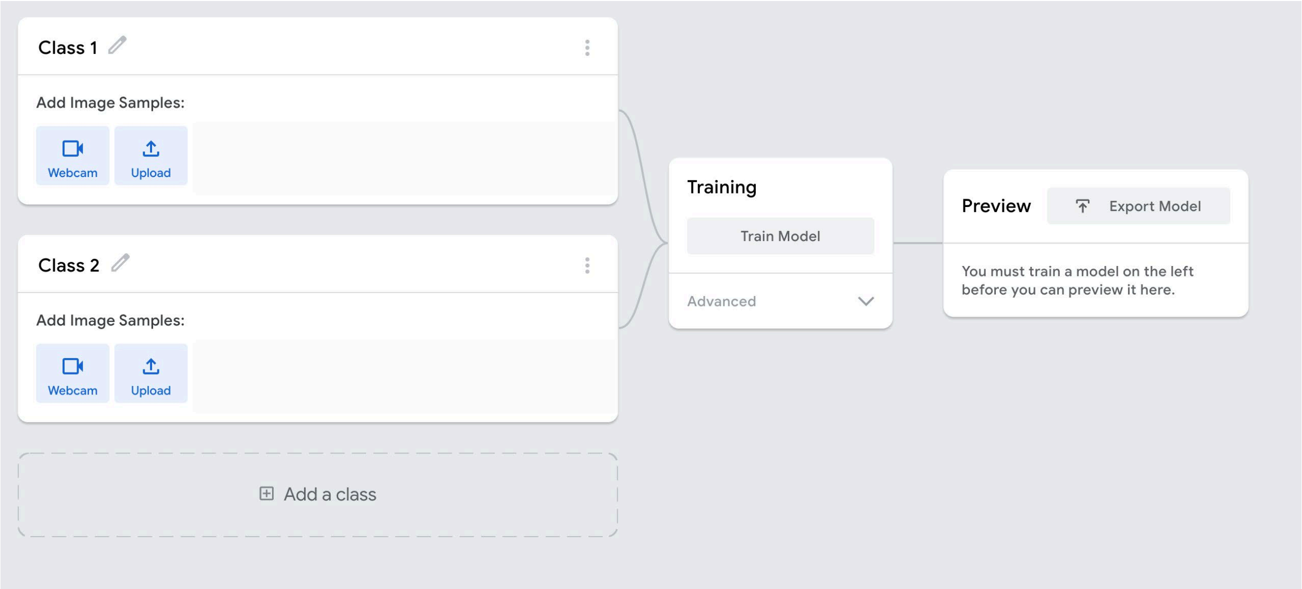
Convolutional Nets



Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

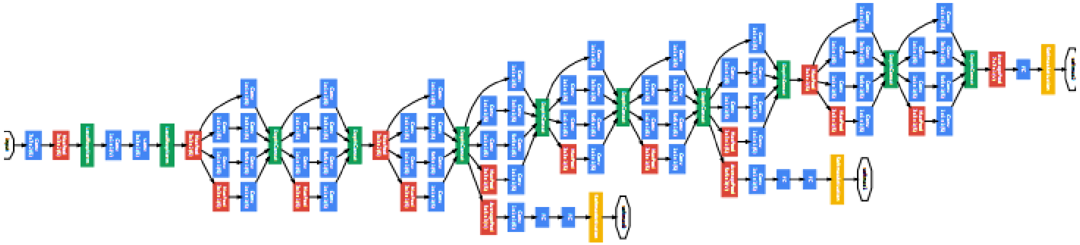
Demo (Teachable Machines)

