Lecture 9 - Applications of CV & NLP

Machine Learning Decal

Hosted by Machine Learning at Berkeley

Agenda

Recap

Image Captioning

DEMO!

Saliency Maps

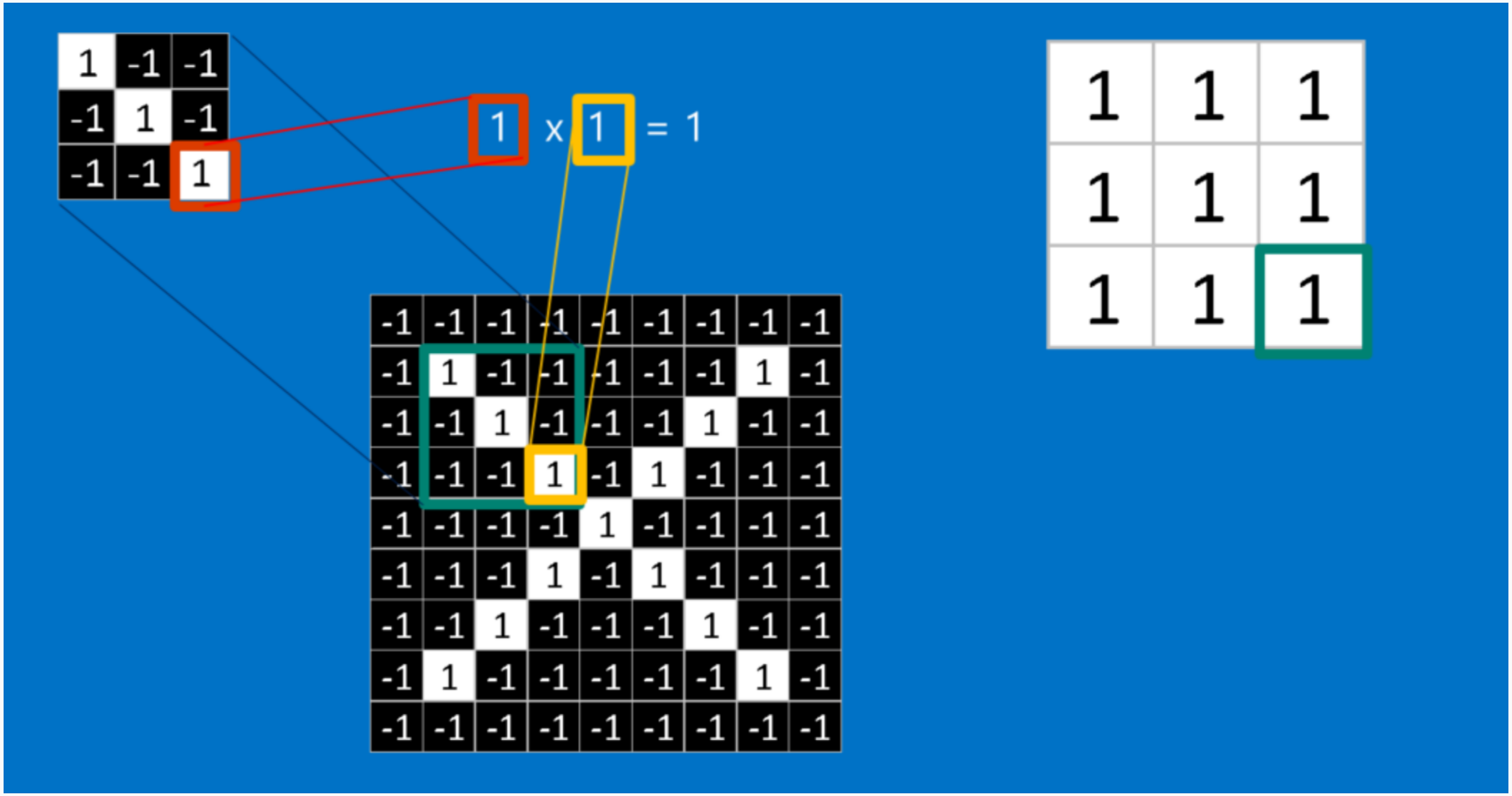
Attention

Image Detection

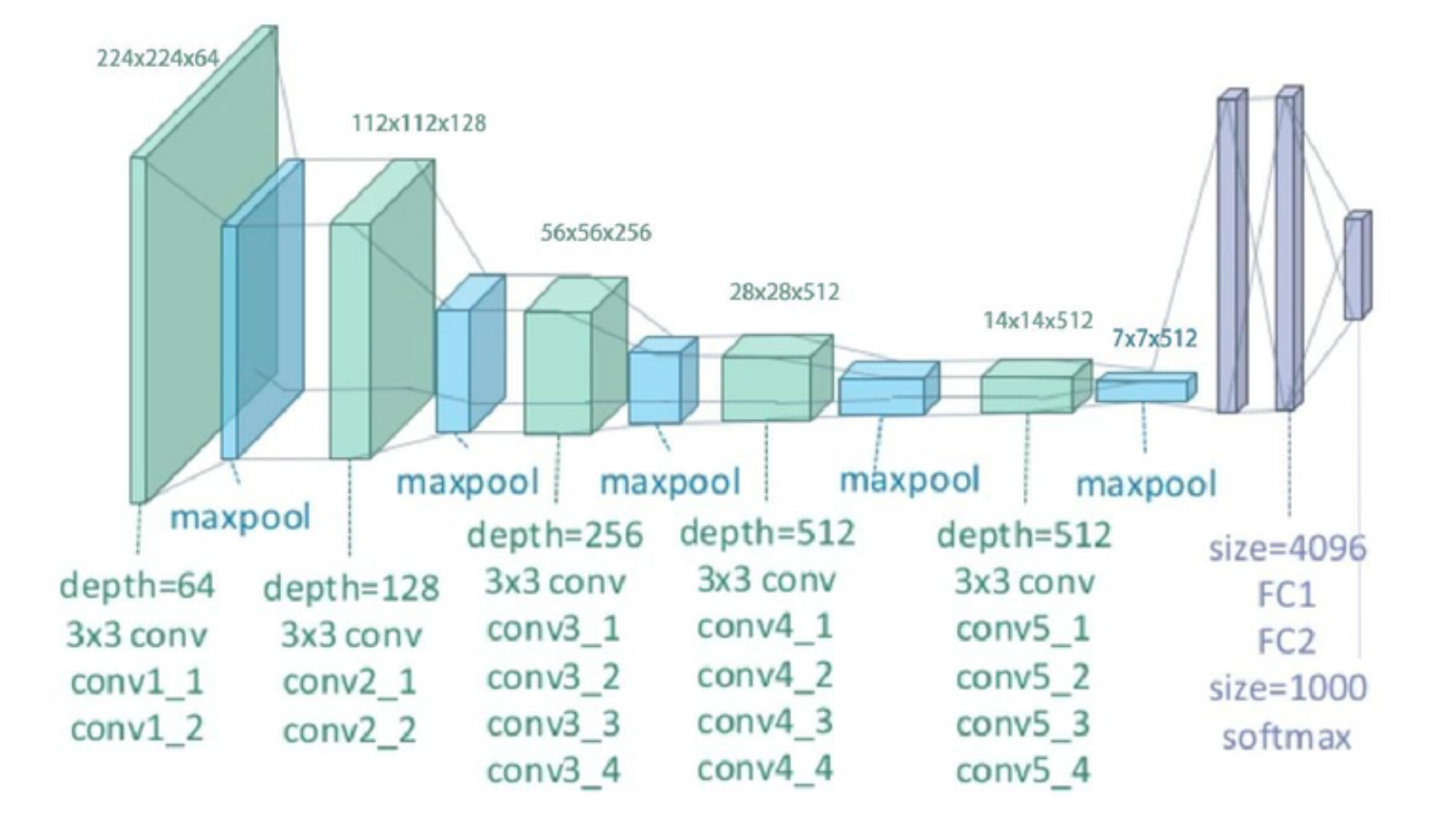
Questions?

Recap

Recap of CV - convolutions



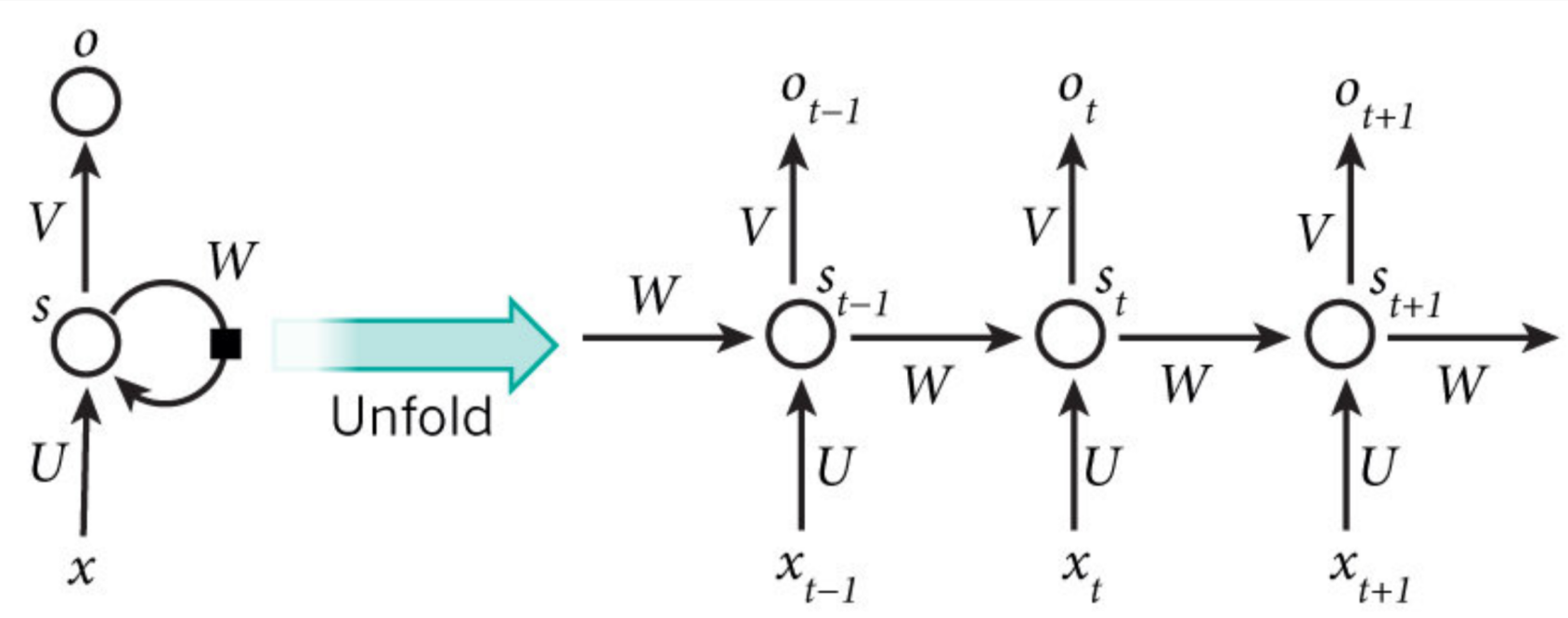
Recap of CV - CNNs



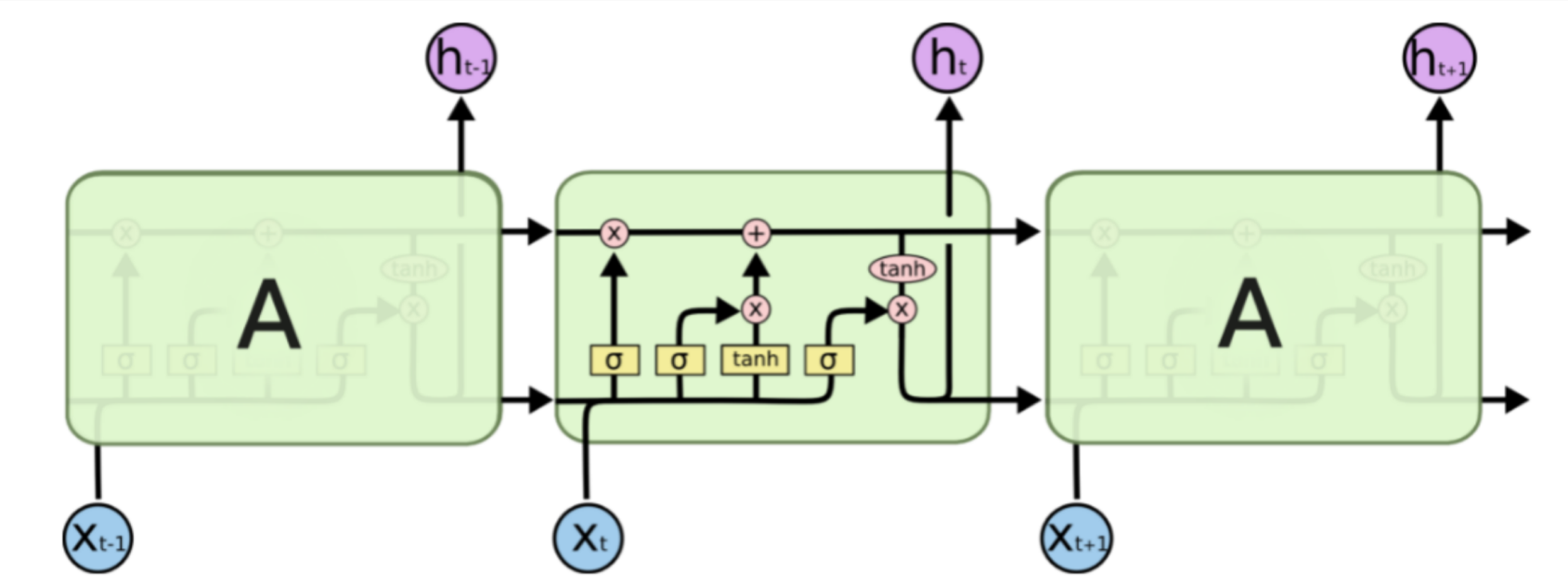
Recap of CV

- Transfer Learning
- Style Transfer
- GANs (Generative Adversarial Networks)

Recap of NLP - RNNs



Recap of NLP - LSTMs



Recap of NLP

- Word embeddings (Word2Vec, GloVe)
- Translation with Seq2Seq networks
- Attention

Goal for today

You've already learned so many techniques. Today is about piecing them together. How can we come up with a solution given what we've learned so far?

Goal for today

You've already learned so many techniques. Today is about piecing them together. How can we come up with a solution given what we've learned so far?

The problems we'll discuss today:

- Image Captioning
- Object Detection

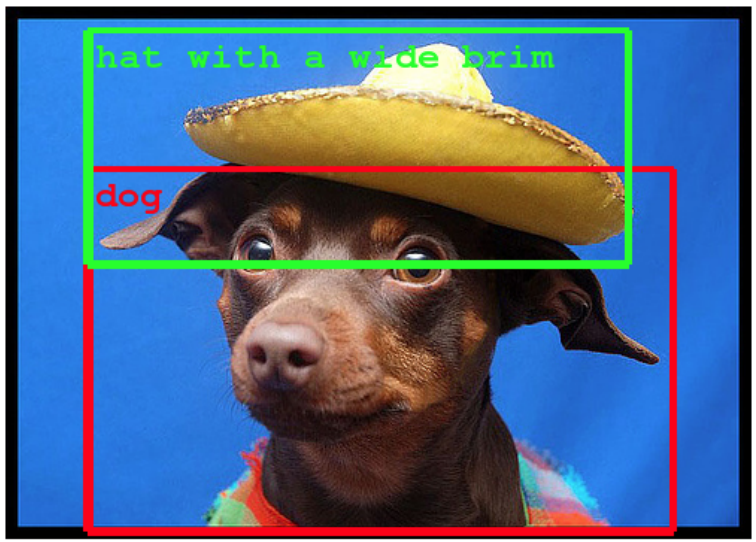


Image Captioning

Image Captioning Problem Statement

Test time: Given new image x, output caption y that best describes x

Image Captioning Problem Statement

Test time: Given new image x, output caption y that best describes x

Training time: $N \times \{(x_i, y_i)\}$, where x_i is an image and y_i is a corresponding caption for the image (MSCOCO dataset)

Image Captioning Problem Statement

Test time: Given new image x, output caption y that best describes x

Training time: $N \times \{(x_i, y_i)\}$, where x_i is an image and y_i is a corresponding caption for the image (MSCOCO dataset)

