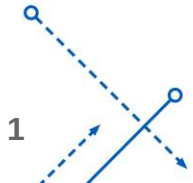
University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

CSE 4/573

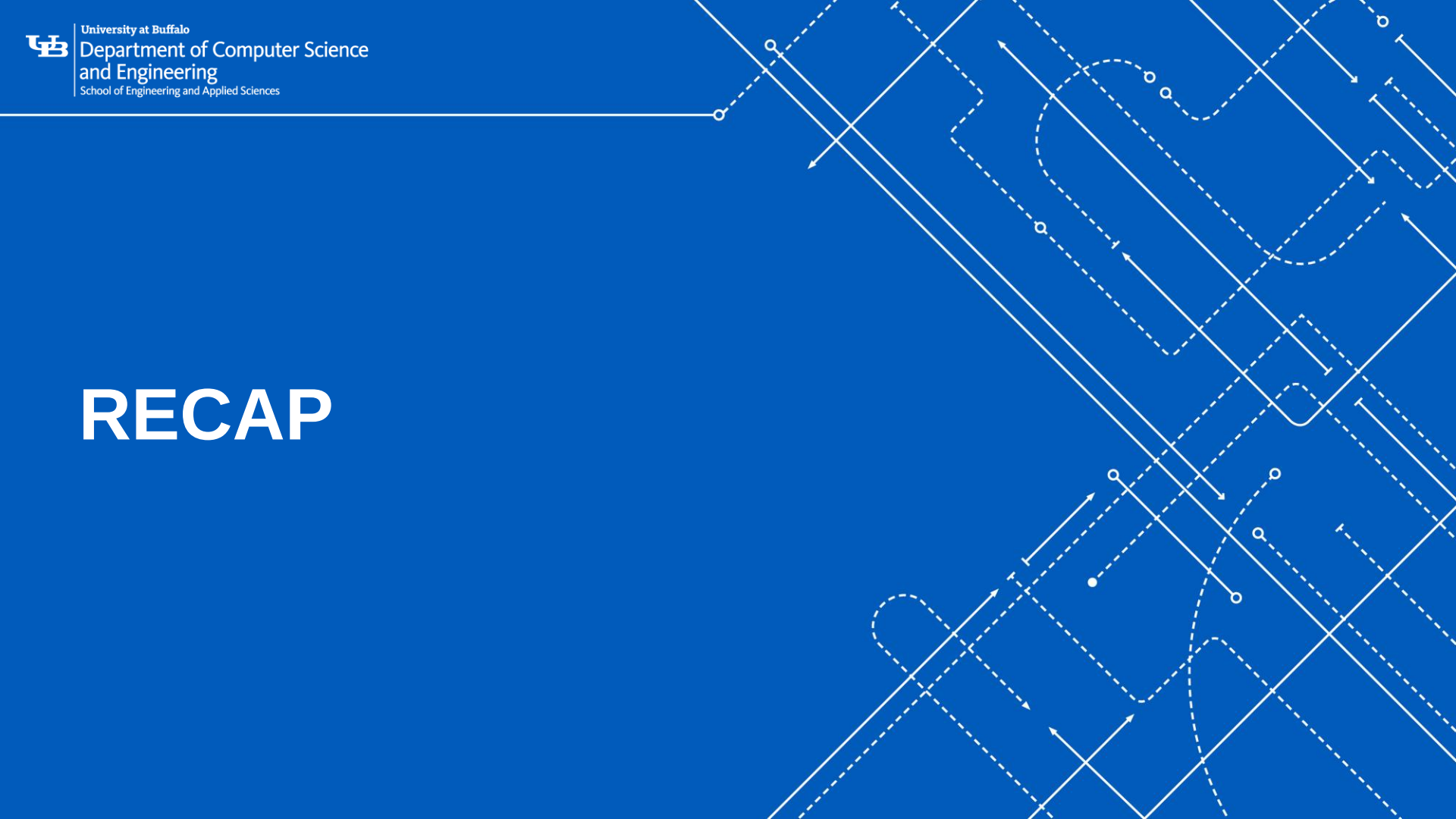
COMPUTER VISION AND IMAGE PROCESSING

Fall 2024

Instructor: Varun Shijo



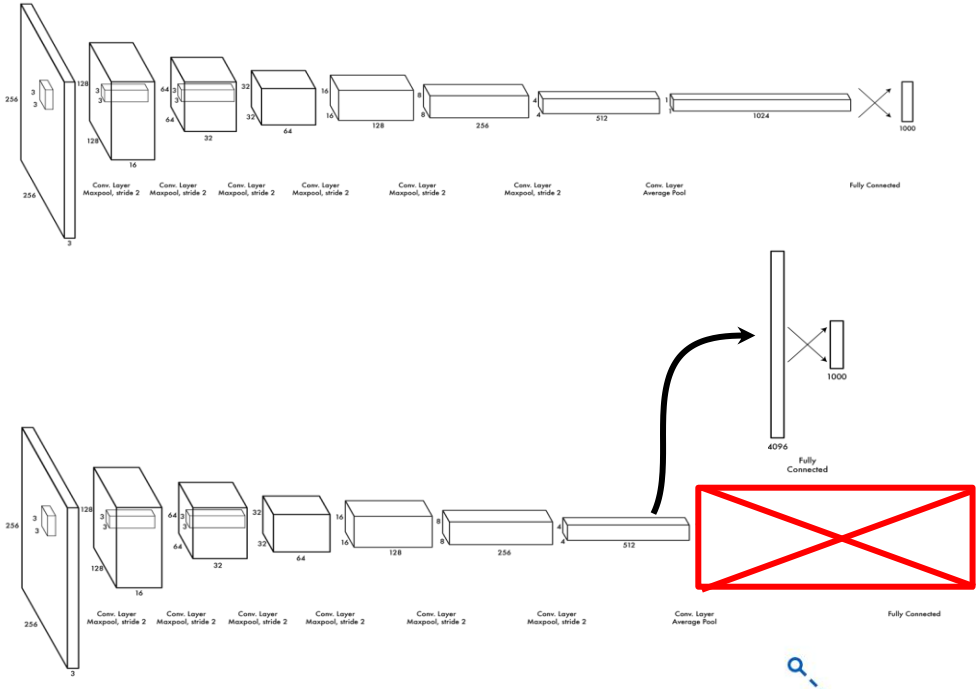
University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences



RECAP

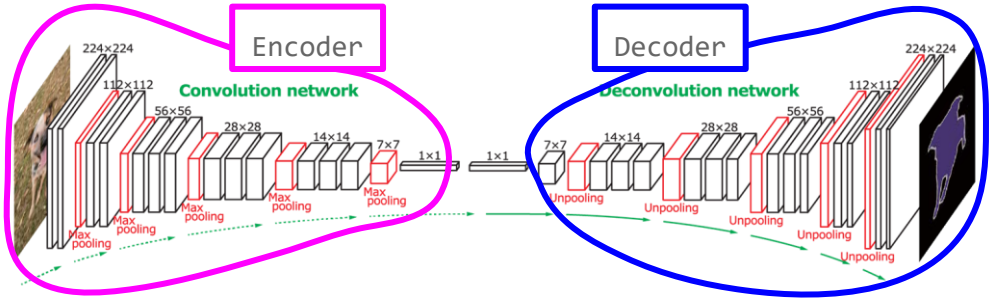
Transfer Learning

- We can assume that models pretrained on ImageNet, PASCAL, COCO etc. have good feature extraction weights.
- We should be able to use these to solve other problems.
  - Especially useful when working with small datasets



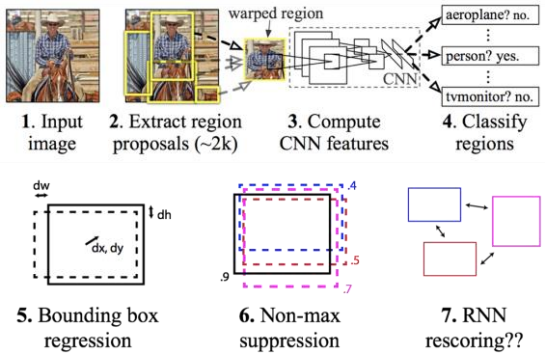
Deep Learning Formulations

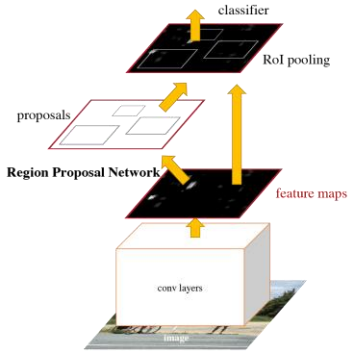
- Semantic Segmentation
  - Pixel level labels
    - Classes are integers or channels
  - Encoder Decoder pattern
    - Encoder extracts features that distill fine-grained information from the input
    - Decoder reconstructs fine-grained features from coarse decoder output
- Object Detection
  - Two stage
  - Single stage



Deep Learning Formulations

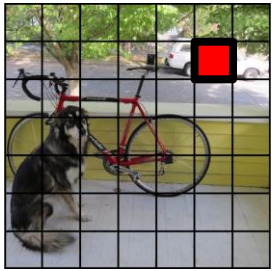
- Semantic Segmentation
- Object Detection
  - Two stage
    - Region proposal
      - Sliding window
      - Selective search
      - Region Proposal Network
    - Classification
  - Single stage
    - Joint prediction of bounding box and class with confidence

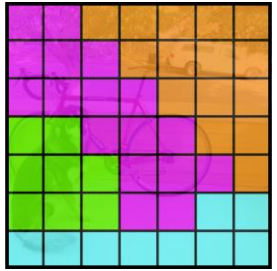


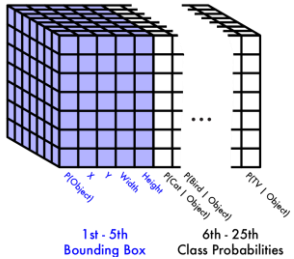


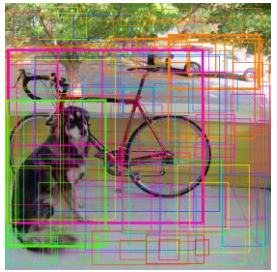
Deep Learning Formulations

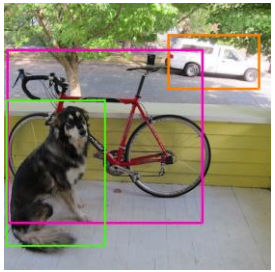
- Semantic Segmentation
- Object Detection
  - Two stage
    - Region proposal
      - Sliding window
      - Selective search
      - Region Proposal Network
    - Classification
  - Single stage
    - Joint prediction of bounding box and class with confidence