https://teachablemachine.withgoogle.com/



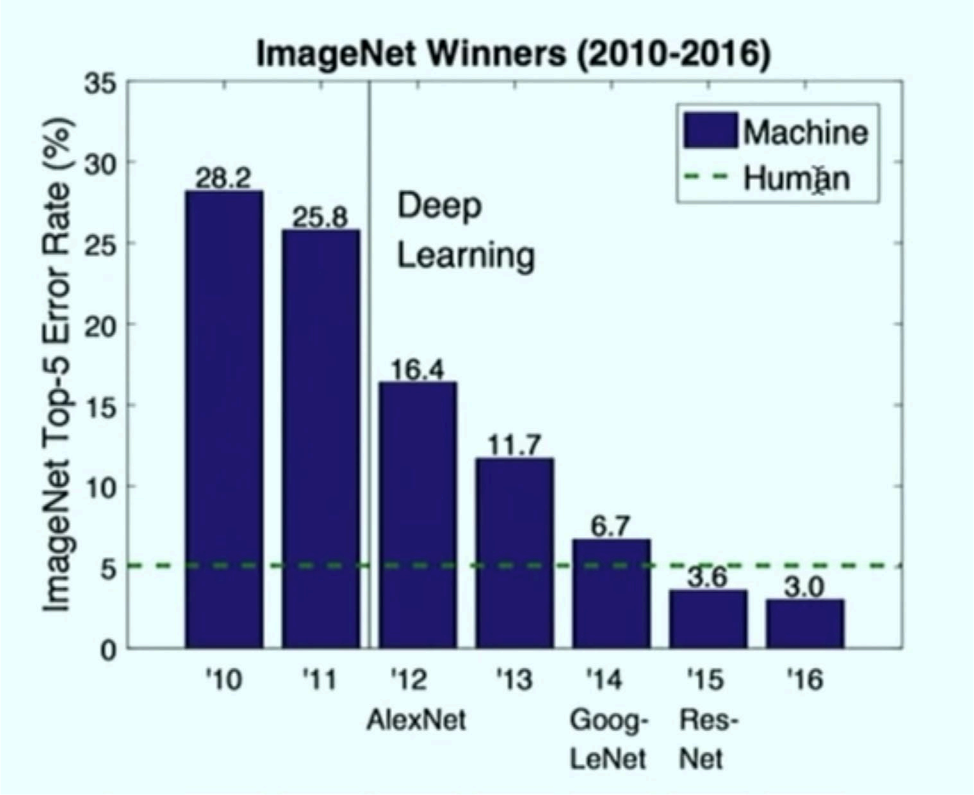
ConvNet roots

- Fukushima, 1980s designed network with same basic structure but did not train by backpropagation.
- The first successful applications of Convolutional Networks by Yann LeCun in 1990's (LeNet)
  - Was used to read zip codes, digits, etc.
- Many variants nowadays, but the core idea is the same
  - Example: a system developed in Google (GoogLeNet)
    - Compute different filters
    - Compose one big vector from all of them
    - Layer this iteratively



See more: http://arxiv.org/pdf/1409.4842v1.pdf

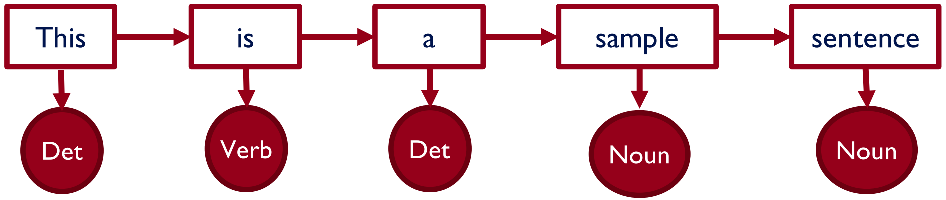
Depth matters



Slide from Michael Collins

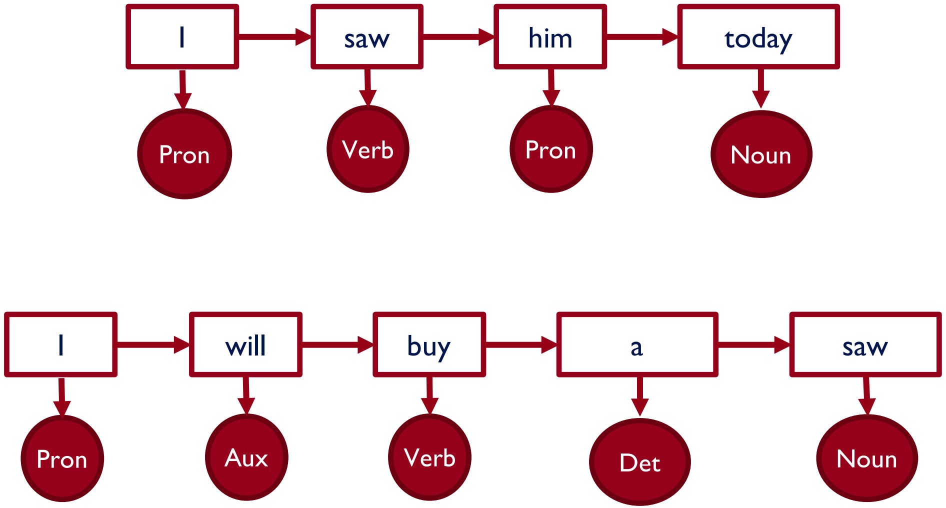
Natural Language Processing

- Word-level prediction on natural language:
  - Example: Part of Speech tagging words in a sentence



