

RECENT ADVANCES IN DNN

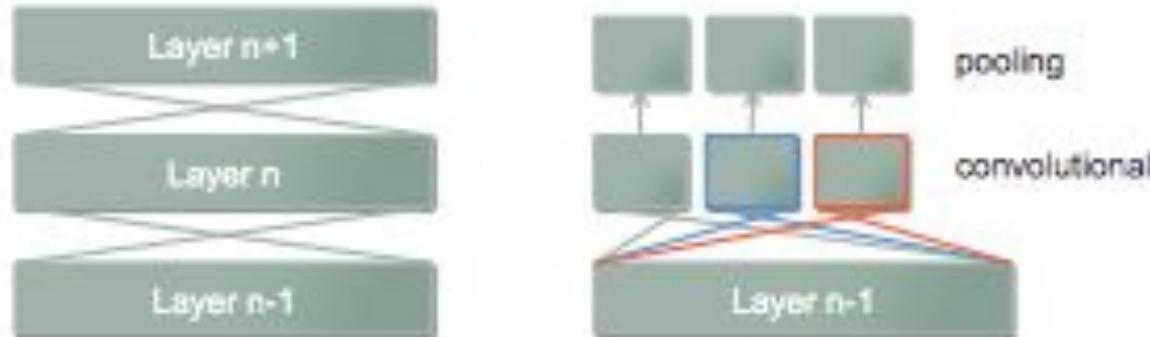
Neural networks

- Fully connected networks
 - Neuron
 - Non-linearity
 - Softmax layer
- DNN training
 - Loss function and regularization
 - SGD and backprop
 - Learning rate
 - Overfitting – dropout, batchnorm

CNNs & RNNs

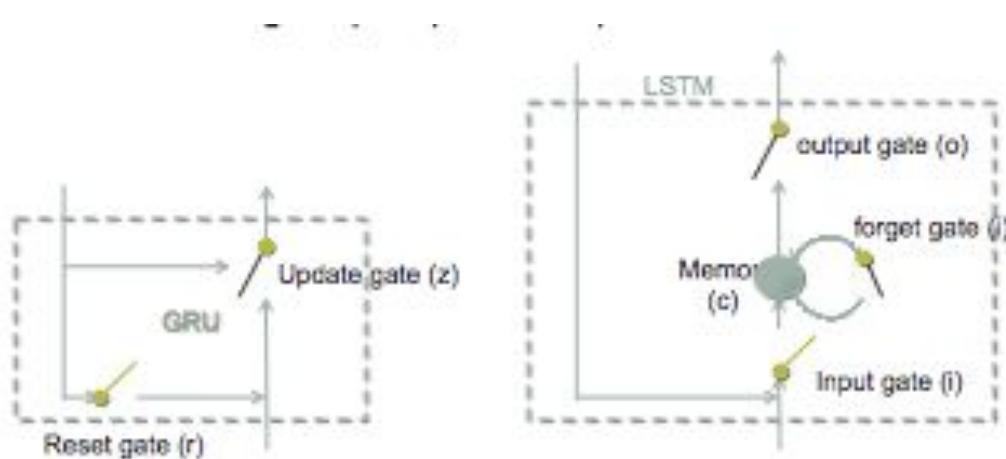
- CNNs

- Matched filters
- Convolution layer
- Subsampling layer
 - Maxout, 1x1 convolution

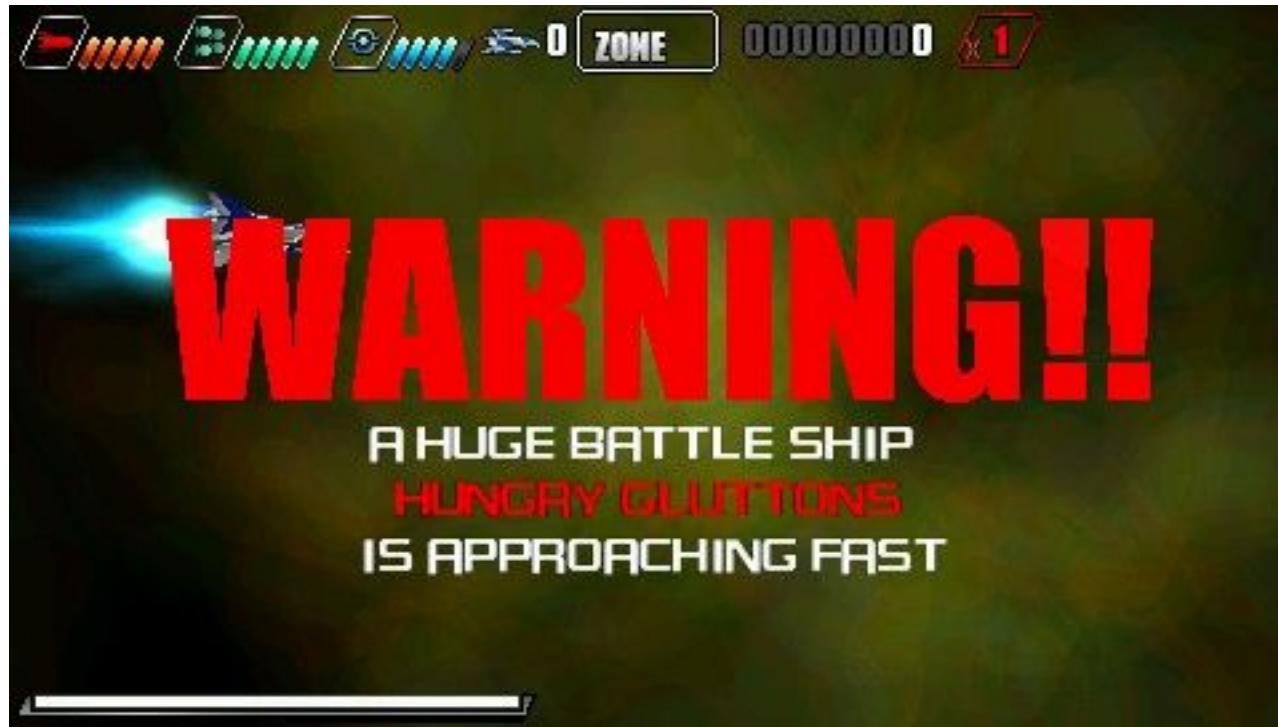


- RNNs

- Backpropagation Through time
- GRUs (2 Gates)
- LSTMs (3 Gates, explicit memory cell)



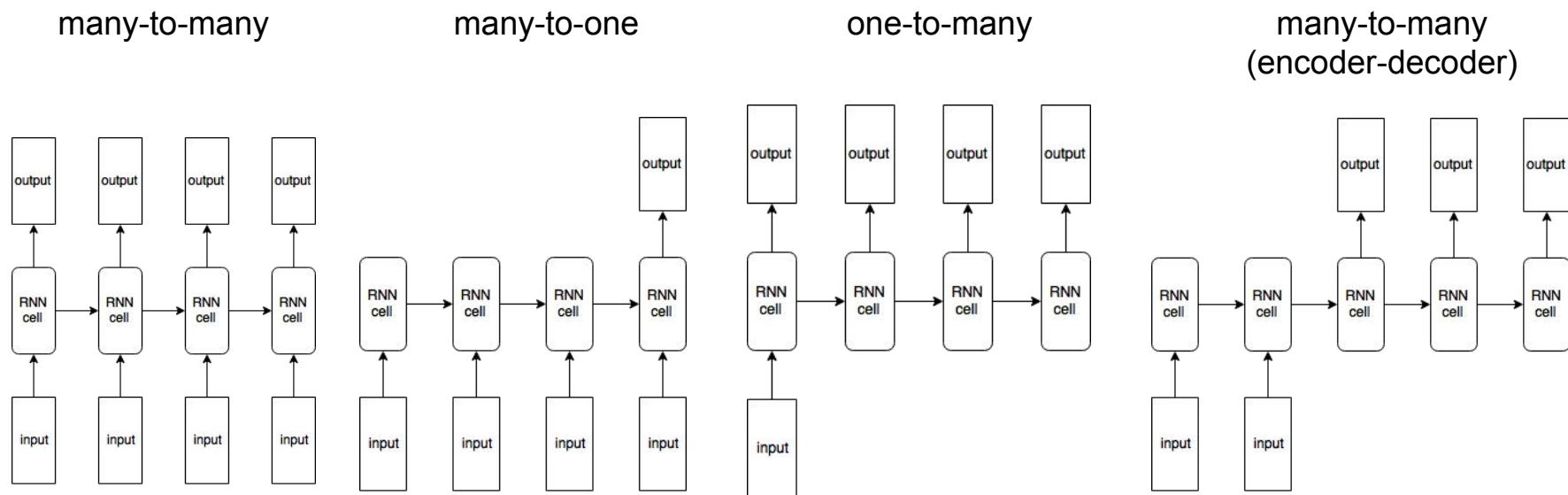
- You are not expected to understand any of the following parts in this lecture.
- This lecture serves as an overview of things going on with deep learning.
- If you are curious about any terms that gets thrown out there, see the relevant papers.



Misc topics

- Encoder-Decoder
- Attention modeling
- Transfer Learning, Multi-task models, Multi-models
- Object detection
- Alternatives to Backprop
 - 2nd order methods
 - Synthetic gradients
 - Genetic Algo
- Reinforcement Learning <- Next lecture!
- GAN <- Next Next lecture!
 - Adversarial inputs

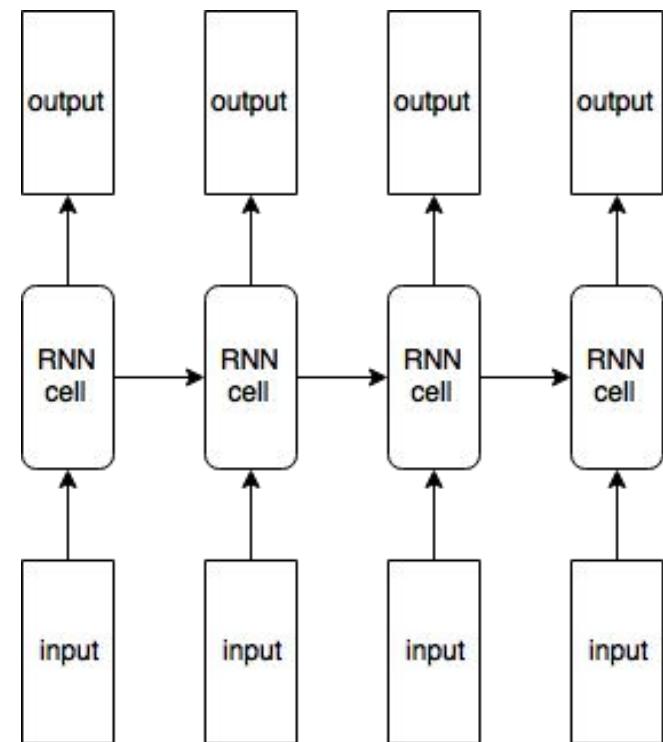
Different types of RNN architectures



Reference: The Unreasonable Effectiveness of Recurrent Neural Networks,
<http://karpathy.github.io/2015/05/21/rnn-effectiveness/>

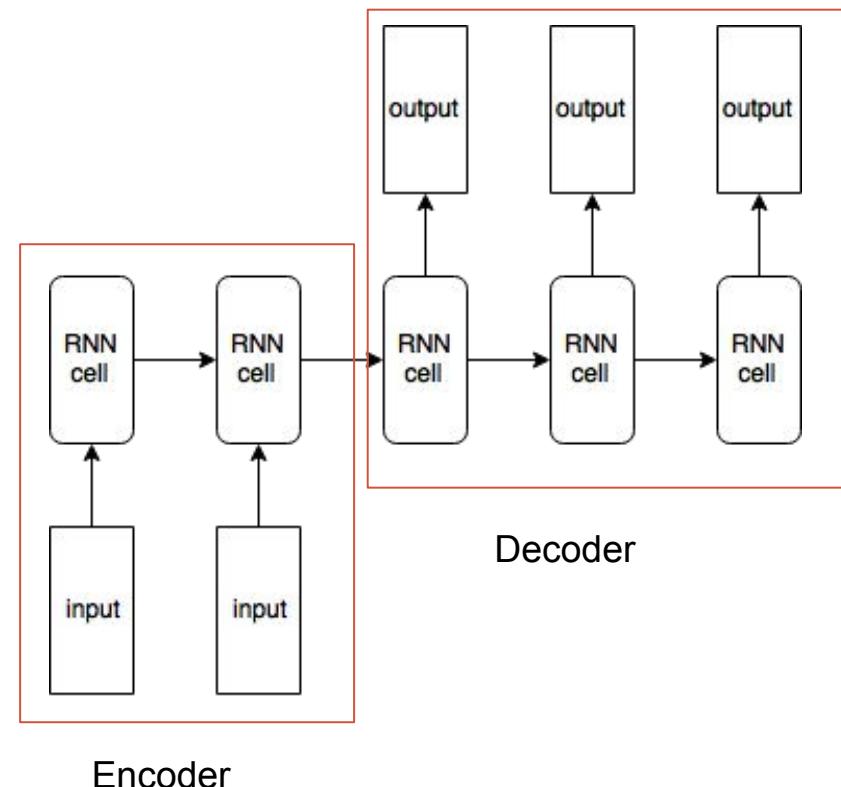
Many-to-many

- You have seen and implemented this type of RNN architecture in your homework (nowcasting)
- Sequence Input, Sequence Output



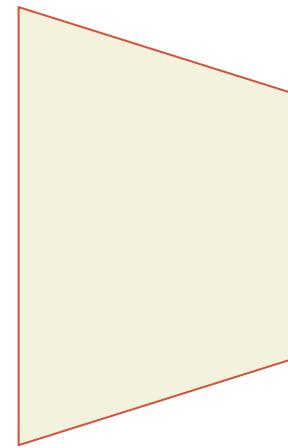
Many-to-many (encoder-decoder)

- Sequence Input, Sequence output
- These two sequences can be of different length
- E.g. Machine Translation
 - Input: English Sentence
 - Output: Thai Sentence

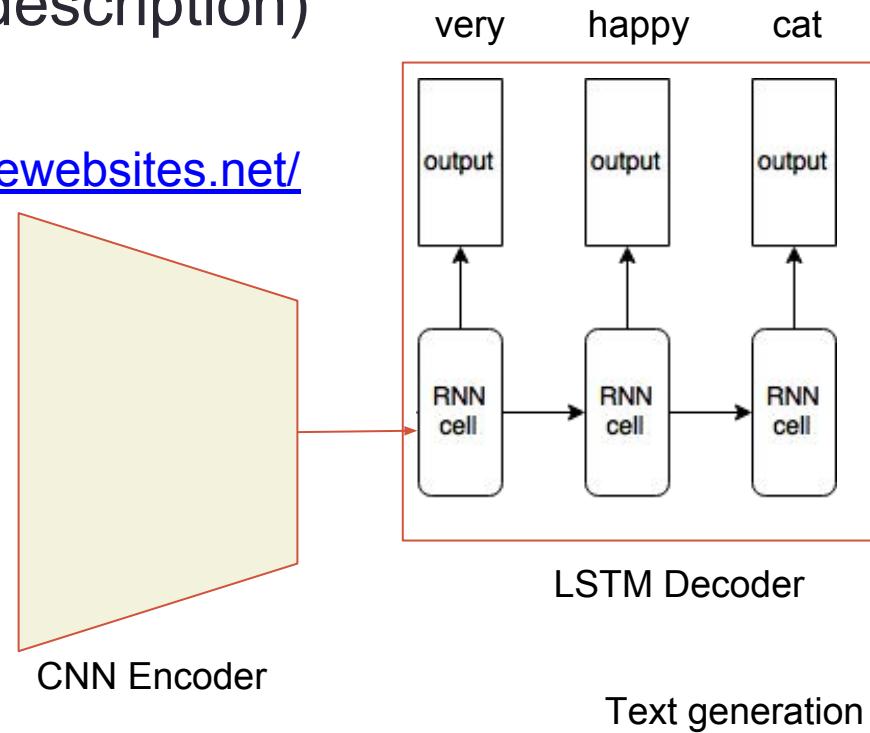


Encoder-decoder

- Input and output can be of different modality
 - Speech, text, image, etc.
- Speech to image (image generation)
- Image to text (image description)
- Image to code
 - <https://sketch2code.azurewebsites.net/>

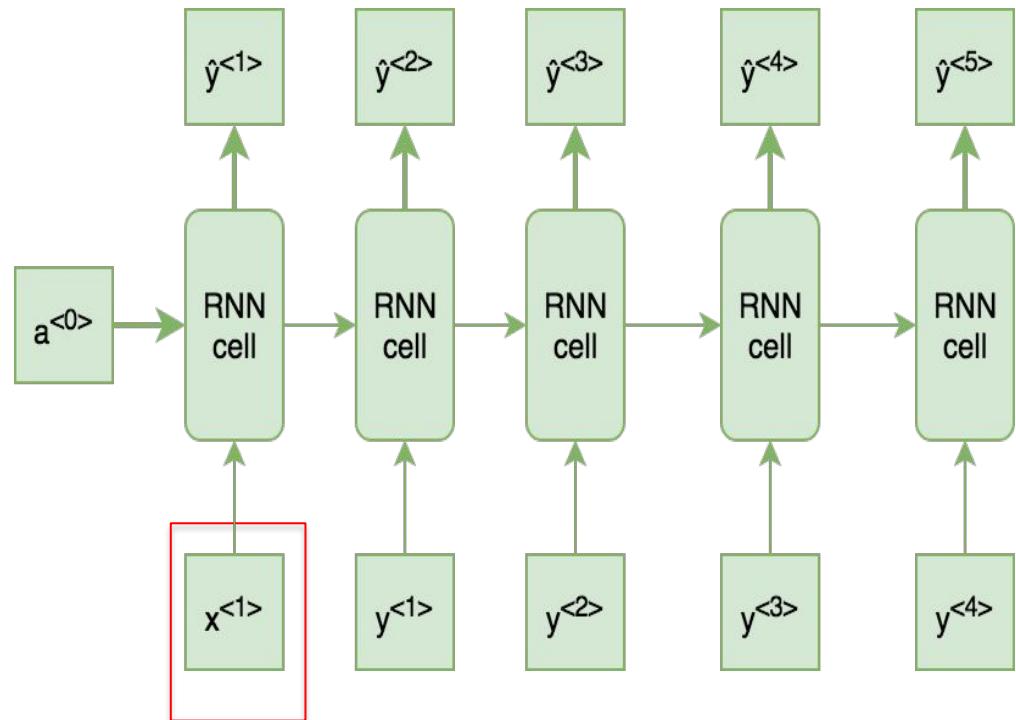


CNN Encoder



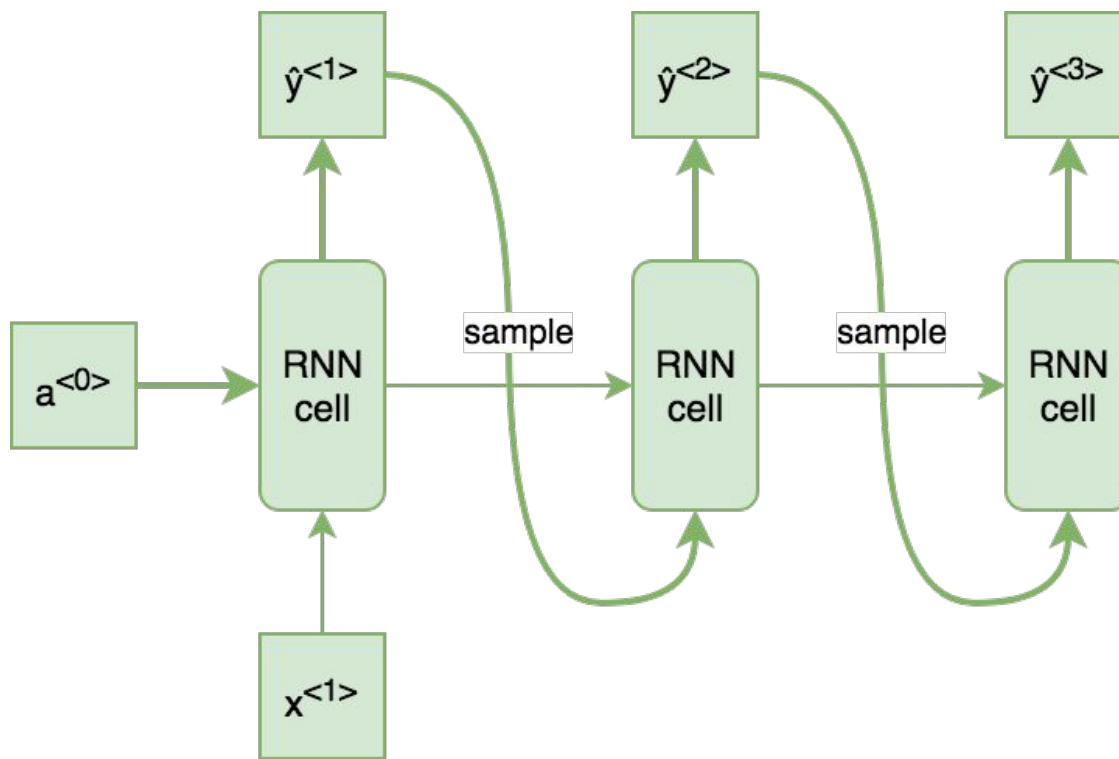
Text generation model (training)

- One-to-Many RNN
- The only real input is $x^{<1>}$
- $a^{<0>}$ is the initial hidden state.
- \hat{y} is the predicted output.
- y is an actual output.
- During **the training phase**, instead of using the predicted output to feed into the next time-step, we use the actual output.