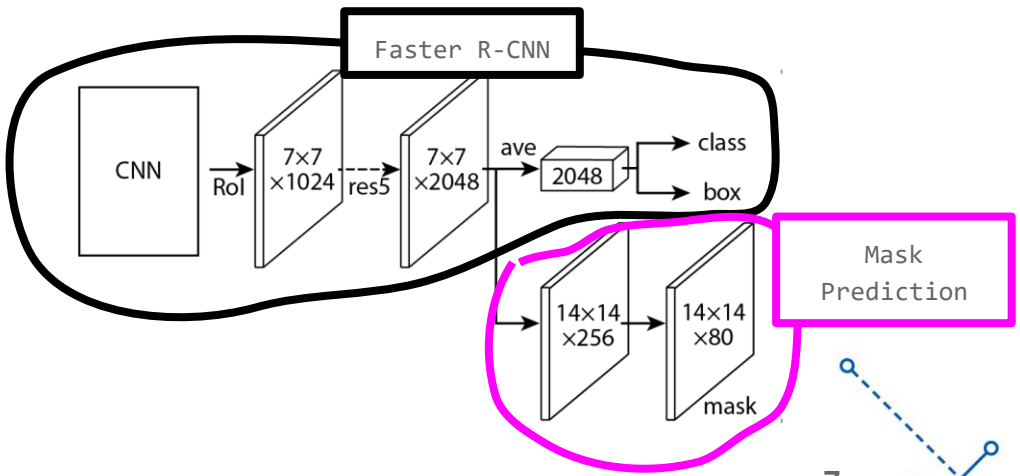






Deep Learning Formulations

- Semantic Segmentation
- Object Detection
- Instance Segmentation
  - Builds on the ability to identify ROIs containing objects and the ability to classify pixels



University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

BACKGROUND

Building up to Image Generation – Language Modeling

- COCO dataset also has captions!
  - 5 captions per image
  - Detection/segmentation is (maybe) just pattern matching
  - To caption an image maybe you really have to *understand* it
  - Need to model both visual information and *language*

The man at bat readies to swing at the pitch while the umpire looks on.

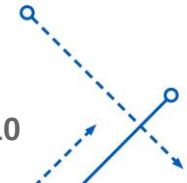
A large bus sitting next to a very tall building.

A horse carrying a large load of hay and two people sitting on it.

Bunk bed with a narrow shelf sitting underneath it.

Why is language modeling challenging?

- *Temporal dependence*

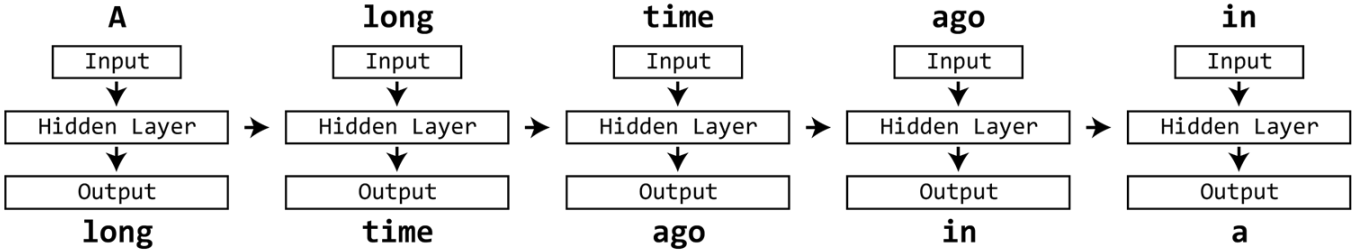


A long time ago in a _____
far,
far away….

A long time ago in a galaxy
far,
far away….

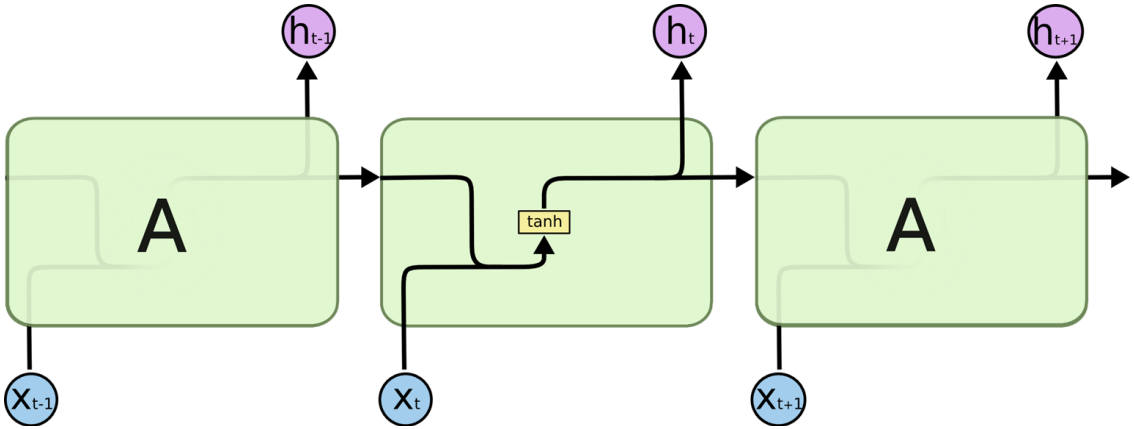
Recurrent neural network

- Handle sequential data
- Idea:
  - Read one token (word, character, etc) at a time.
  - Produce output
  - Also update internal memory



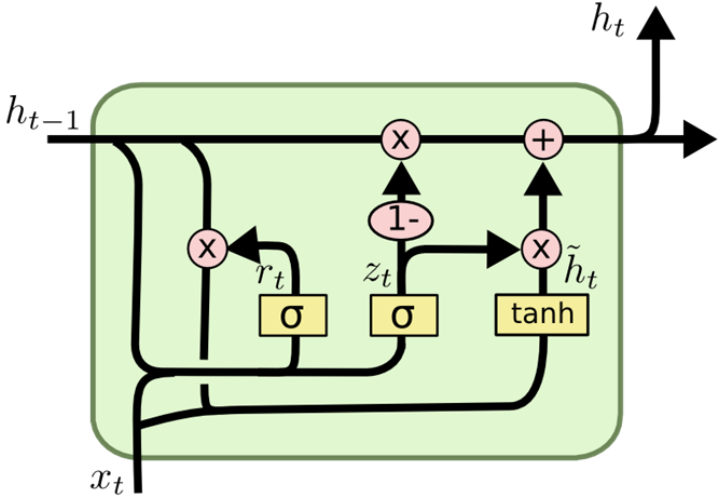
Vanilla RNN

- Given input $\mathbf{x}_t$, previous memory $\mathbf{h}_{t-1}$; produce output $\mathbf{y}_t$
- In practice, append x_t to h_{t-1} and use one set of weights
- $\mathbf{h}_t = \varphi(\mathbf{w}\cdot[\mathbf{x}_t,\mathbf{h}_{t-1}])$
- $\mathbf{y}_t = \mathbf{h}_t$



GRU: Gated recurrent units

Lots to unpack



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

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

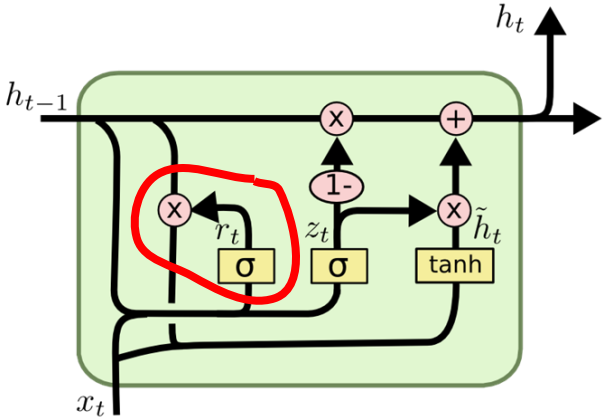
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

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

https://arxiv.org/pdf/1406.1078.pdf, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Reset gate: ignore some memory

- Rather than computing memory from scratch every recurrence, gate memory to only update parts.



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

Figure out what parts of memory to pay attention to, what to ignore

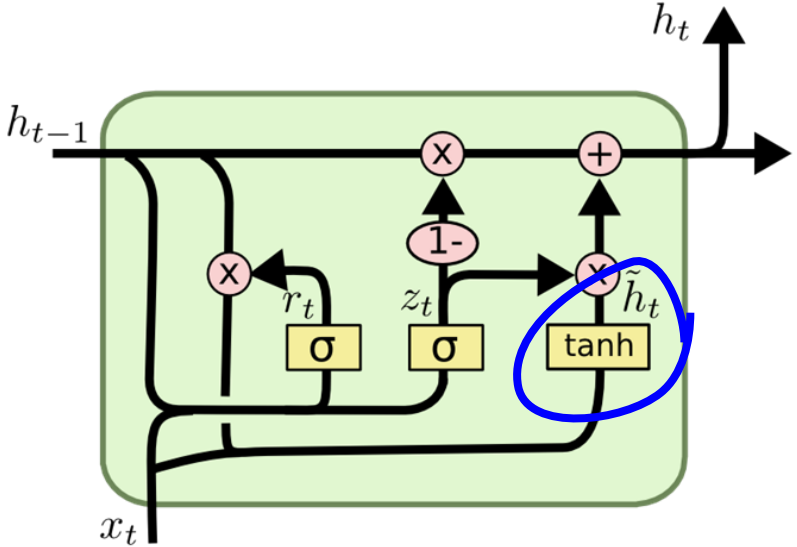
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

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

https://arxiv.org/pdf/1406.1078.pdf, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Calculate update



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

$$r_t = \sigma\left(W\right.$$

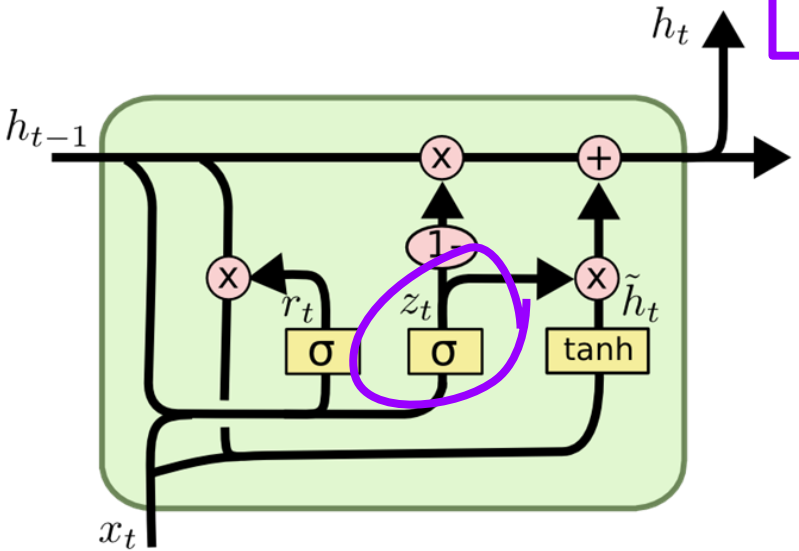
Calculate update using input and the important parts of memory