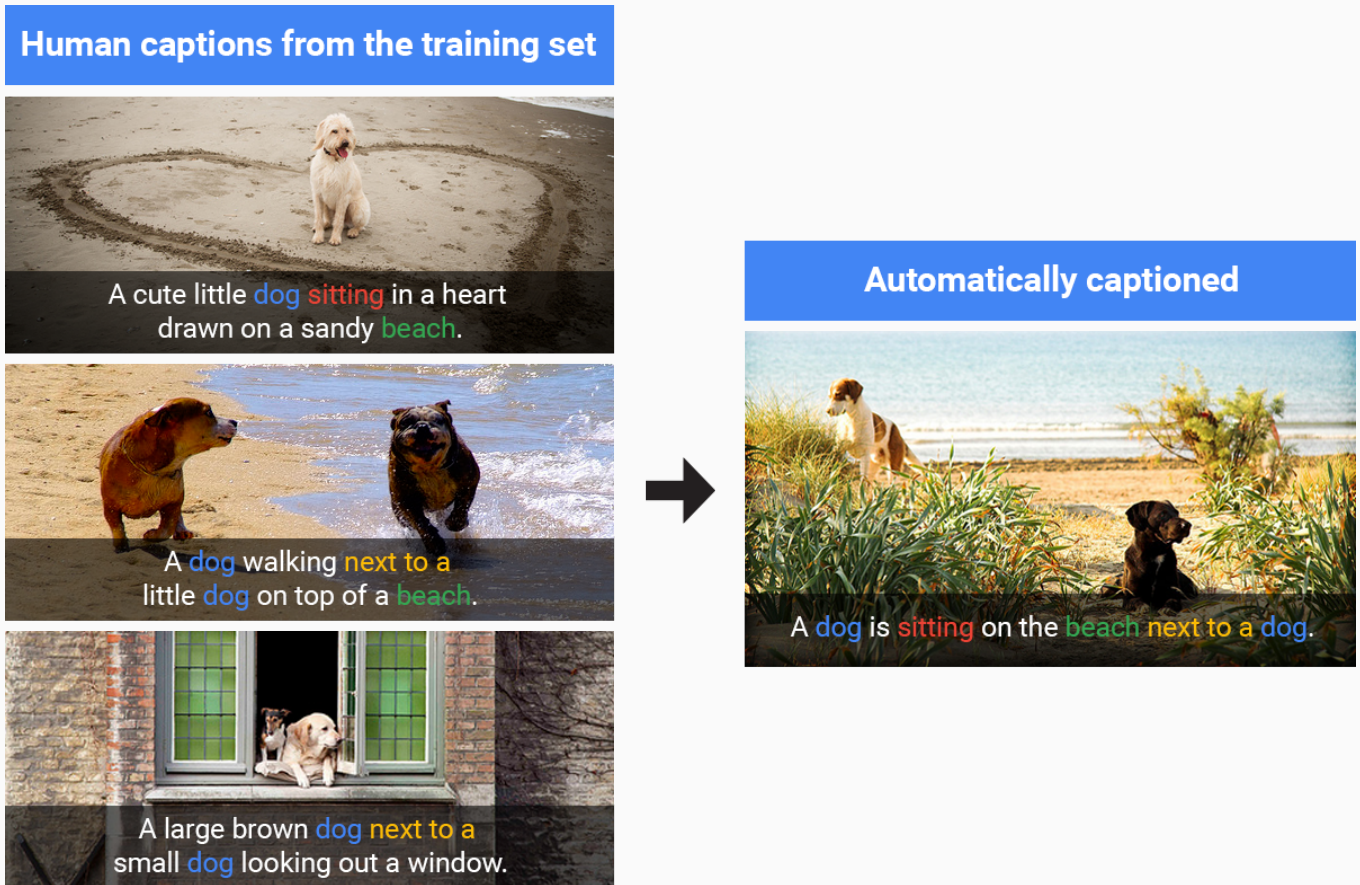


Image Captioning Solution Outline

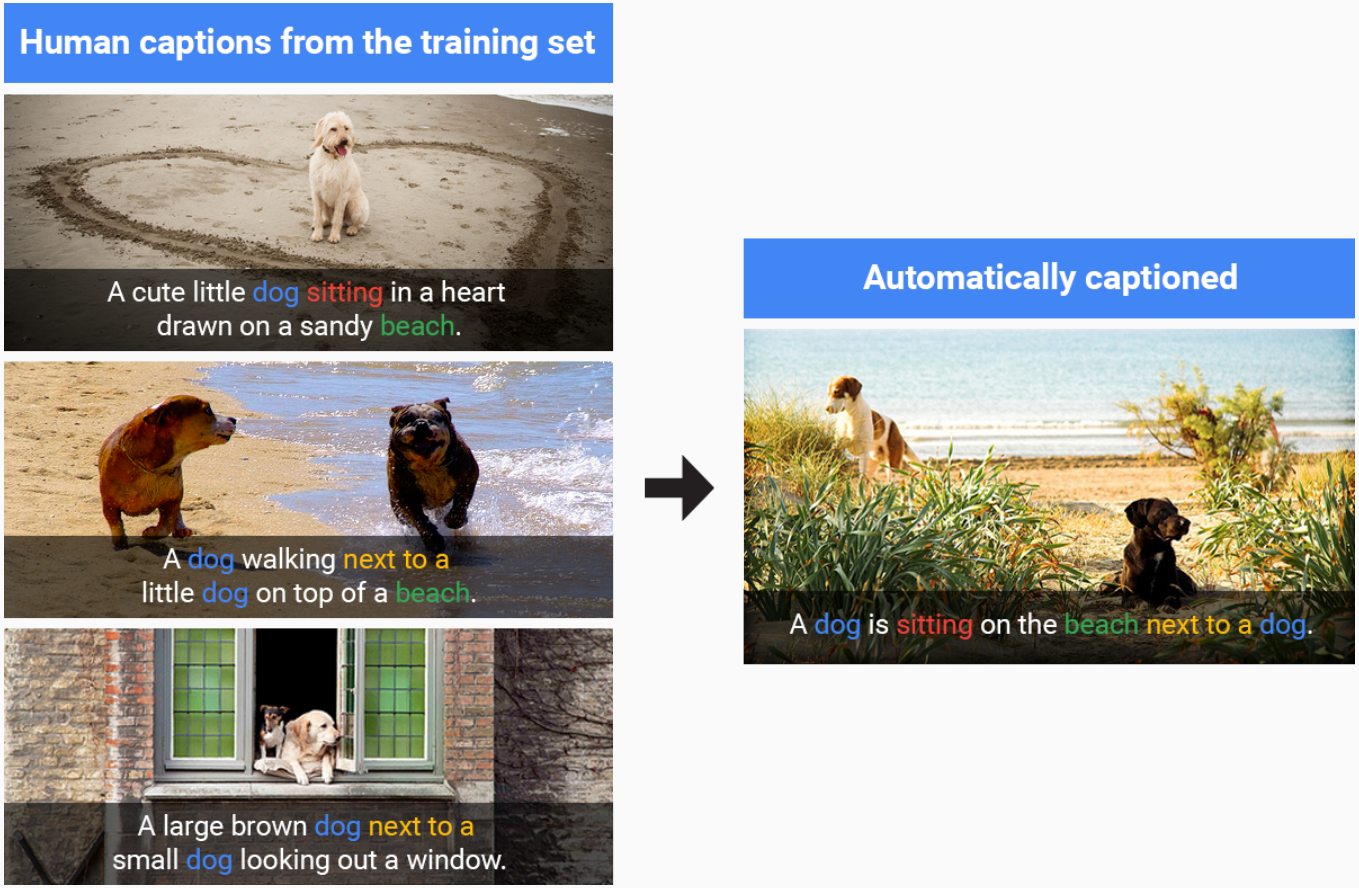
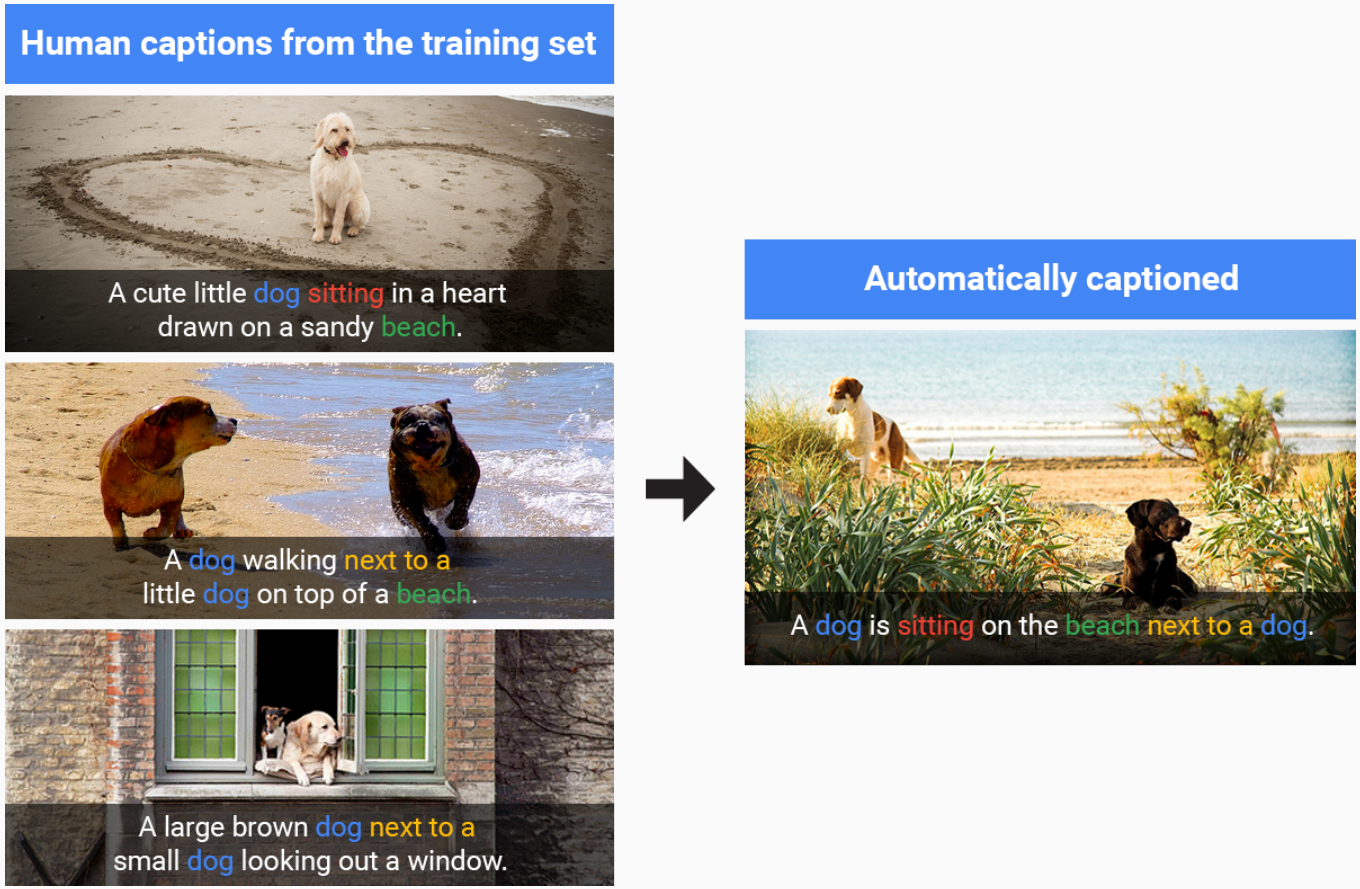


Image Captioning Solution Outline



We want to create a model that can do this (duh). For that, we'll need an **architecture** and a **loss function**.

Image Captioning Architecture

What we'll need:

- Some CV stuff - we have images. some sort of convnet

Image Captioning Architecture

What we'll need:

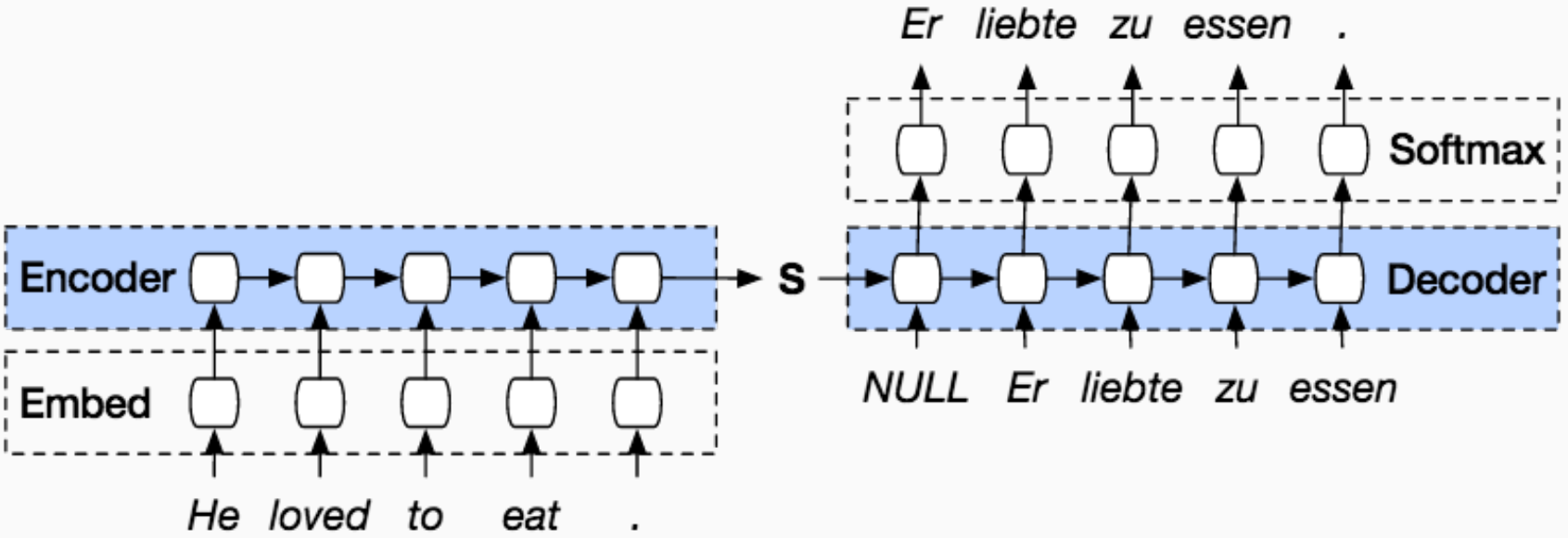
- Some CV stuff - we have images. some sort of convnet
- We also need to generate words. RNNs would be a good idea

Image Captioning Architecture

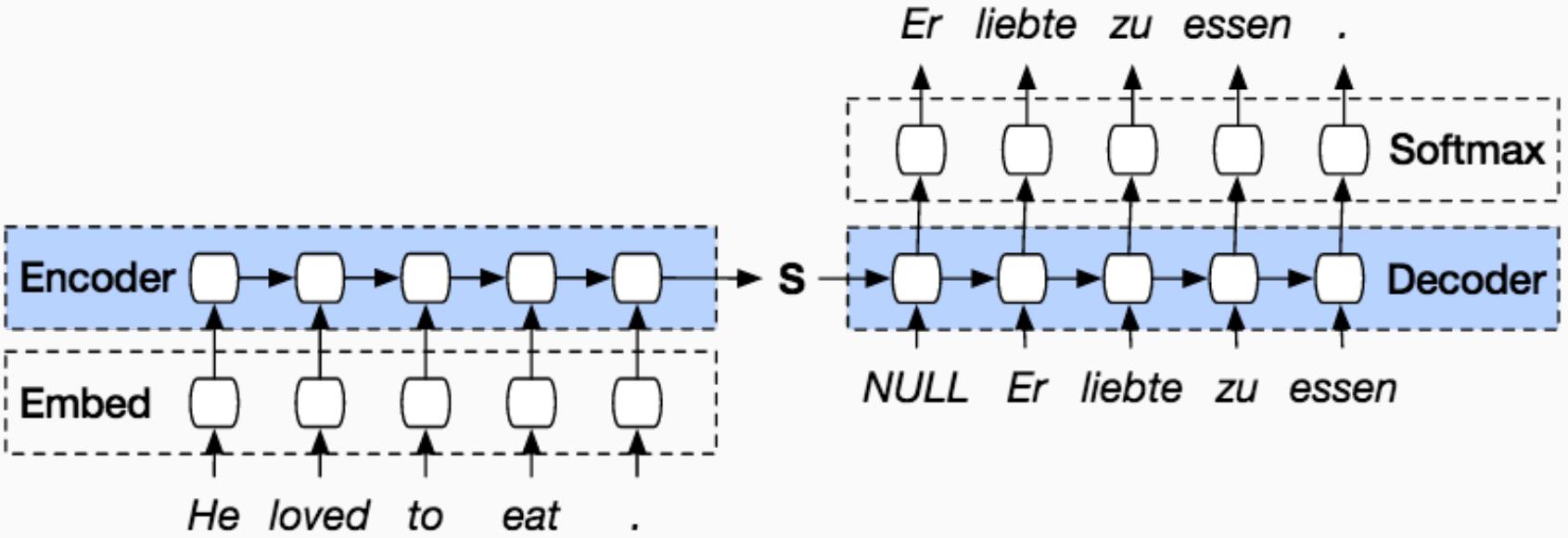
What we'll need:

- Some CV stuff - we have images. some sort of convnet
- We also need to generate words. RNNs would be a good idea
- How will these two interplay? think about the **encoder-decoder** framework we learned previously

Encoder Decoder for Translation (review)