  - Challenges:
    - Structure in the input: Dependence between different parts of the inputs
    - Structure in the output: Correlations between labels
    - Variable size inputs: e.g. sentences differ in size

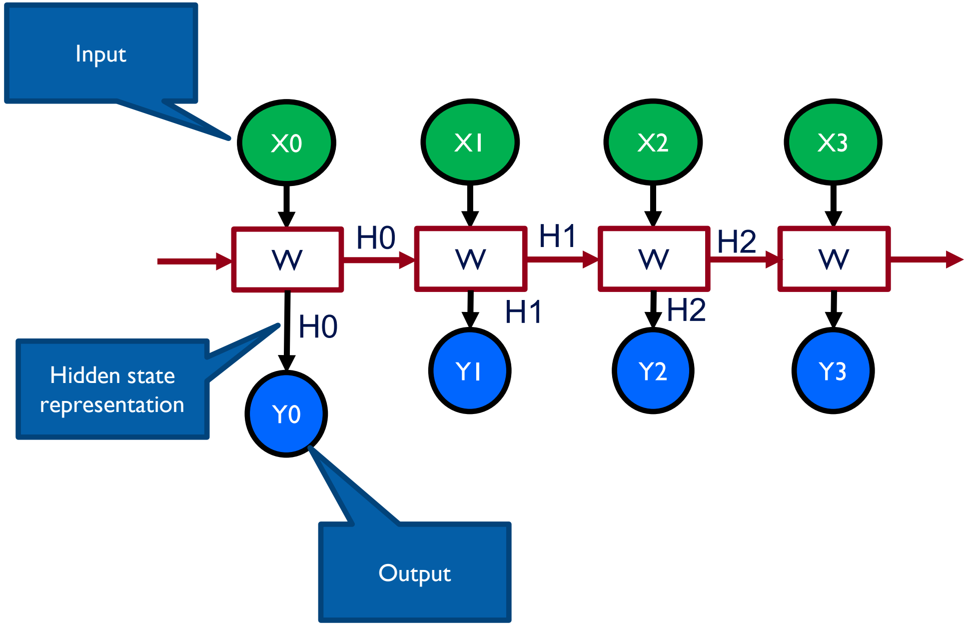
Natural Language Processing



How would you go about solving this task?

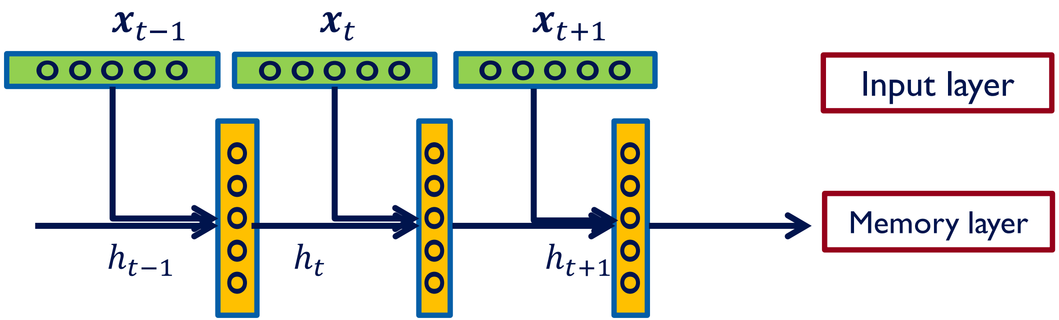
Recurrent Neural Networks

- Infinite uses of finite structure



Recurrent Neural Networks

- A chain RNN:
  - Each input is replaced with its vector representation $\boldsymbol{x}_t$
  - Hidden (memory) unit h_t contain information about previous inputs and previous hidden units h_{t-1}, h_{t-2}, etc
    - Computed from the past memory and current word. It summarizes the sentence up to that time.

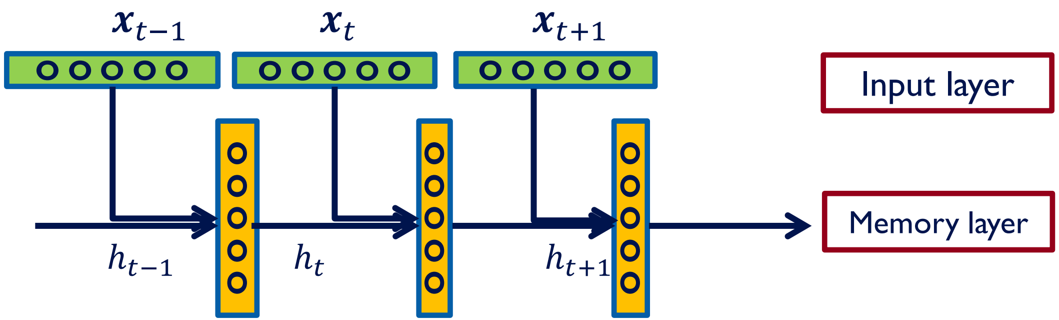


Recurrent Neural Networks

- A popular way of formalizing it:

$$h_t = f(W_h h_{t-1} + W_i x_t)$$

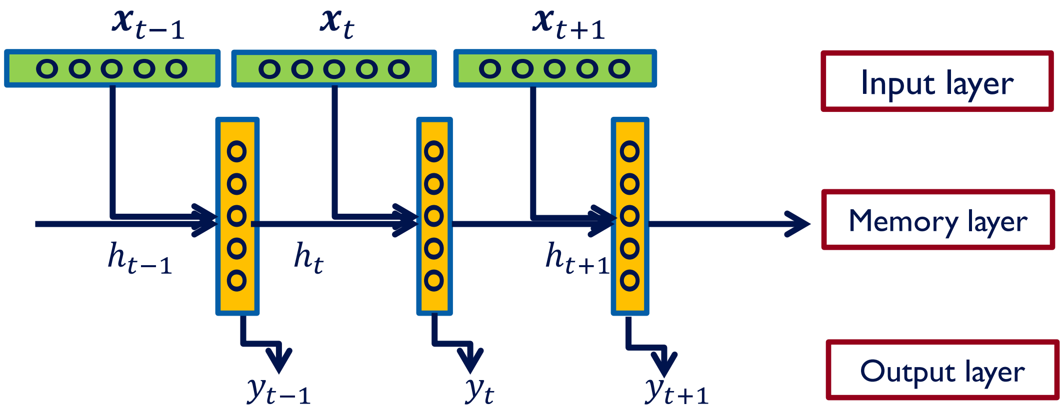
  - Where f is a nonlinear, differentiable (why?) function.
- Outputs?
  - Many options; depending on problem and computational resource



Recurrent Neural Networks

- Prediction for $\boldsymbol{x}_t$, with h_t:

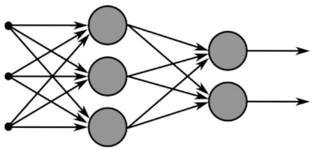
$$y_t = \text{softmax}(W_o h_t)$$



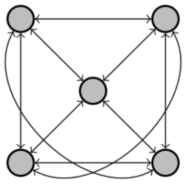
- Some inherent issues with RNNs:
  - Recurrent neural nets cannot capture phrases without prefix context
  - They focus too much on last words in final vector
    - A slightly more sophisticated solution: Long Short-Term Memory (LSTM) units

Recurrent Neural Networks

- Multi-layer feed-forward NN: DAG
  - Just computes a fixed sequence of
  - non-linear learned transformations to convert an input patter into an output pattern

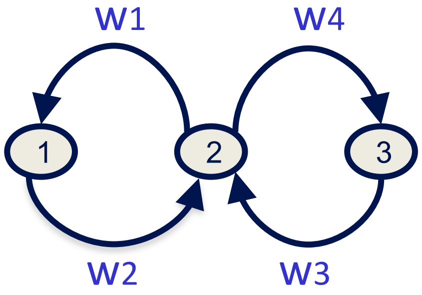


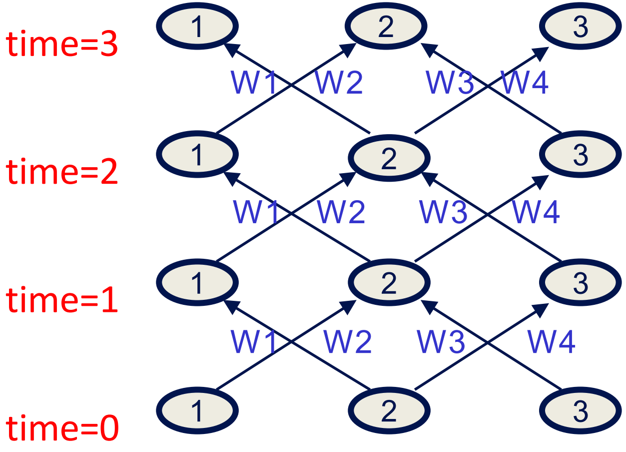
- Recurrent Neural Network: Digraph
  - Has cycles.
  - Cycle can act as a memory;
  - The hidden state of a recurrent net can carry along information about a “potentially” unbounded number of previous inputs.
  - They can model sequential data in a much more natural way.