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

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

https://arxiv.org/pdf/1406.1078.pdf, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Update gate: what to save/update



Calculate what parts of memory to save, what to replace

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

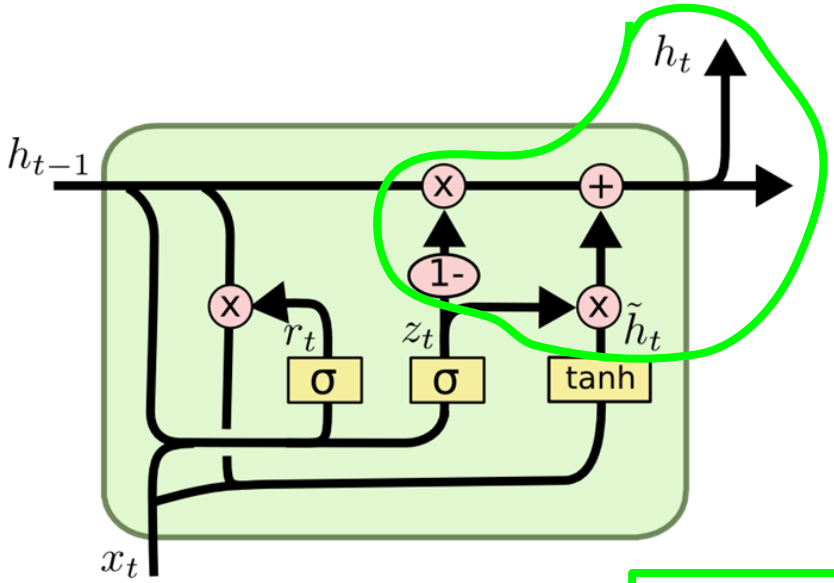
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

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

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

https://arxiv.org/pdf/1406.1078.pdf, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Output: weighted sum



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

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

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

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

Output is weighted sum of previous output and "new" output

https://arxiv.org/pdf/1406.1078.pdf, http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Language modeling: what's next

Given a string of words/characters, what's the next one

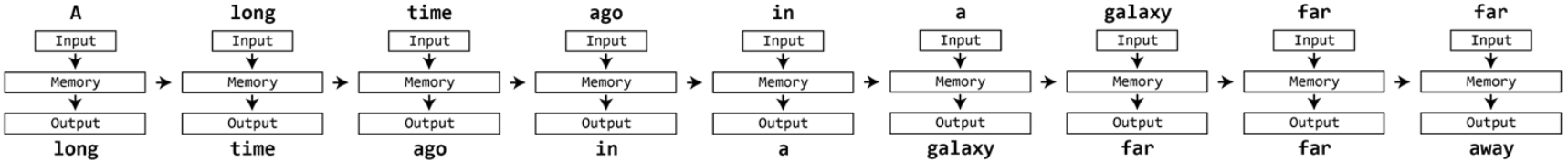
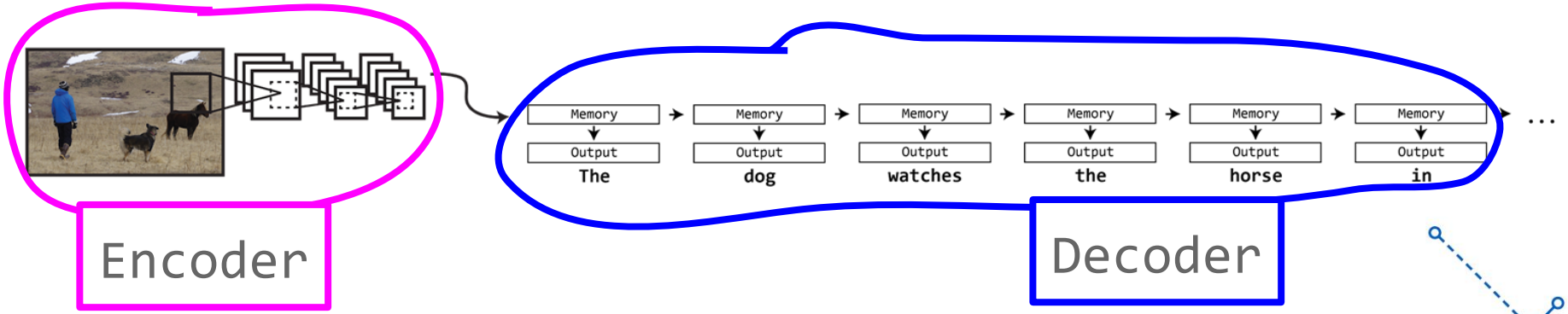


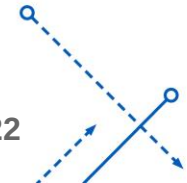
Image captioning

- Given an image:
  - Extract features using CNN
  - Feed into RNN
  - Generate sentences!



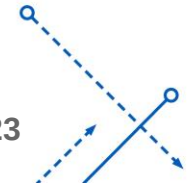
Why is language modeling challenging?

- *Temporal dependence*
- *Scoring*



Evaluating caption predictions

- How do we know what a “good” caption even is? Not like scoring detection or classification!
- Automated scoring, BLEU, METEOR, etc.
- “Demo” scoring methods



https://arxiv.org/pdf/1505.04467.pdf

Visual Question Answering

- Given image and question...
  - Answer it!
- Harder than captioning?
- Requires more understanding?
- Easier to evaluate!

Situation Recognition

- Images often have one main thing going on, one verb
- Recognize that verb and what sense it's being used in, fill in the other important objects and how they relate in a linguistic *frame*




CLIPPING

ROLE	VALUE
AGENT	MAN
SOURCE	SHEEP
TOOL	SHEARS
ITEM	WOOL
PLACE	FIELD

ROLE	VALUE
AGENT	VET
SOURCE	DOG
TOOL	CLIPPER
ITEM	CLAW
PLACE	ROOM

JUMPING

ROLE	VALUE
AGENT	BOY
SOURCE	CLIFF
OBSTACLE	-
DESTINATION	WATER
PLACE	LAKE

ROLE	VALUE
AGENT	BEAR
SOURCE	ICEBERG
OBSTACLE	WATER
DESTINATION	ICEBERG
PLACE	OUTDOOR

Why is language modeling challenging?

- *Temporal dependence*
- *Scoring*
- *Representation*

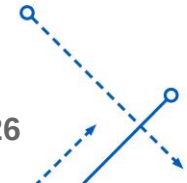
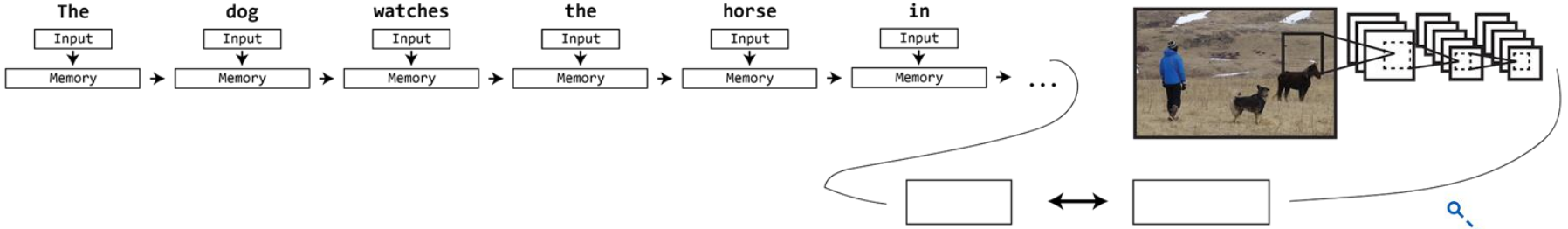


Image retrieval

- Given a sentence:
- Extract representation using RNN
- Find matching representations from images processed with CNN



Word2Vec: Trained word embeddings

- 1 layer neural network
- Predict next word from previous (1-hot encoding)
- Hidden layer << vocabulary size
- Train w/ SGD, etc.

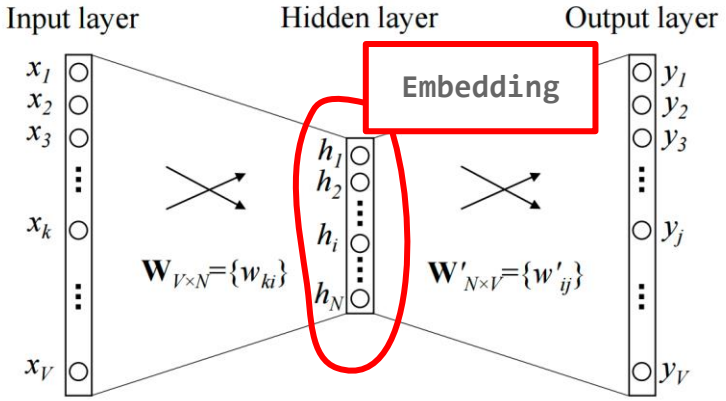
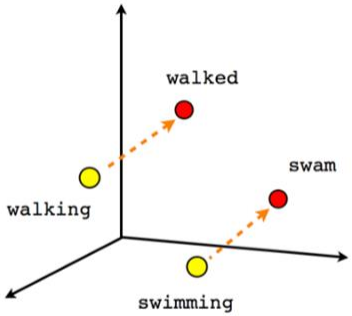


Figure 1: A simple CBOW model with only one word in the context

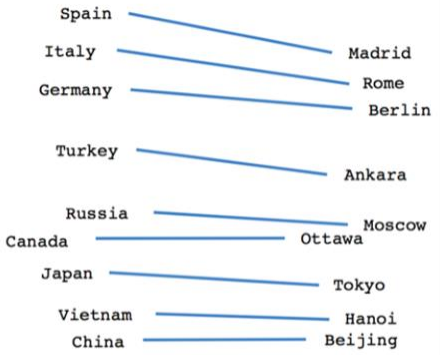
https://arxiv.org/pdf/1301.3781.pdf

Word2Vec: Trained word embeddings

- Embedding “math” works!
- Nearest(f(Madrid) - f(Spain) + f(Italy)) = f(Rome)