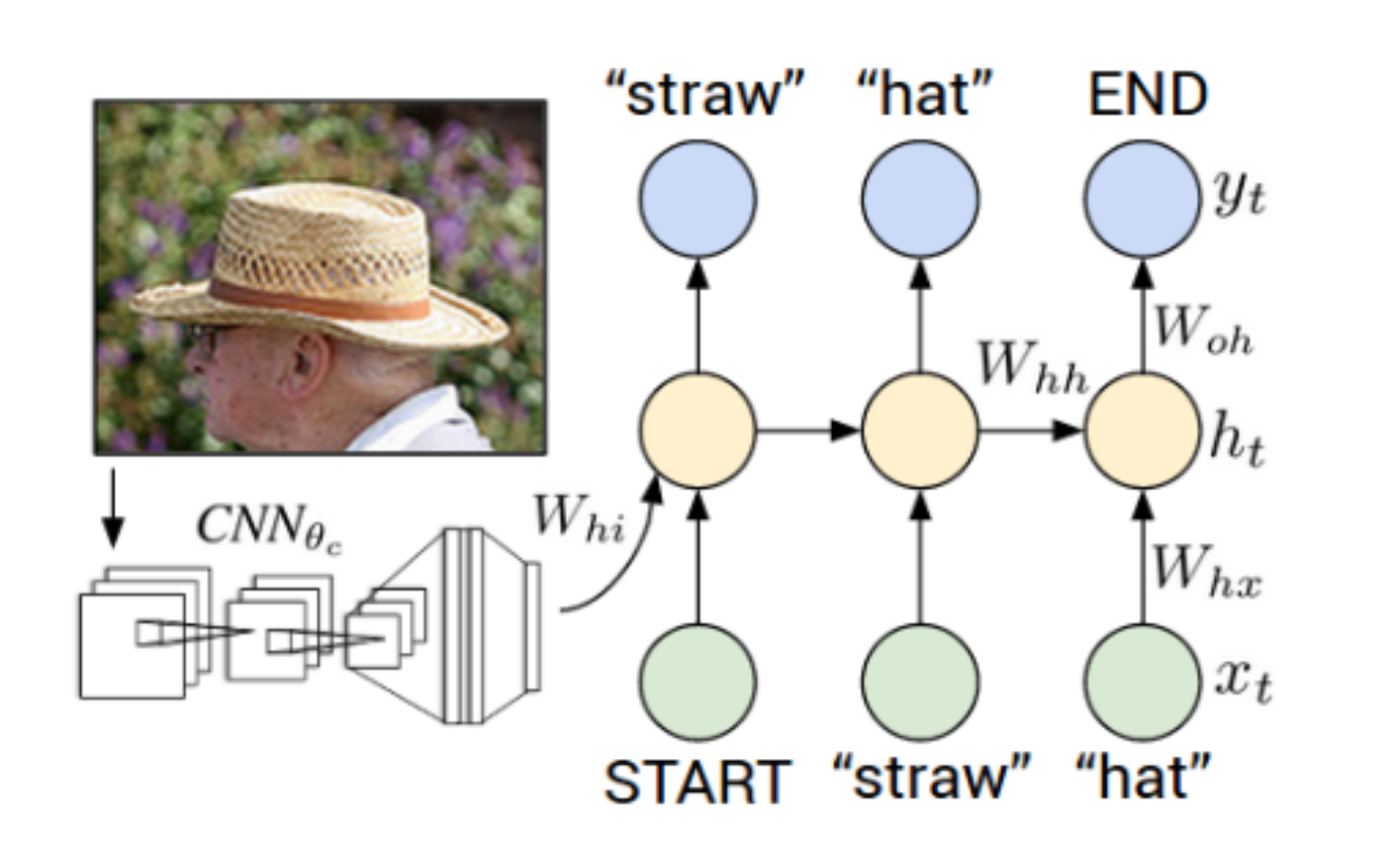


Encoder Decoder for Translation (review)

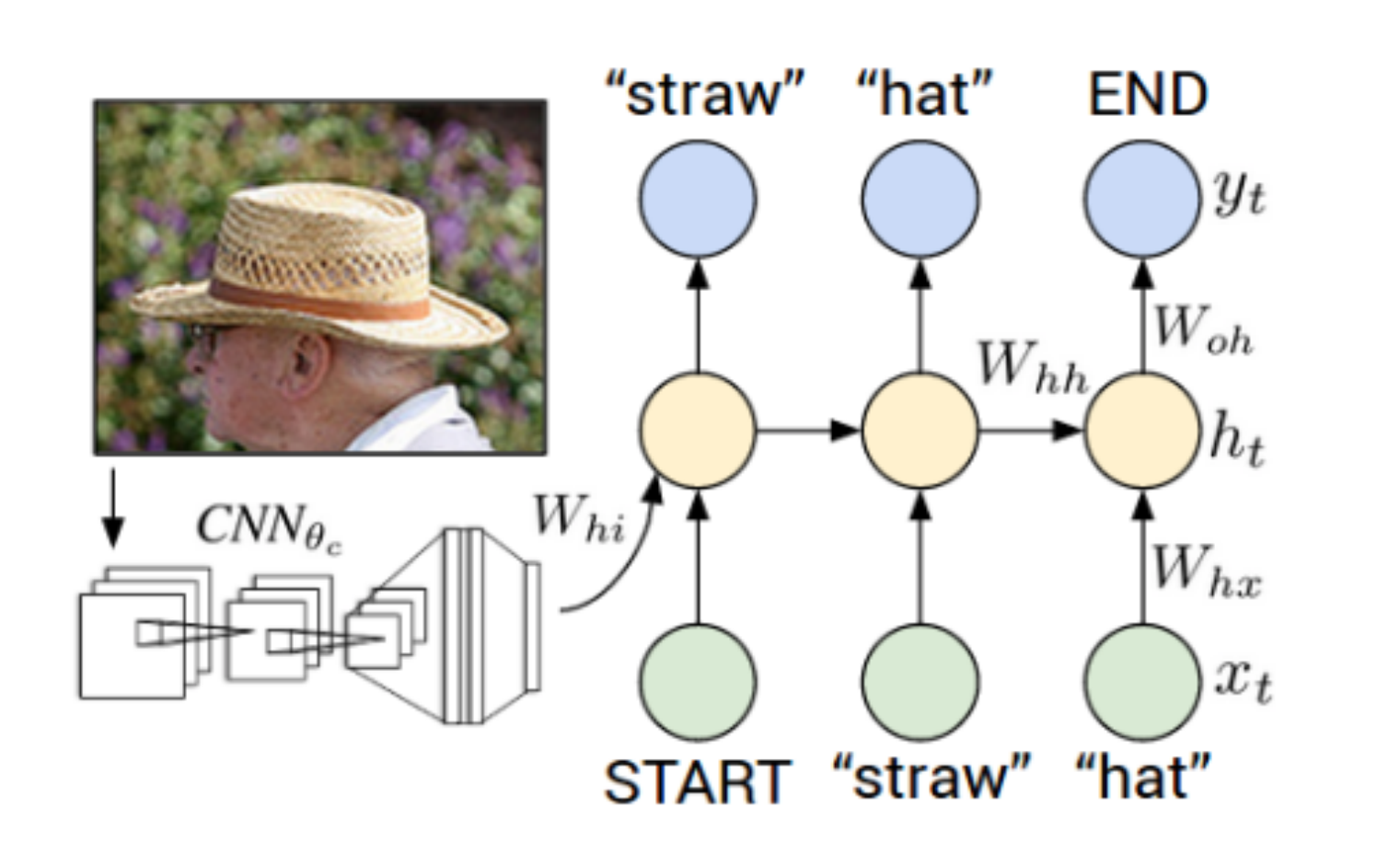


Main idea: we want to move from one domain (english) to another (german). The state **S**, which contains the encoded meaning of the sentence, acts as a bridge between the two.

Architecture Overview

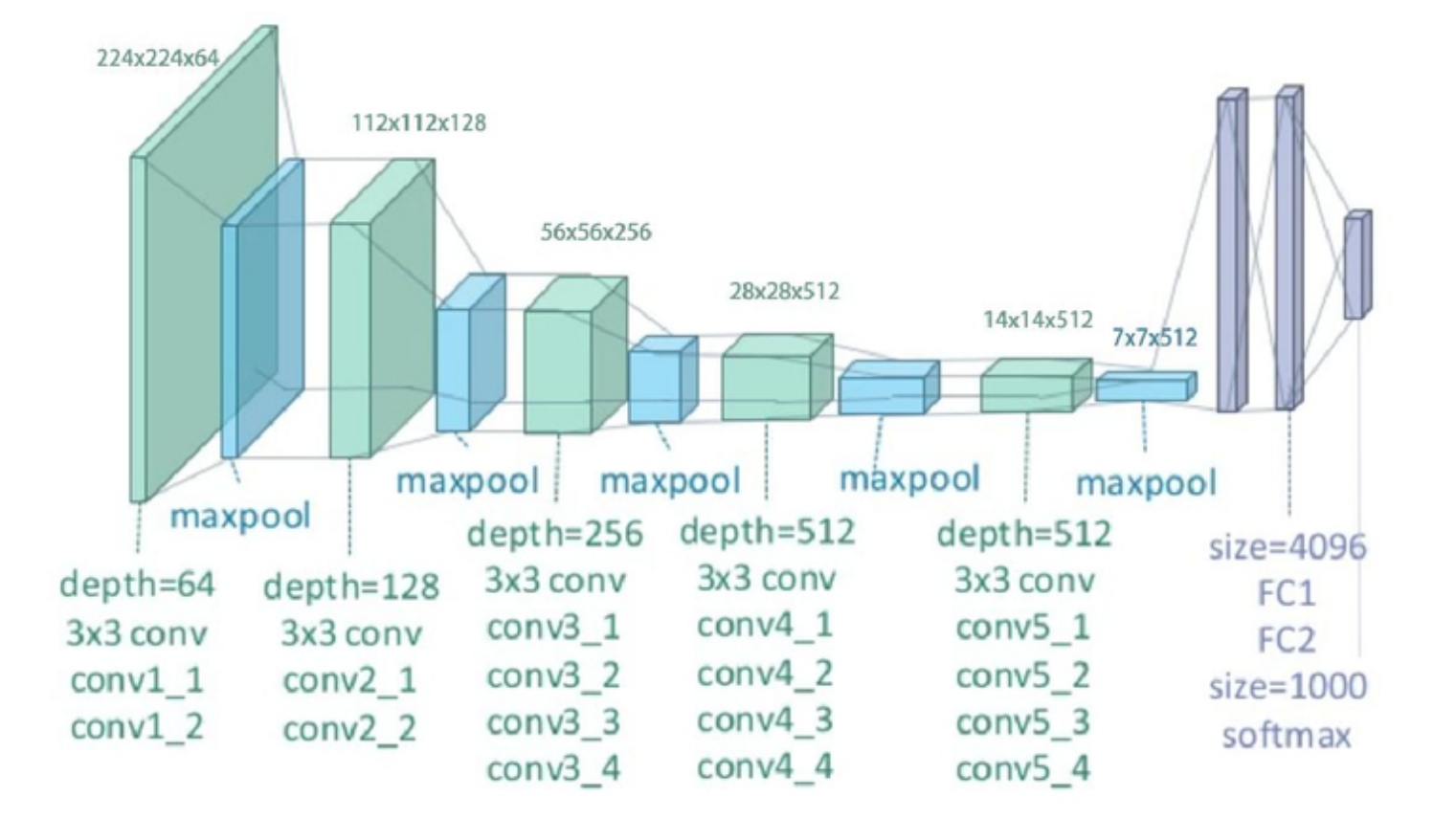


Architecture Overview



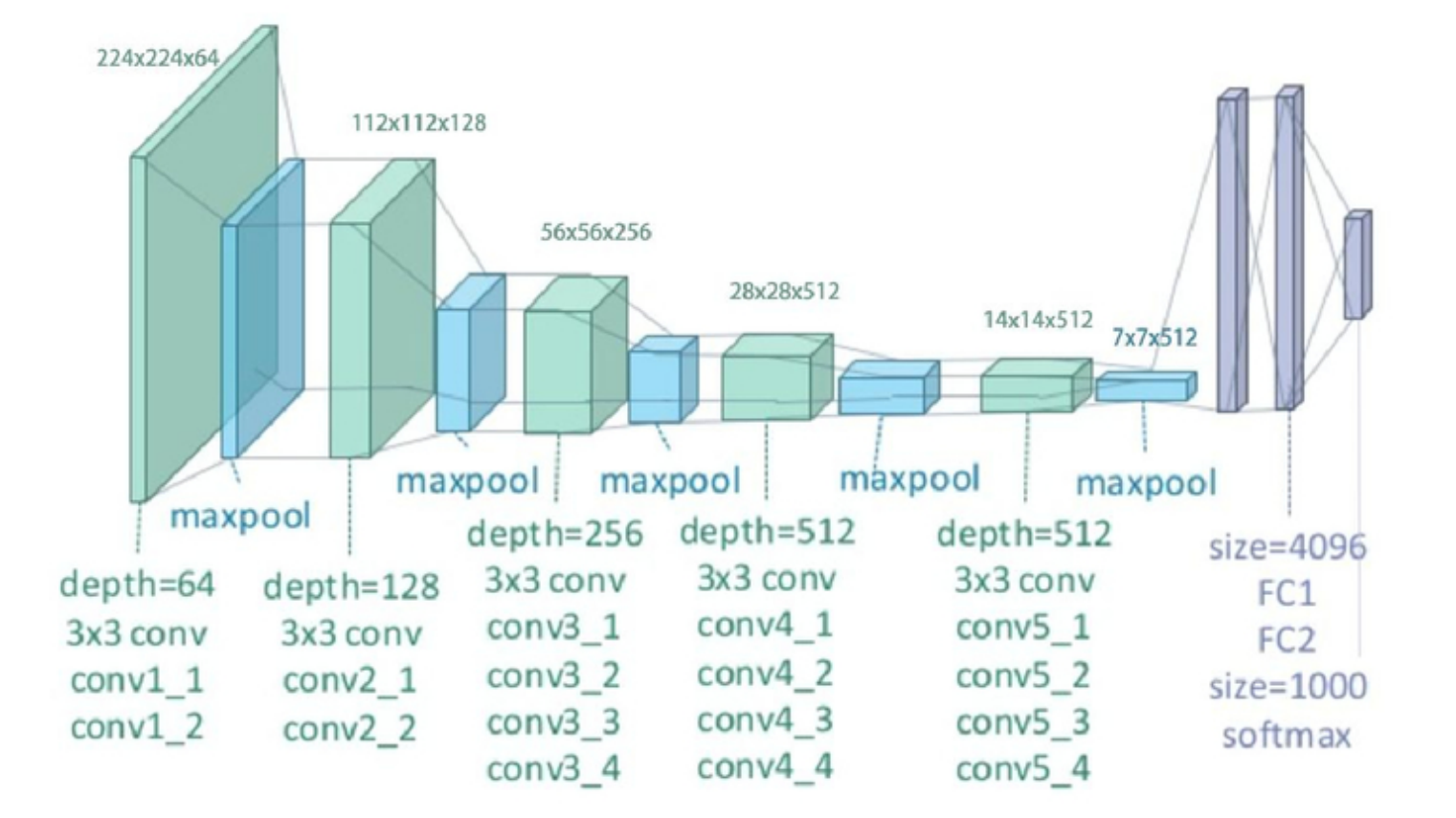
Encode the image with a CNN, **decode** with an RNN using the state passed over from CNN.

What state should the CNN pass over?



What layer contains the best representation of the image?

What state should the CNN pass over?