Equivalence between RNN and Feed-forward NN

- Assume that there is a time delay of 1 in using each connection.
- The recurrent net is just a layered net that keeps reusing the same weights.

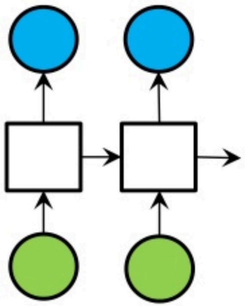




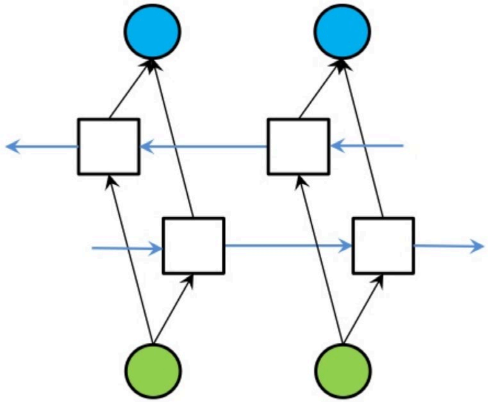
Slide Credit: Geoff Hinton

Bi-directional RNN

- One of the issues with RNN:
  - Hidden variables capture only one side context
- A bi-directional structure

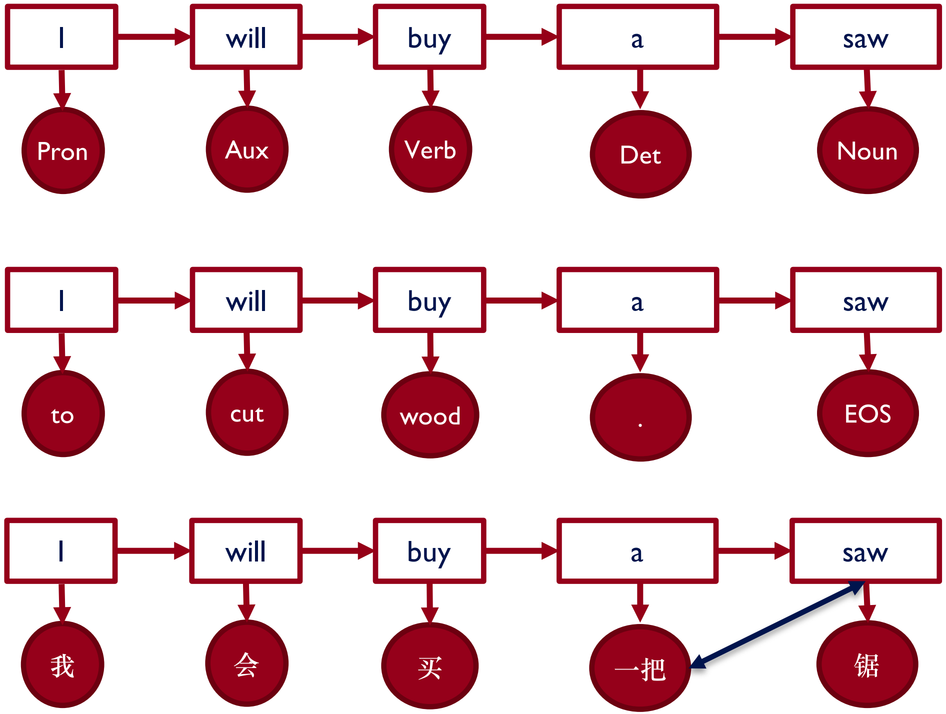


RNN



Bi-directional RNN

Sequence to sequence models

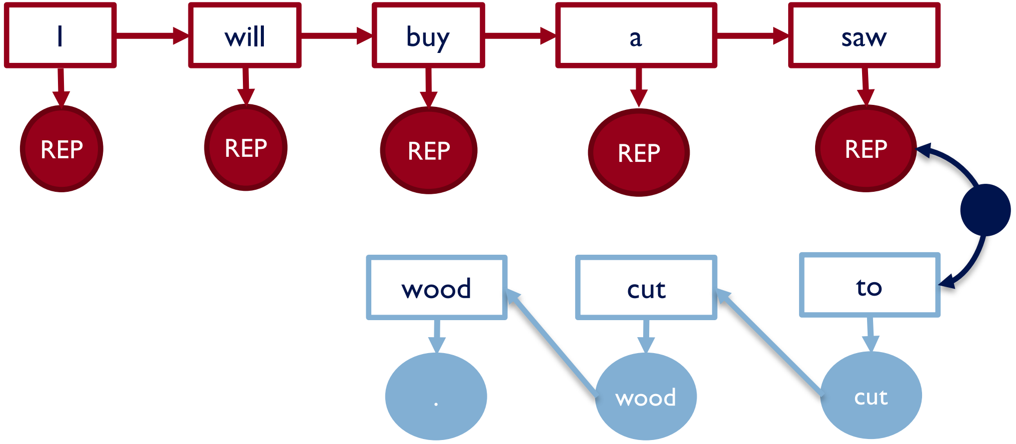


Works!