Verb tense

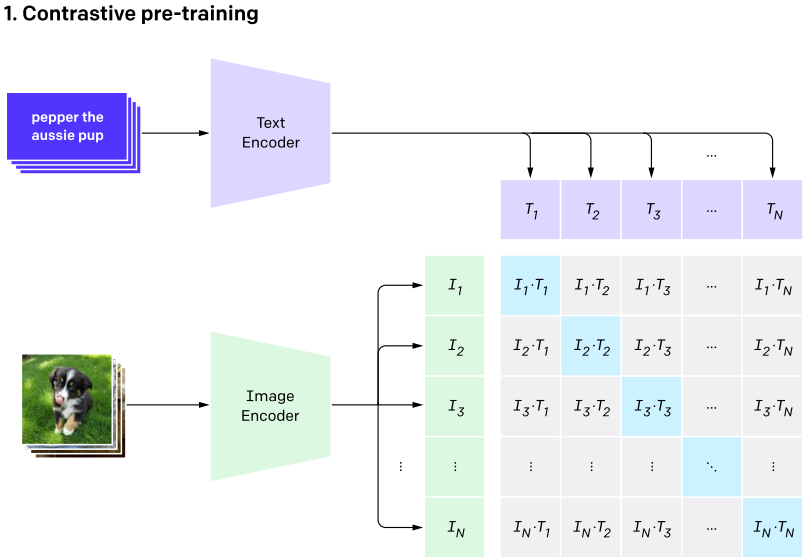


Country-Capital

https://arxiv.org/pdf/1301.3781.pdf

Joint Vision-Language Embedding Space

- CLIP



University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

GENERATIVE ADVERSARIAL NETWORKS

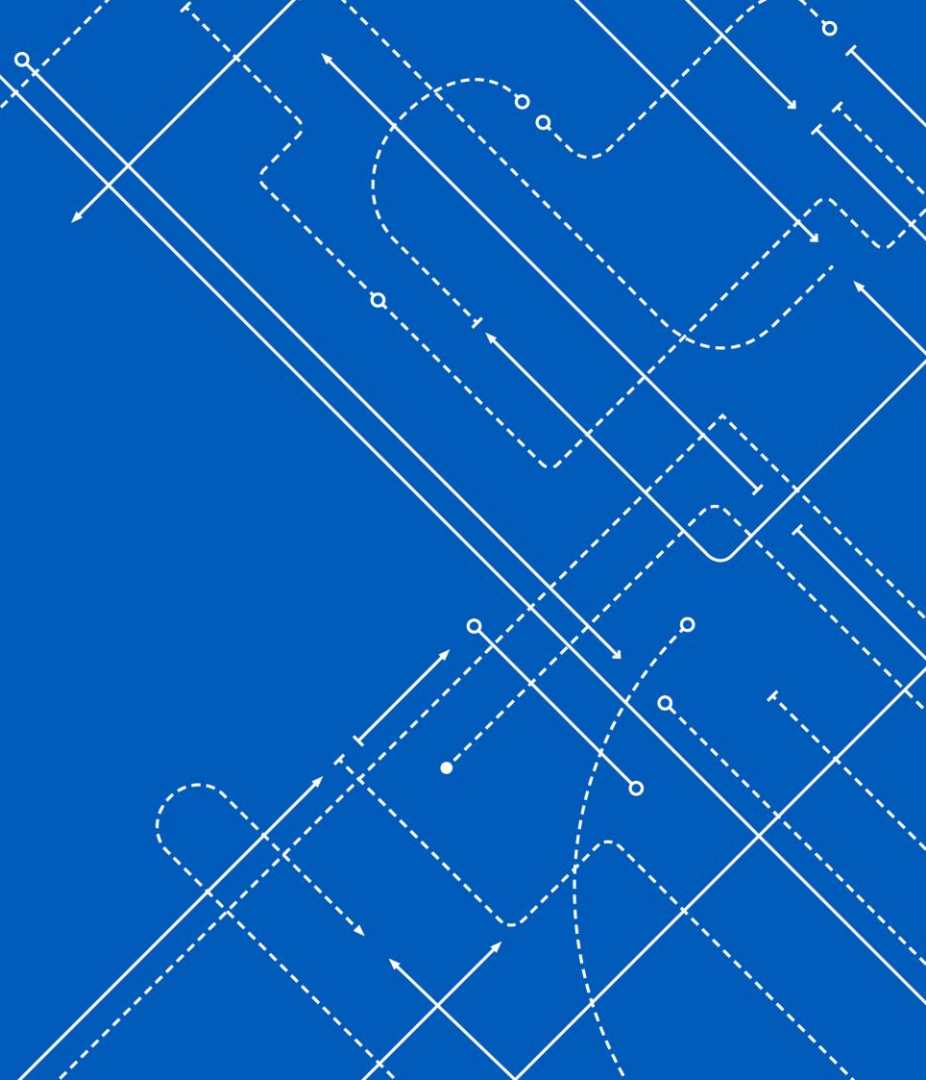
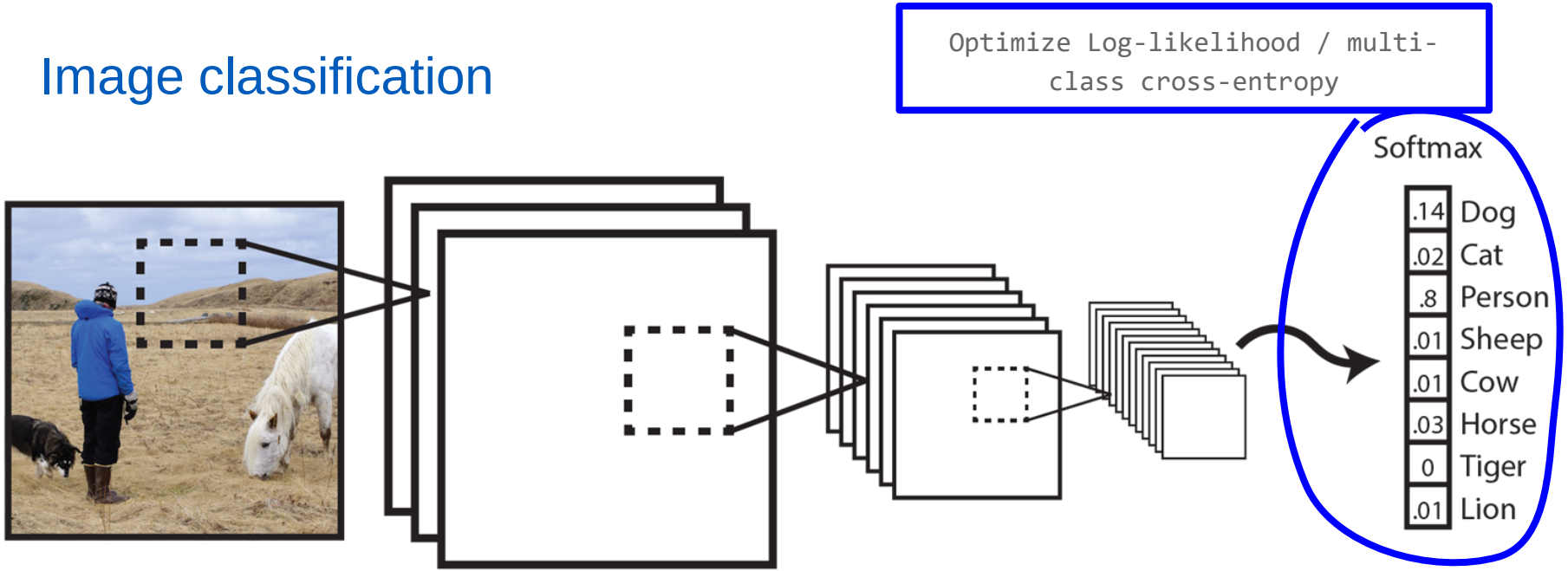
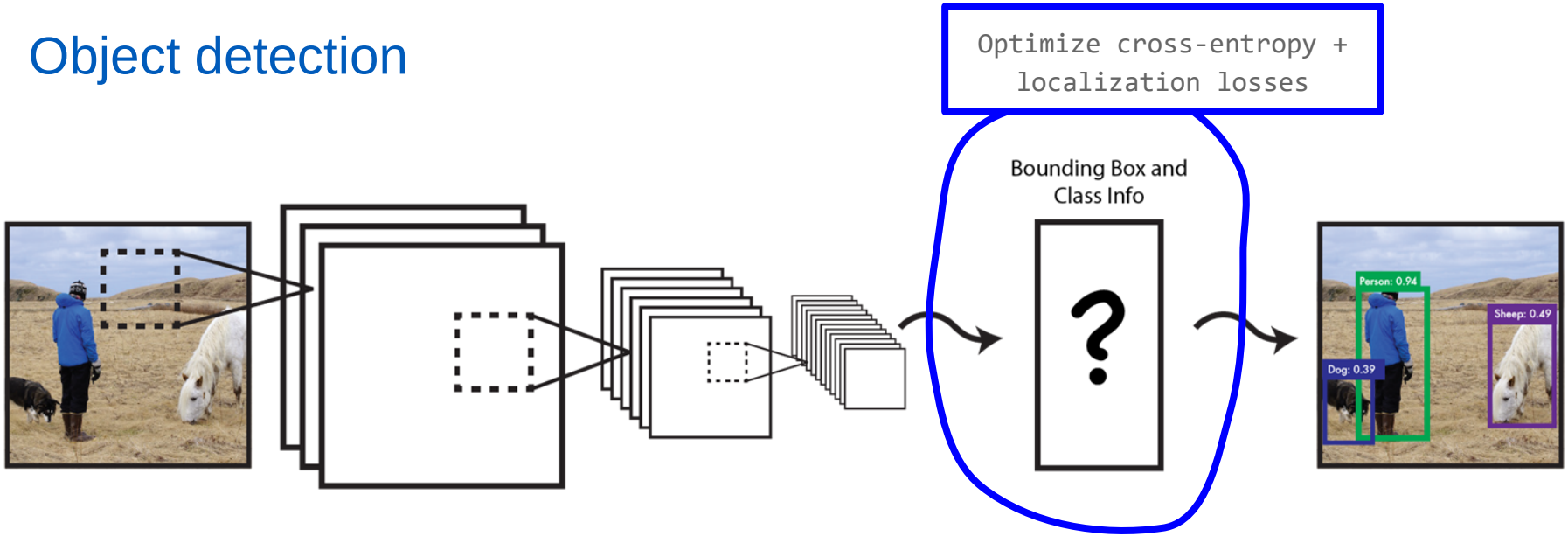