What layer contains the best representation of the image? **Last FC layer.**

Image Captioning Architecture

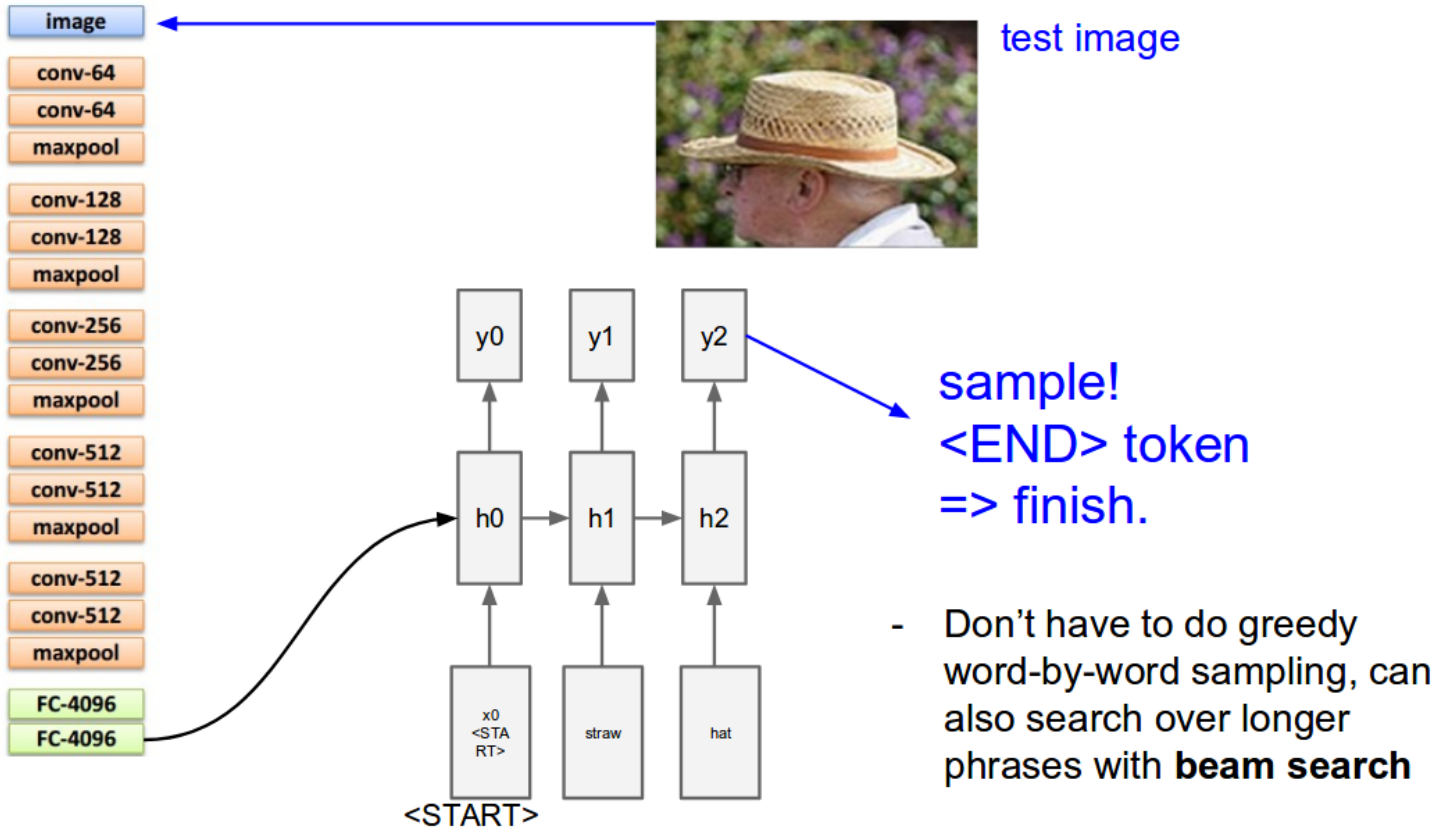
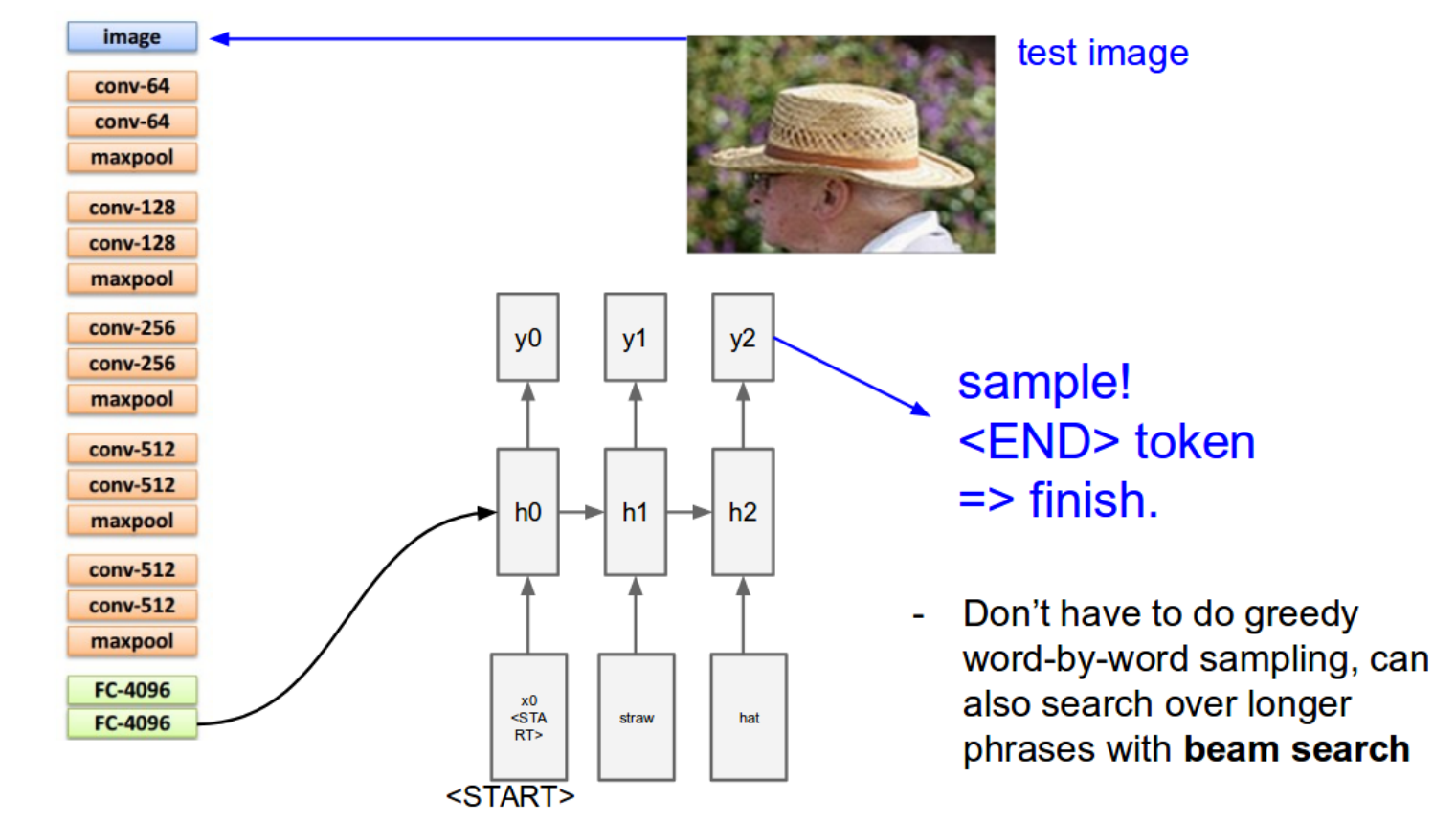
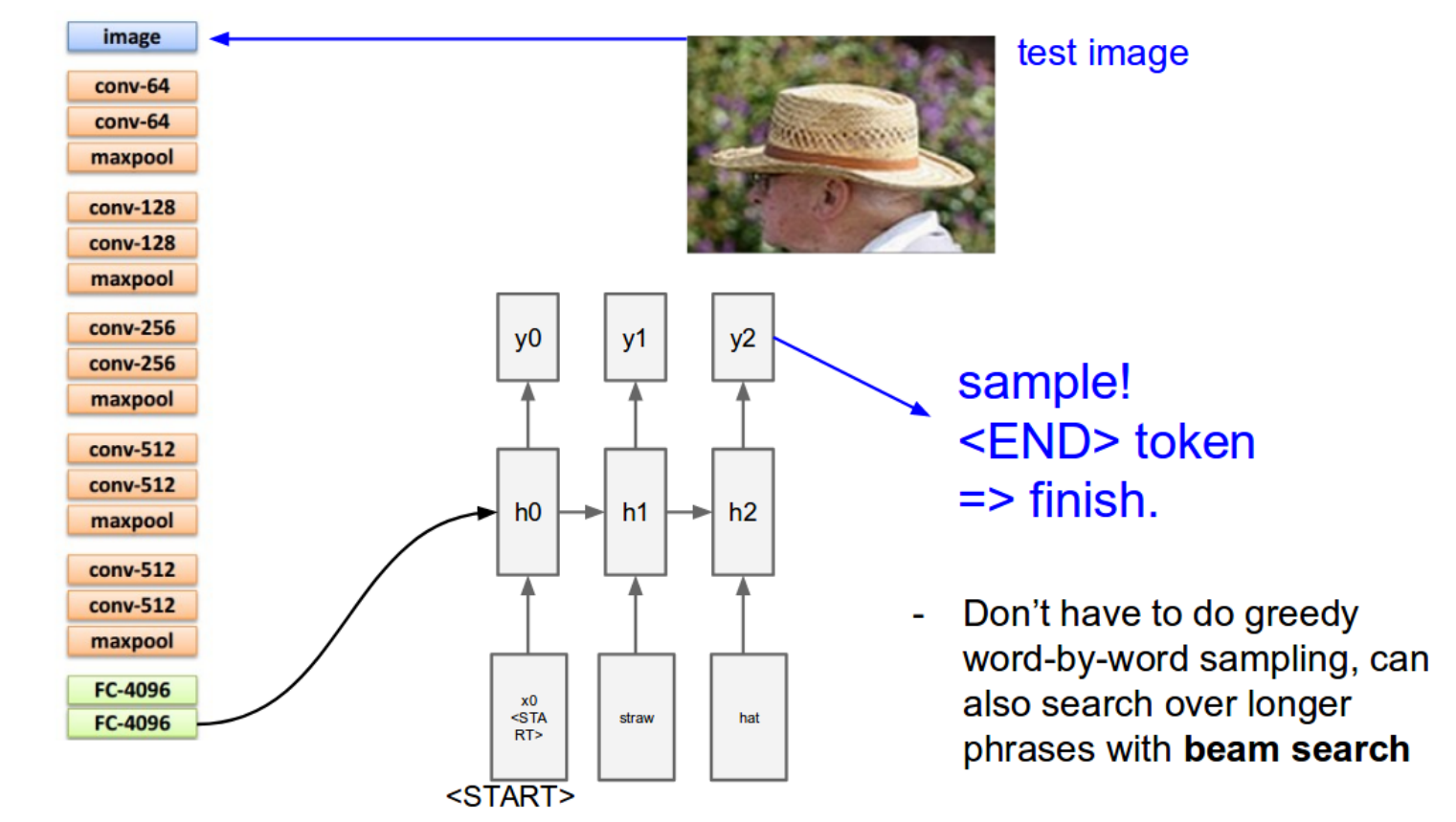


Image Captioning Architecture



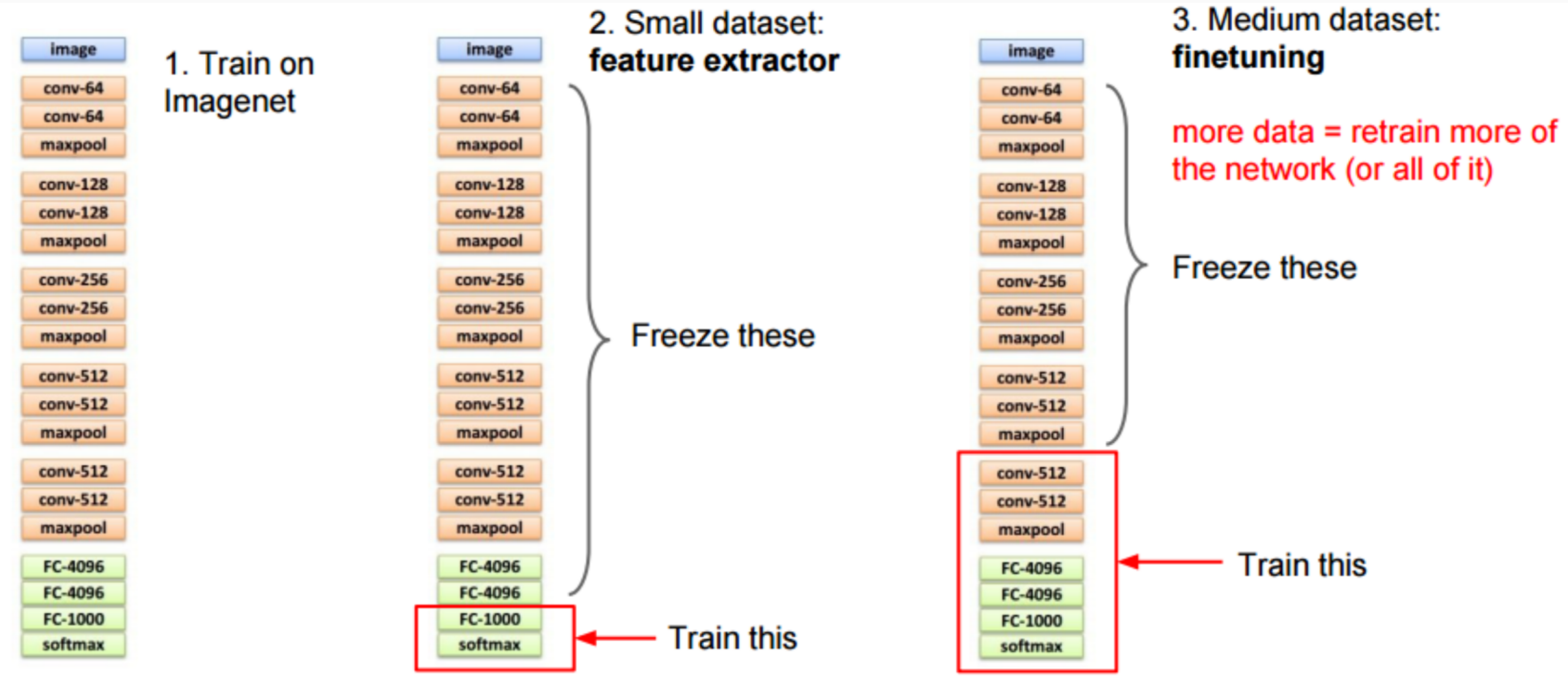
Can anyone see a potential problem with this?

Image Captioning Architecture



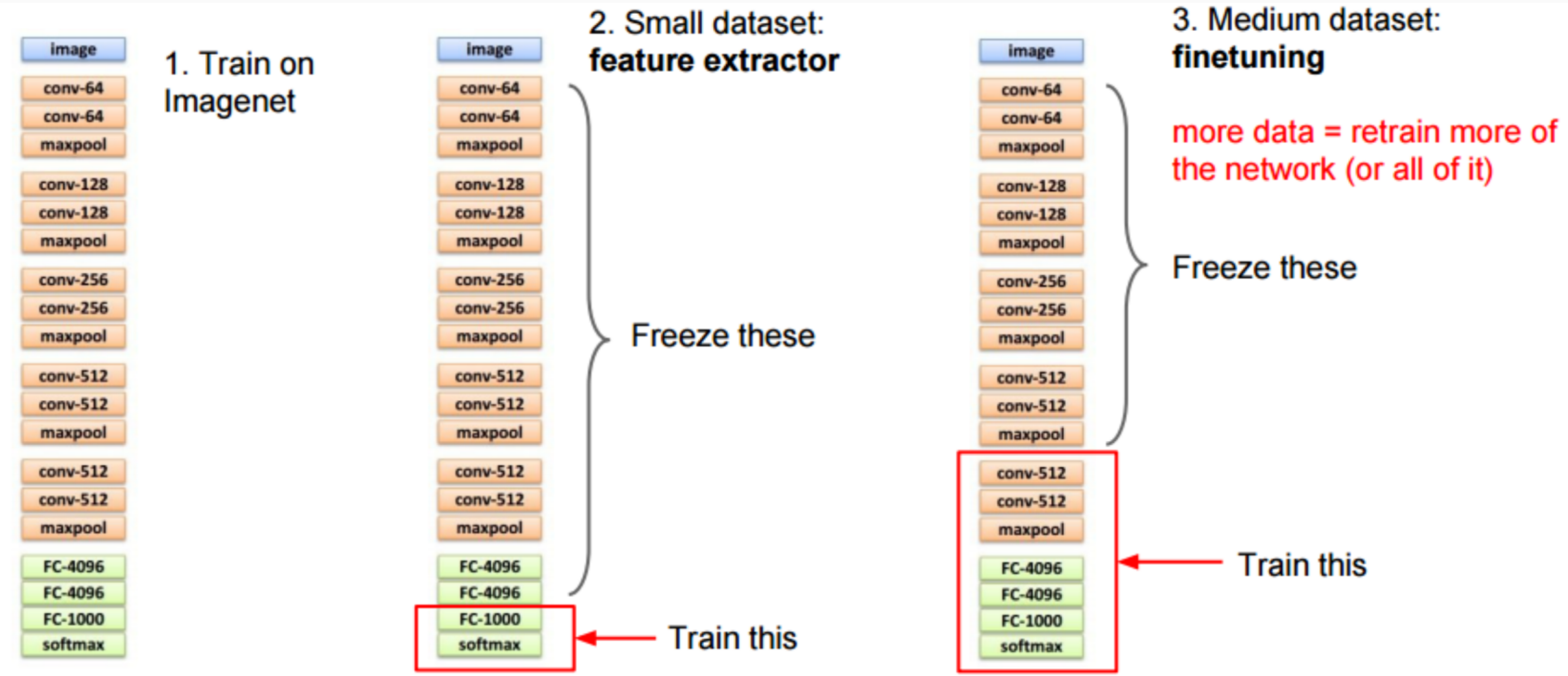
Can anyone see a potential problem with this? There's so many weights to train!

Transfer Learning



Let's revisit transfer learning. In our situation, which case is most suitable?

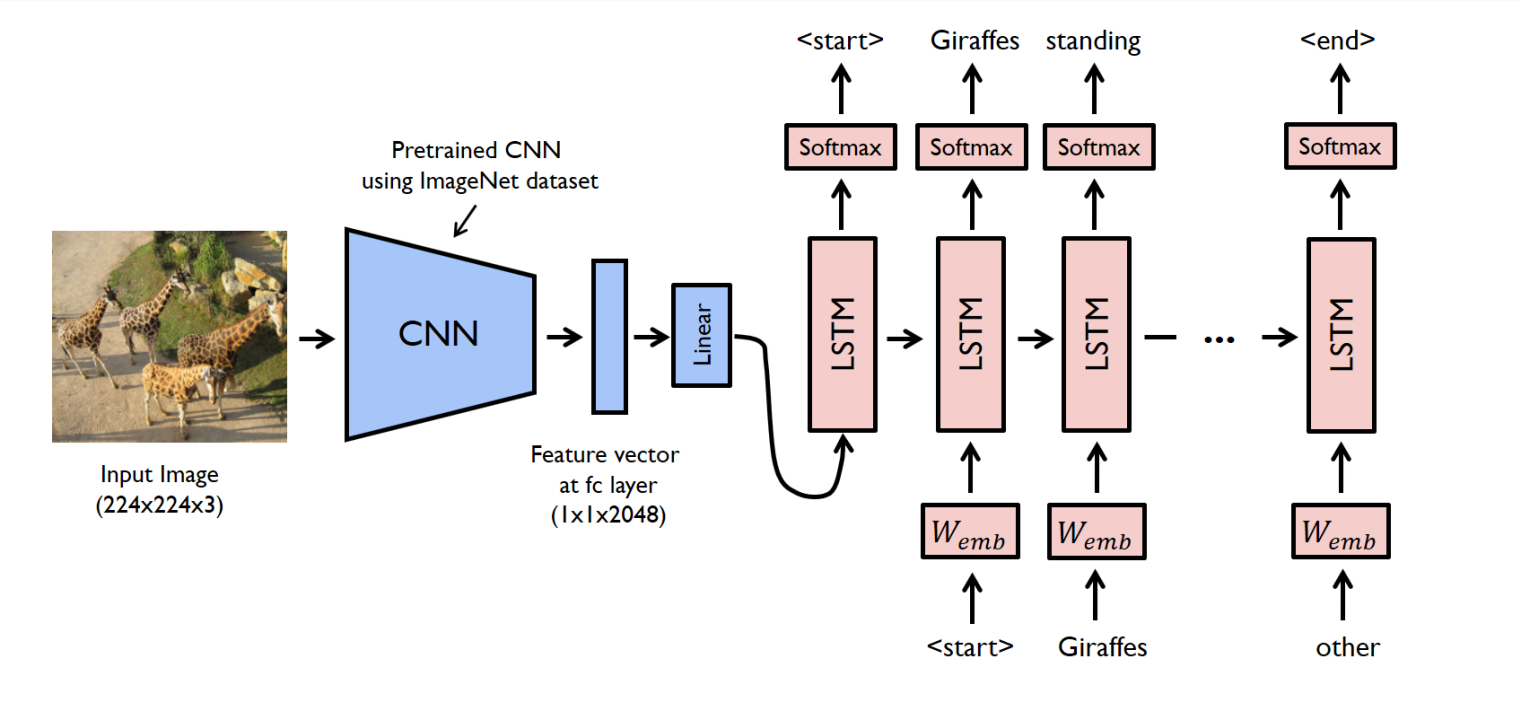
Transfer Learning



Let's revisit transfer learning. In our situation, which case is most suitable?