Text generation model (inference)

- To generate a novel sequence, the inference model (testing phase) randomly samples an output from a softmax distribution.



We are forced to pick one answer then proceed to the next step.

Auto-regressive decoding

Can reduce the error by doing **beamsearch** (try different possibilities but still pick at answer for each search space)

Text generation example

<https://medium.com/@samim/obama-rnn-machine-generated-political-speeches-c8abd18a2ea0>

SEED: Democracy

Good morning. One of the borders will be able to continue to be here today. We have to say that the partnership was a partnership with the American people and the street continually progress that is a process and distant lasting peace and support that they were supporting the work of concern in the world. They were in the streets and communities that could have to provide steps to the people of the United States and Afghanistan. In the streets—the final decade of the country that will include the people of the United States of America. Now, humanitarian crisis has already rightly achieved the first American future in the same financial crisis that they can find reason to invest in the world.

Thank you very much. God bless you. God bless you. Thank you.

Music generation

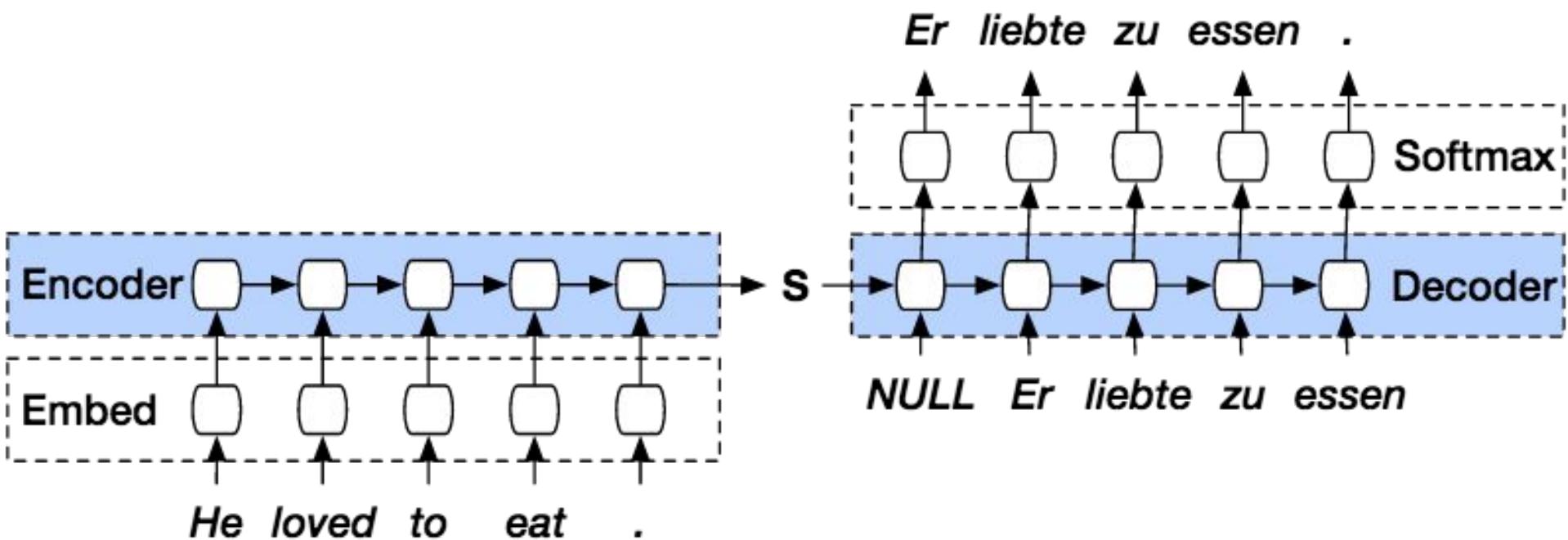
Input: initial note
Predict: note differences
and chord



https://www.researchgate.net/publication/327811979_Algorithmic_Music_Composition_Comparison

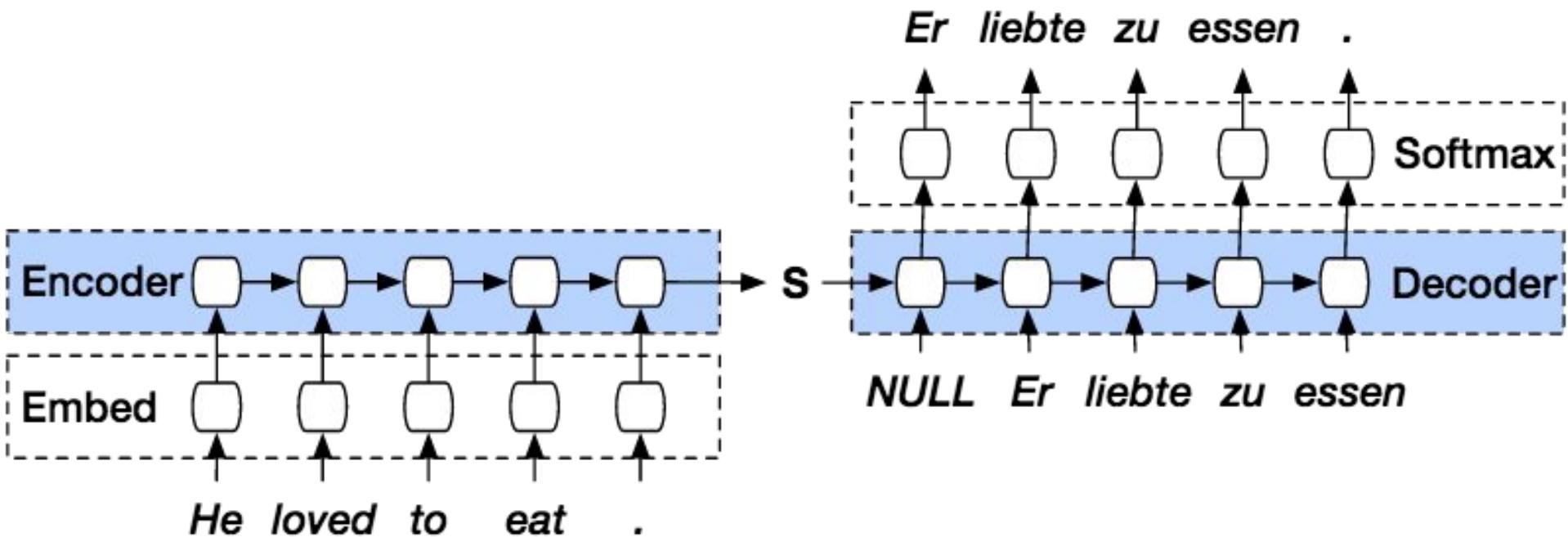
Machine translation using encoder-decoder

- Use another LSTM as the decoder
- Input to the LSTM is the encoded sentence and the previously output word



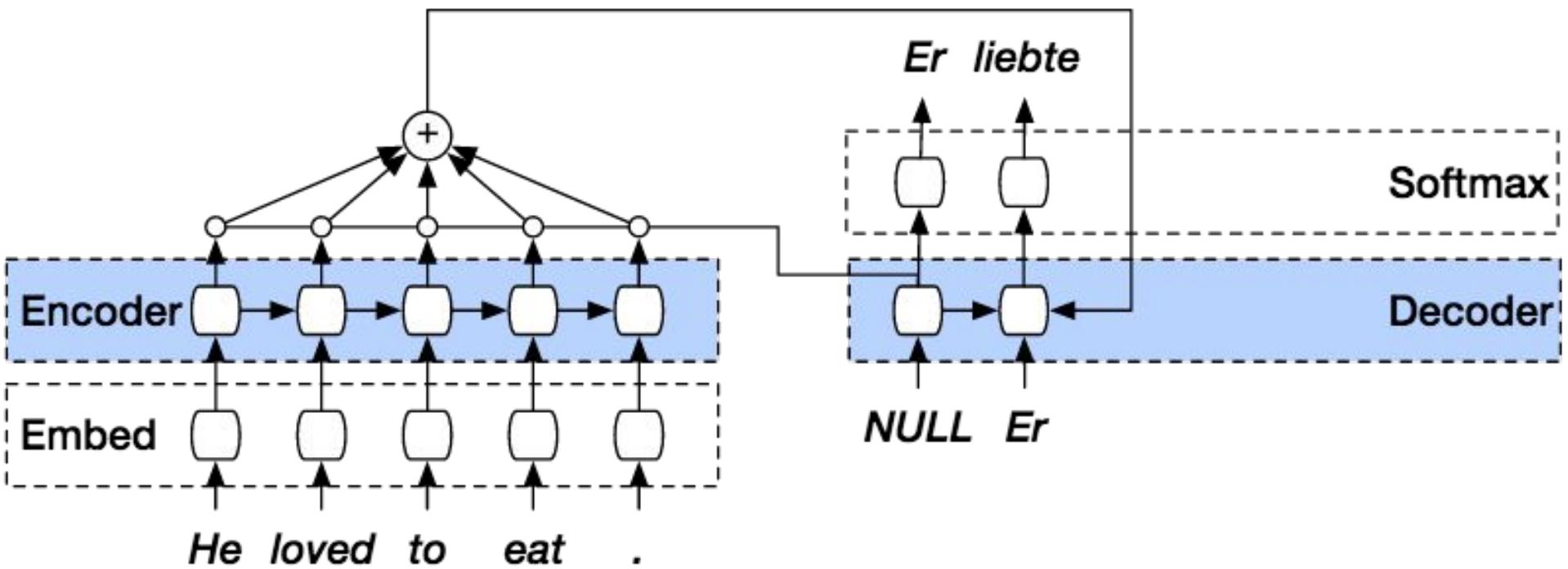
Problems with LSTM/GRU

- Cannot handle long sequence well (limited fixed size memory S)
 - Also vanishing gradients

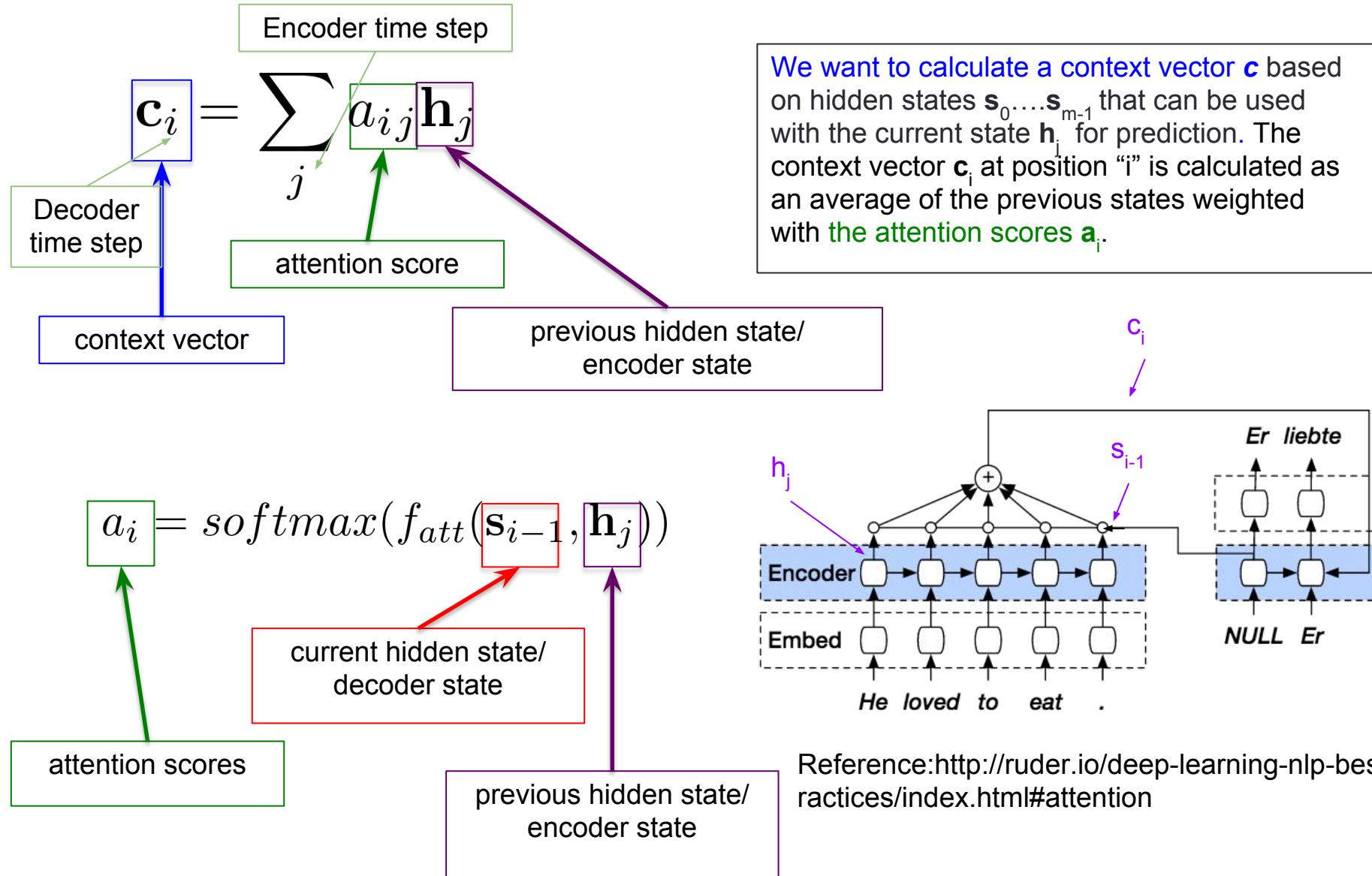


Attention Mechanism

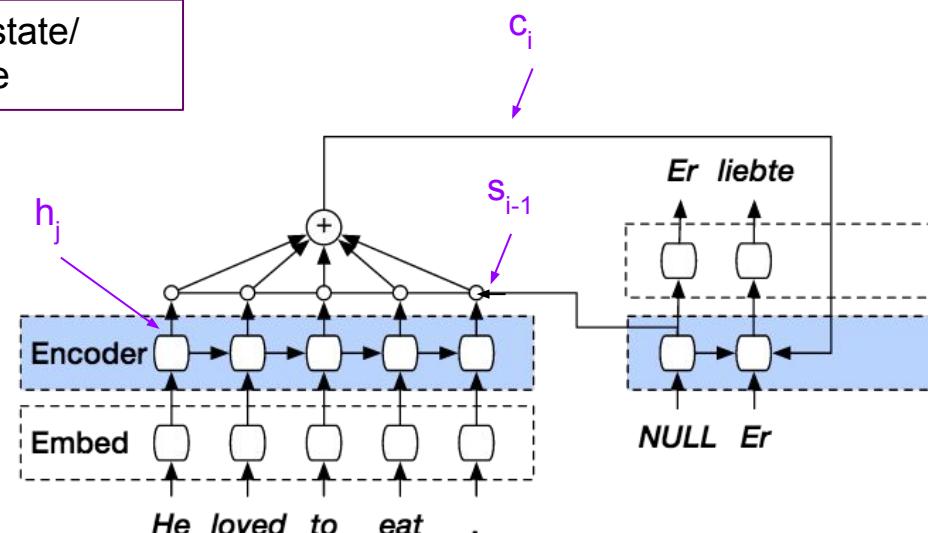
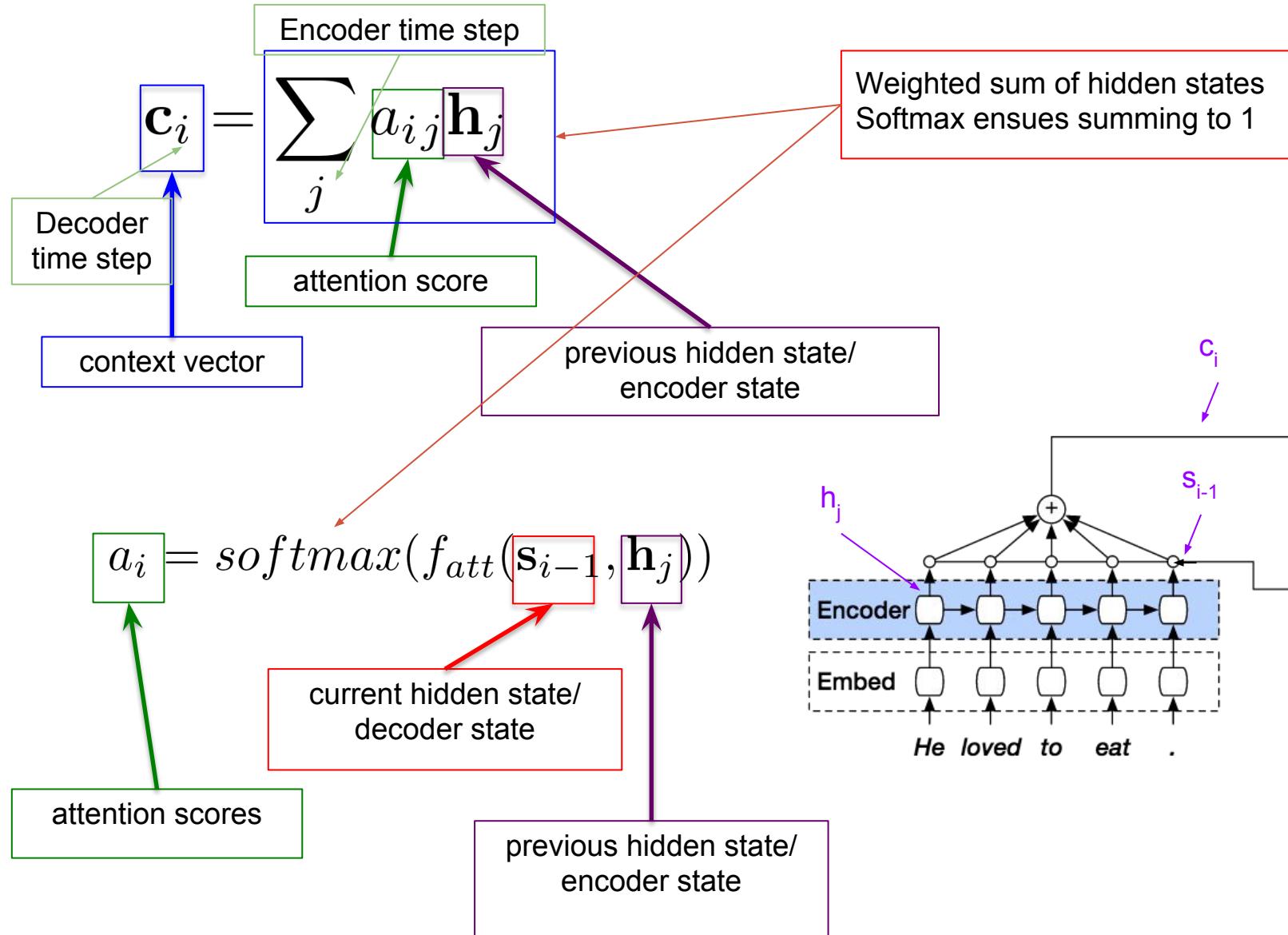
Attention allows the decoder part of the network to focus/attend on a different part of the encoder's outputs for every step of the decoder's own outputs.