Image classification



Object detection



Semantic segmentation

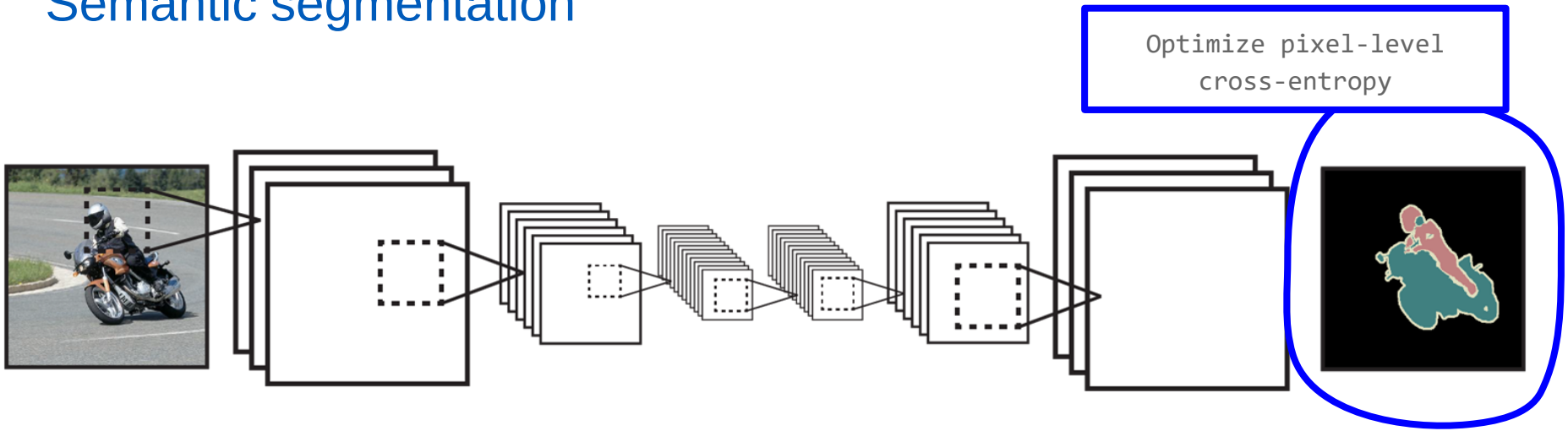
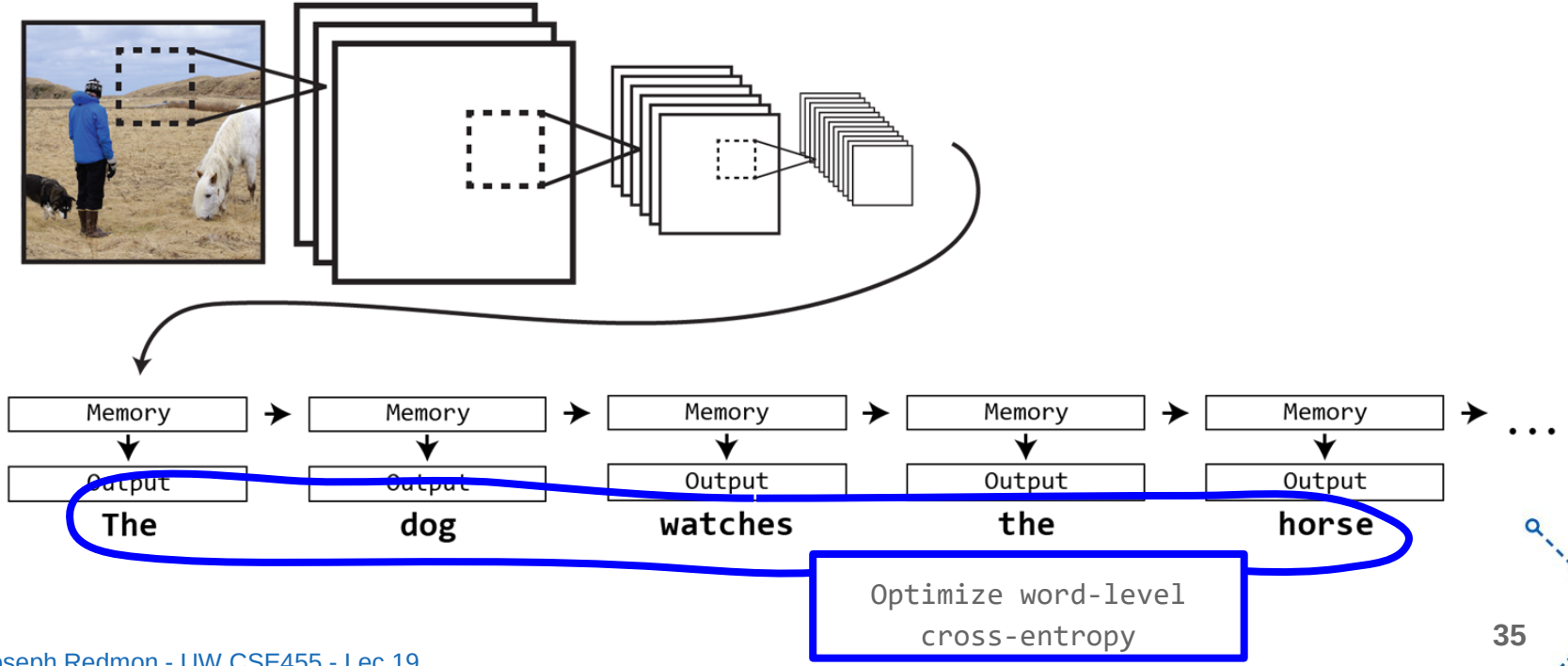
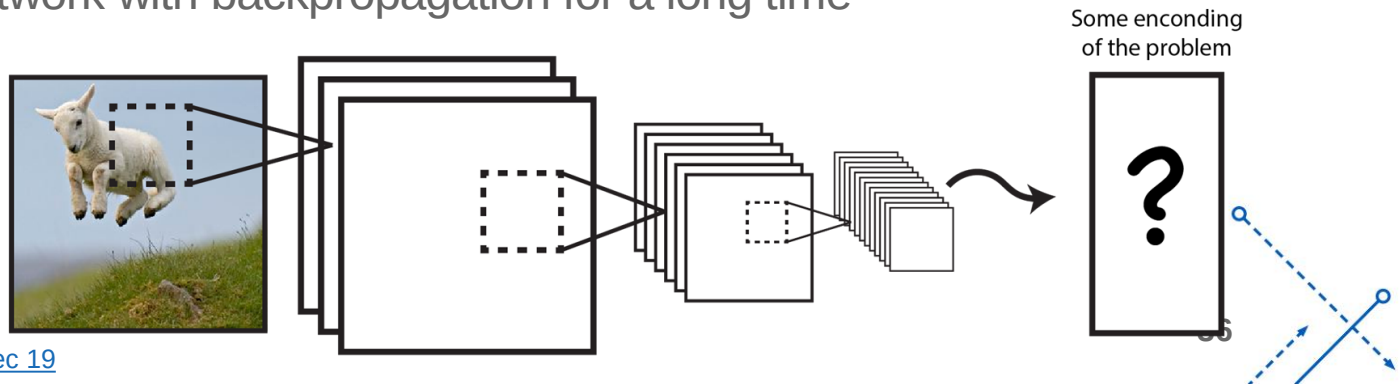


Image captioning



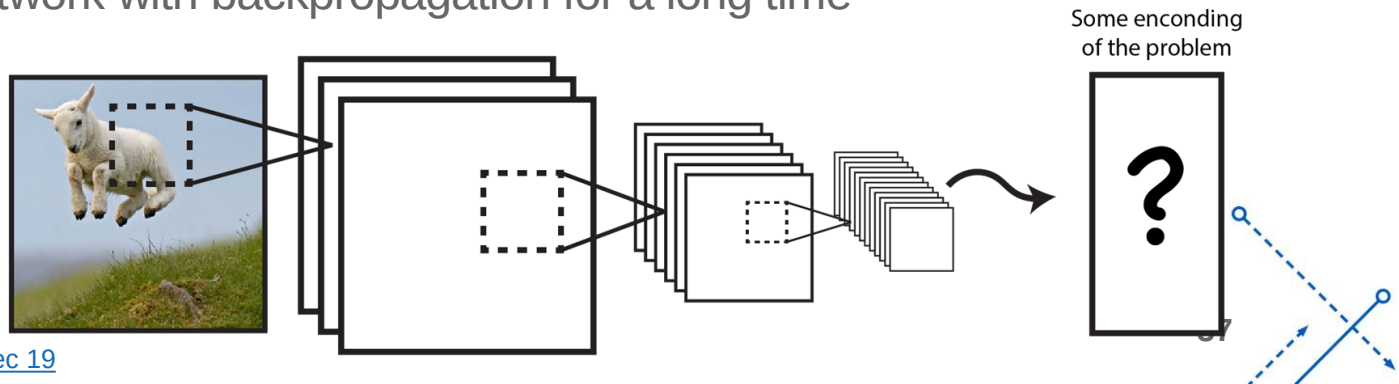
How to solve any vision problem

1. Pick a neural network architecture
2. Design an encoding of the expected output
3. Pick a loss function for that encoding (squared error? Log-likelihood?)
4. Gather a bunch of training data (and label it)
5. Train your network with backpropagation for a long time



How to solve any vision problem

1. Pick a neural network architecture
2. Design an encoding of the expected output
3. Pick a loss function for that encoding (squared error? Log-likelihood?)
4. Gather a bunch of training data (and label it)
5. Train your network with backpropagation for a long time



How to solve any vision problem

1. Pick a neural network architecture
2. Design an encoding of the expected output
3. Pick a loss function for that encoding (squared error? Log-likelihood?)
4. Gather a bunch of training data (and label it)
5. Train your network with backpropagation for a long time

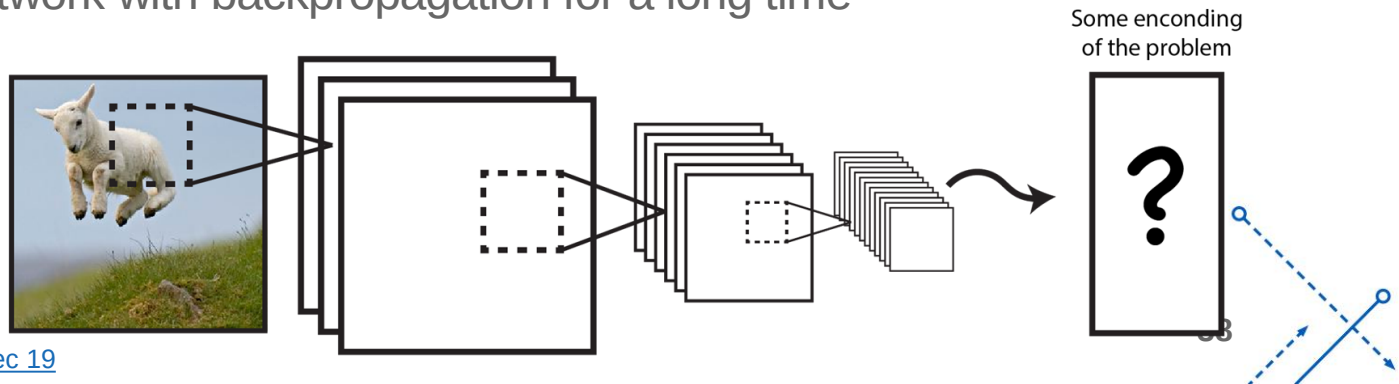


Image colorization

- Training is easy, grayscale a bunch of images, try to predict the original one!
- But there's a problem...