Case #2 **without any finetuning** - we'll use it simply as a static feature extractor

Image Captioning Final Architecture



The convnet weights are frozen. We learn an extra linear layer to allow us to go from convnet FC dimension to LSTM input dimension.

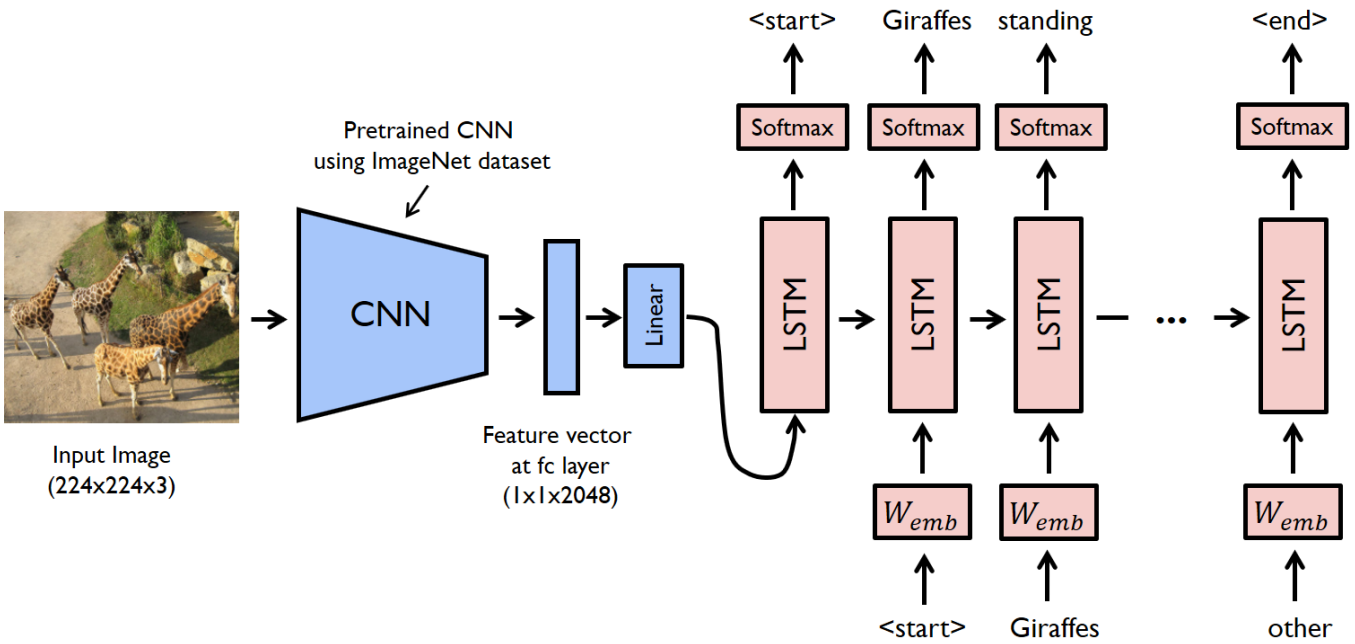
Note we also switched plain RNN to LSTM.

Loss Function

BLEU Score

Compare generated caption with given caption in labeled dataset:

$$\frac{\text{max correct number of n-grams in any reference sentence}}{\text{number n-grams in candidate sentence}}$$



DEMO!

Saliency Maps

Saliency Maps

Key idea: influence (saliency) of a pixel = magnitude * gradient

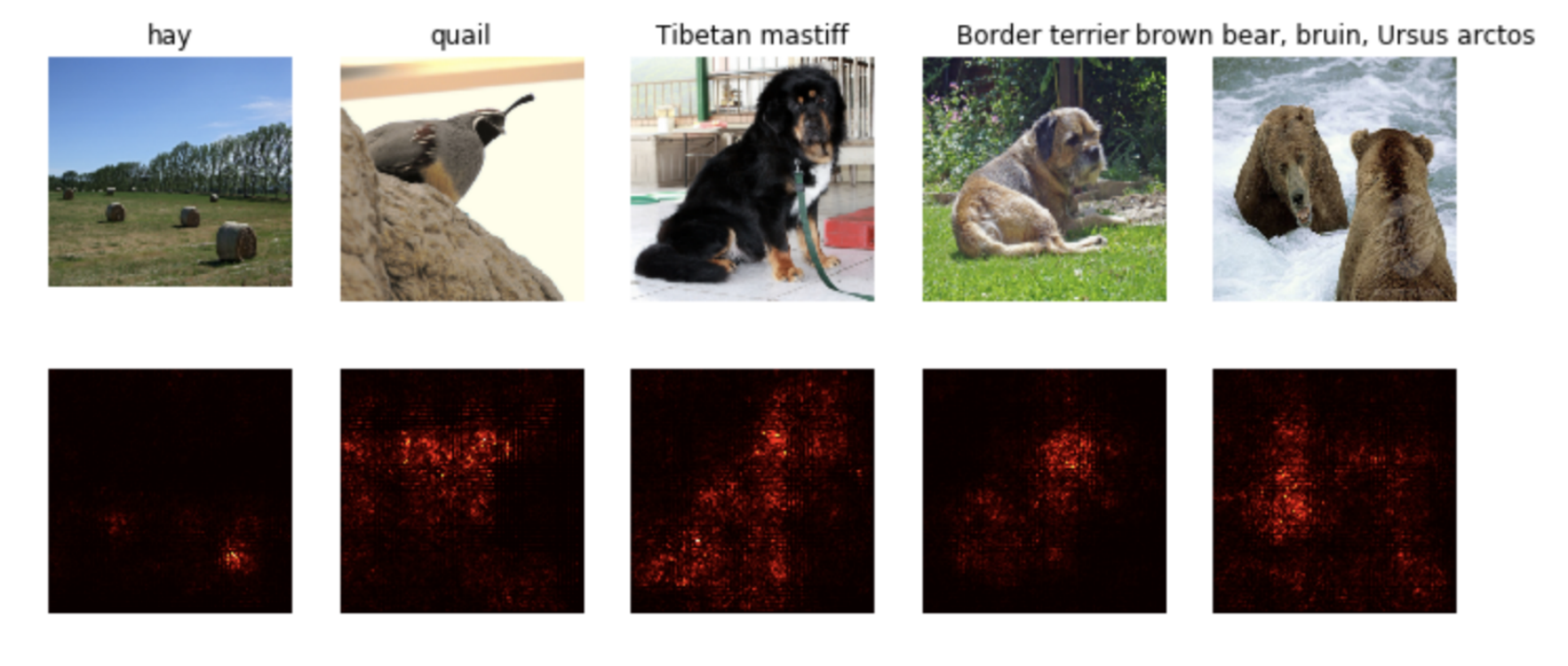
$$Saliency(i) = Magnitude(i) * Gradient(i)$$

$$Magnitude(i) = max(abs(R_i), abs(G_i), abs(B_i))$$

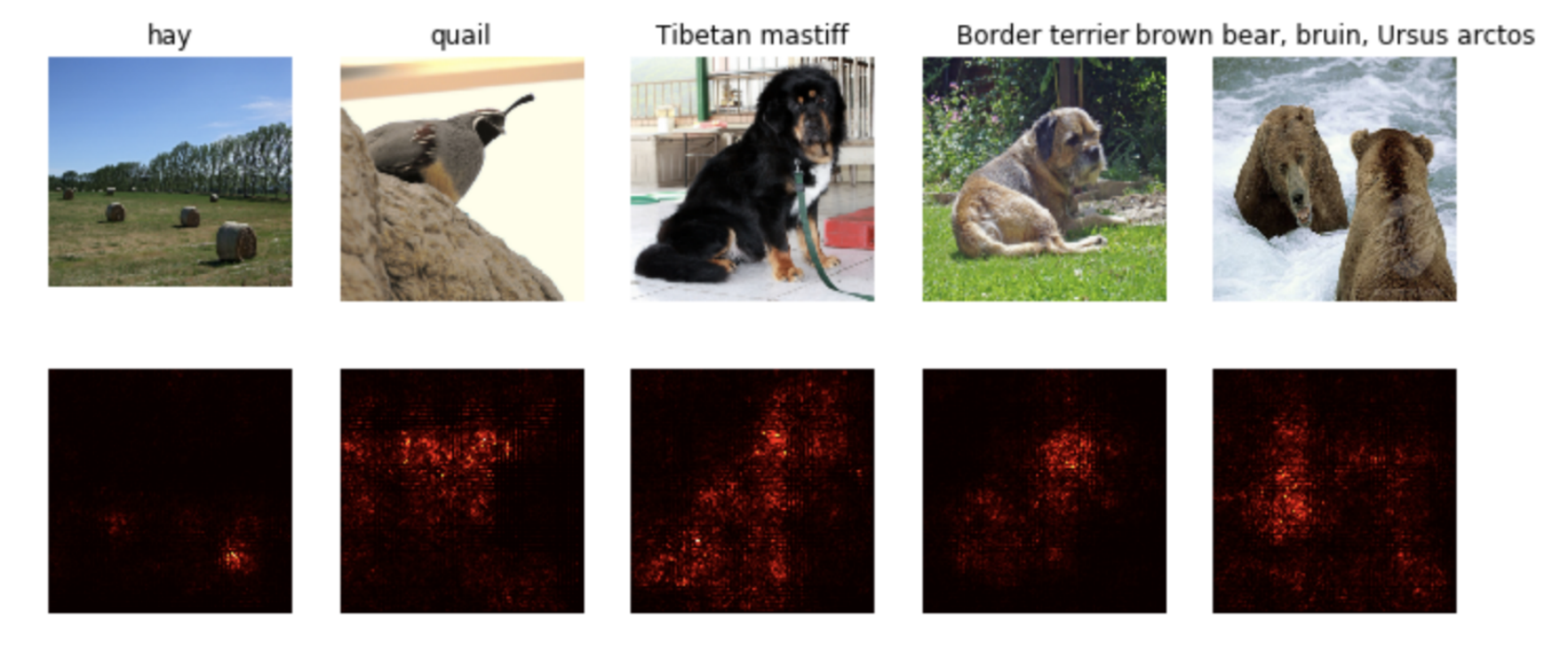
$$Gradient(i) = max(abs(\frac{\partial L}{\partial R_i}), abs(\frac{\partial L}{\partial G_i}), abs(\frac{\partial L}{\partial B_i}))$$

Repeat for all spatial pixel locations i

Saliency Maps



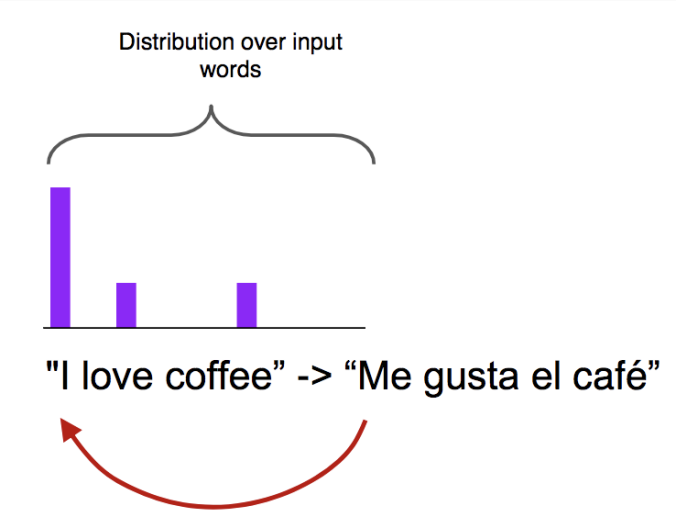
Saliency Maps

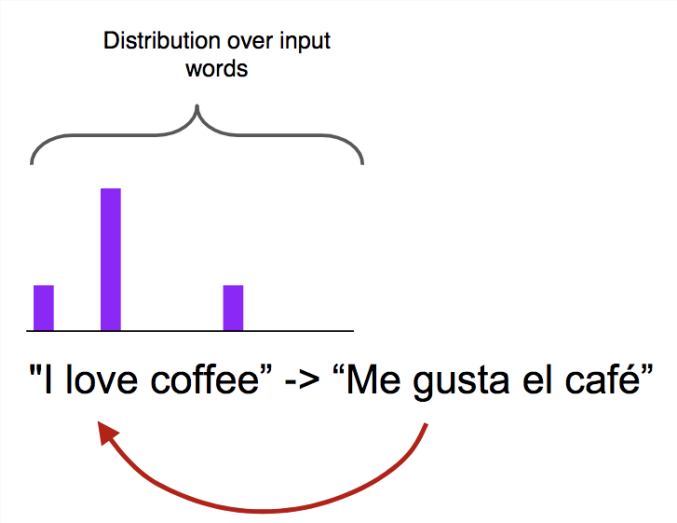


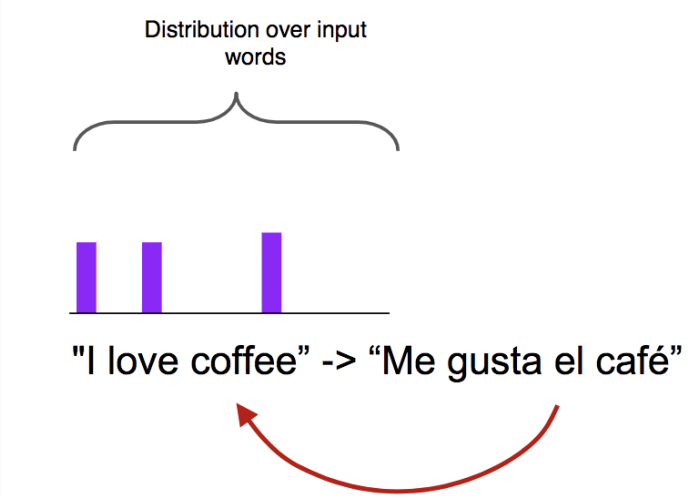
Is there we can let our image captioning model use this additional information? i.e. - how can we make it refer back to our image somehow?

Attention

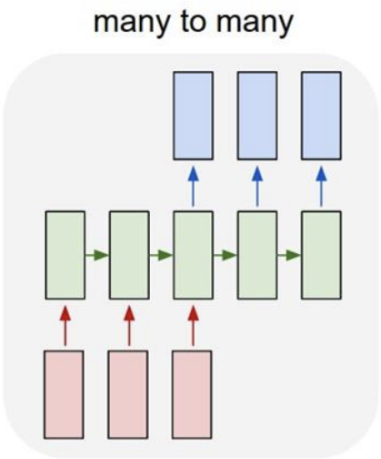
Remember: Attention for Translation



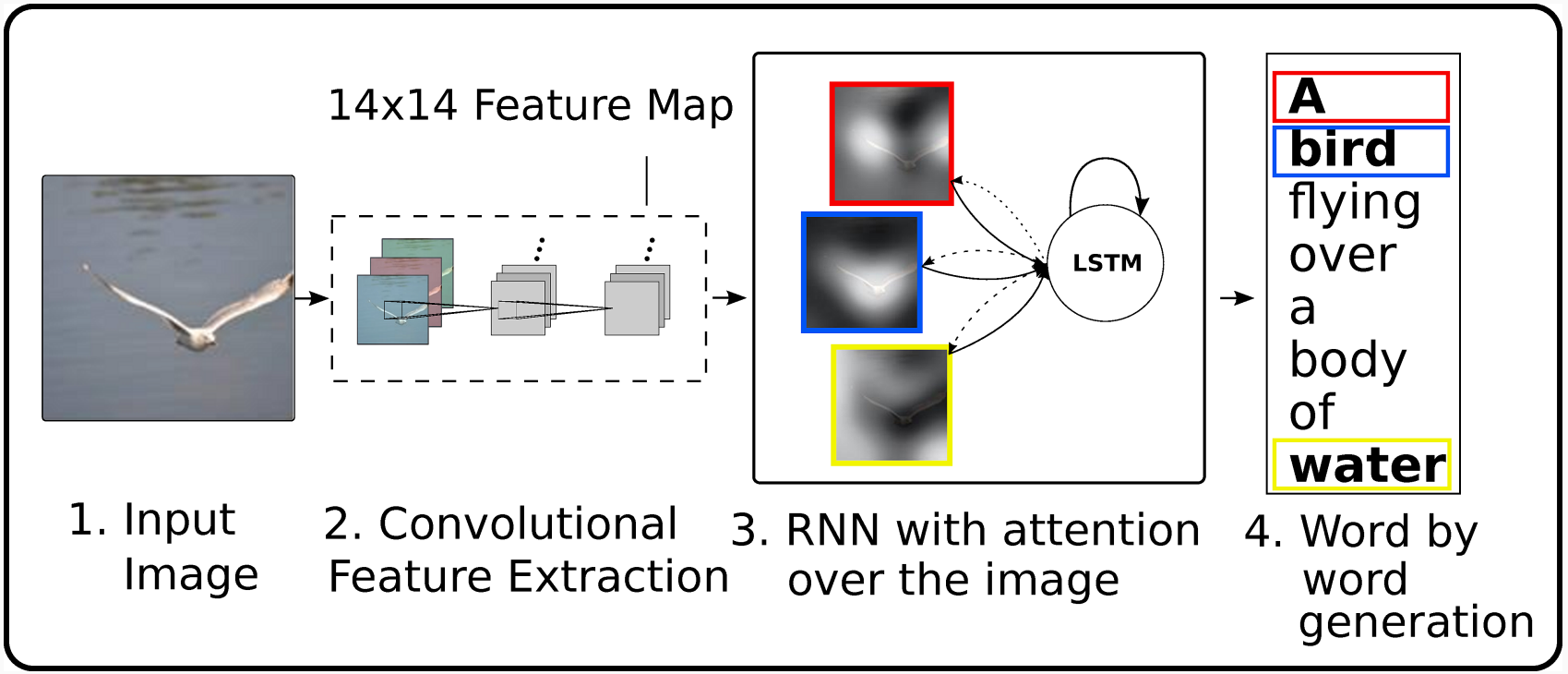




Bahdanau et al, "Neural Machine Translation by Jointly Learning to Align and Translate", ICLR 2015



Attention for Images

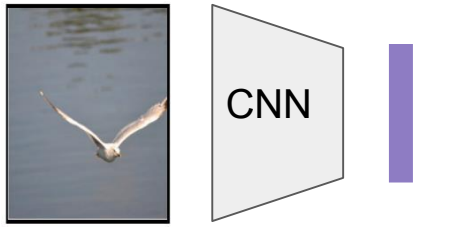


Key idea: we want to retain some spacial information about the image so we can use it in our attention.