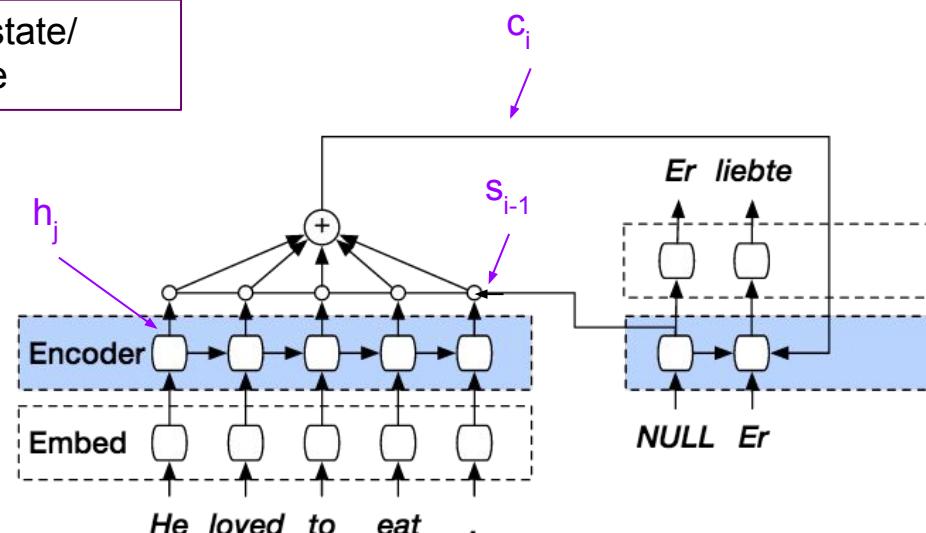
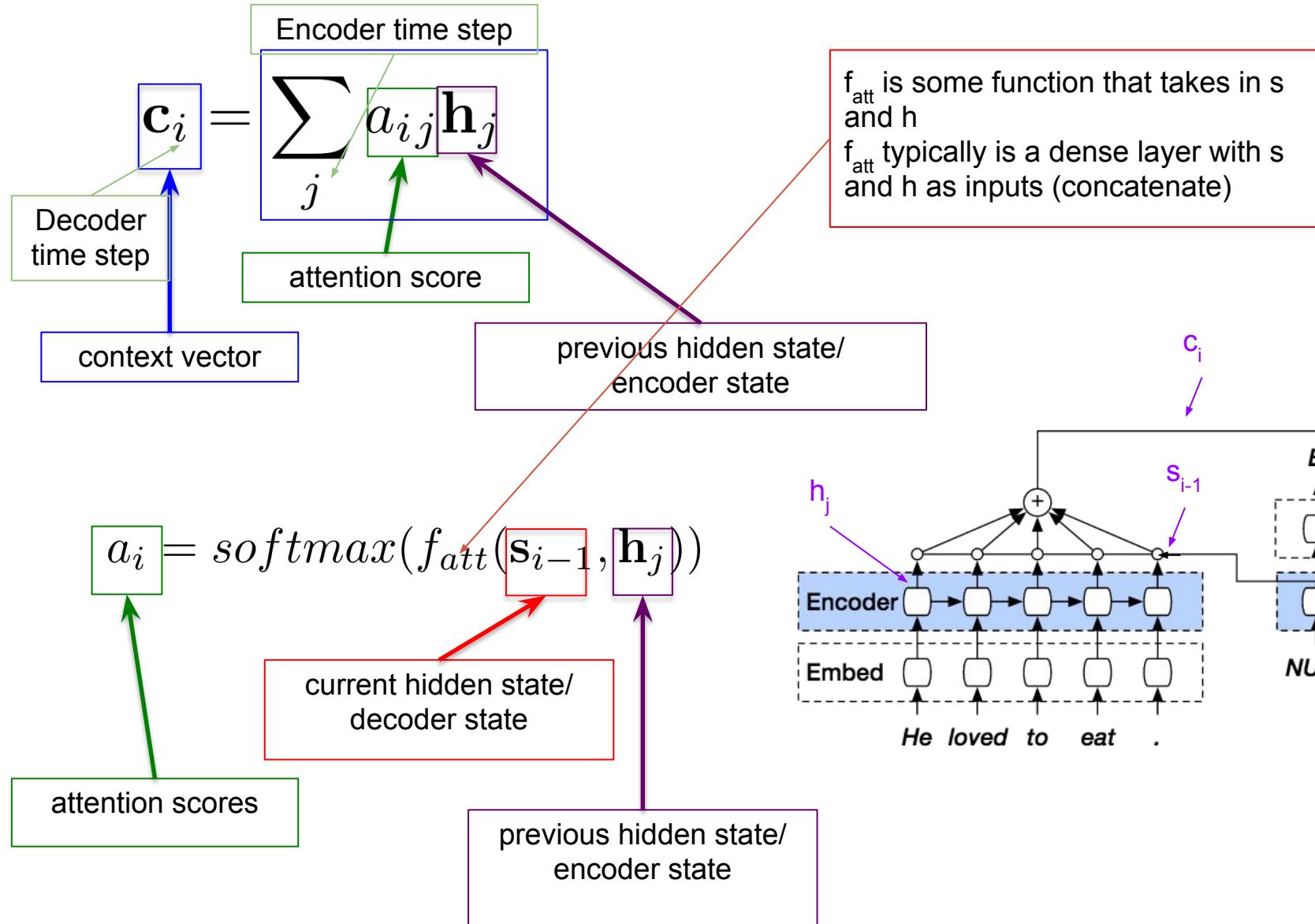
Attention Mechanism



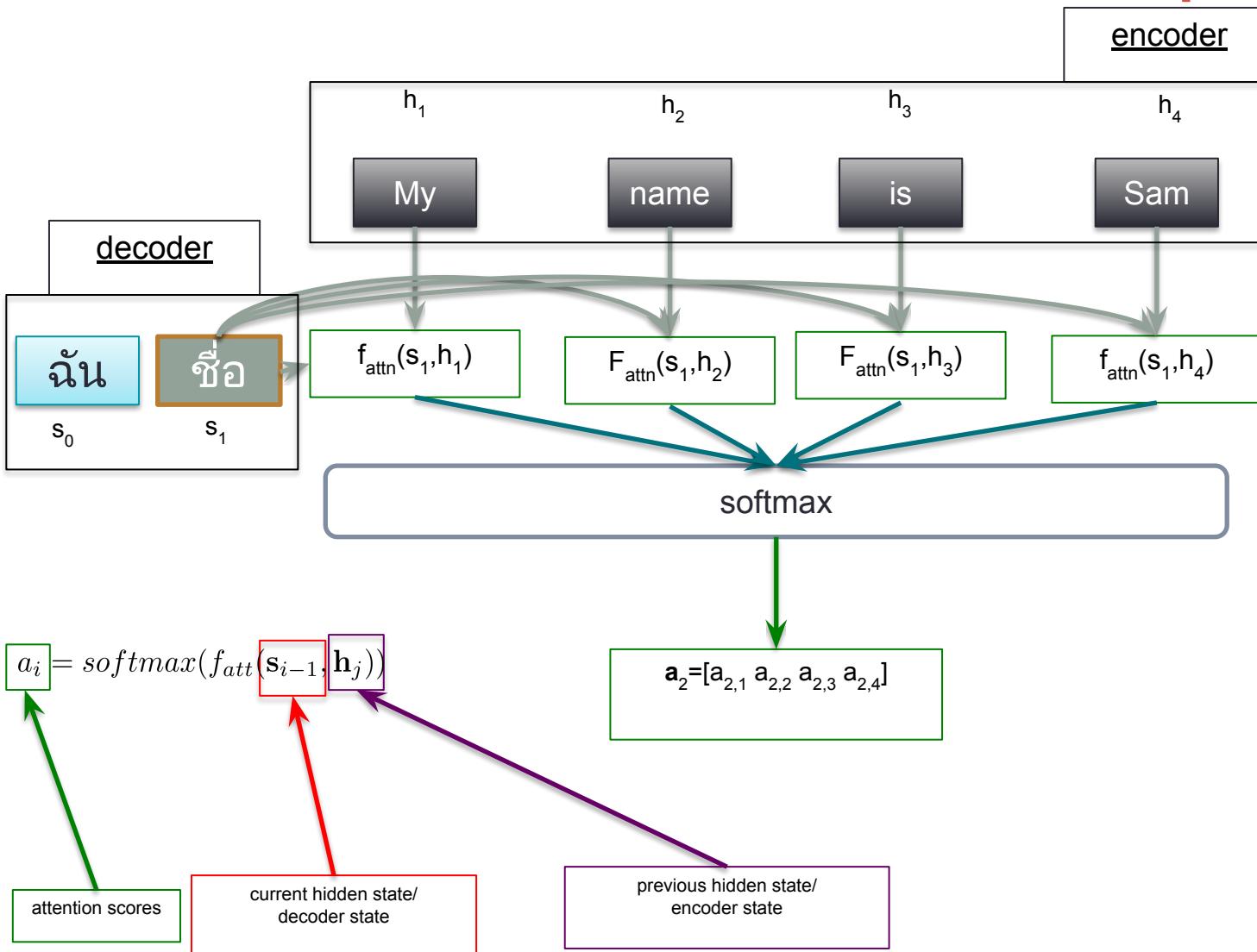
Attention Mechanism



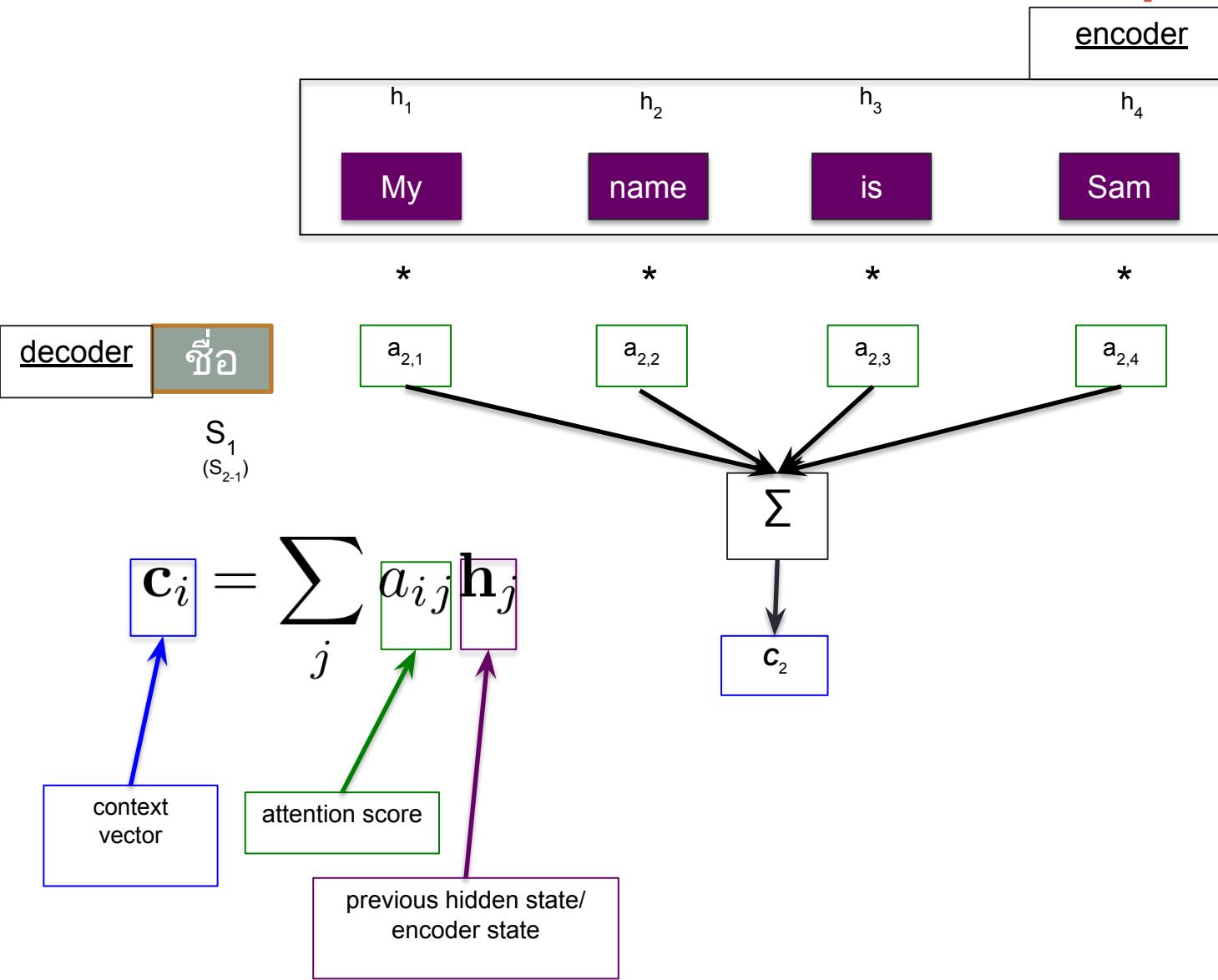
Attention Mechanism



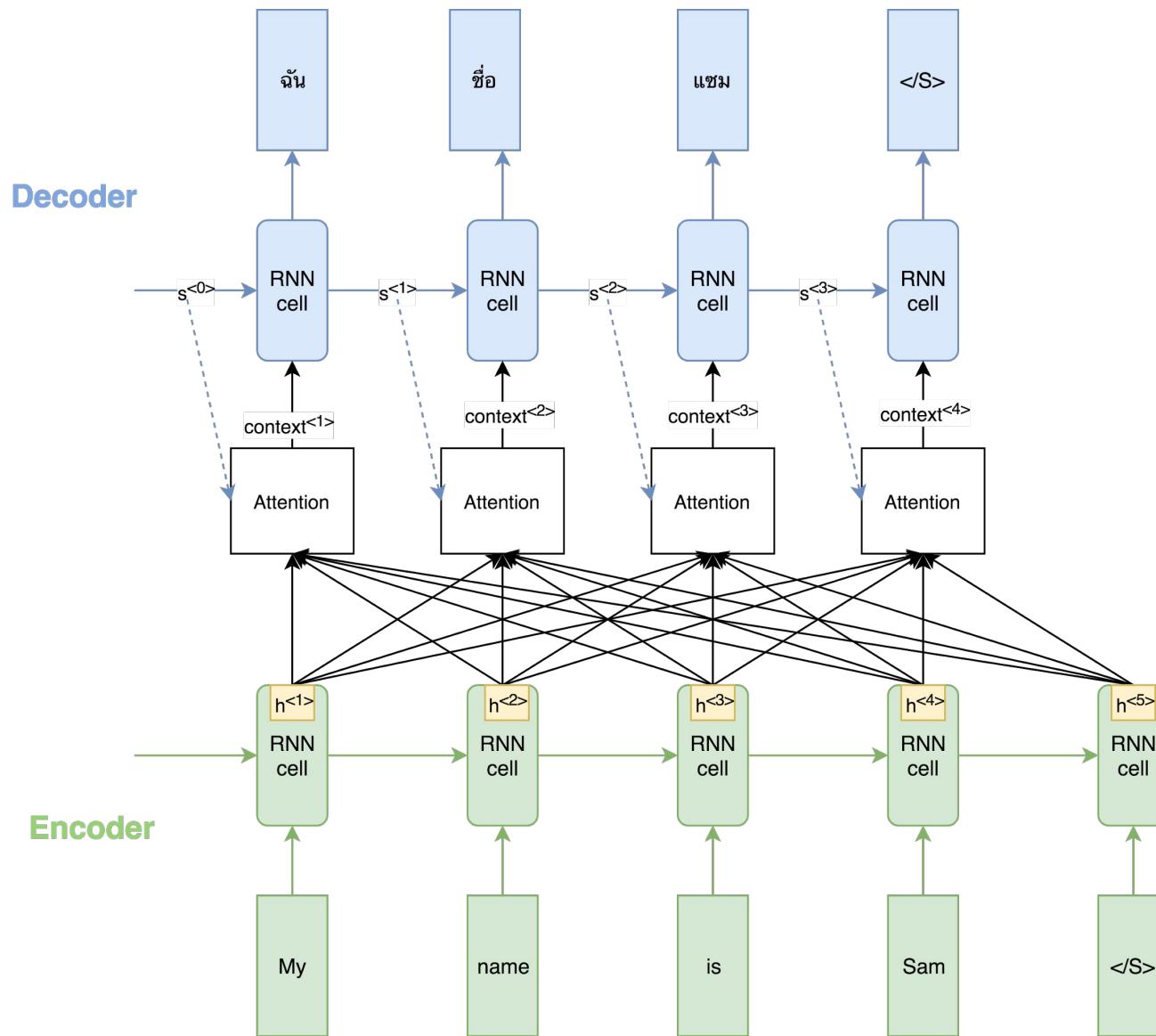
Attention Calculation Example



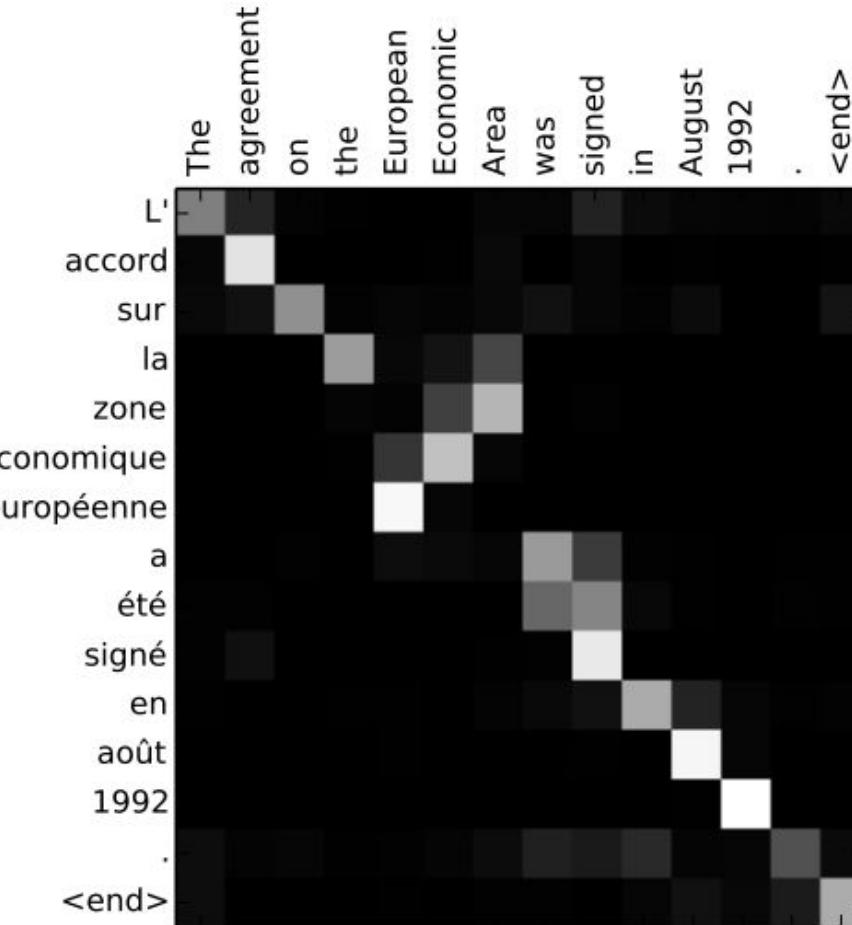
Attention Calculation Example



RNN and attention mechanism



Graphical Example: English-to-French machine translation



Reference: Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." ICLR(2015).