What about other endings?

What if prediction depends on the future?

Sequence to sequence models

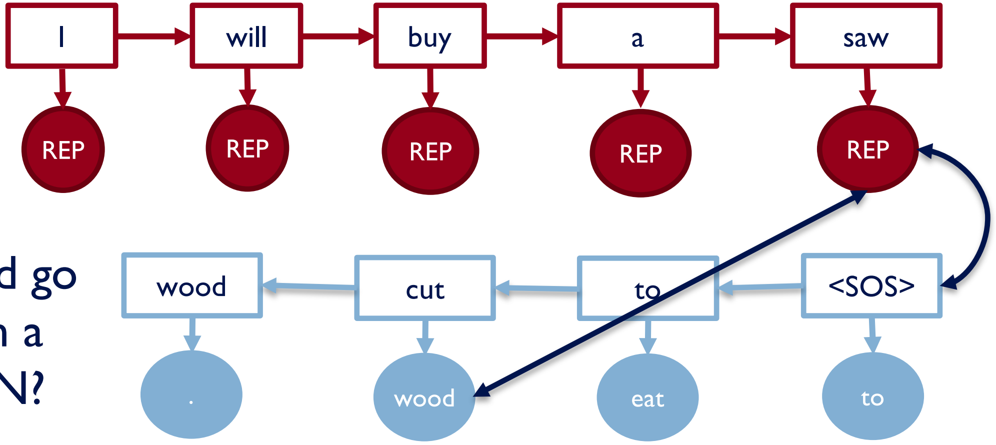


encoder

decoder

How do we train?

Sequence to sequence models



encoder

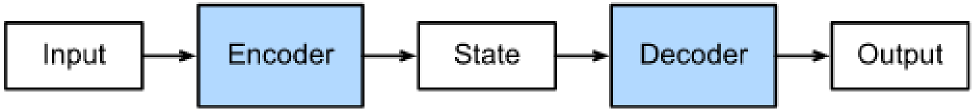
decoder

What could go wrong with a simple RNN?

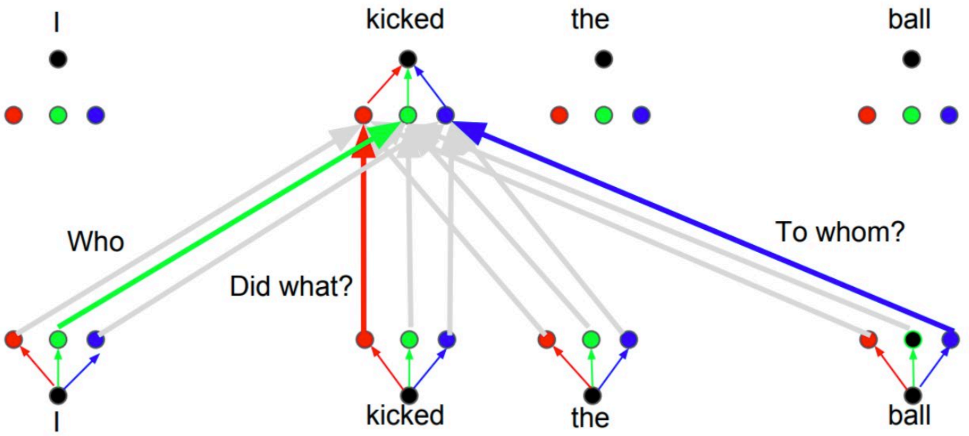
Long range dependencies!

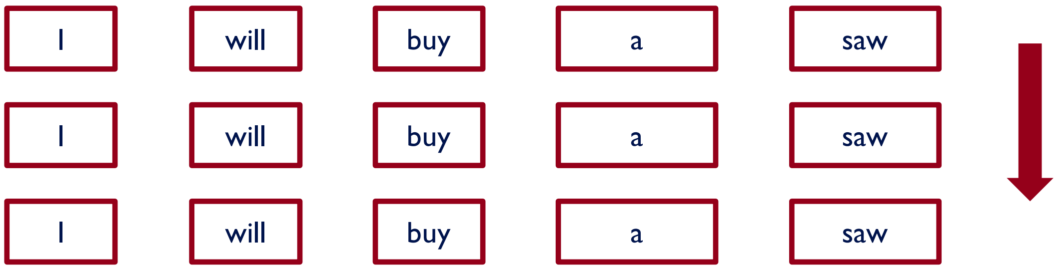
Long Short Term Memory networks (LSTM)

Doesn't have to be an RNN!