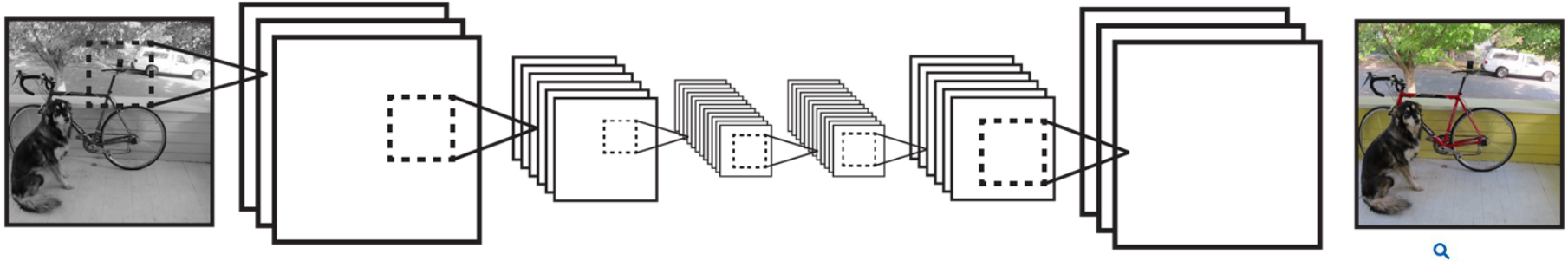


Image colorization

- Training is easy, grayscale a bunch of images, try to predict the original one!
- But there's a problem...

Image colorization

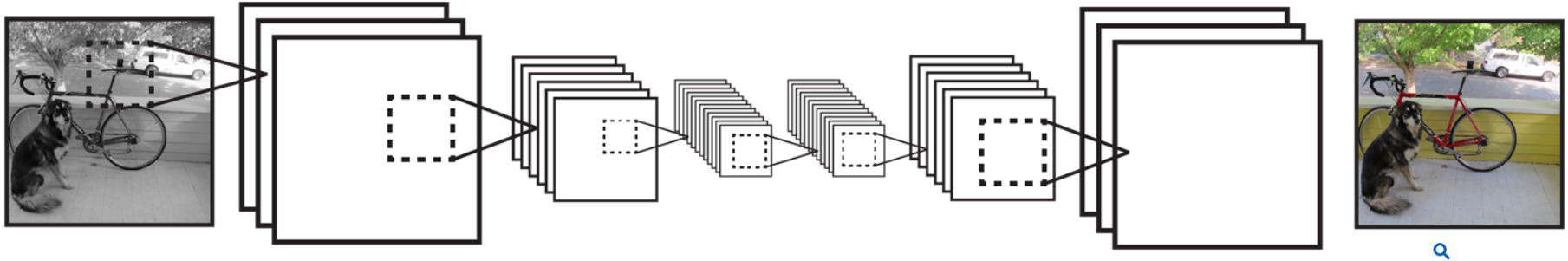
- Training is easy, grayscale a bunch of images, try to predict the original one!
- But there's a problem...

Image colorization

- Training is easy, grayscale a bunch of images, try to predict the original one!
- But there's a problem…
- Big difference between green and red according to L_2 loss, perceptually we don't really care
- Don't want prediction to be "right", we just want it to look good!

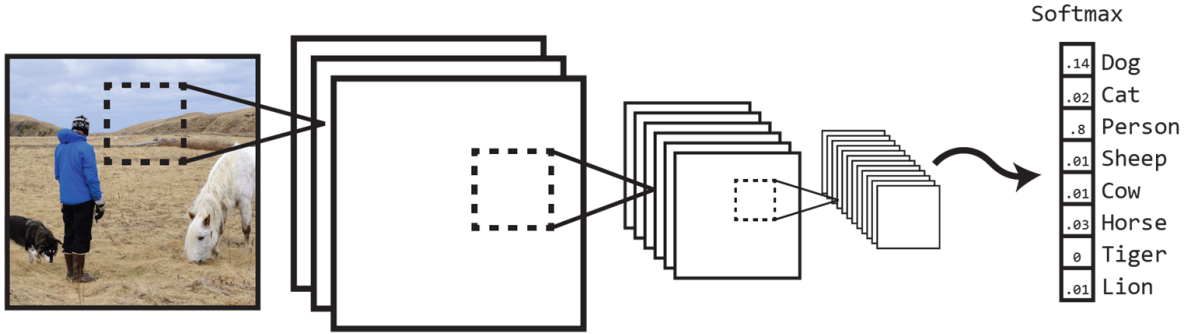
We want a perceptual loss function

- What we really want is… does the image look good?
- And if not, how do we change it to look better!
- Well, we just spent a long time learning how to train neural networks to do stuff...



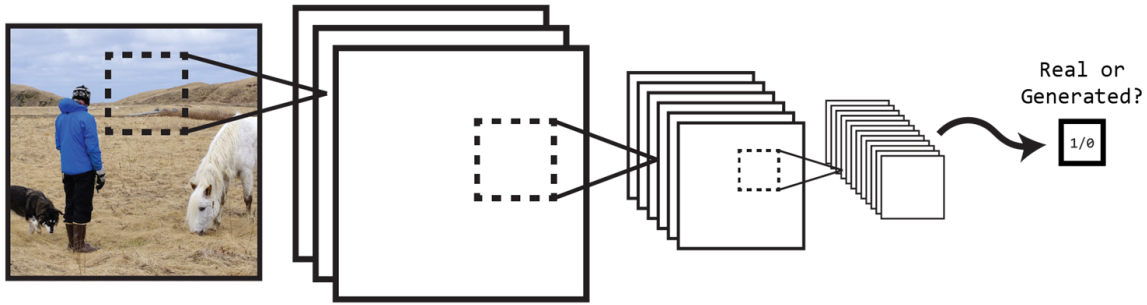
We want a perceptual loss function

- What we really want is… does the image look good?
- And if not, how do we change it to look better!
- Well, we just spent a long time learning how to train neural networks to do stuff...



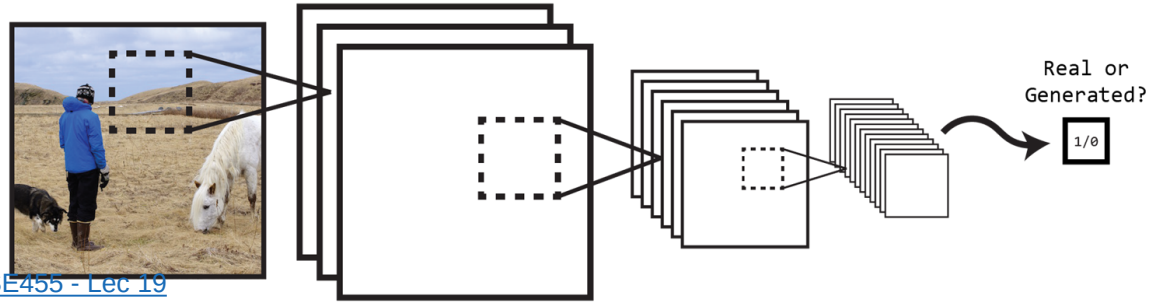
We want a perceptual loss function

- What we really want is… does the image look good?
- And if not, how do we change it to look better!
- Well, we just spent a long time learning how to train neural networks to do stuff...