Type of Attention mechanisms

There are many variants of attention function f_{attn}

Additive attention: The original attention mechanism (Bahdanau et al., 2015) uses a one-hidden layer feed-forward network to calculate the attention alignment:

$$f_{\text{attn}}(\mathbf{s}_{i-1}, \mathbf{h}_j) = \tanh(\mathbf{W}_a[\mathbf{s}_{i-1}; \mathbf{h}_j])$$

Multiplicative attention: Multiplicative attention (Luong et al., 2015) simplifies the attention operation by calculating the following function:

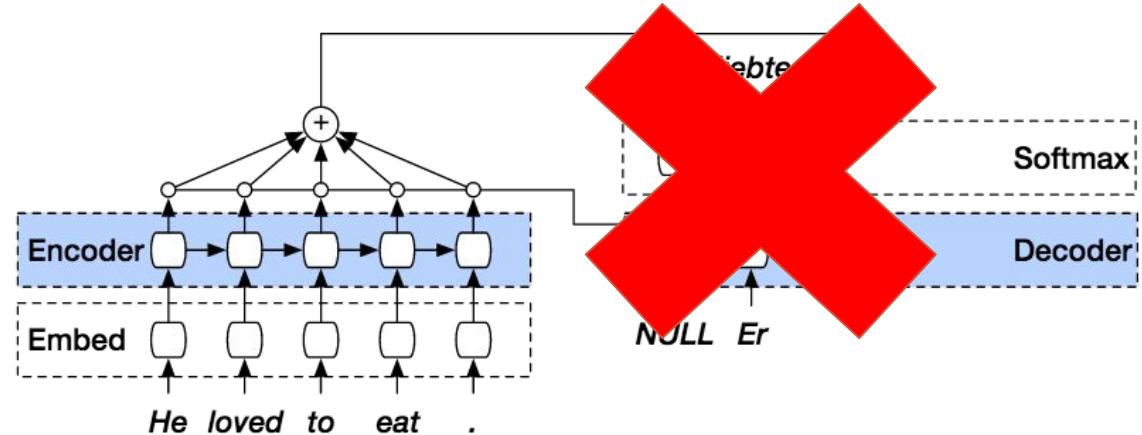
$$f_{\text{attn}}(\mathbf{s}_{i-1}, \mathbf{h}_j) = \mathbf{s}_{i-1}^\top \mathbf{W}_a \mathbf{h}_j$$

Self-attention: Without any additional information, however, we can still extract relevant aspects from the sentence by allowing it to attend to itself using self-attention (Lin et al., 2017)

$$\mathbf{a} = \text{softmax}(\mathbf{w}_{s_2} \tanh(\mathbf{W}_{s_1} \boxed{\mathbf{H}^T}))$$

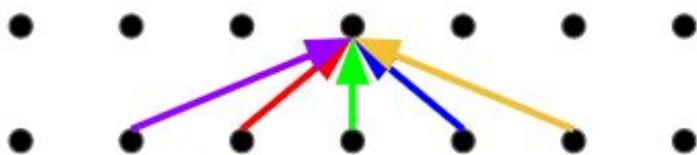
No \mathbf{s} from encoder
(Encoder side can also self-attend)

Self-attention



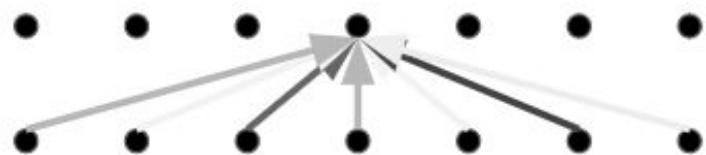
No need for additional information in order to select where to attend

Convolution



Similar to CNN in flow

Self-Attention



Self-attention example

- if I can give this restaurant a 0 I will we be just ask our waitress leave because someone with a reservation be wait for our table my father and father-in-law be still finish up their coffee and we have not yet finish our dessert I have never be so humiliad do not go to this restaurant their food be mediocre at best if you want excellent Italian in a small intimate restaurant go to dish on the South Side I will not be go back
- this place suck the food be gross and taste like grease I will never go here again ever sure the entrance look cool and the waiter can be very nice but the food simply be gross taste like cheap 99cent food do not go here the food shot out of me quick then it go in
- everything be pre cook and dry its crazy most Filipino people be used to very cheap ingredient and they do not know quality the food be disgusting I have eat at least 20 different Filipino family home this not even mediocre
- seriously f *** this place disgust food and shitty service ambience be great if you like dine in a hot cellar engulf in stagnate air truly be over rate over price and they just under deliver forget try order a drink here it will take forever get and when it finally do arrive you will be ready pass out from heat exhaustion and lack of oxygen how be that a head change you do not even have pay for it I will not disgust you with the detailed review of everything I have try here but make it simple it all suck and after you get the bill you will be walk out with a sore ass save your money and spare your self the disappointment
- i be so angry about my horrible experience at Medusa today my previous visit be amaze 5/5 however my go to out of town and I land an appointment with Stephanie I go in with a picture of roughly what I want and come out look absolutely nothing like it my hair be a horrible ashy blonde not anywhere close to the platinum blonde I request she will not do any of the pop of colour I want and even after specifically tell her I do not like blunt cut my hair have lot of straight edge she do not listen to a single thing I want and when I tell her I be unhappy with the colour she basically tell me I be wrong and I have do it this way no no I do not if I can go from Little Mermaid red to golden blonde in 1 sitting that leave my hair fine I shall be able go from golden blonde to a shade of platinum blonde in 1 sitting thanks for ruin my New Year's with 1 the bad hair job I have ever have

(a) 1 star reviews

- really enjoy Ashley and Ami salon she do a great job be friendly and professional I usually get my hair do when I go to MI because of the quality of the highlight and the price the price be very affordable the highlight fantastic thank Ashley i highly recommend you and ill be back
- love this place it really be my favorite restaurant in Charlotte they use charcoal for their grill and you can taste it steak with chimichurri be always perfect Fried yucca cilantro rice pork sandwich and the good tres lech I have had.The desert be all incredible if you do not like it you be a mutant if you will like diabeetus try the Inca Cola
- this place be so much fun I have never go at night because it seem a little too busy for my taste but that just prove how great this restaurant be they have amazing food and the staff definitely remember us every time we be in town I love when a waitress or waiter come over and ask if you want the cab or the Pinot even when there be a rush and the staff be run around like crazy whenever I grab someone they instantly smile acknowledge us the food be also killer I love when everyone know the special and can tell you they have try them all and what they pair well with this be a first last stop whenever we be in Charlotte and I highly recommend them
- great food and good service what else can you ask for everything that I have ever try here have be great
- first off I hardly remember waiter name because its rare you have an unforgettable experience the day I go I be celebrate my birthday and let me say I leave feel extra special our waiter be the best ever Carlos and the staff as well I be with a party of 4 and we order the potato salad shrimp cocktail lobster amongst other thing and boy be the food great the lobster be the good lobster I have ever eat if you eat a dessert I will recommend the cheese cake that be also the good I have ever have it be expensive but so worth every penny I will definitely be back there go again for the second time in a week and it be even good this place be amazing

(b) 5 star reviews

Red means word is often attended to

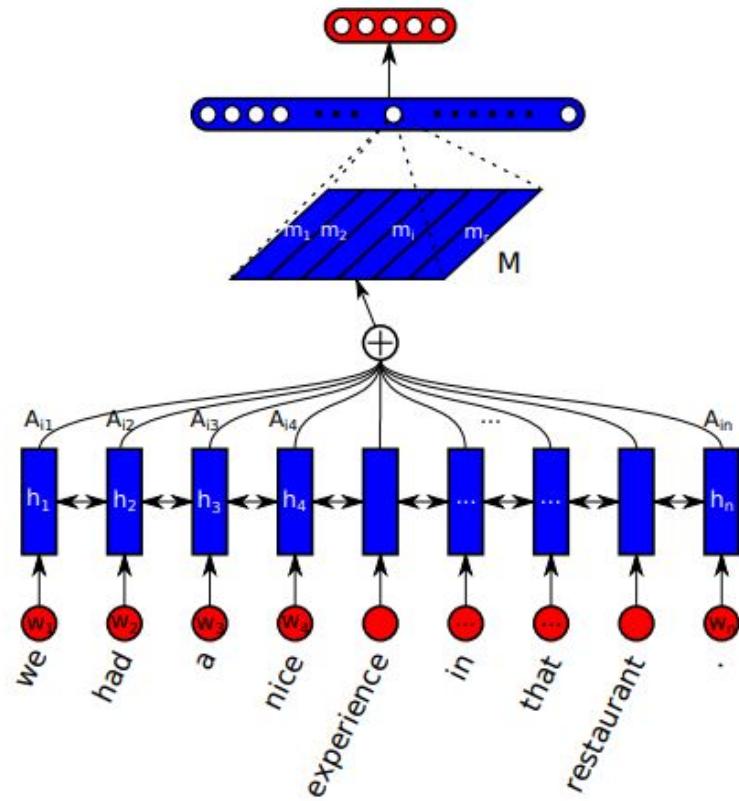


Figure 2: Heatmap of Yelp reviews with the two extreme score.

Type of Attention mechanisms

There are many variants of attention function f_{attn}

Additive attention: The original attention mechanism (Bahdanau et al., 2015) uses a one-hidden layer feed-forward network to calculate the attention alignment:

$$f_{\text{attn}}(\mathbf{s}_{i-1}, \mathbf{h}_j) = \tanh(\mathbf{W}_a[\mathbf{s}_{i-1}; \mathbf{h}_j])$$

Multiplicative attention: Multiplicative attention (Luong et al., 2015) simplifies the attention operation by calculating the following function:

$$f_{\text{attn}}(\mathbf{s}_{i-1}, \mathbf{h}_j) = \mathbf{s}_{i-1}^\top \mathbf{W}_a \mathbf{h}_j$$

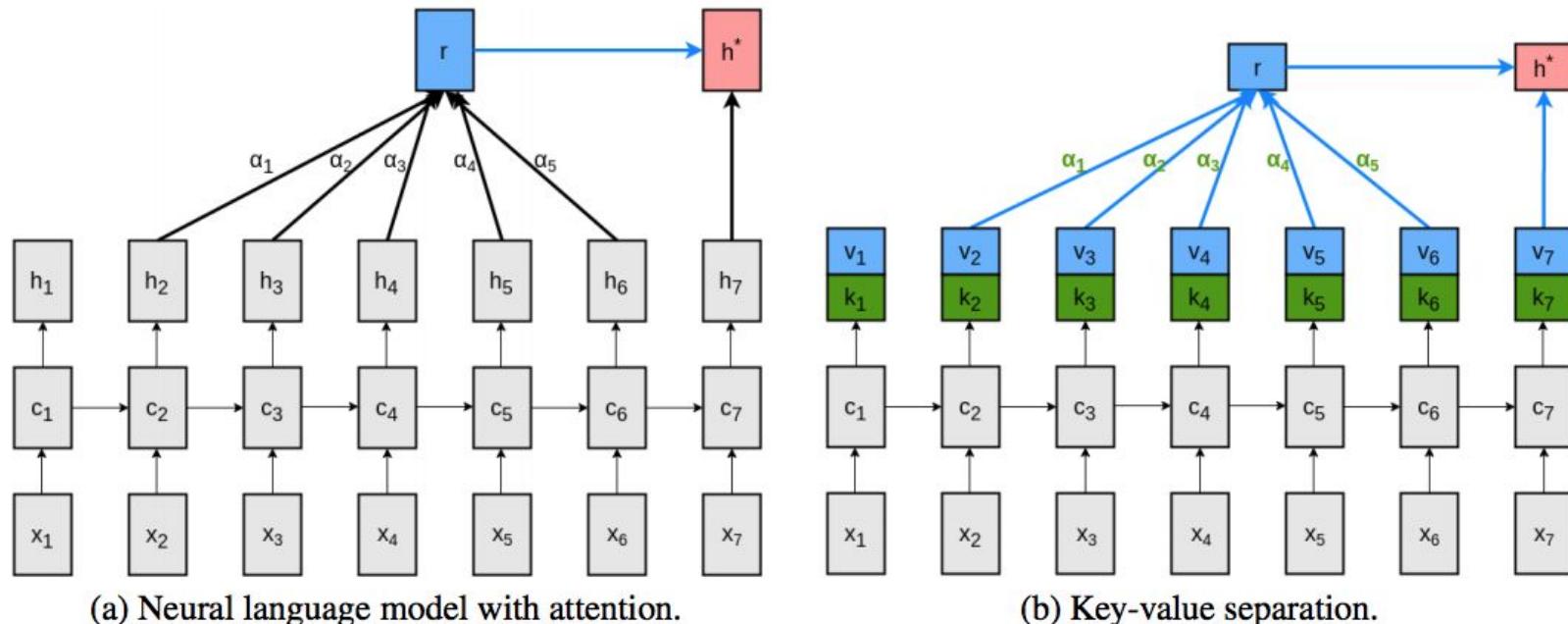
Self-attention: Without any additional information, however, we can still extract relevant aspects from the sentence by allowing it to attend to itself using self-attention (Lin et al., 2017)

$$\mathbf{a} = \text{softmax}(\mathbf{w}_{s_2} \tanh(\mathbf{W}_{s_1} \boxed{\mathbf{H}^T}))$$

No \mathbf{s} from encoder
(Encoder side can also self-attend)

Key-value attention: key-value attention (Danyluk et al., 2017) is a recent attention variant that separates form from function by keeping separate vectors for the attention calculation.

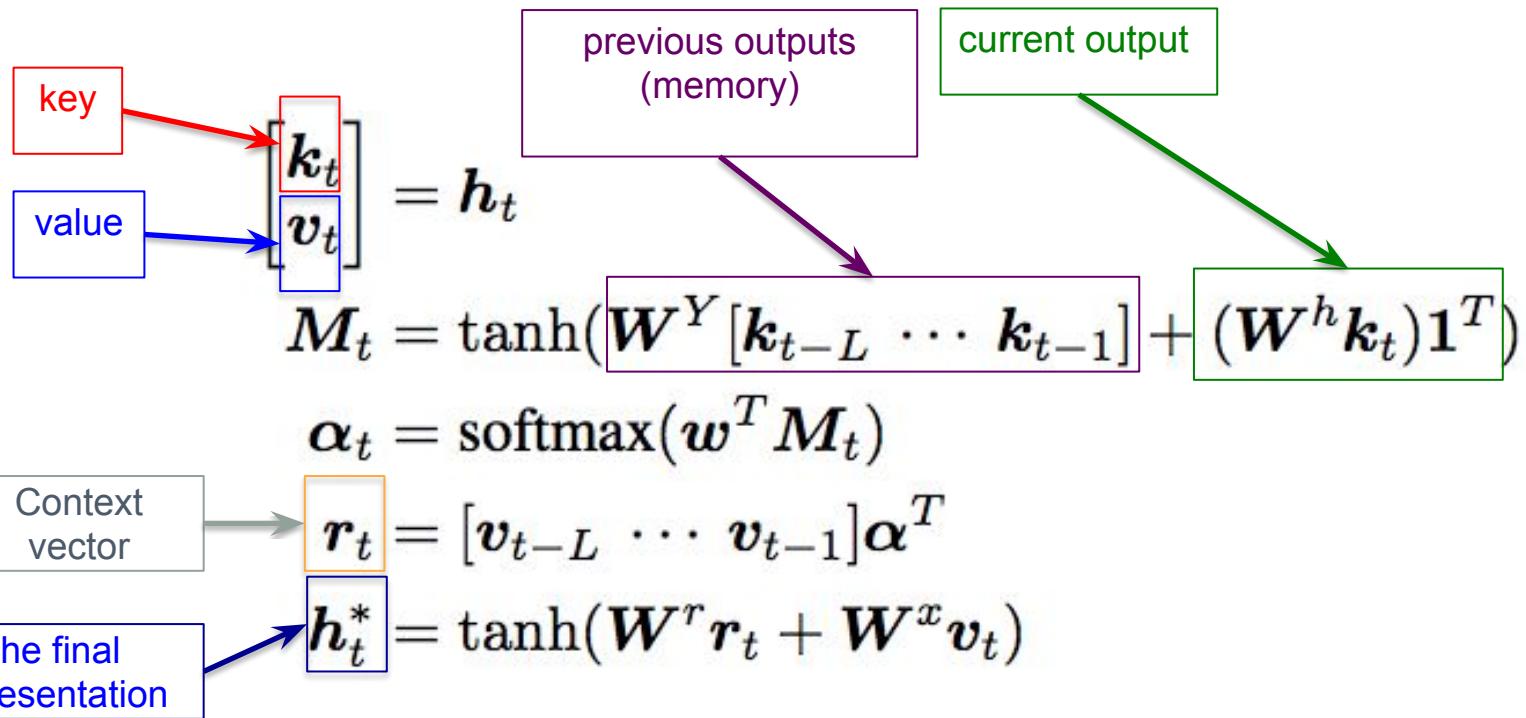
Key-value attention



K (keys) for finding attention weights
V (value) for using as features to sum

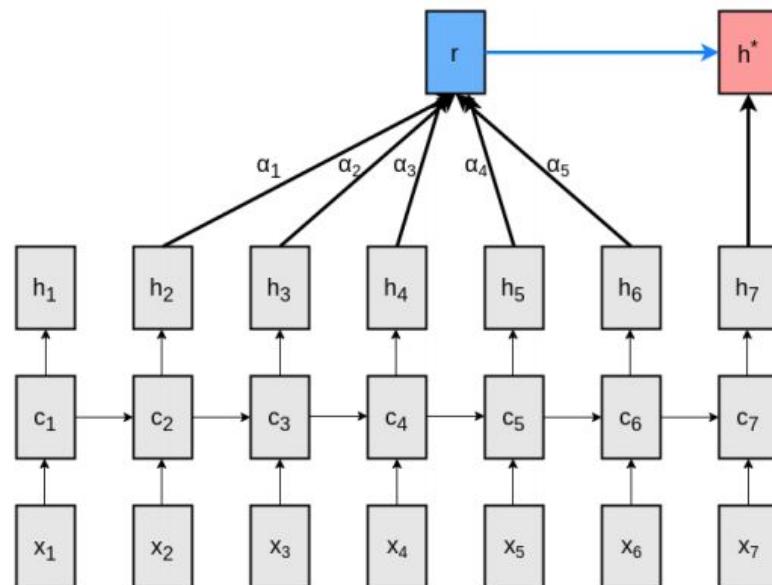
Reference: Daniluk, M., Rockt, T., Welbl, J., & Riedel, S. (2017). Frustratingly Short Attention Spans in Neural Language Modeling. In ICLR 2017.