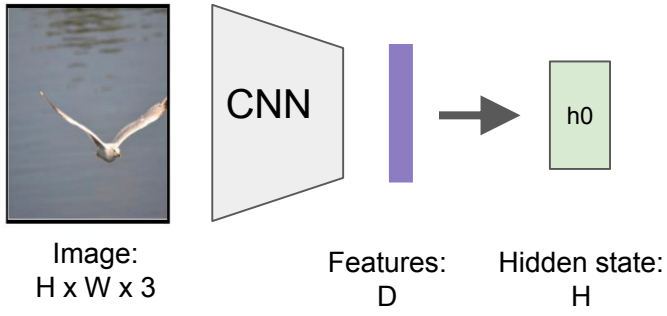
RNN for Captioning

Image:
H x W x 3

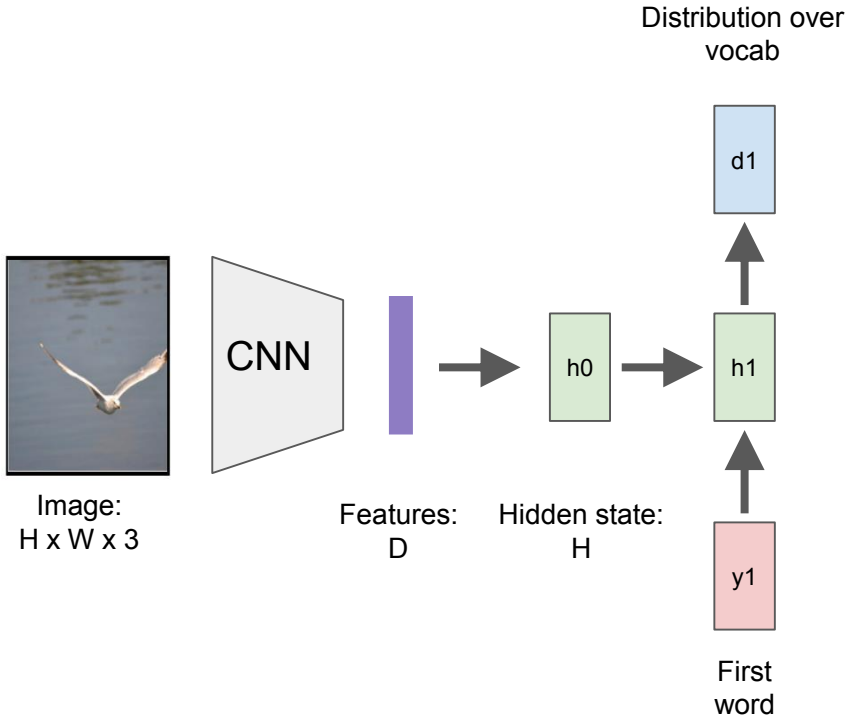
RNN for Captioning



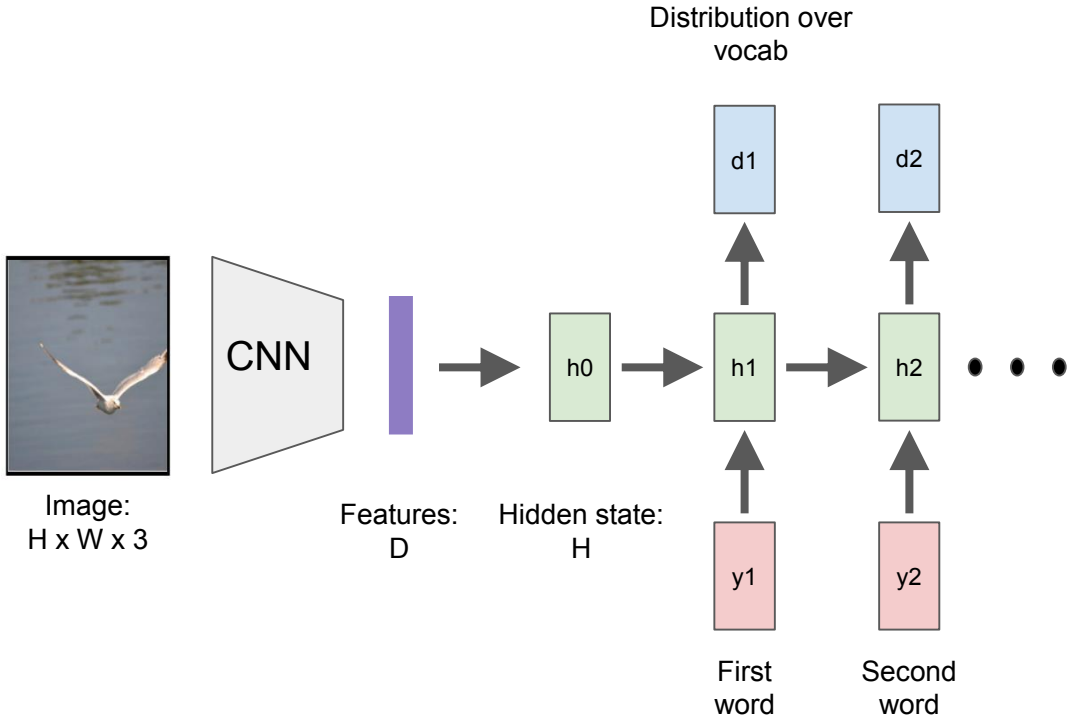
RNN for Captioning



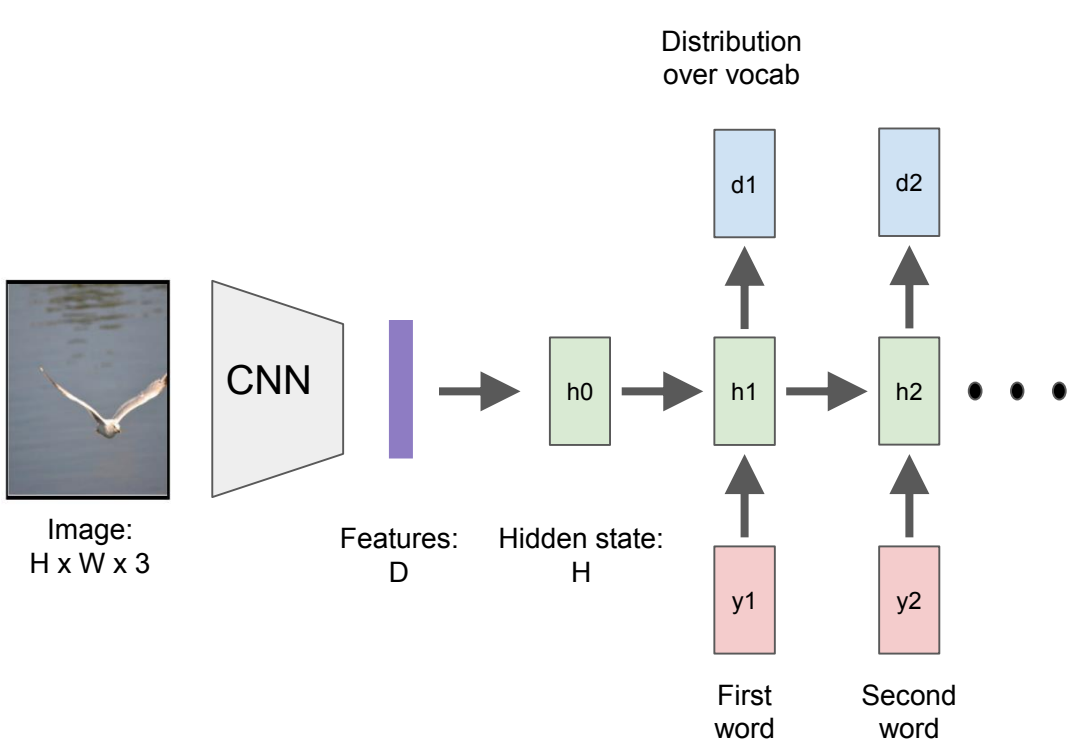
RNN for Captioning



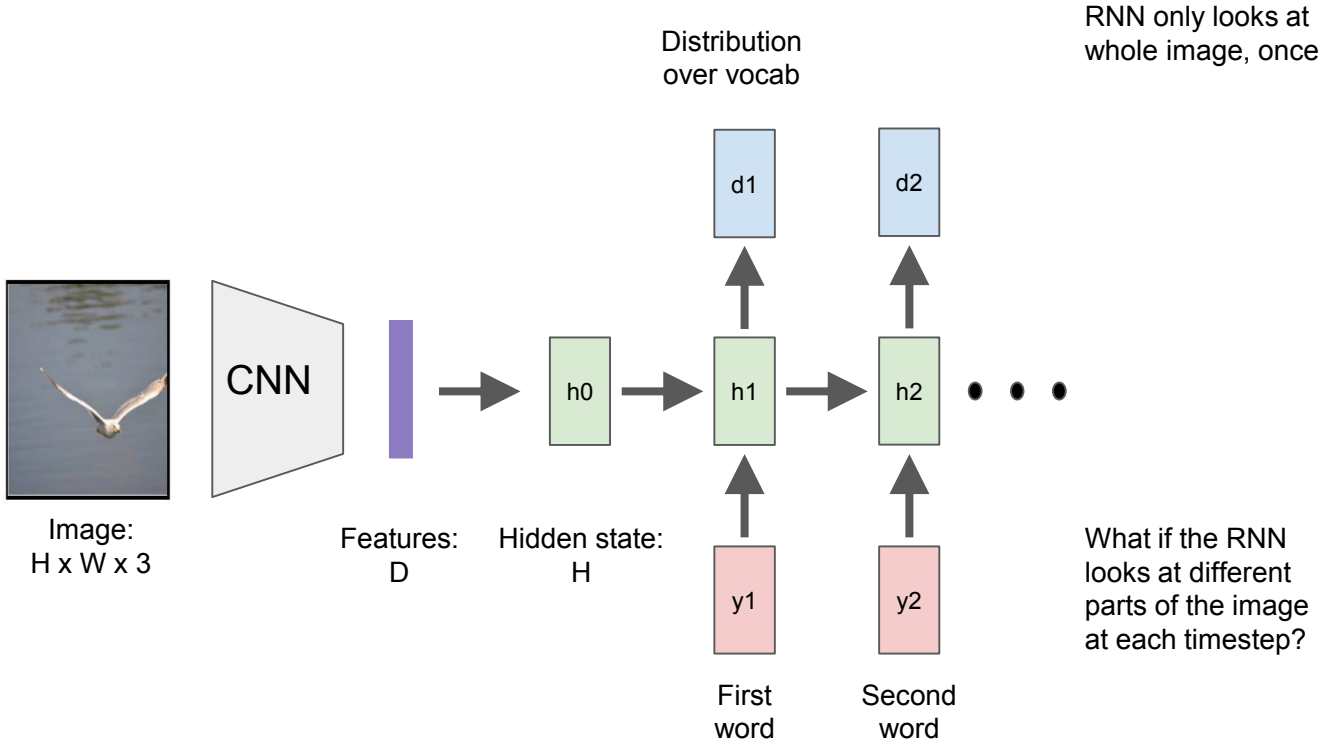
RNN for Captioning



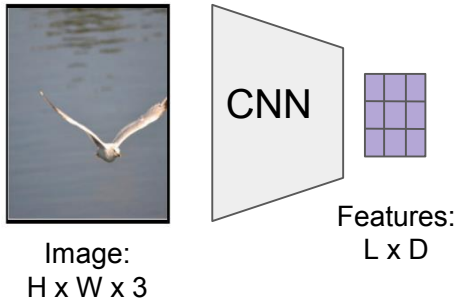
RNN for Captioning



RNN for Captioning

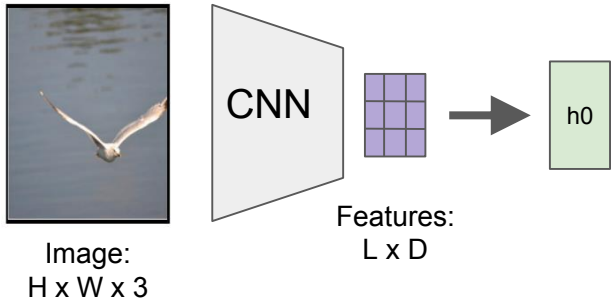


RNN for Captioning



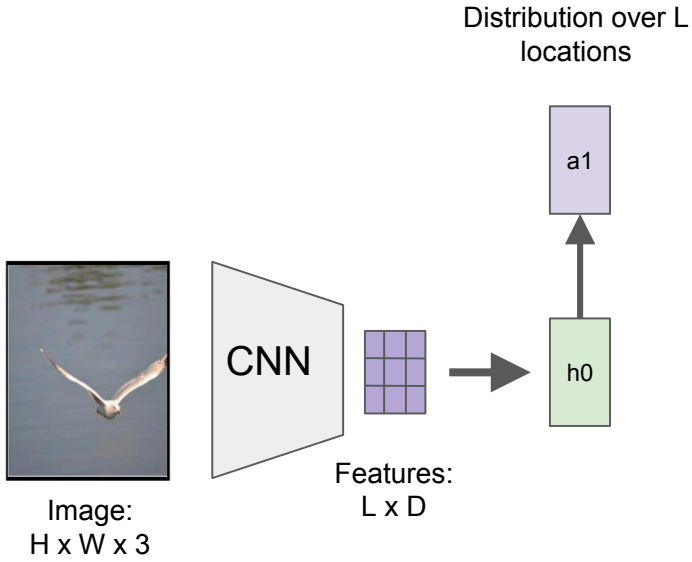
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

RNN for Captioning



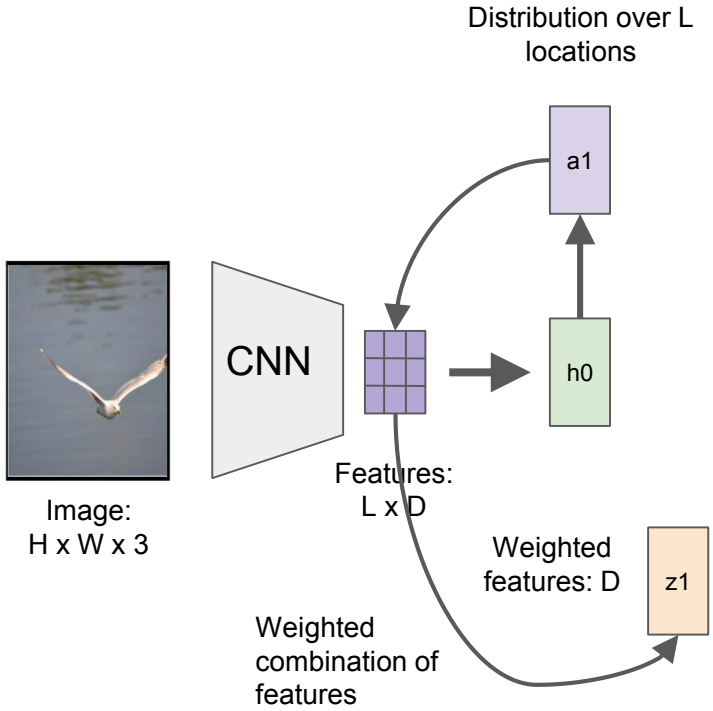
Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