Self-Attention and Transformers





Transformers: Many attention layers stacked

Seq2seq with attention

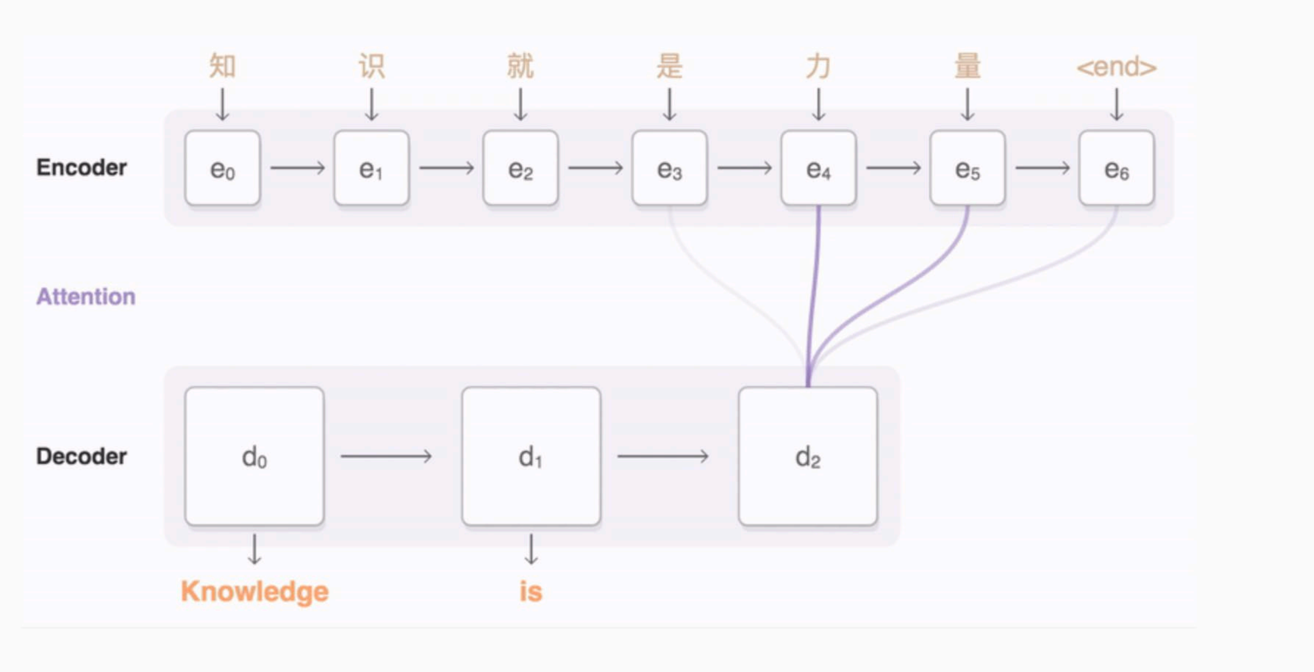
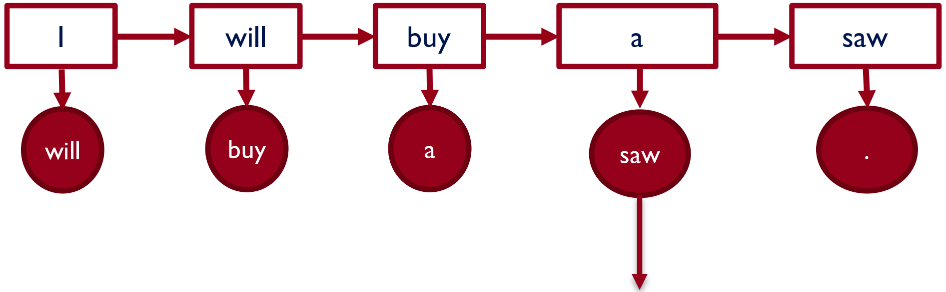