Key-value attention

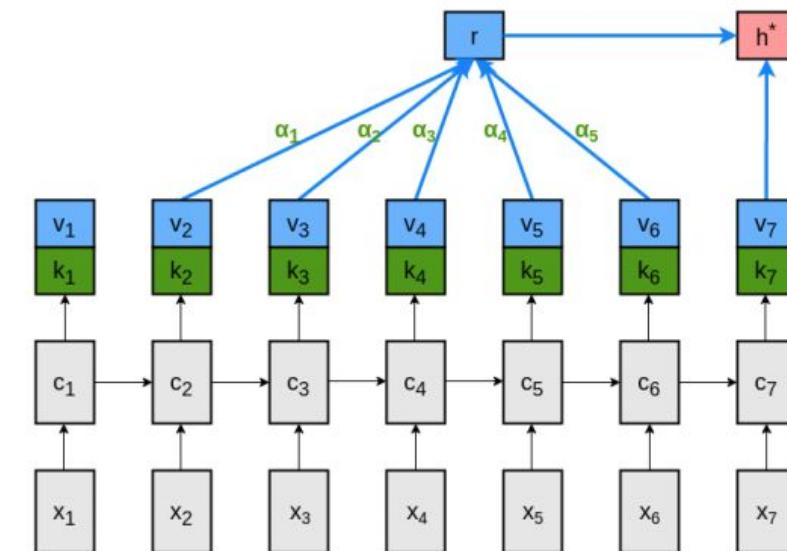


Problems with Recurrent networks

Slow. Feed forward needs to wait for output from previous output

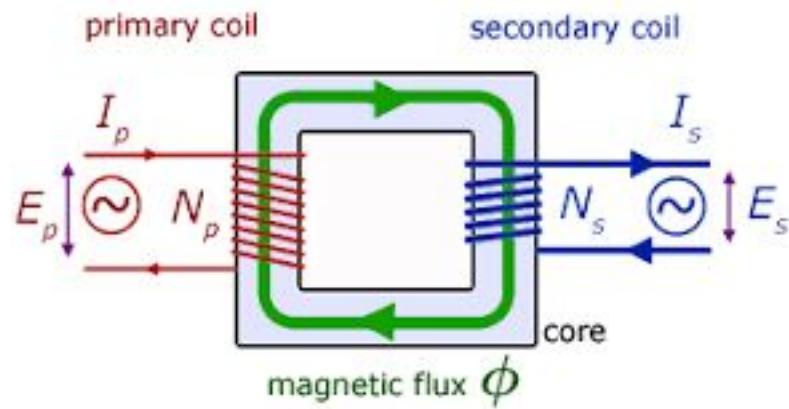


(a) Neural language model with attention.



(b) Key-value separation.

Transformer?



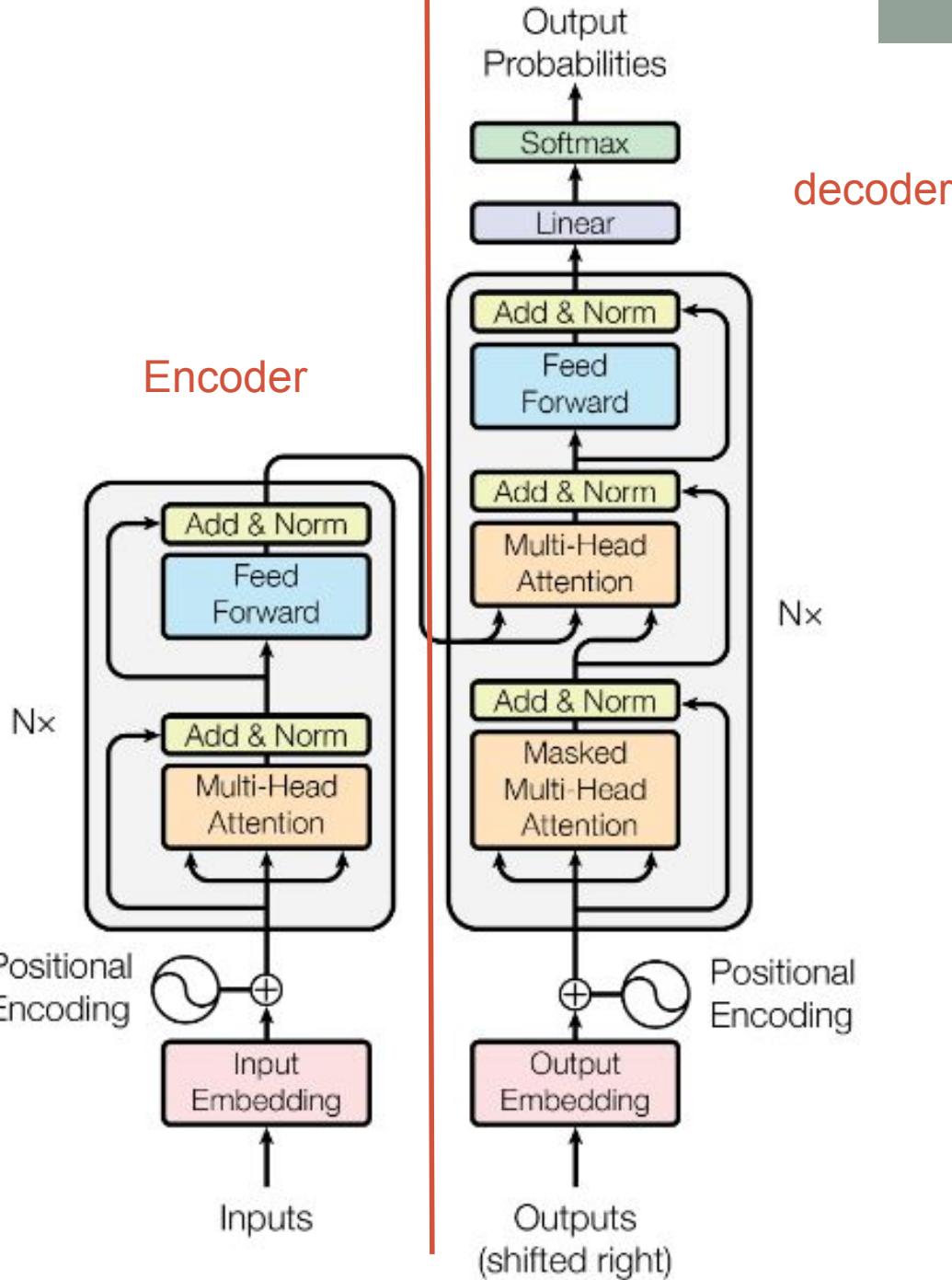
Attention is all you need

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

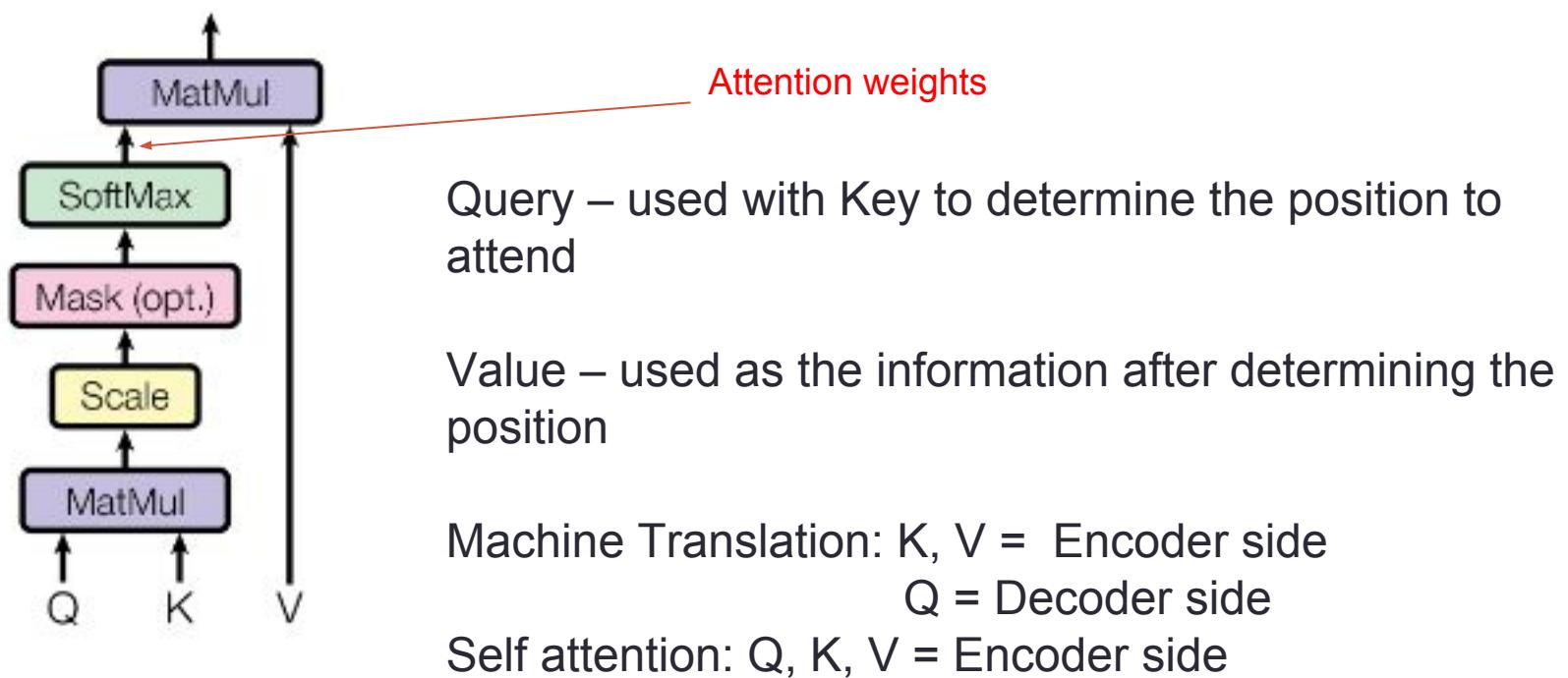
To eliminate GRU which remembers time, encode position instead

Done by adding some fixed number for each location of the word in the sentence



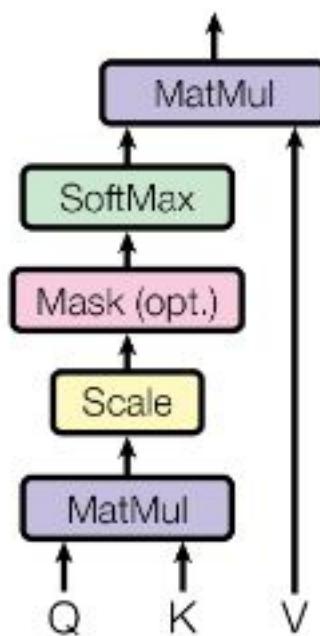
Key Value Query attention

Scaled Dot-Product Attention

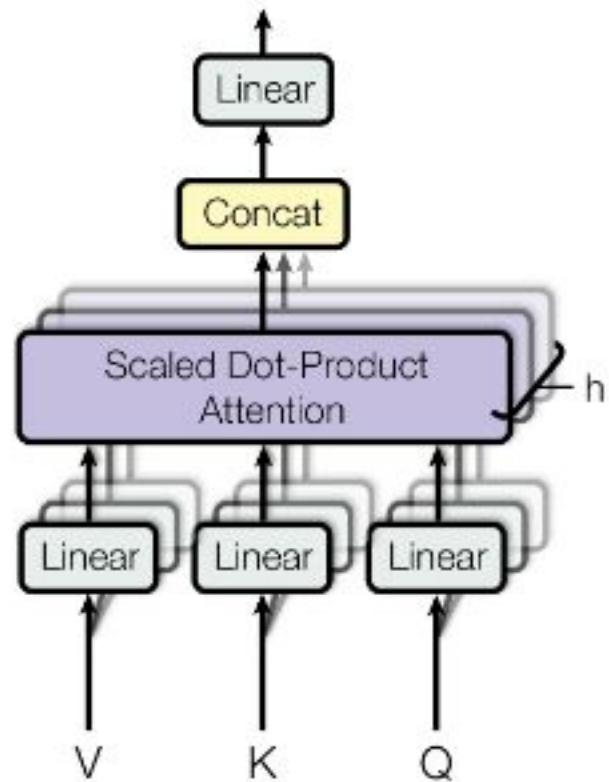


Multi-head attention

Scaled Dot-Product Attention



Multi-Head Attention



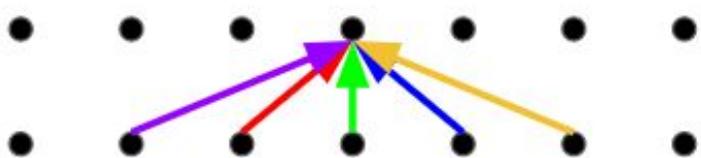
Query – used with Key to determine the position

Value – used as the information after determining the position

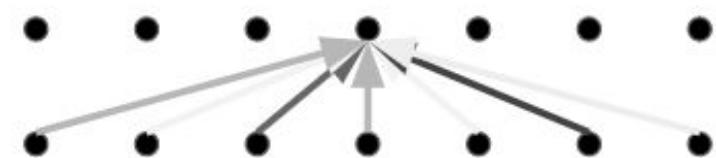
Attention drawback

- Convolution: weights * input. Each weights are different.
So position is encoded.
- Self-attention: a weighted average. Position information is lost at the output

Convolution



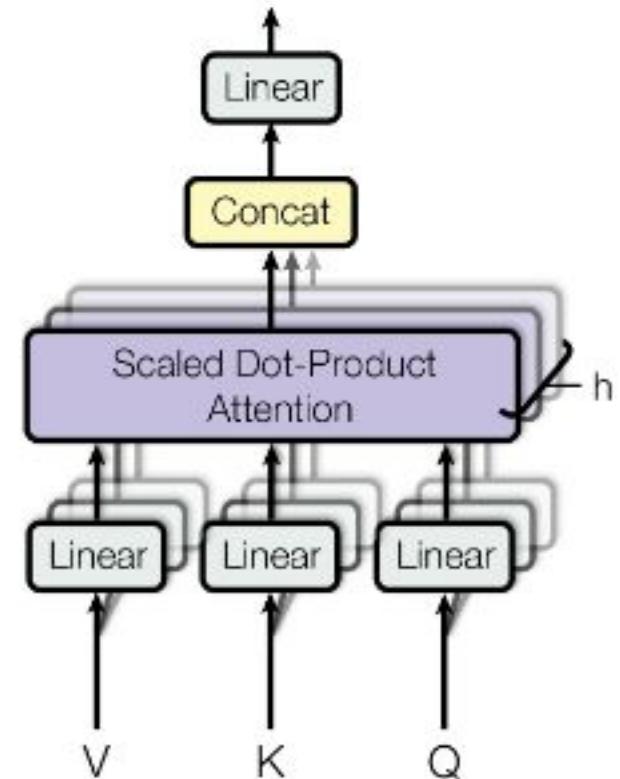
Self-Attention



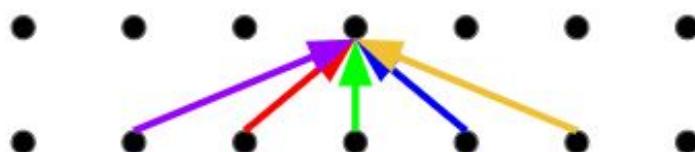
Multi-head attention

- Multiple attention layers (heads) that run in parallel
- Each head use different weights
- Each head can learn different relationship

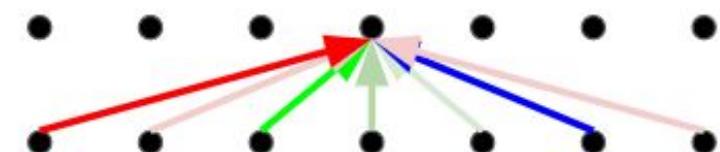
Multi-Head Attention



Convolution



Multi-Head Attention

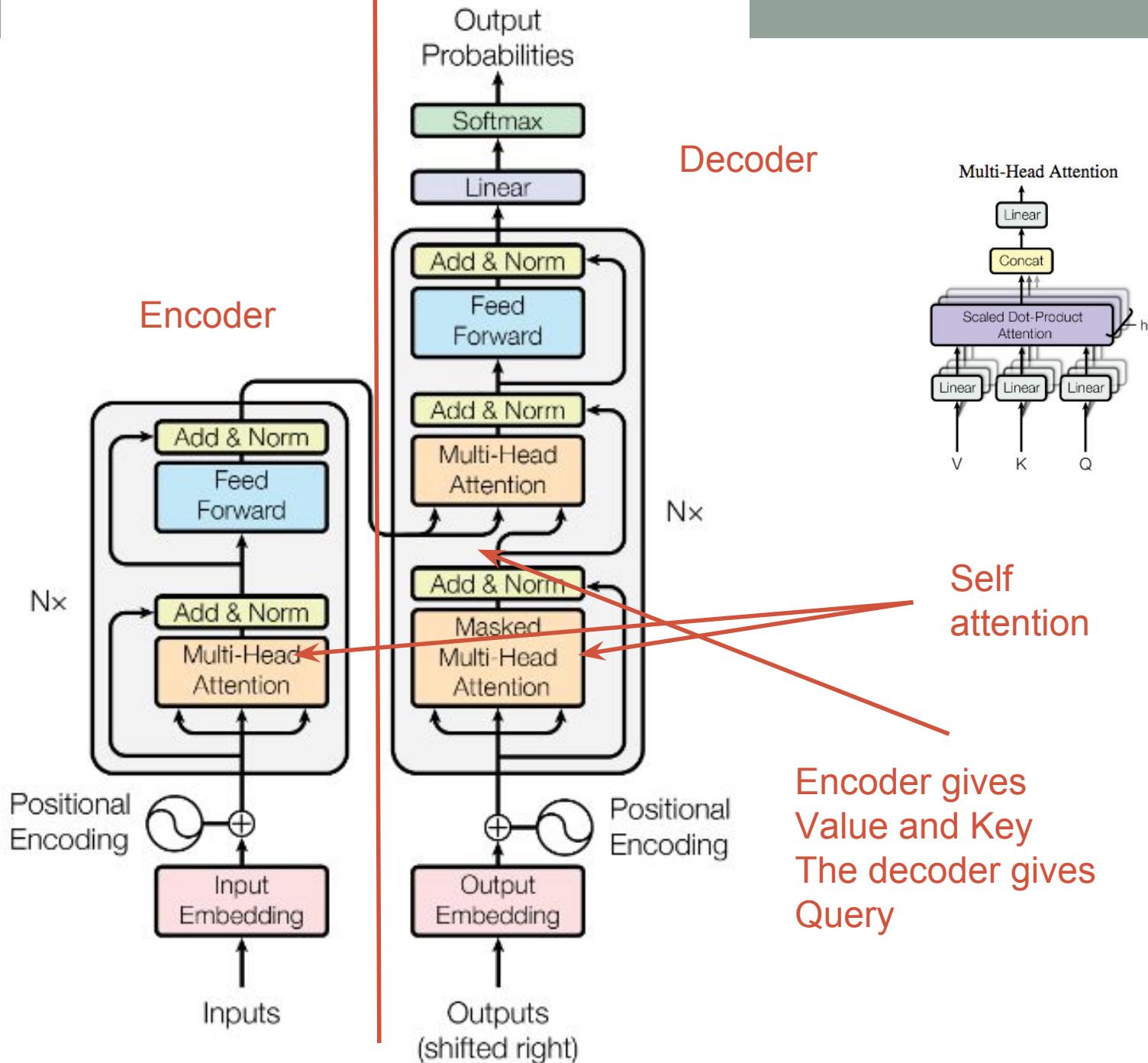


Multi-head visualization

It is in this spirit that a majority of American governments have passed new laws since 2009 making the registration or voting process more difficult .