RNN for Captioning

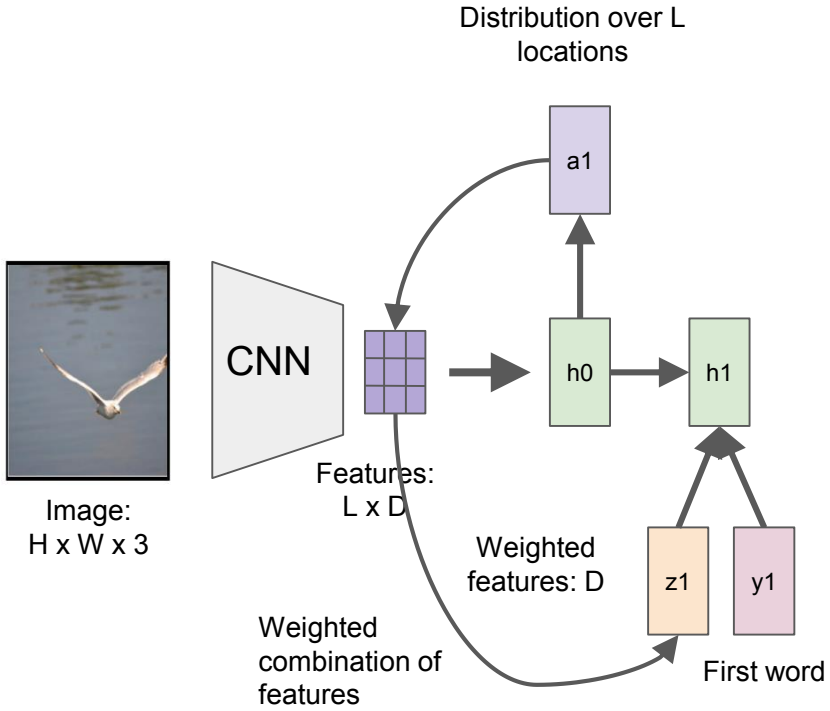


Xu et al, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention”, ICML 2015

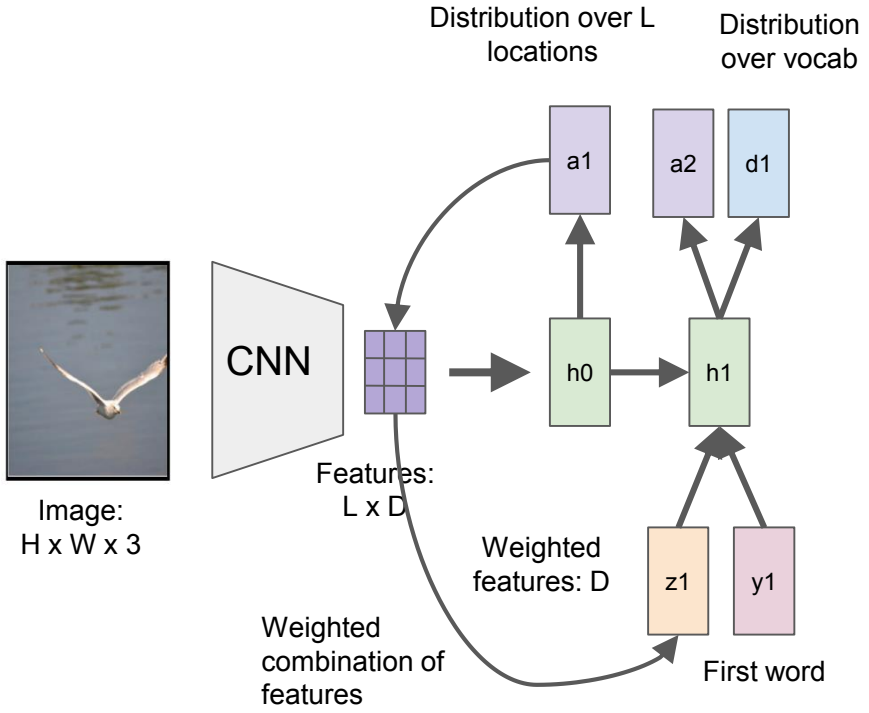
RNN for Captioning



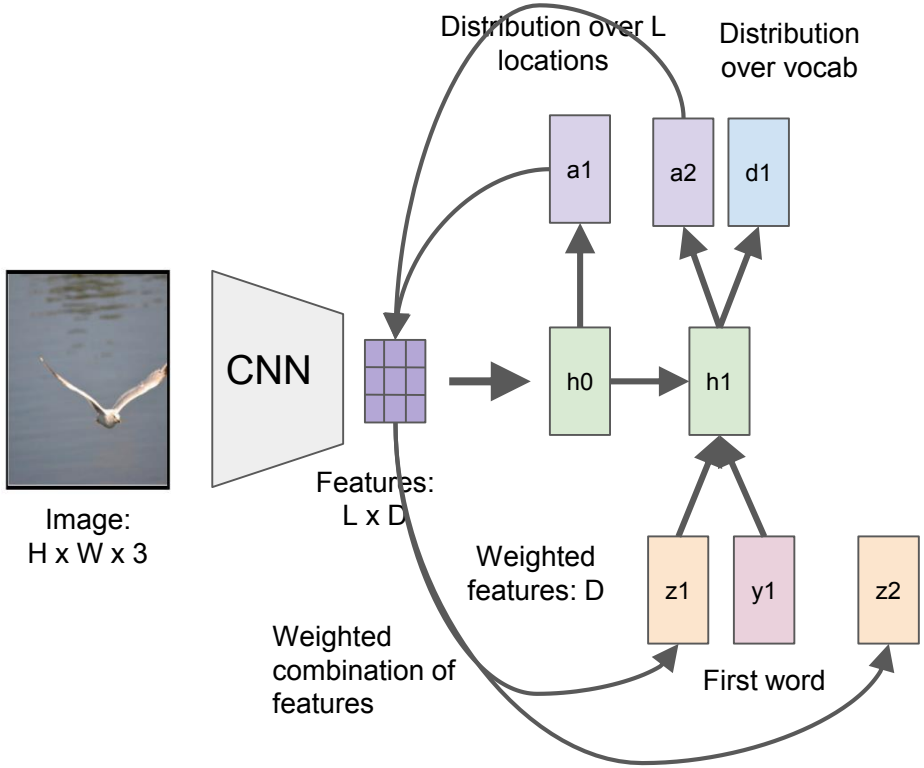
RNN for Captioning



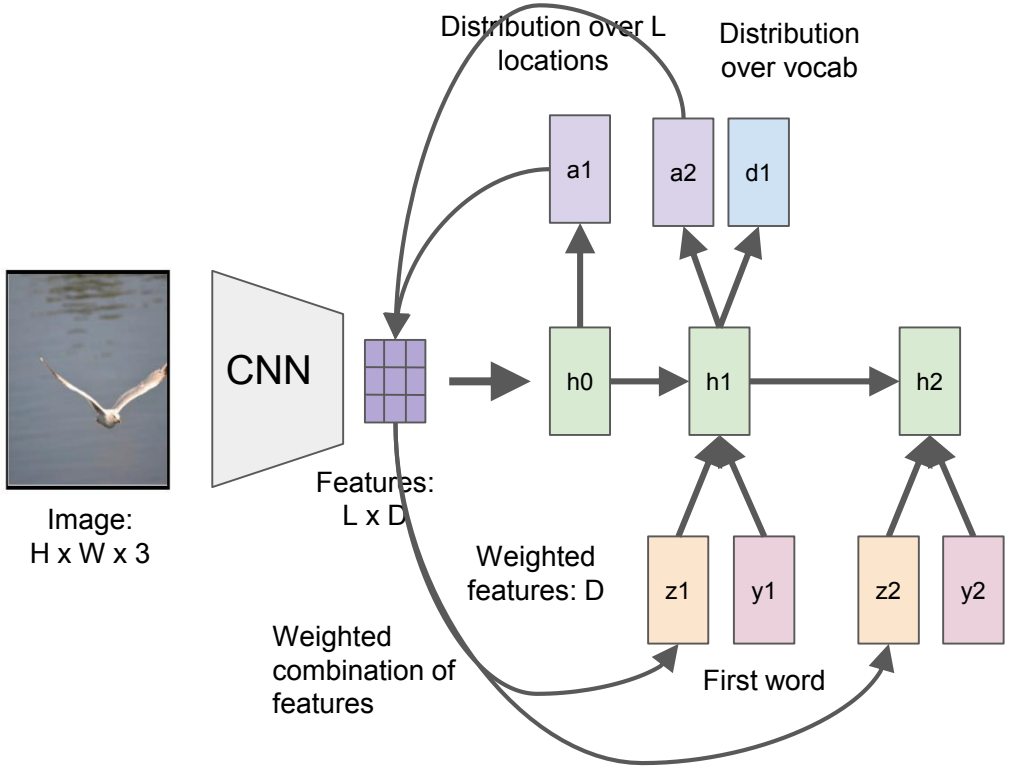
RNN for Captioning



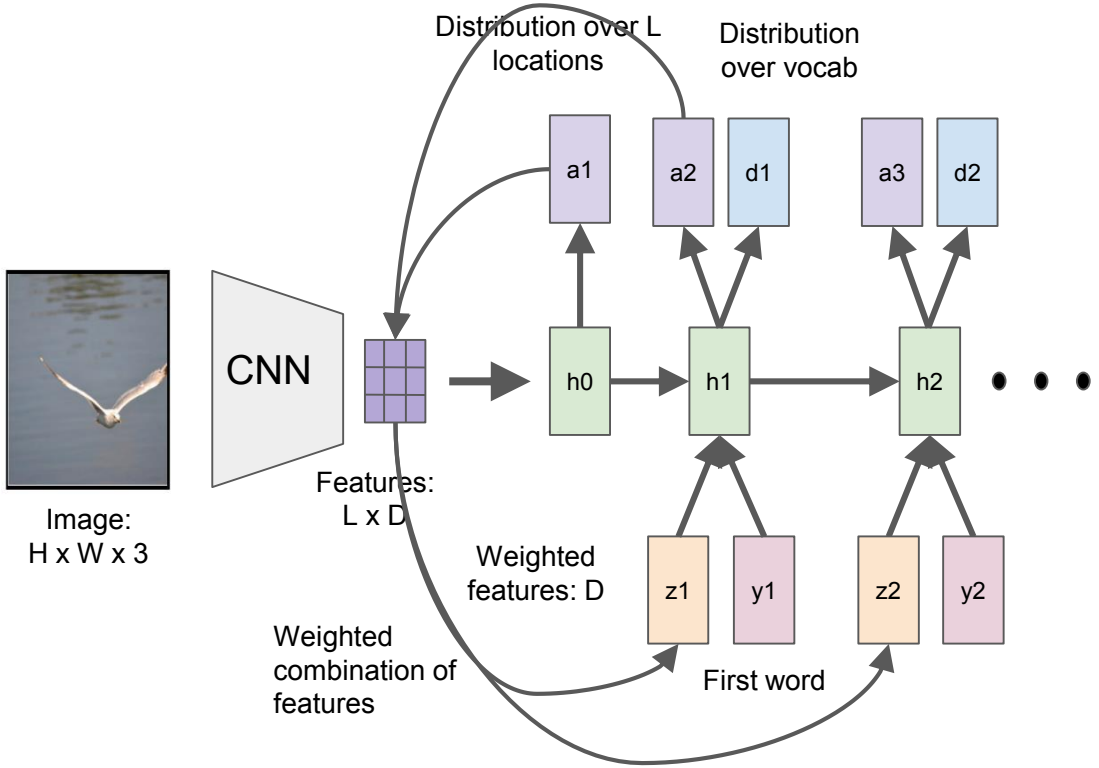
RNN for Captioning



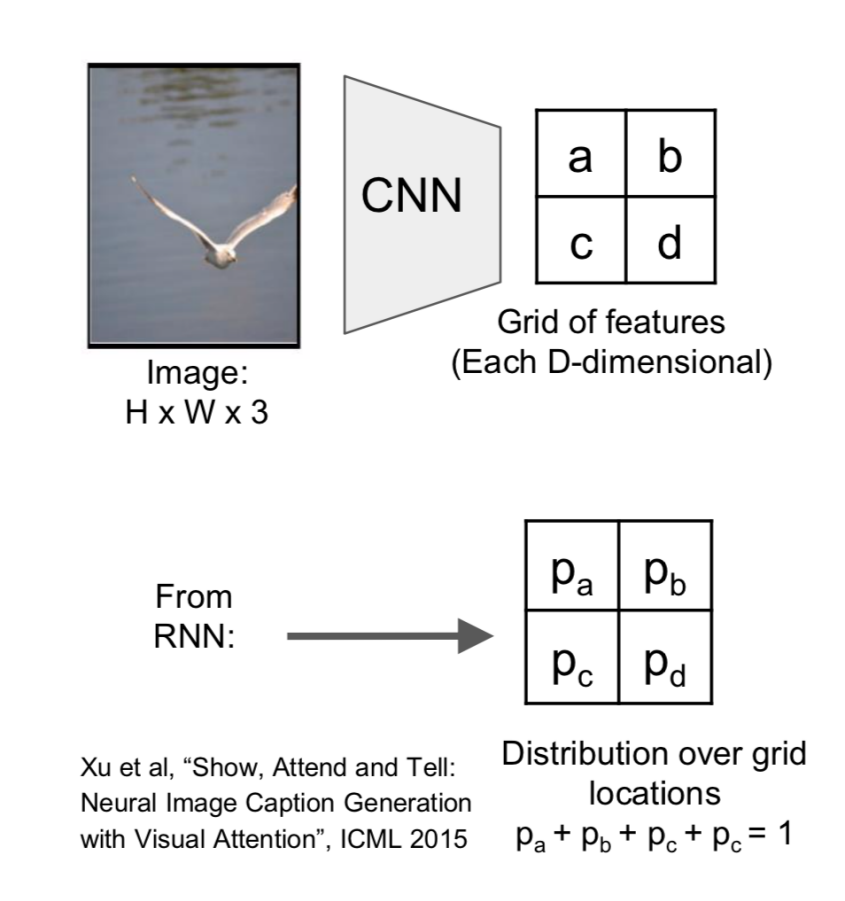
RNN for Captioning



RNN for Captioning

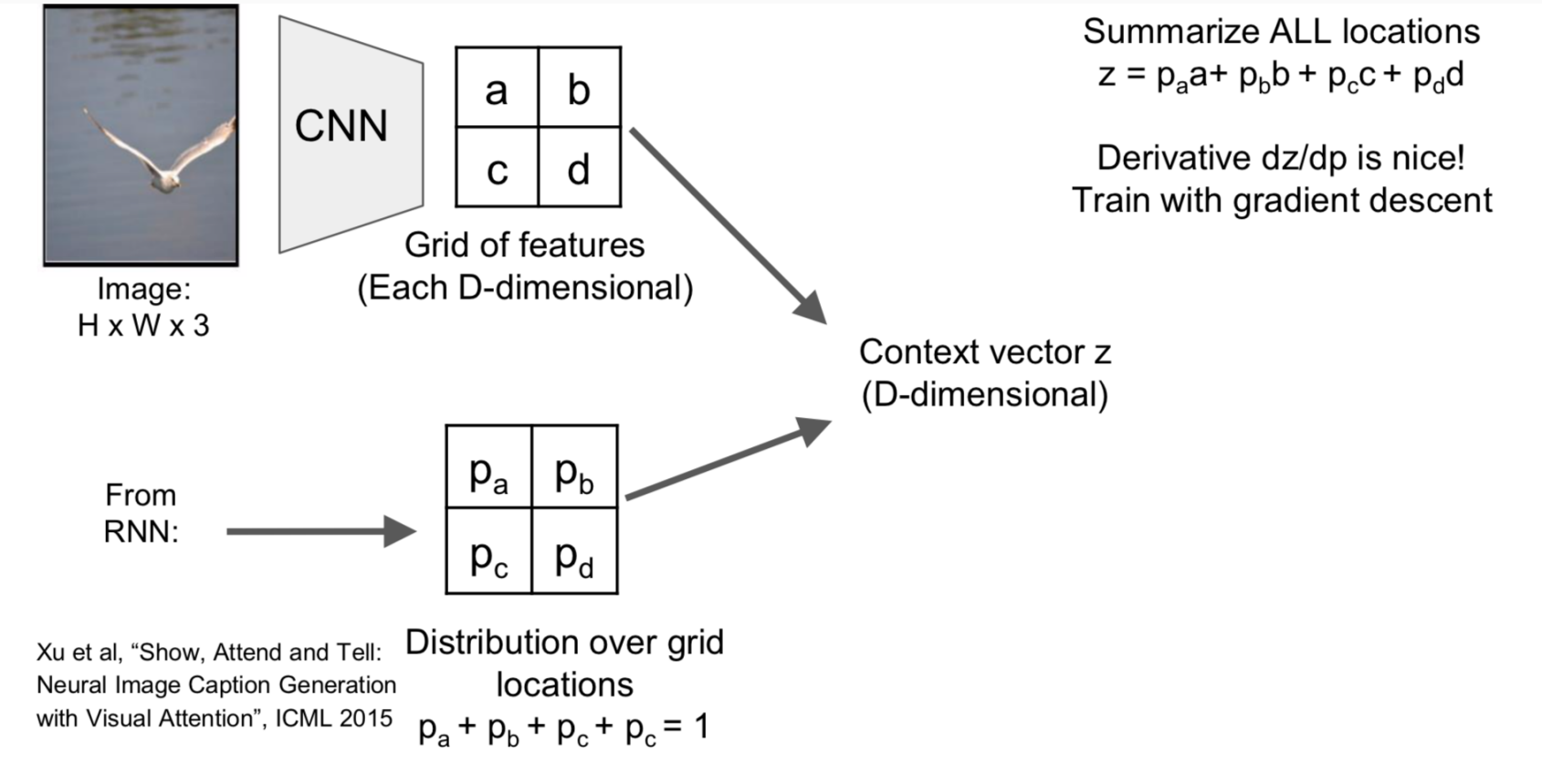


Attention for Images



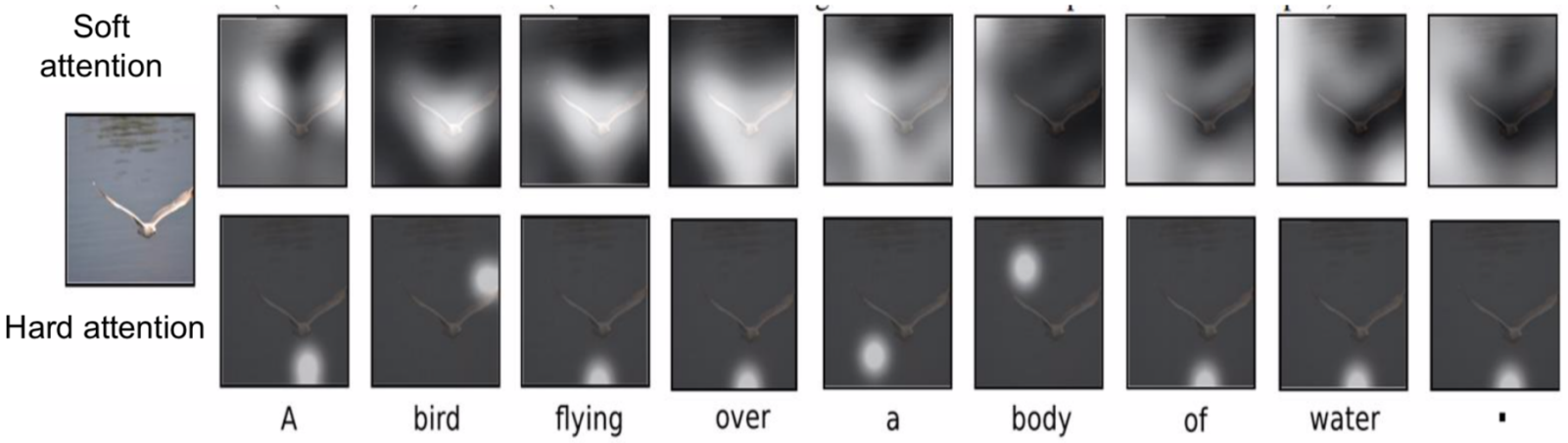
Our network will output a score for each grid cell, and we can use softmax to normalize them to 1.