Figure Credit: Google Open Source

Unsupervised (Pre-) training

- Motivation: representation learning and transfer learning

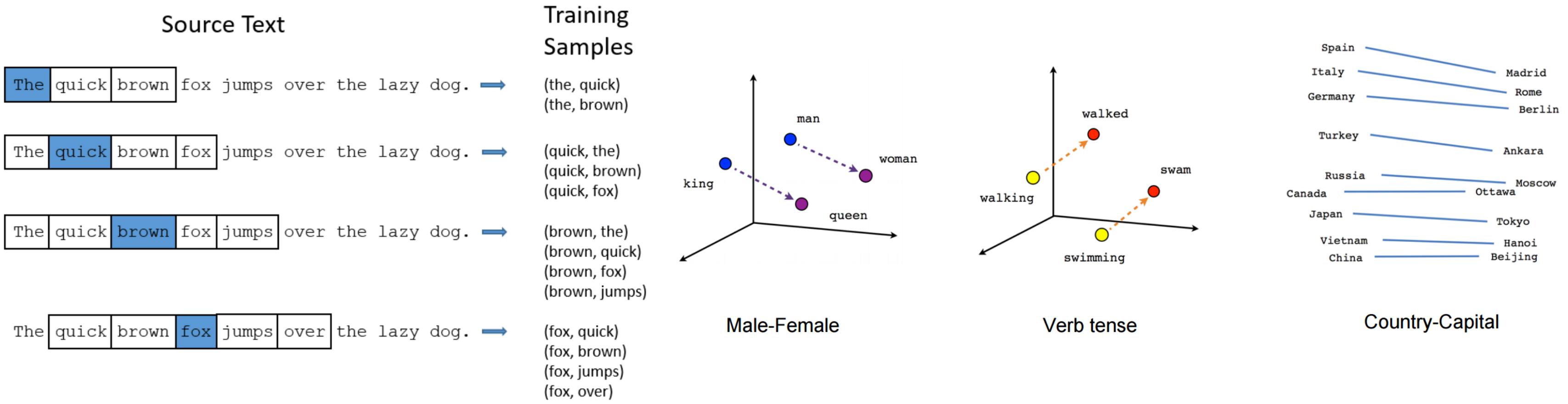


A vector that represents something buyable.

In part-of-speech: a noun!

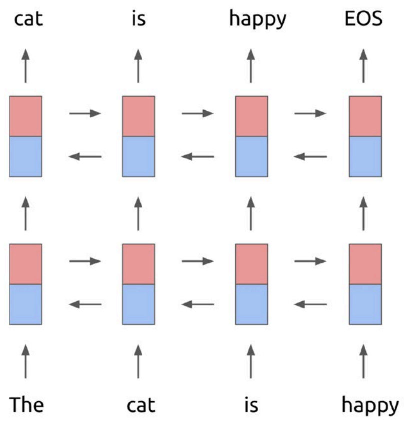
Unsupervised (Pre-) training

- Early works
  - Word embeddings from N-grams (Mikolov 2013)



Unsupervised (Pre-) training

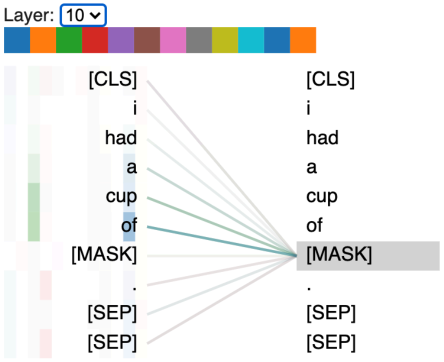
- Early works
  - Word embeddings from N-grams (Mikolov 2013)
- Contextualized embeddings
  - ELMo (Peters et al. 2018), a bi-directional RNN



In training: only predict next words in the forward run or previous words in the backward run.

Unsupervised (Pre-) training

- Early works
  - Word embeddings from N-grams (Mikolov 2013)
- Contextualized embeddings
  - ELMo (Peters et al. 2018), a bi-directional RNN
  - Bert (Devlin et al. 2018), a transformer (many layers of attentions)



→ some drink that goes in a cup

77.9% **coffee**
17.1% **tea**
0.8% **water**
0.7% **wine**
0.3% **milk**

Reflects a problem! Ideas?

Unsupervised (Pre-) training

- Early works
  - Word embeddings from N-grams (Mikolov 2013)
- Contextualized embeddings
  - ELMo (Peters et al. 2018), a bi-directional RNN
  - Bert (Devlin et al. 2018), a transformer (many layers of attentions)
- All of them (any many others)
  - Unsupervised; used as much data as there is
  - Contributed to a big part of NLP progress in the past decade

Unsupervised (Pre-) training in vision

- The computer vision community also uses a similar spirit to learn general representations of images before a specific task
- ImageNet
  - 14 million images of objects, 21,841 potential fine-grained labels
  - Initializes "good" convolution filters or other layers in a model
- Transfers to many other tasks
  - Even chest radiology!