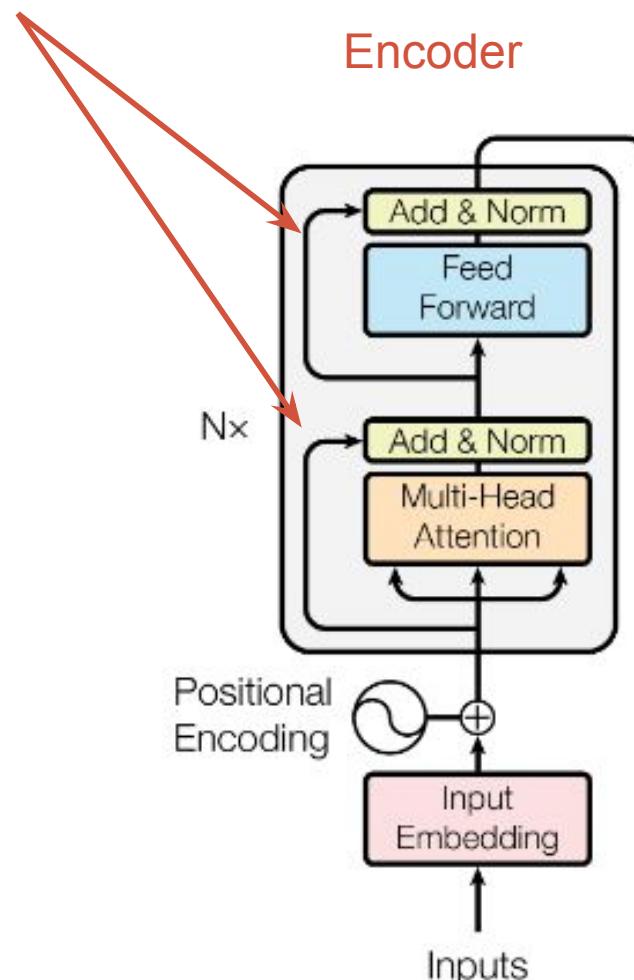
<EOS>

The diagram illustrates a sequence of words from a sentence being processed by multiple neural network heads. The words are arranged in a timeline from left to right. A pink head processes 'making' and 'the'. A brown head processes 'registration' and 'or'. A light purple head processes 'voting' and 'process'. A blue head processes 'more' and 'difficult'. A grey head processes the final punctuation. A heatmap at the bottom shows the activation levels of these heads across the sequence.



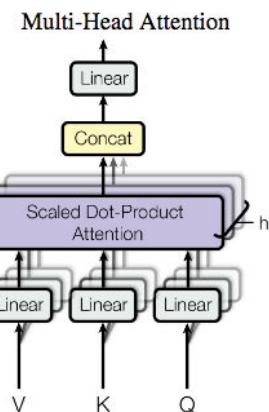
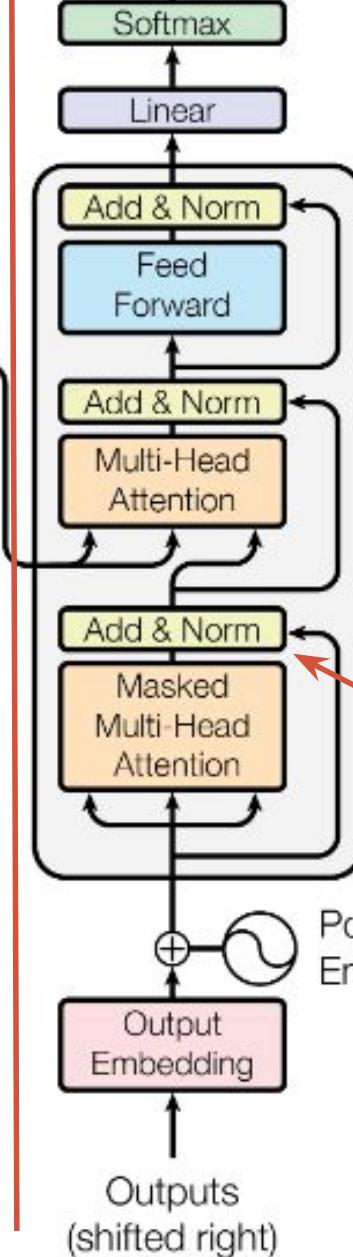
Residual connection

Encoder



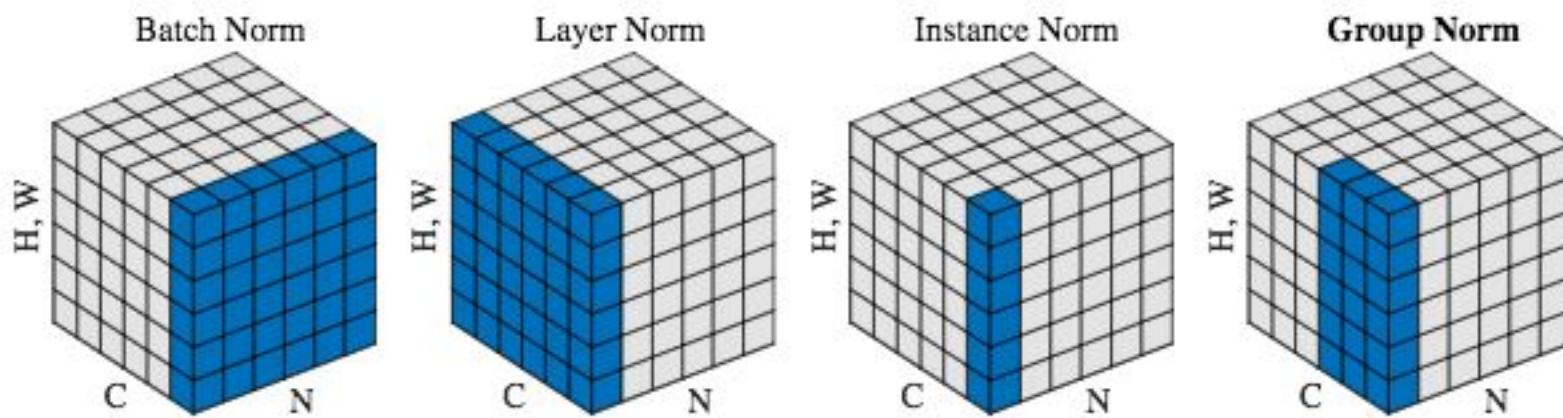
Output
Probabilities

Decoder



Layer norm

- Normalize the mean and SD
- Batch norm vs layer norm vs Instance norm vs group norm
Group is used to distributed models into multiple GPUs



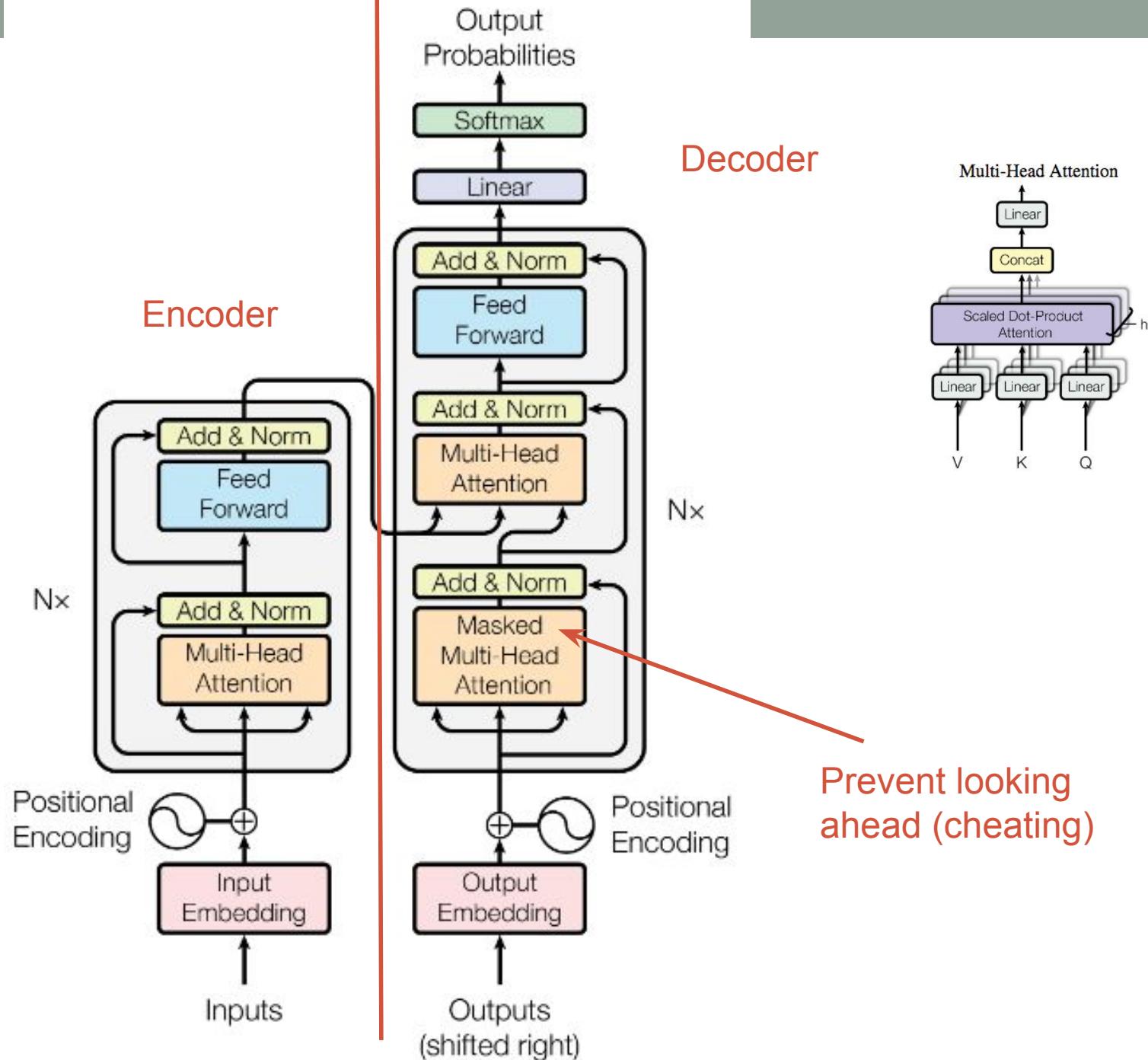
N – example in mini batch

C – Channel output

H,W – spatial coordinates (x,y)

Box is output tensor from CNN

BN and GN are usually best, GN is better when batch size is small (Vision task)



MT results

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

| Model | BLEU | | Training Cost (FLOPs) | |
|---------------------------------|-------------|--------------|-----------------------|---------------------|
| | EN-DE | EN-FR | EN-DE | EN-FR |
| ByteNet [18] | 23.75 | | | |
| Deep-Att + PosUnk [39] | | 39.2 | | $1.0 \cdot 10^{20}$ |
| GNMT + RL [38] | 24.6 | 39.92 | $2.3 \cdot 10^{19}$ | $1.4 \cdot 10^{20}$ |
| ConvS2S [9] | 25.16 | 40.46 | $9.6 \cdot 10^{18}$ | $1.5 \cdot 10^{20}$ |
| MoE [32] | 26.03 | 40.56 | $2.0 \cdot 10^{19}$ | $1.2 \cdot 10^{20}$ |
| Deep-Att + PosUnk Ensemble [39] | | 40.4 | | $8.0 \cdot 10^{20}$ |
| GNMT + RL Ensemble [38] | 26.30 | 41.16 | $1.8 \cdot 10^{20}$ | $1.1 \cdot 10^{21}$ |
| ConvS2S Ensemble [9] | 26.36 | 41.29 | $7.7 \cdot 10^{19}$ | $1.2 \cdot 10^{21}$ |
| Transformer (base model) | 27.3 | 38.1 | | $3.3 \cdot 10^{18}$ |
| Transformer (big) | 28.4 | 41.8 | | $2.3 \cdot 10^{19}$ |

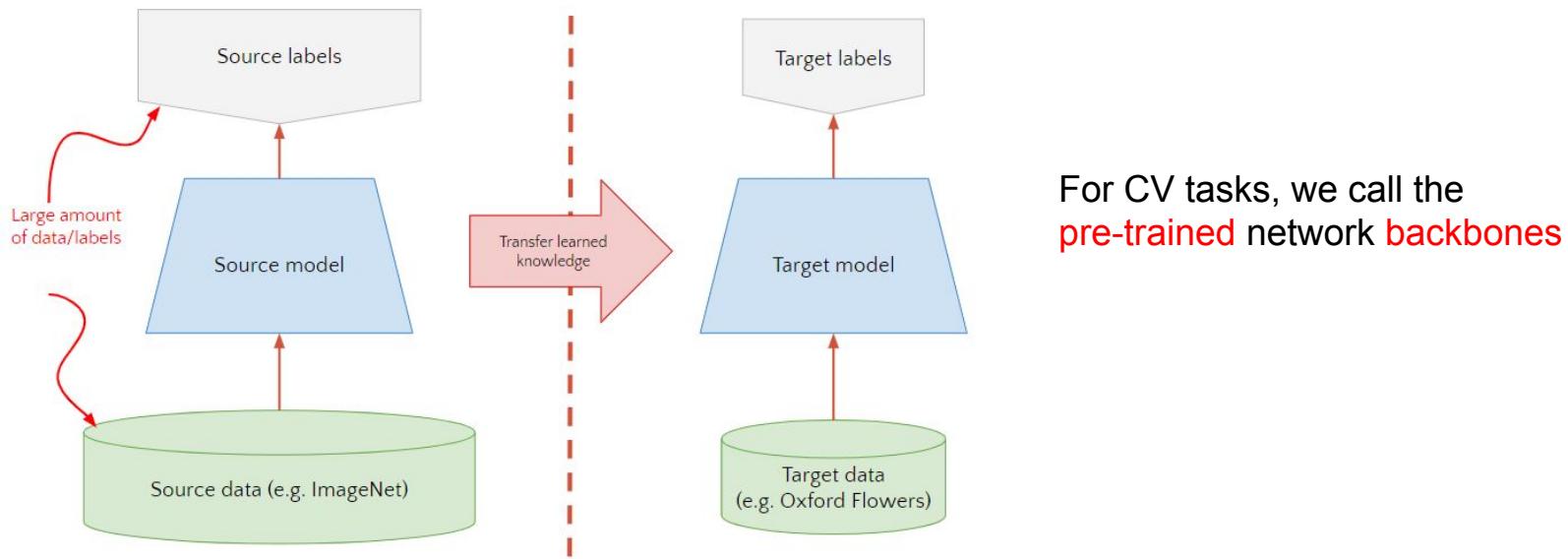
Can use for other tasks, like ASR, parsing, etc.

Misc topics

- Encoder-Decoder
- Attention modeling
- Transfer Learning, Multi-task models, Multi-models
- Object detection
- Alternatives to Backprop
 - 2nd order methods
 - Synthetic gradients
 - Genetic Algo
- Reinforcement Learning <- Next lecture!
- GAN <- Next Next lecture!
 - Adversarial inputs

Transfer learning (basics)

- We know networks captures good representations
- Can we use it for other tasks?
- Use trained networks to initialize a new network for a different task.
- Re-train the network using SGD on new data.



Transfer learning idea

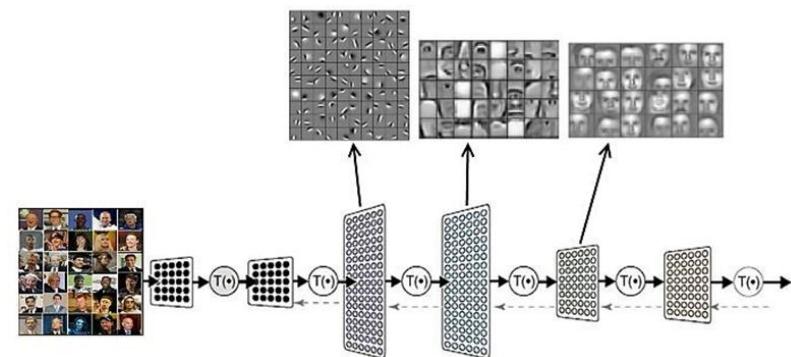
Instead of training a deep network from scratch for your task, you can

- Take a network trained on **a different domain** for a **different source task**
- **Adapt (fine-tune)** it for your domain and your **target task**

This lecture will talk about how to do this.

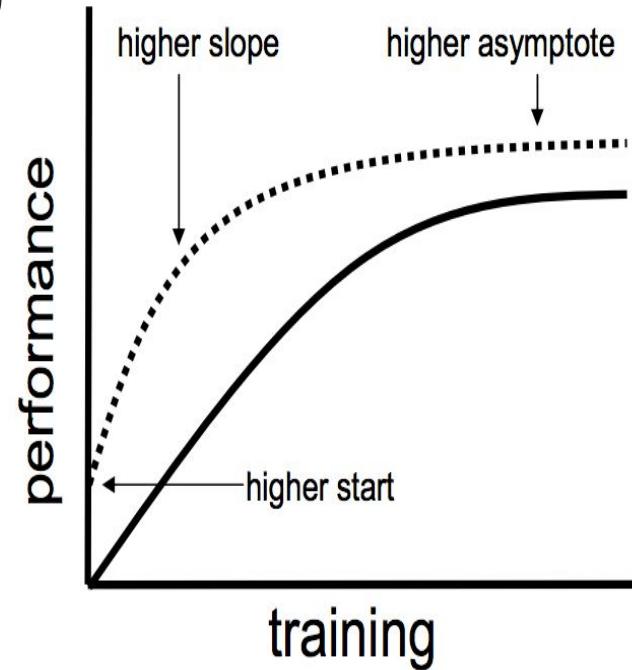
Variations:

- Different domain, same task
- Different domain, different task



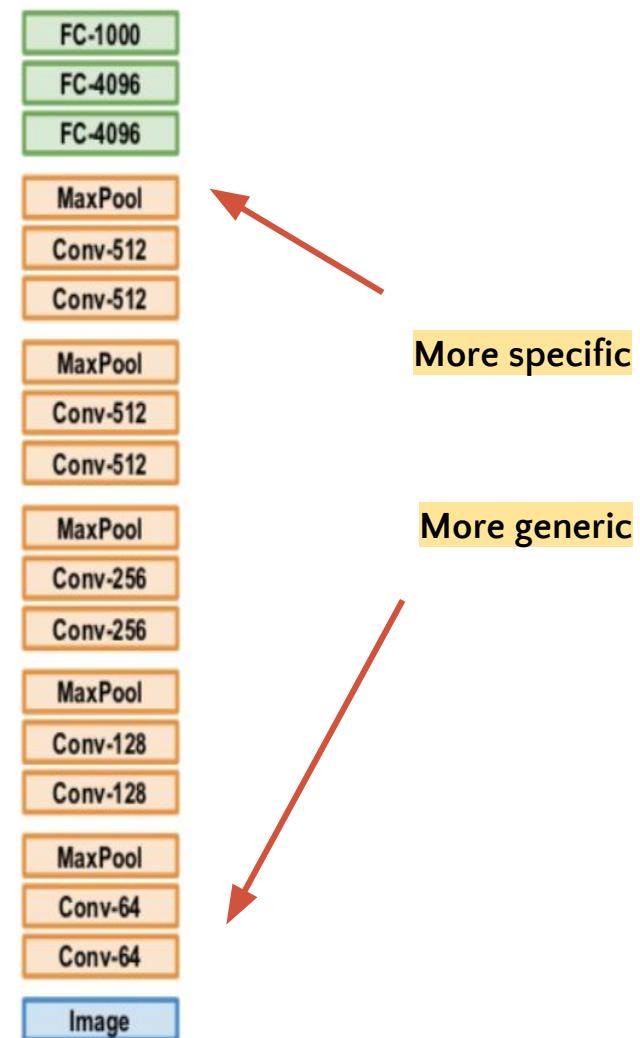
Benefits to transfer learning

1. Higher start
2. Higher slope (converge faster)
3. Higher asymptote (if small data)



Layers

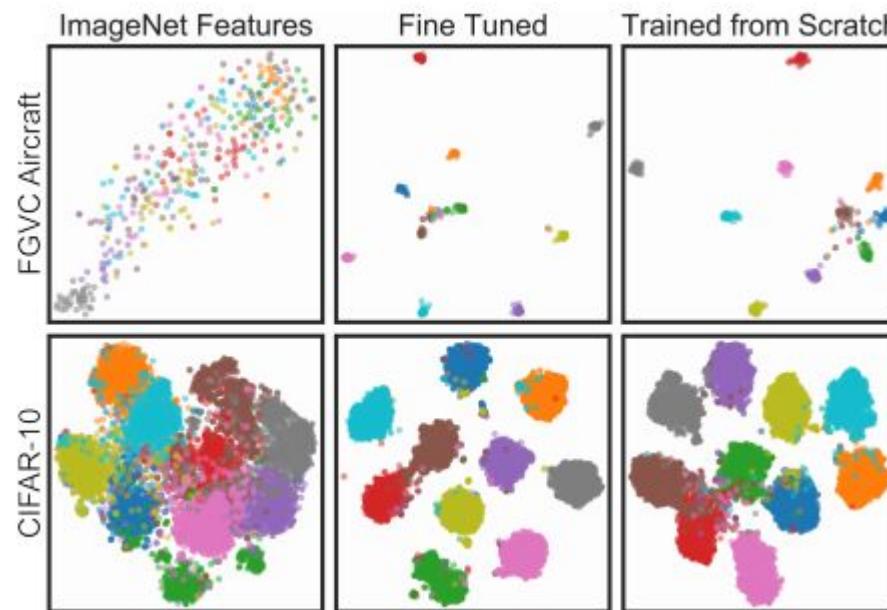
Lower layers: more general
Higher layers: more specific