Attention for Images



Using our scores we take a weighted average of our grid - this will produce a scalar. We use the same scores across the D dimensions of the grid, and we get a d-dim vector z.

Visualization of Attention - Image Captioning



Visualization of Attention - Image Captioning

A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

A giraffe standing in a forest with trees in the background.

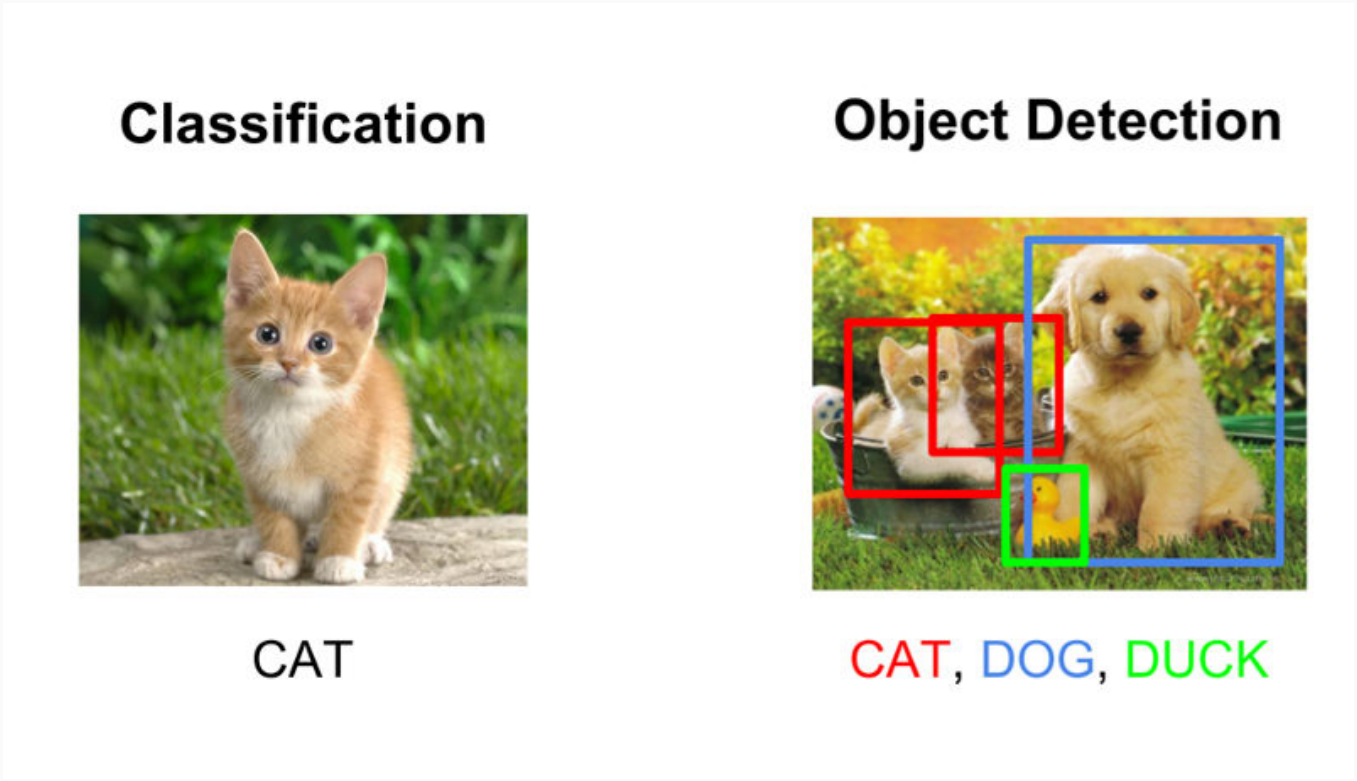
Importance of Attention

Attention has become a cornerstone of model architectures, on par with CNNs/RNNs. In fact, in many research areas, people are replacing recurrent models with attention since it's more effective!

Further reading: Attention Is All You Need, Transformer Networks

Image Detection

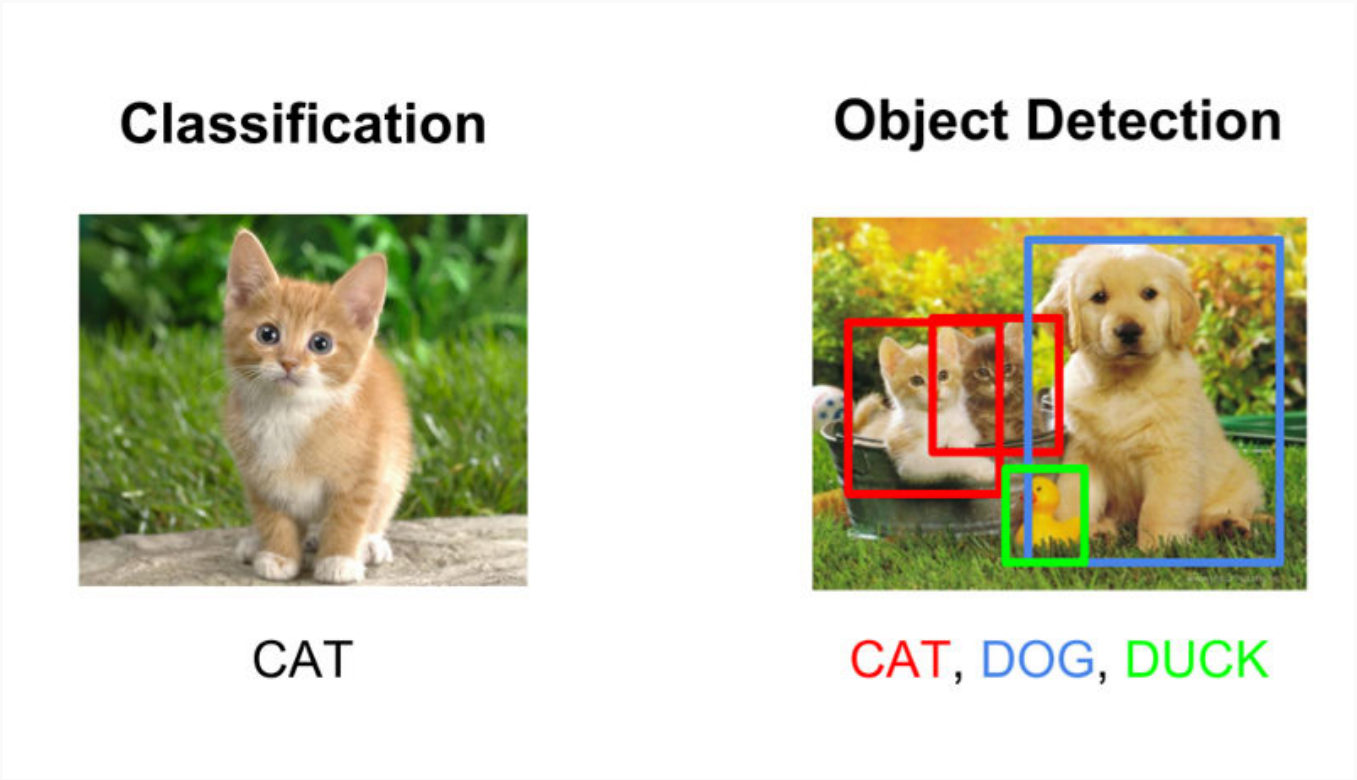
Image Detection Setup



For each image, output the **classes and bounding boxes** for each object in the image.

This is difficult - there can be a varying number of objects, all at different scales and poses, in each image.

Image Detection Setup



But deep learning has had much success! The first deep learning method increased accuracy 3x, and since then it's been increased another 3x - for a total of 9x improvement in the last 5 years!

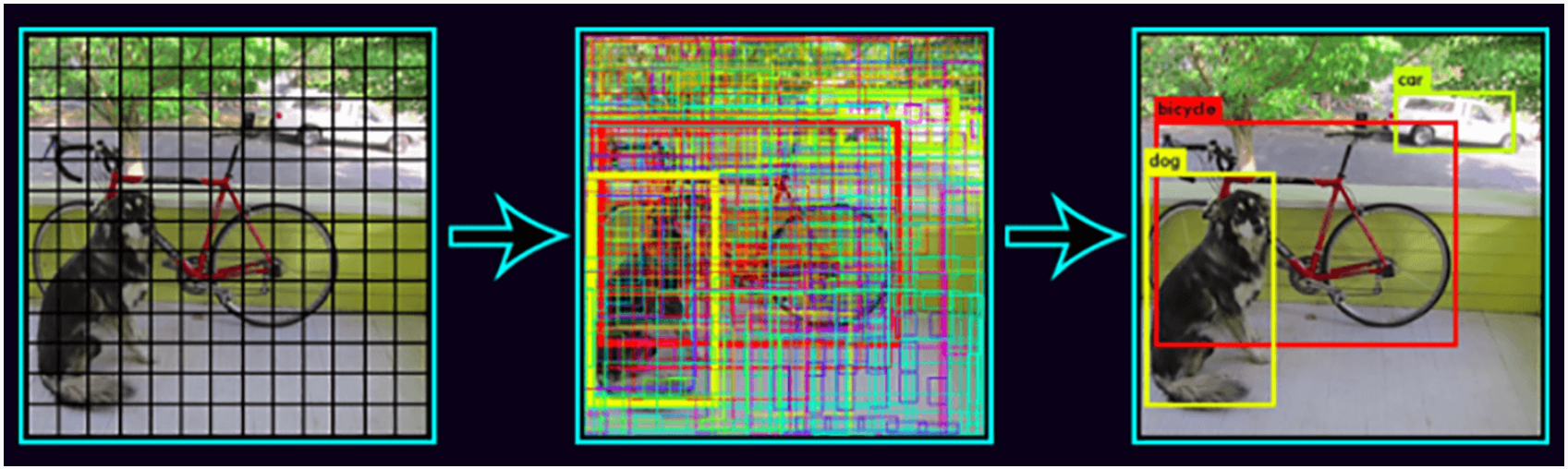
YOLO

YOLO - You Only Live Once

Superfast - can do object detection in real time!

https://youtu.be/MPU2HistivI

YOLO

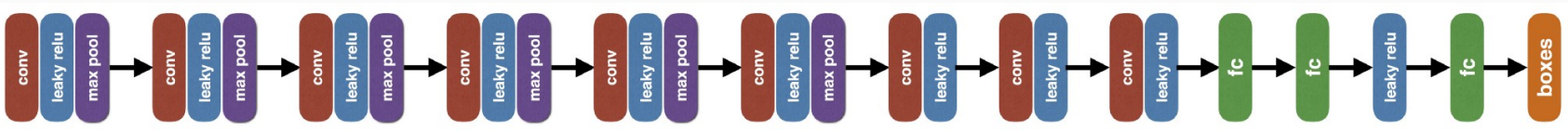


Key idea: divide image up into a grid, for each grid cell predict:

- Is there an object, and if so which? (class & confidence score)
- Coordinates of bounding box for that image

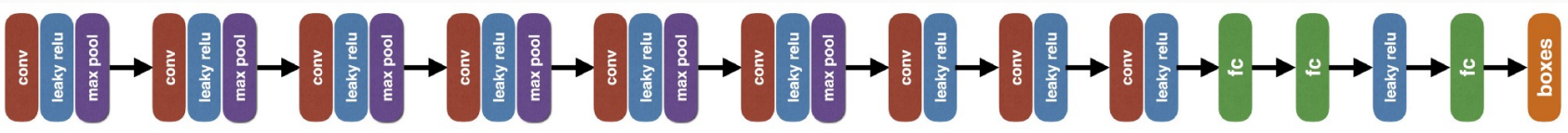
Then, only keep the bounding boxes with highest confidence.

YOLO Architecture

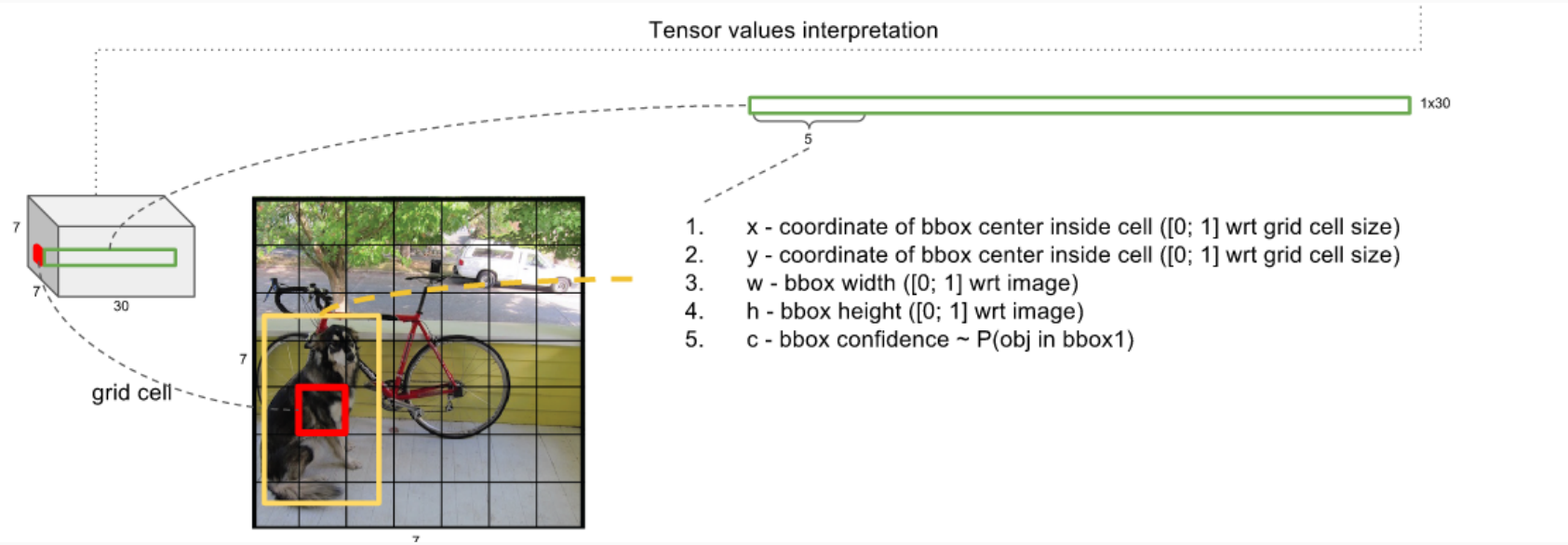


Typical convolutional network, at the end the output = 7 * 7 * 30

YOLO Architecture



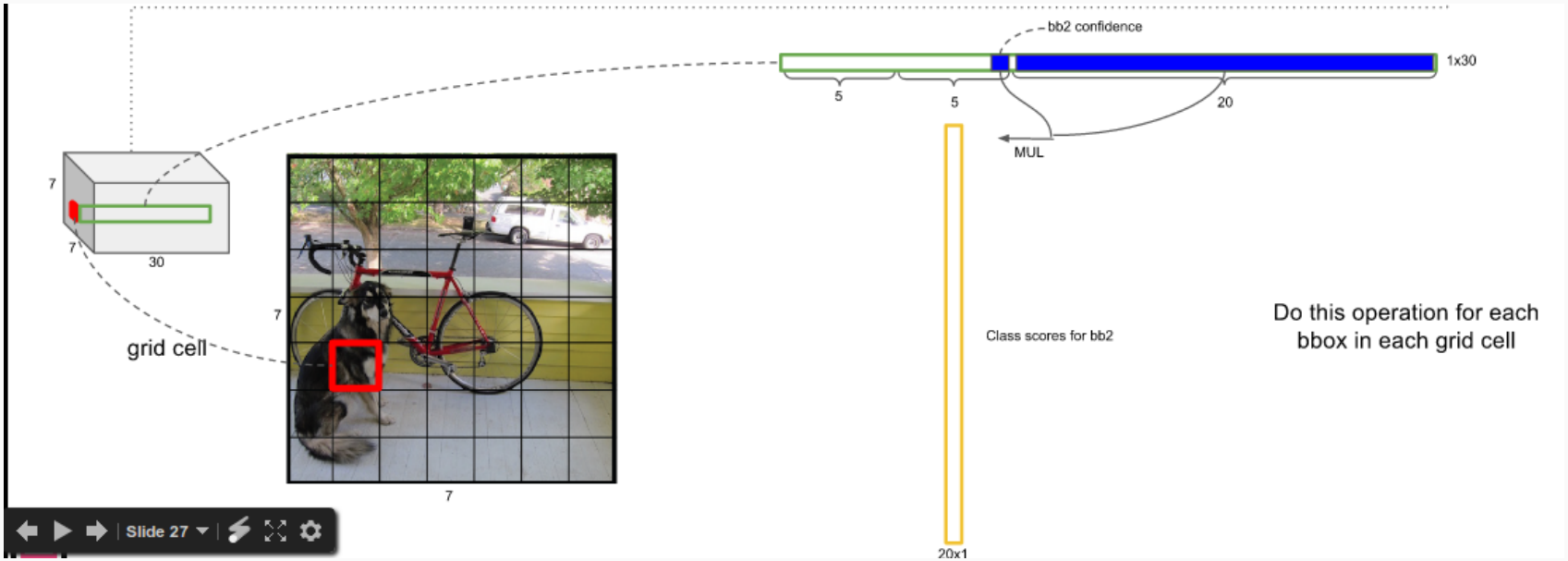
Typical convolutional network, at the end the output = 7 * 7 * 30



For each cell in 7x7 grid, use the corresponding 30 outputs to predict class & confidence scores and bounding boxes

YOLO Architecture

For each cell in 7x7 grid, use the corresponding 30 outputs to predict class & confidence scores and bounding boxes



Then apply non-maximal suppression to only show the most confident bounding boxes

Questions?