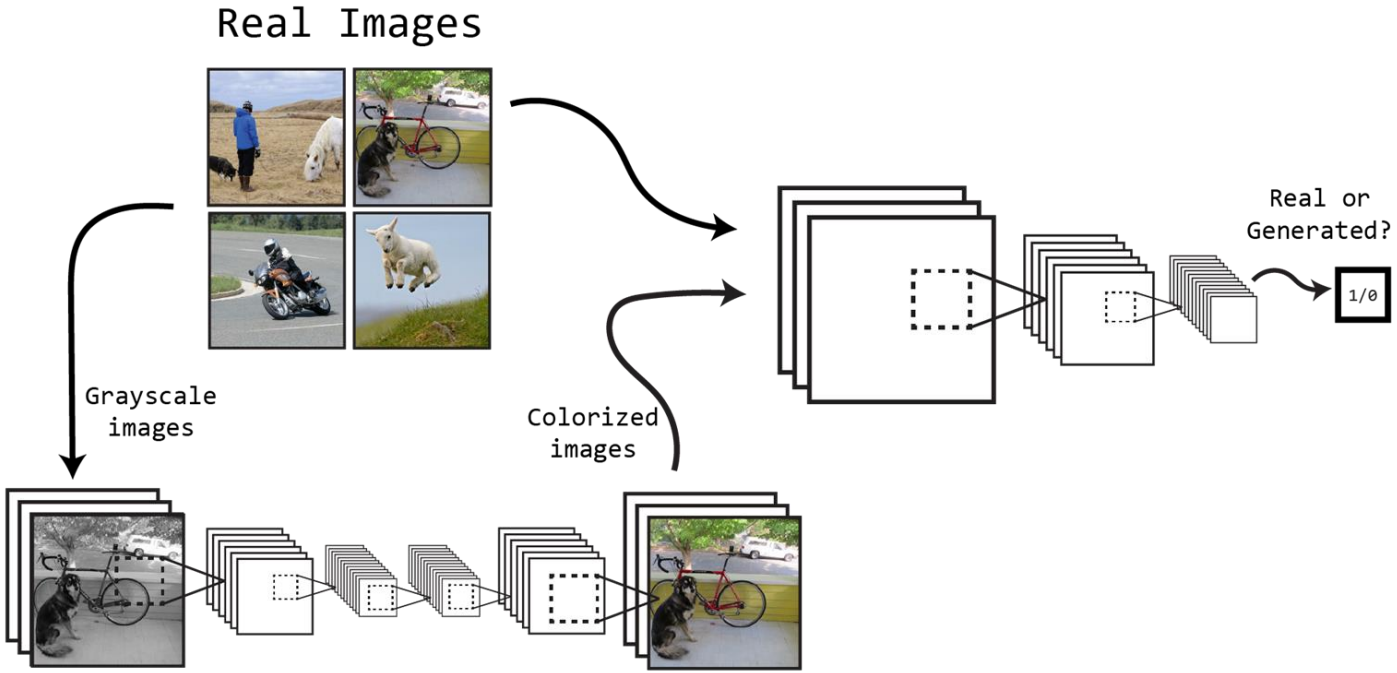


Discriminator network

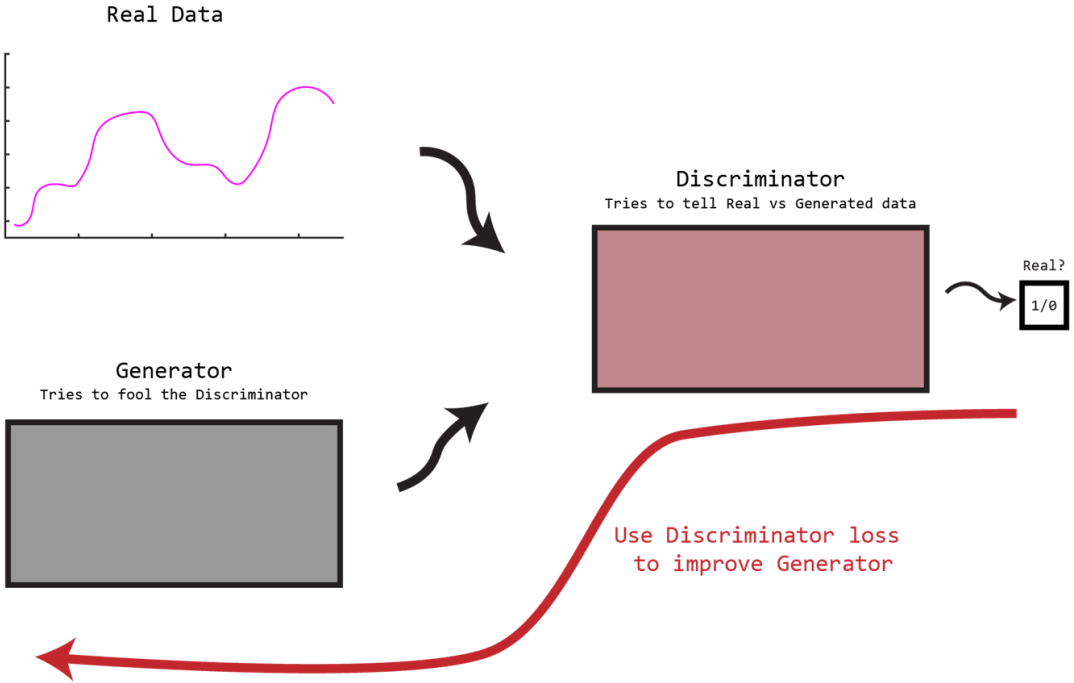
- Trained to tell apart real and generated images
- How do we make an image more “real”?
- Run image through
- Calculate loss with real = 1
- Backpropagate loss through the network to the image itself
- We get “error” for the image that would make it more real!



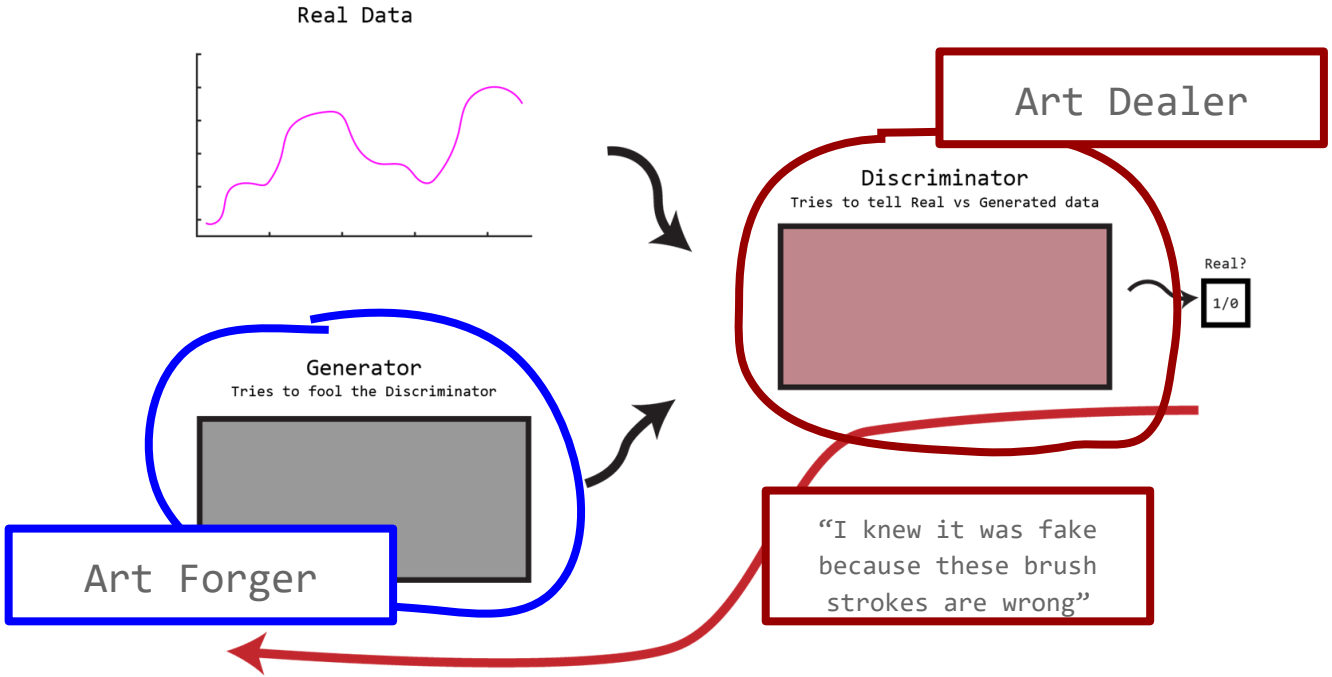
Generative adversarial network



Generative adversarial network



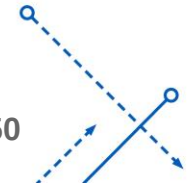
Generative adversarial network



https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

Why are GANs so cool?

- Because we are learning our loss function!



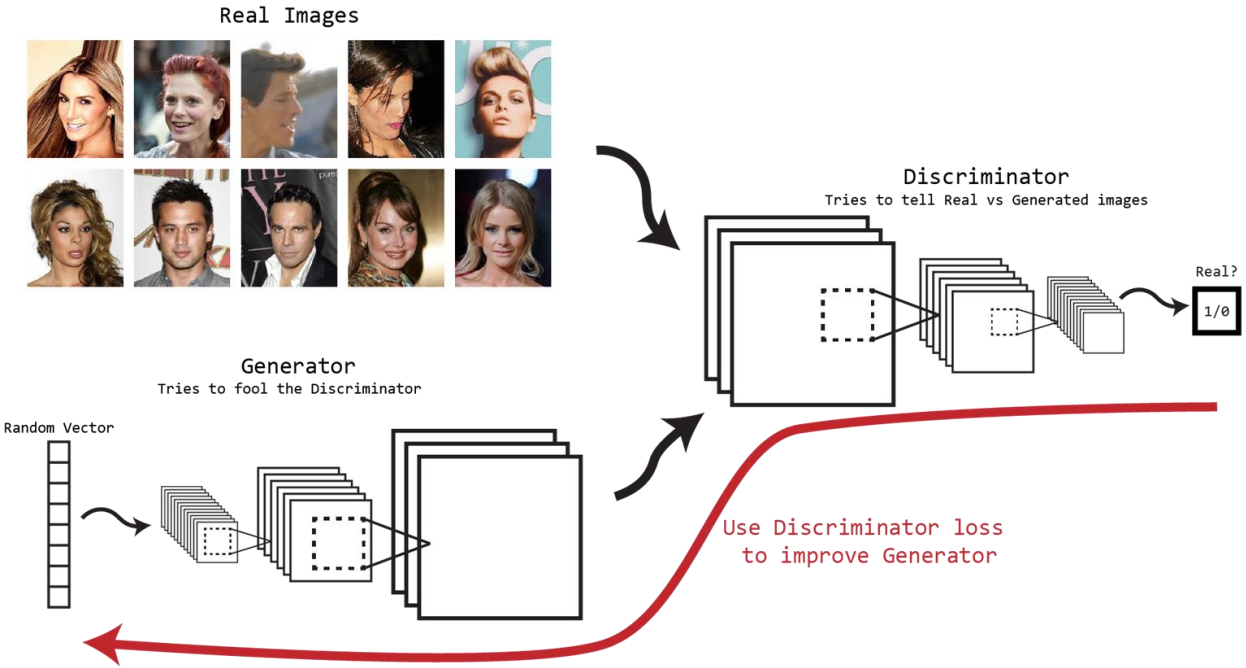
https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

But how do we simultaneously optimize D and G?