Domains

Similar domain can transfer easier
Fine-tuning (adaptation) is crucial

Aircraft images
Different from ImageNet



Natural images
Similar to ImageNet

Transfer learning in ASR

Assamese and Bengali language



Bengali



Assamese

| Bengali | WER |
|---|-------|
| No pre-training (10 hr Bengali) | 71.8% |
| No pre-training (60 hr Bengali) | 64.5% |
| Transfer learning (10 hr Assamese -> 10 hr Bengali) | 66.0% |
| Transfer learning (60 hr Assamese -> 10 hr Bengali) | 64.6% |
| + Adaptation | 63.7% |
| Transfer learning (60 hr Bengali -> 10 hr Bengali) | 61.6% |

Closer domain is better

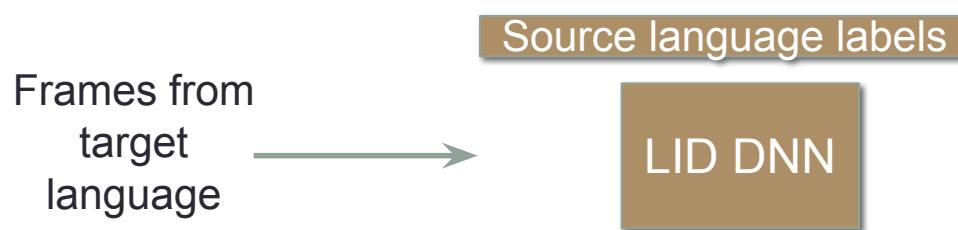
- Pre-train on 60 hours of data, and adapt on languages with 10 hours of data
- Numbers in **blue** is for 10->10 hours (baseline)

| | | Pretrain on | | | |
|----------|----------|-------------|----------|------|---------|
| | | Bengali | Assamese | Lao | Turkish |
| Adapt to | Bengali | 66.0 | 63.8 | 65.1 | 64.2 |
| | Assamese | 61.2 | 65.2 | 62.9 | 62.1 |
| | Lao | 59.8 | 60.1 | 62.3 | 60.0 |
| | Turkish | 61.8 | 63.1 | 63.3 | 63.9 |

- Transfer learning always improves performance.
- Similar languages perform better on transfer learning
- Can we identify this automatically?

Language Identification for data selection

- Train a classifier (LID) on source languages to predict the language given input frames
- Compute posteriors of the target language data using that classifier
- The best language for the target language should have the highest average posterior score



Predicting the best language

| | | Pretrain on | | | |
|--------------|----------|---------------------------|----------|------|---------|
| | | Bengali | Assamese | Lao | Turkish |
| Adapt to | Bengali | 66.0 | 63.8 | 65.1 | 64.2 |
| | Assamese | 61.2 | 65.2 | 62.9 | 62.1 |
| | Lao | 59.8 | 60.1 | 62.3 | 60.0 |
| | Turkish | 61.8 | 63.1 | 63.3 | 63.9 |
| | | Language prediction score | | | |
| | | Bengali | Assamese | Lao | Turkish |
| Input frames | Bengali | 0.57 | 0.21 | 0.09 | 0.13 |
| | Assamese | 0.21 | 0.57 | 0.11 | 0.11 |
| | Lao | 0.08 | 0.11 | 0.71 | 0.10 |
| | Turkish | 0.13 | 0.12 | 0.10 | 0.65 |

The LID scores correspond to the best language to use most of the time.

Which task to transfer?

TASKONOMY

Home API Live Demo Pretrained Task Bank Paper Dataset Team

Sample Images (click to use).



Image to upload:

Choose File No file chosen

Run inference

Input Image



Surface Normals

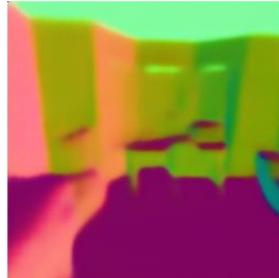
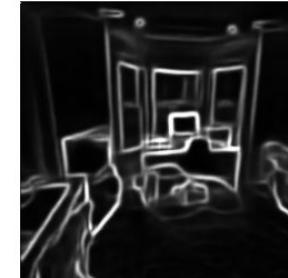


Image Reshading



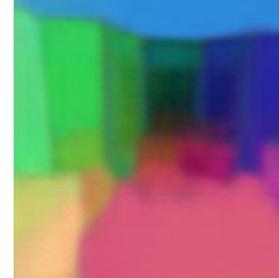
2D Texture Edges



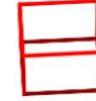
Vanishing Points



Unsupervised 2.5D Segm.



Room Layout



Scene Classification

Top 5 prediction:
home_office
office
television_room
computer_room
office_cubicles

3D Keypoints



3D Occlusion Edges



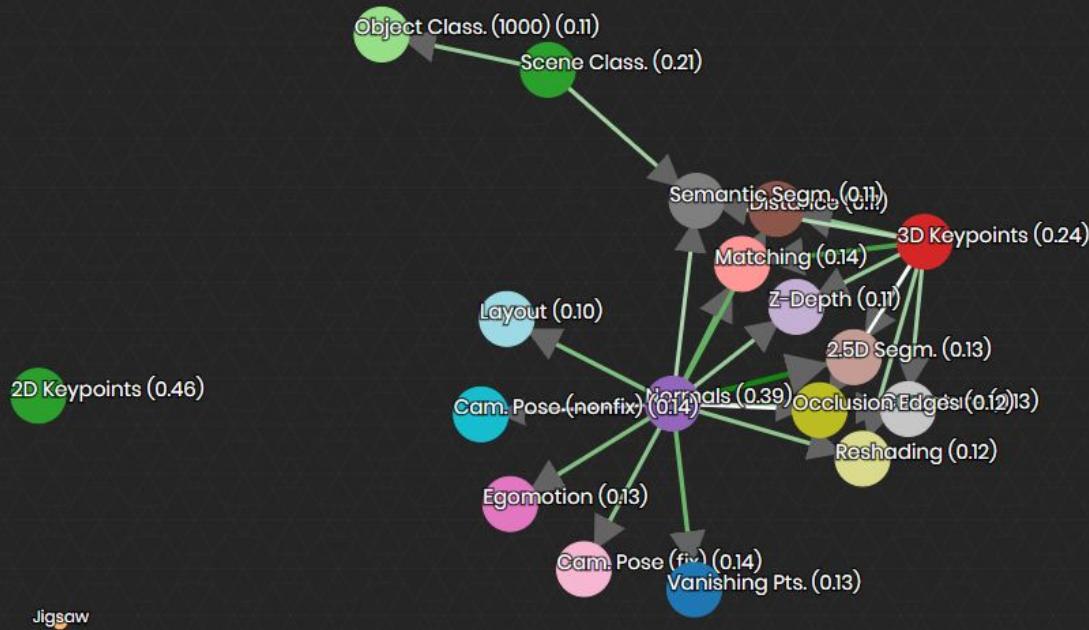
Choose task dictionary ▾



Solve! ↗

Instructions/Definitions ⓘ

Color Nodes:



0% 100%

Adaptation tricks

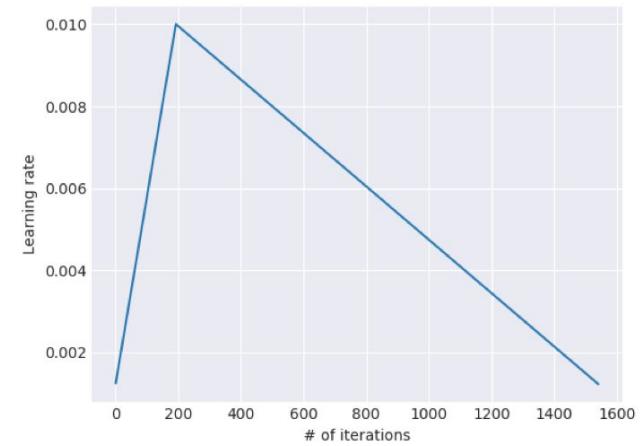
Triangle learning rate

At the start some of the model is randomly initialized. Cannot trust the gradient

Discriminative fine-tuning

Instead of using the same learning rate for all layers of the model, discriminative fine-tuning allows us to tune **each layer with different learning rates**.

- Layers that are closer to inputs → large learning rate
- Layers that are closer to outputs → small learning rate



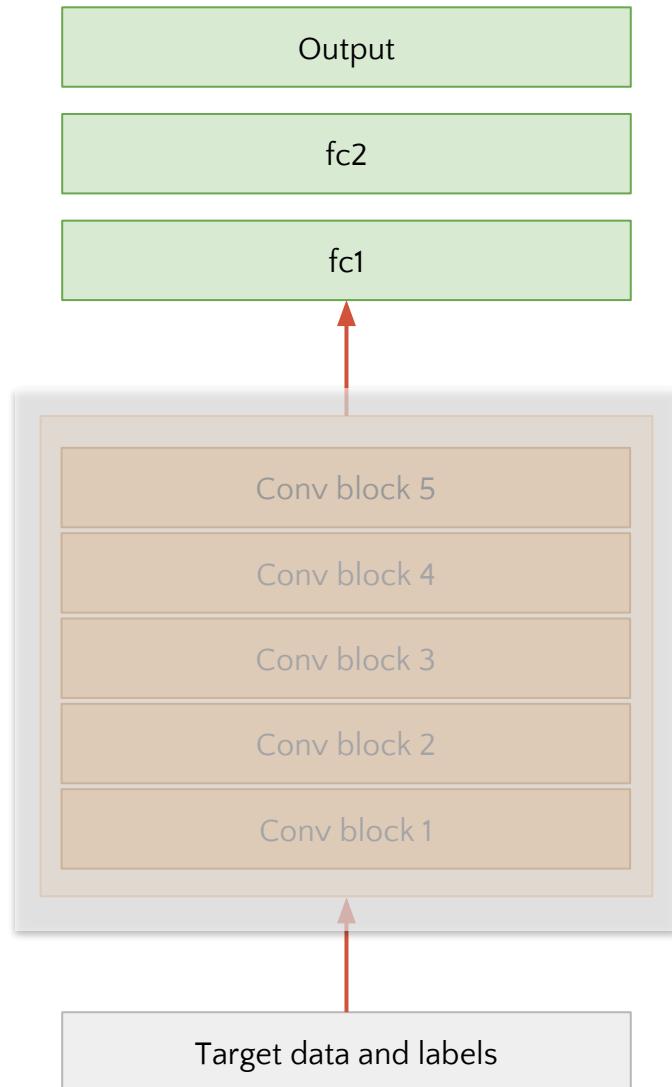
Advance adaptation ideas

- Chain-taw
 - Freeze the old layers, train the new weights for a bit
1. fine-tunes any new layers.
 2. fine-tunes each layer individually of base model starting from the first to the last.
 3. the entire model is trained with all layers.

Chain-taw

1. fine-tunes any new layers.

Freeze weights

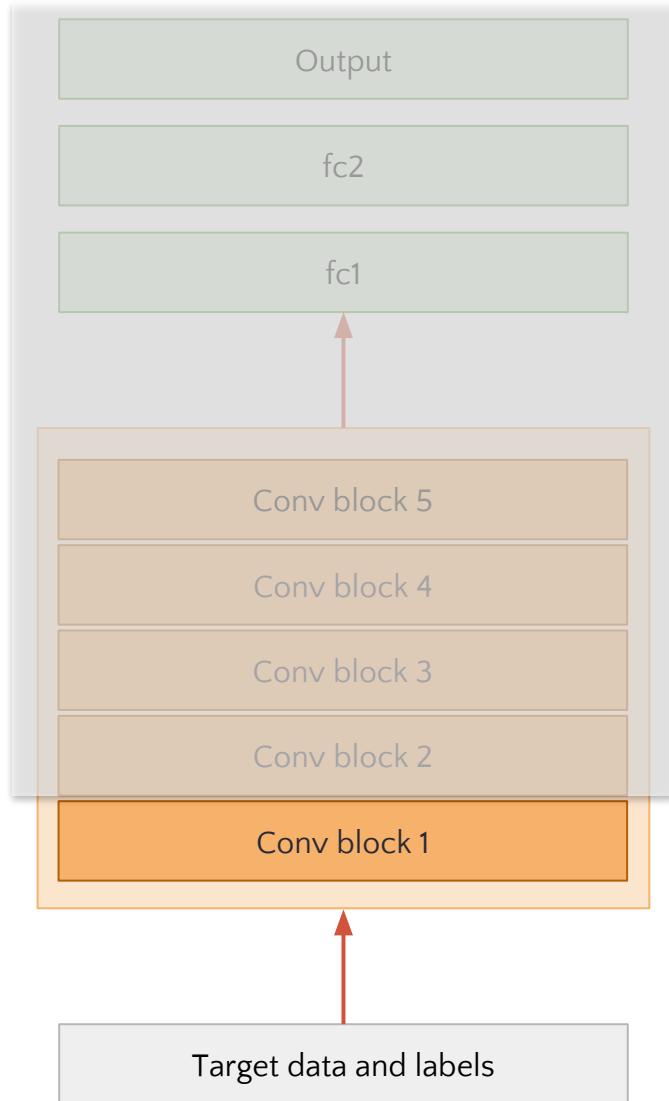


Chain-taw

Freeze weights



2. fine-tunes each layer individually of base model starting from the first to the last.

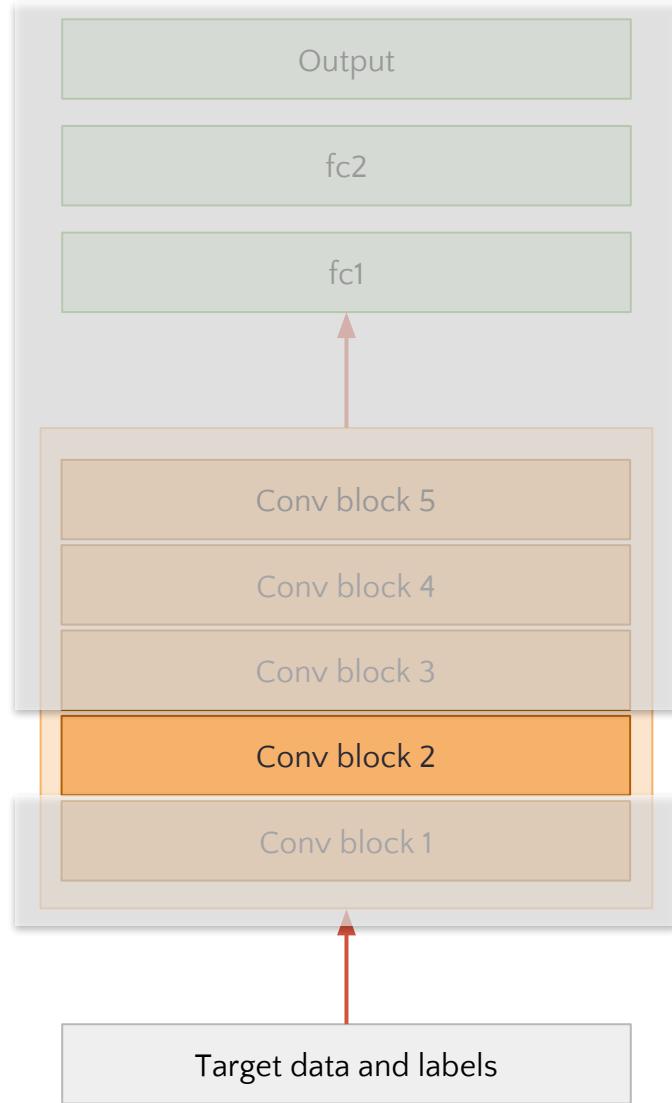


Chain-taw

Freeze weights



2. fine-tunes each layer individually of base model starting from the first to the last.

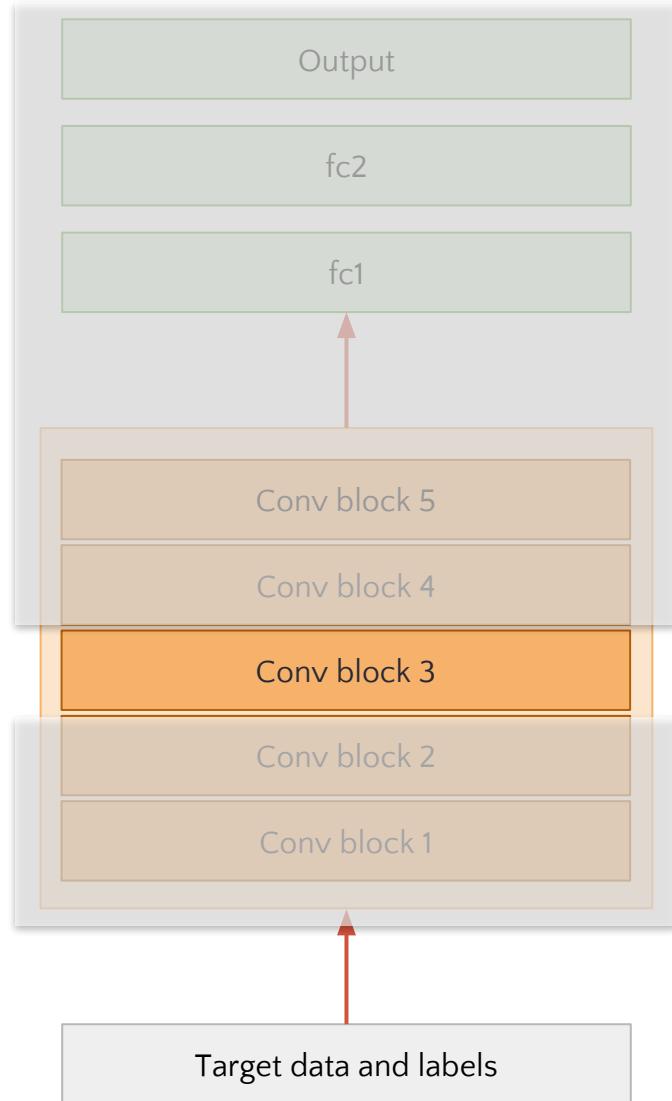


Chain-taw

Freeze weights

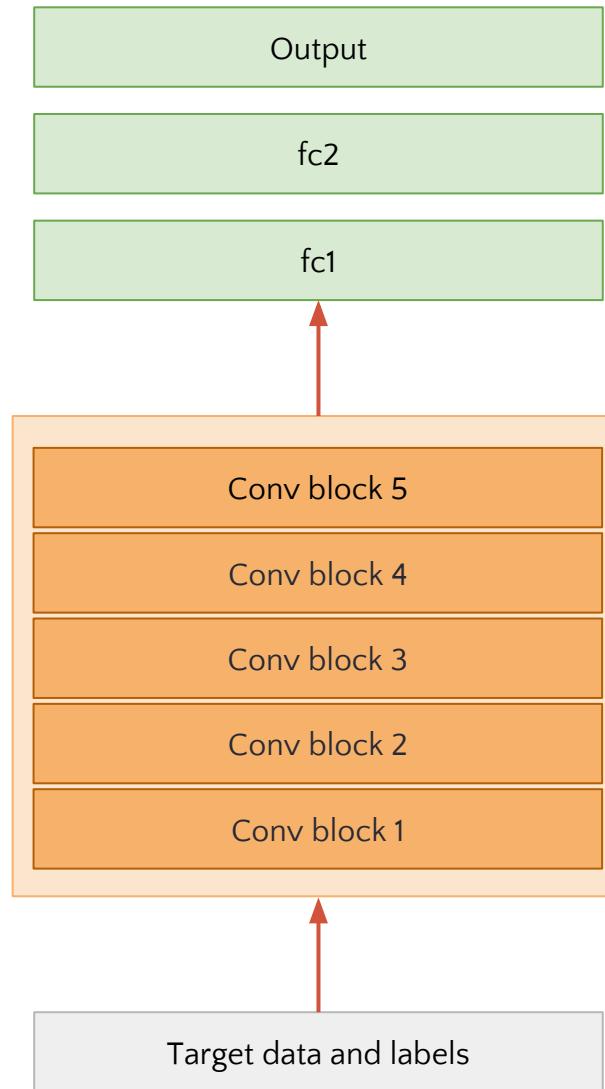


2. fine-tunes each layer individually of base model starting from the first to the last.