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

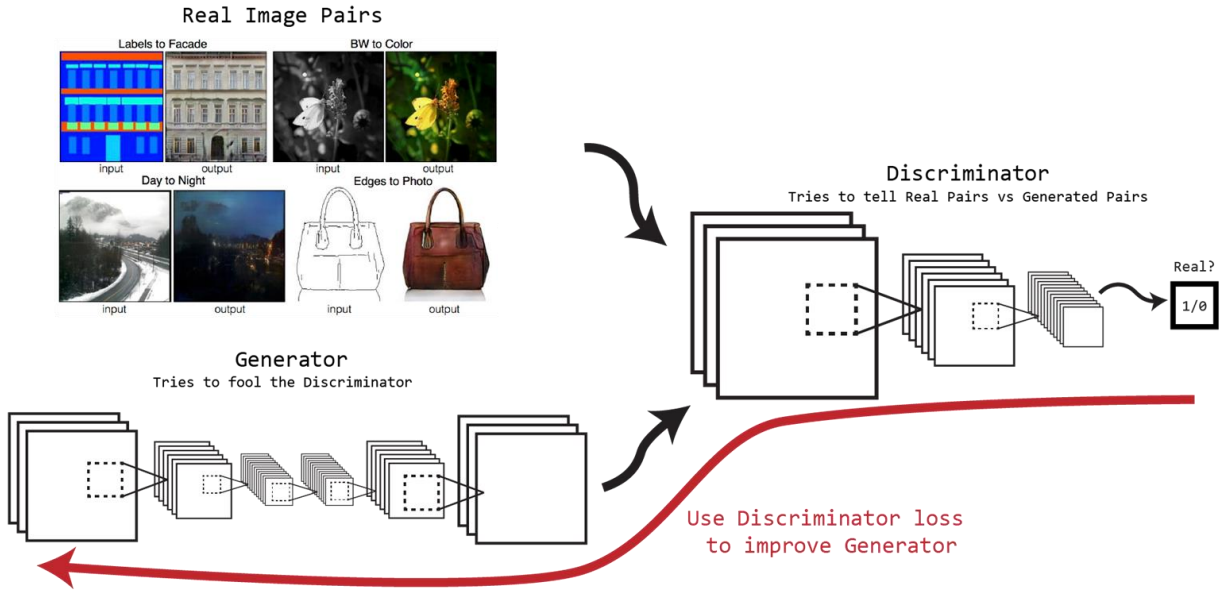
- $\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]$- Expected value over real data samples
  - Measures discriminator's ability to correctly identify real samples
  - $\mathrm{D}(\mathrm{x})$ represents probability that x is real
- $\mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))]$ - Expected value over generated samples
  - Measures discriminator's ability to reject fake samples
  - G(z) generates fake samples from random noise z
  - $\mathrm{D}(\mathrm{G}(\mathrm{z}))$is probability that generated sample is real

DCGAN (deep convolutional GANs)



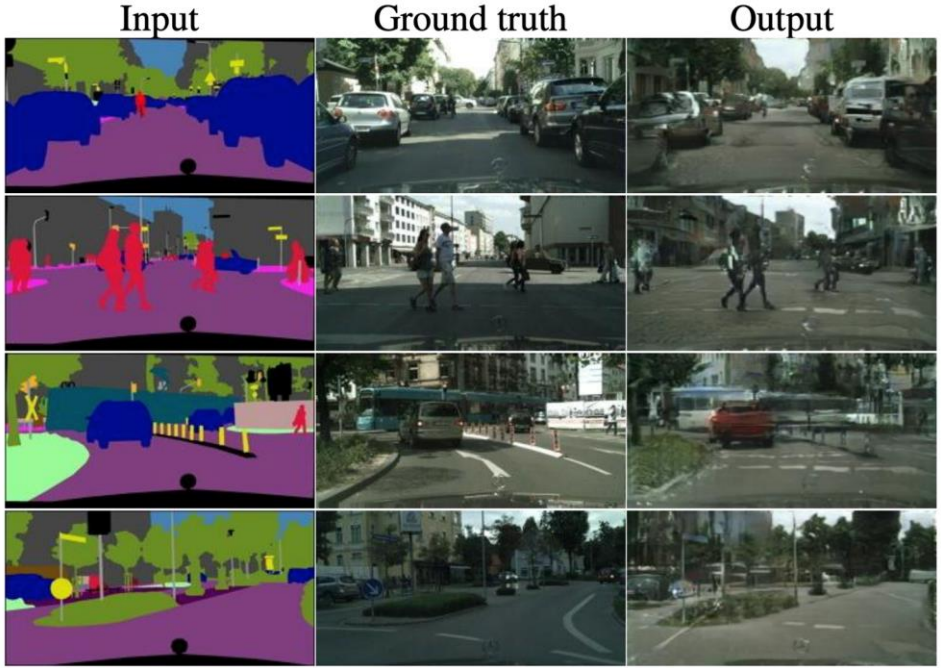
https://arxiv.org/pdf/1511.06434.pdf, https://arxiv.org/pdf/1710.10196.pdf

pix2pix: paired image modification



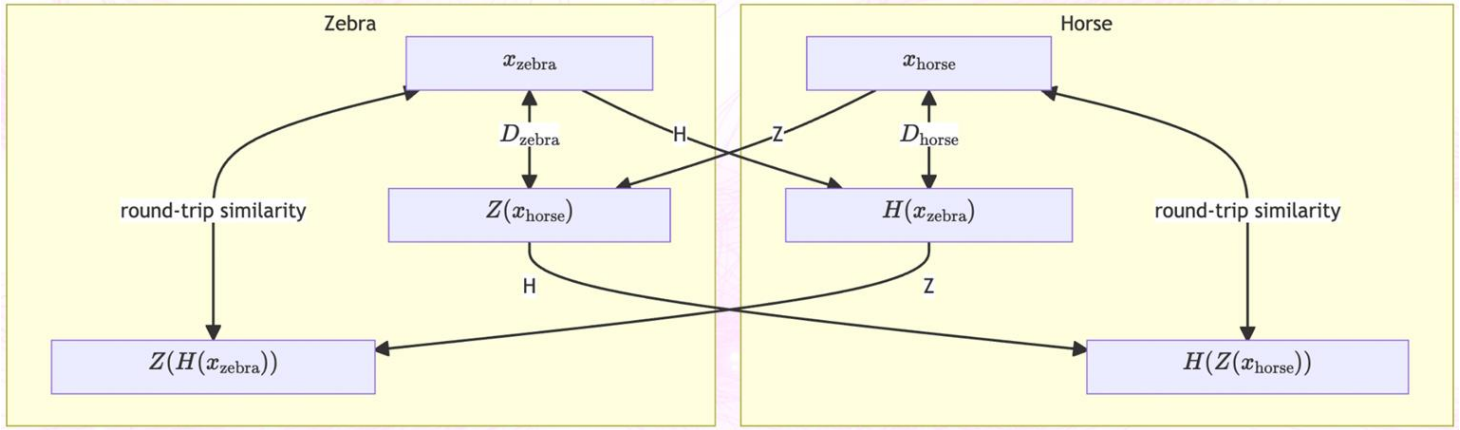
https://affinelayer.com/pixsrv/, https://github.com/phillipi/pix2pix

pix2pix: paired image modification



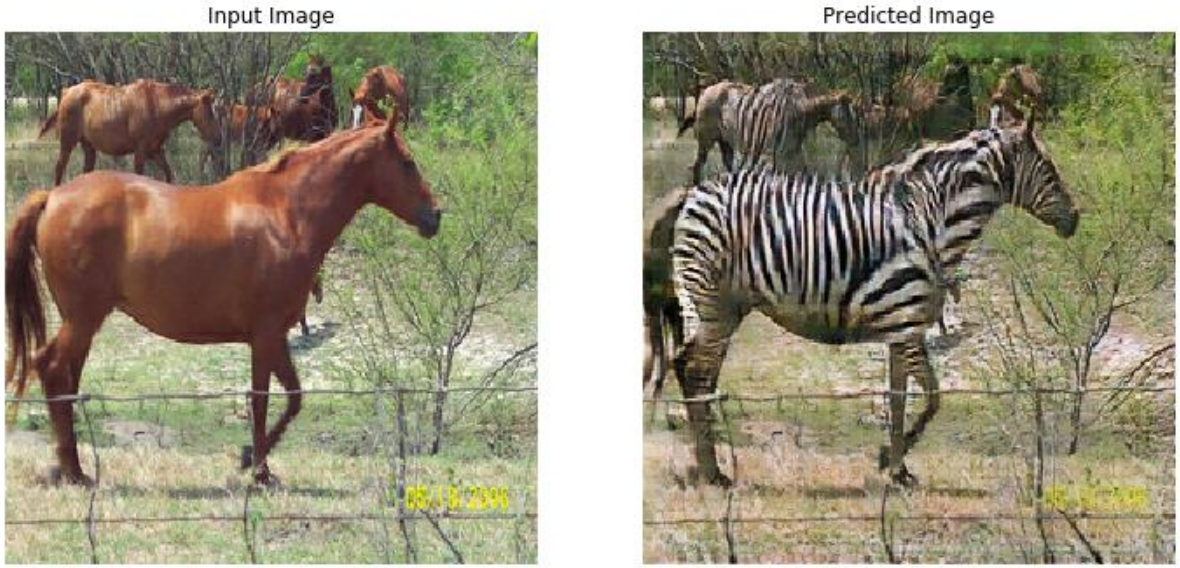
https://affinelayer.com/pixsrv/, https://github.com/phillipi/pix2pix

CycleGan

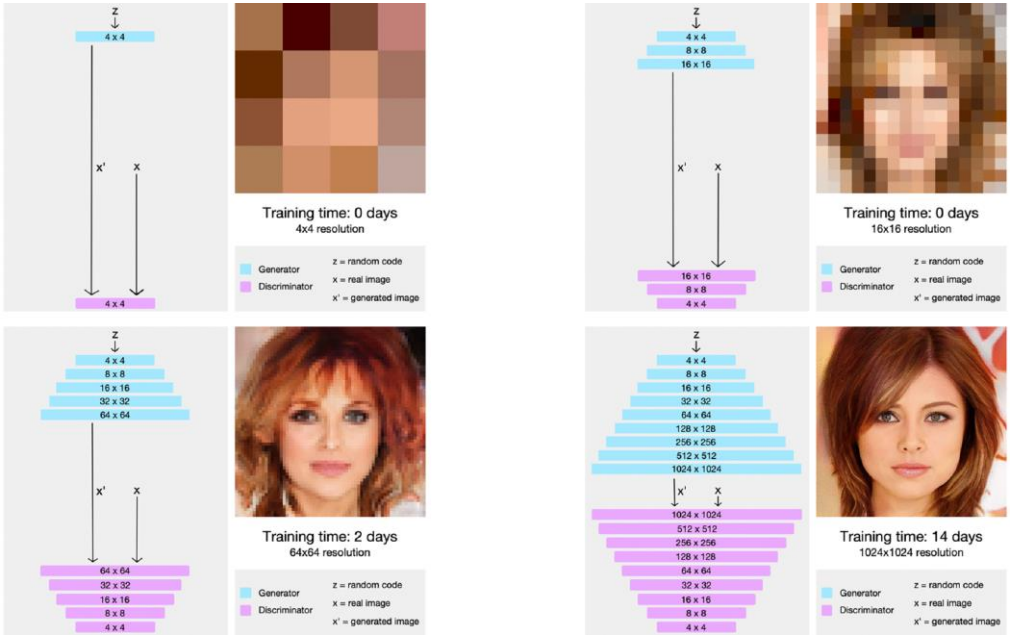


Z: generator for zebras
H: generator for horses
D_z: discriminator for zebras
D_h: discriminator for horses

CycleGan



Progressive growing of GANs



https://arxiv.org/pdf/1710.10196.pdf

Progressive growing of GANs

https://arxiv.org/pdf/1710.10196.pdf

GANs are really hard to train

- Unstable
- Classification easier than generation
- Mode collapse
- …

Diffusion Models: The Future

Forward: Add noise to image (easy)

Backward: Train network to denoise

Diffusion Models: The Future

1. class + noise -> image
2. image + hi res noise -> hi res image