Chain-taw

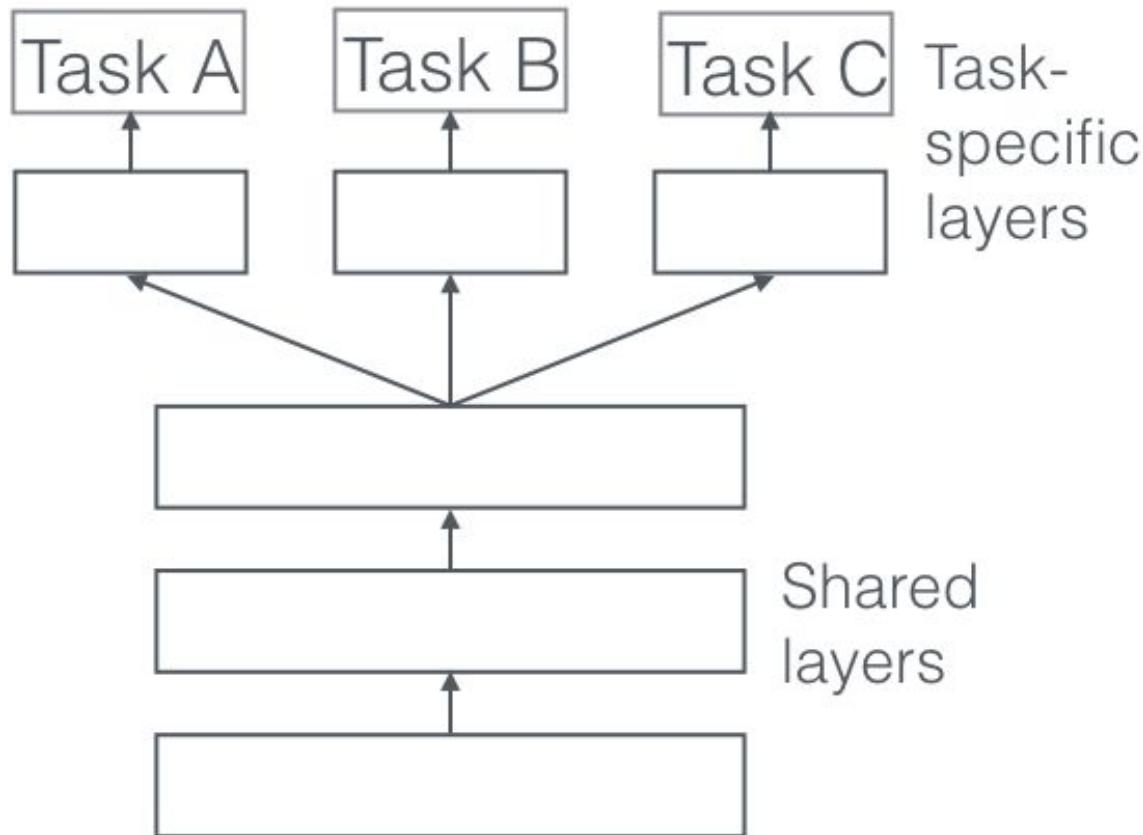
3. the entire model is trained with all layers.



Multi-task training (Joint models)

- Training a neural network on multiple tasks at the same time can help the network distinguish between noise and important information
- Helps partition the output space
- Example: Input speech, output text vs Input speech, output text and language.

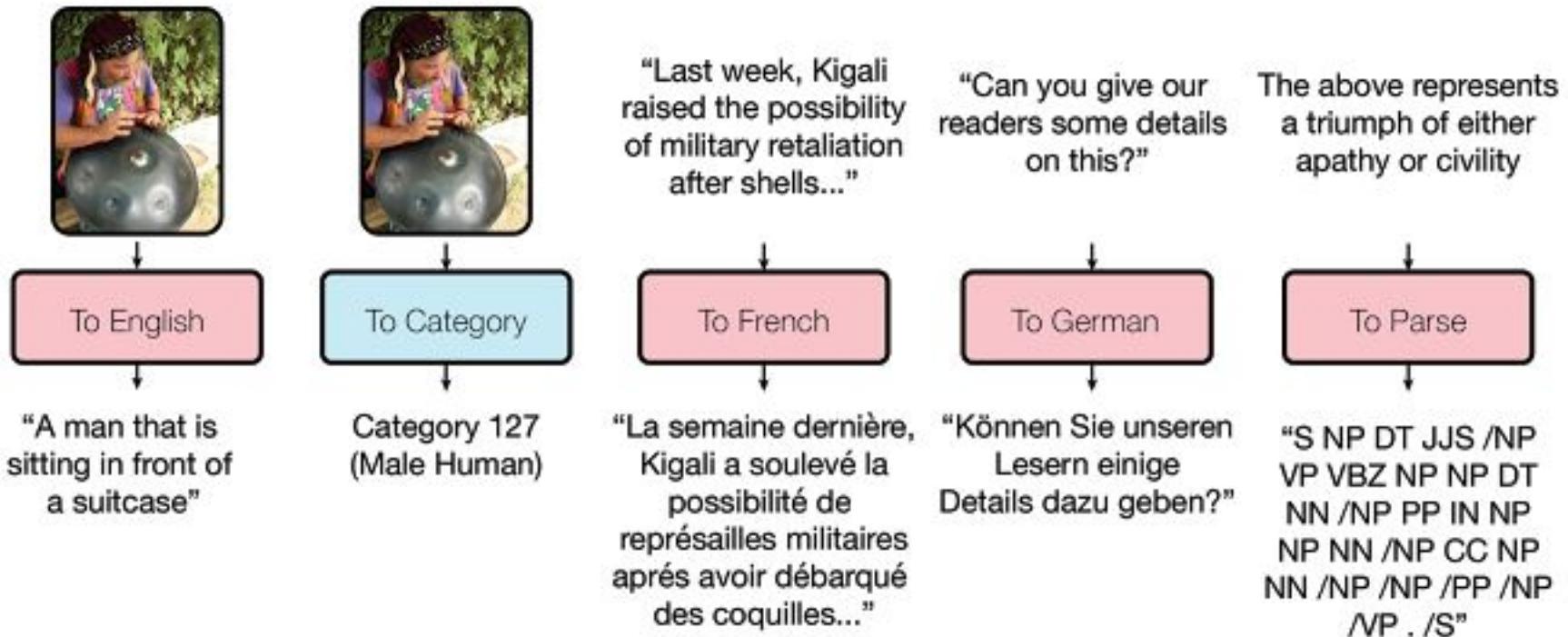
Multi-task learning (simplest form)

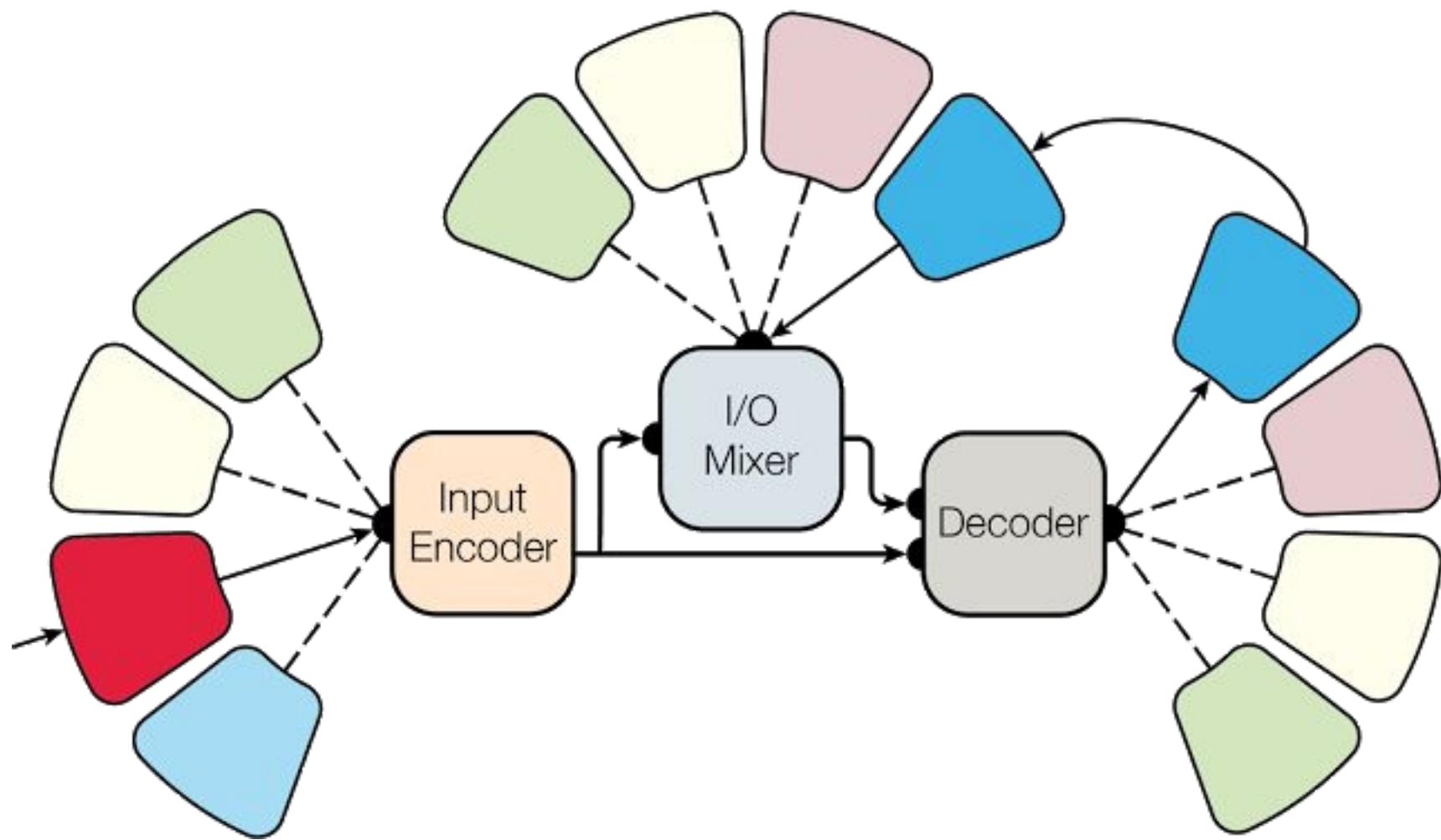


$$\text{Total Loss} = W_0 \text{ Loss A} + W_1 \text{ Loss B} + W_2 \text{ Loss C}$$

Multimodels

- A single model that can handle different kinds of inputs
- Joint representation of meaning improves performance



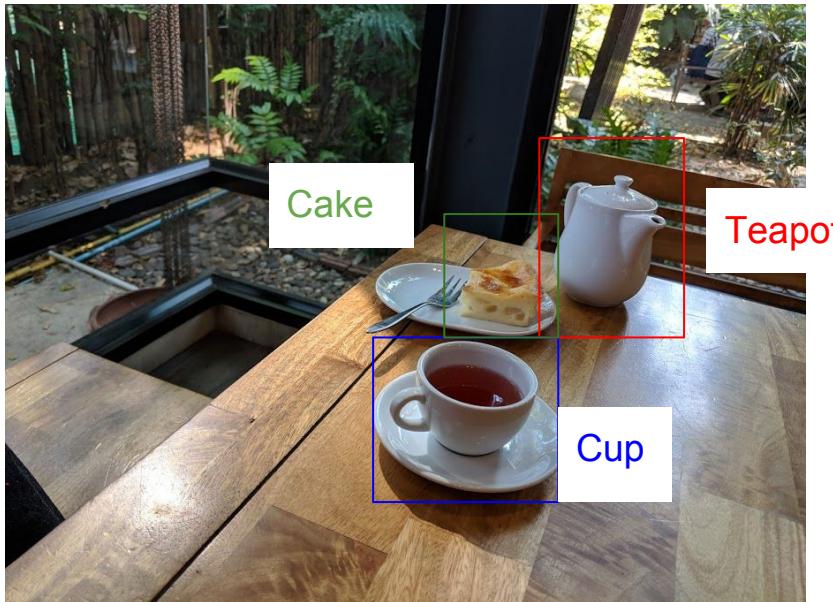


Misc topics

- Encoder-Decoder
- Attention modeling
- Transfer Learning, Multi-task models, Multi-models
- Object detection
- Alternatives to Backprop
 - 2nd order methods
 - Synthetic gradients
 - Genetic Algo
- Reinforcement Learning <- Next lecture!
- GAN <- Next Next lecture!
 - Adversarial inputs

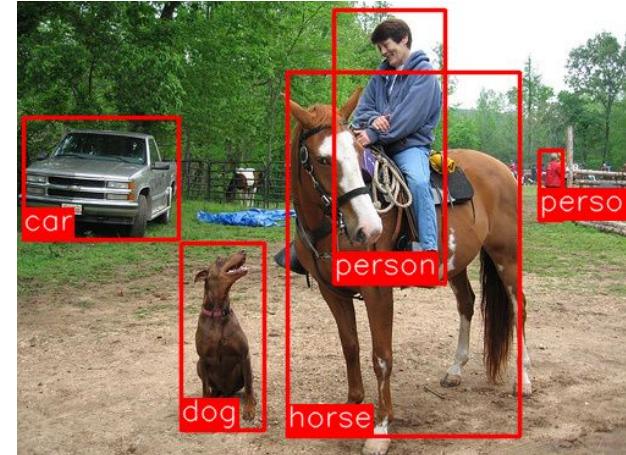
Object detection

Locate every object in an image and identify them



Object detection

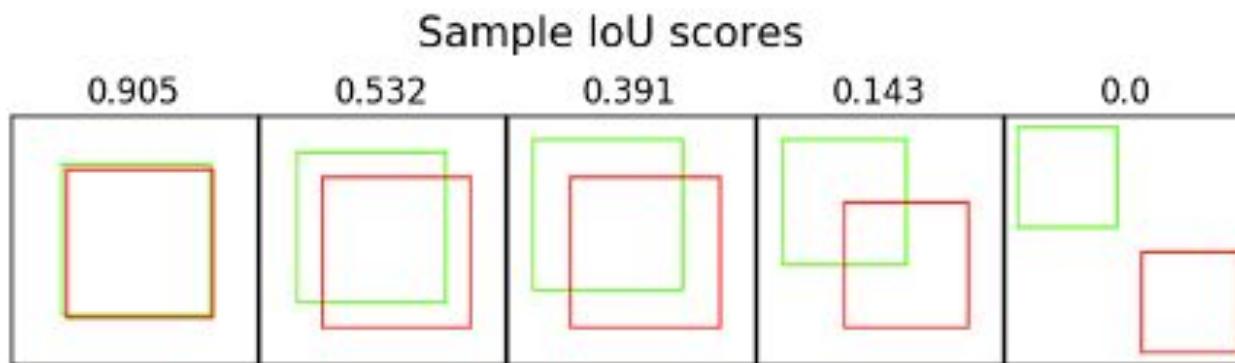
- Identify all objects in an image
- Two sub-task
 - Determine what objects
 - Determine where objects are



IoU (Intersect over Union)



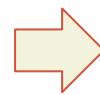
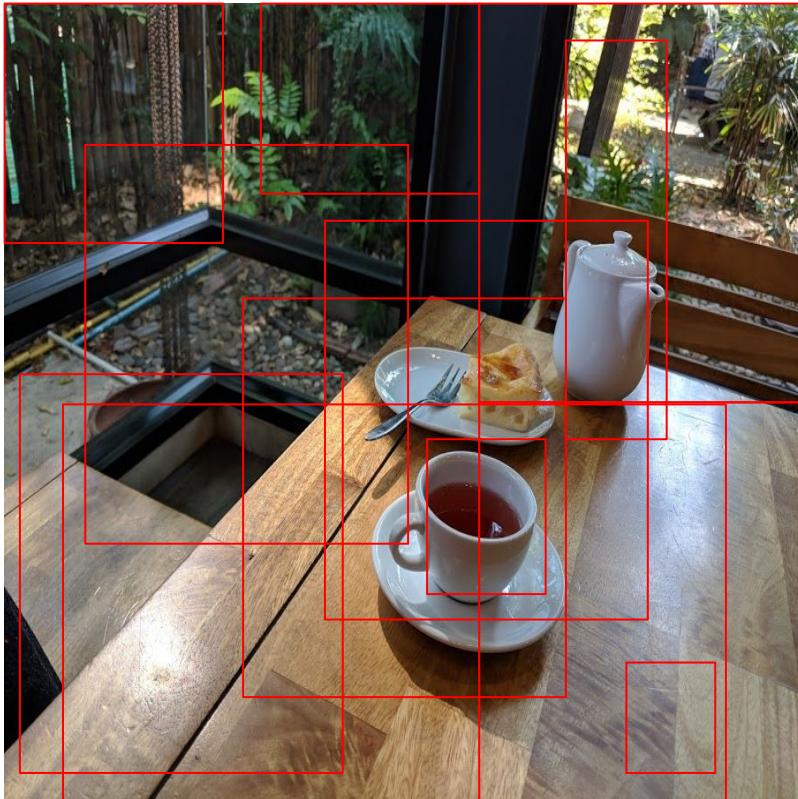
What IoU score should be considered a detection?



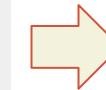
Simplest solution: sliding window

Slide a object classifier over every possible region and scale

Expensive. Can we do better?



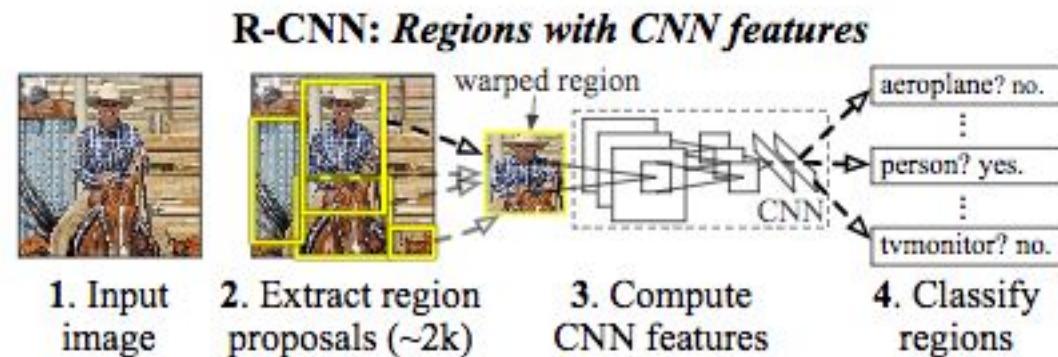
Object
classifier
(backbone)



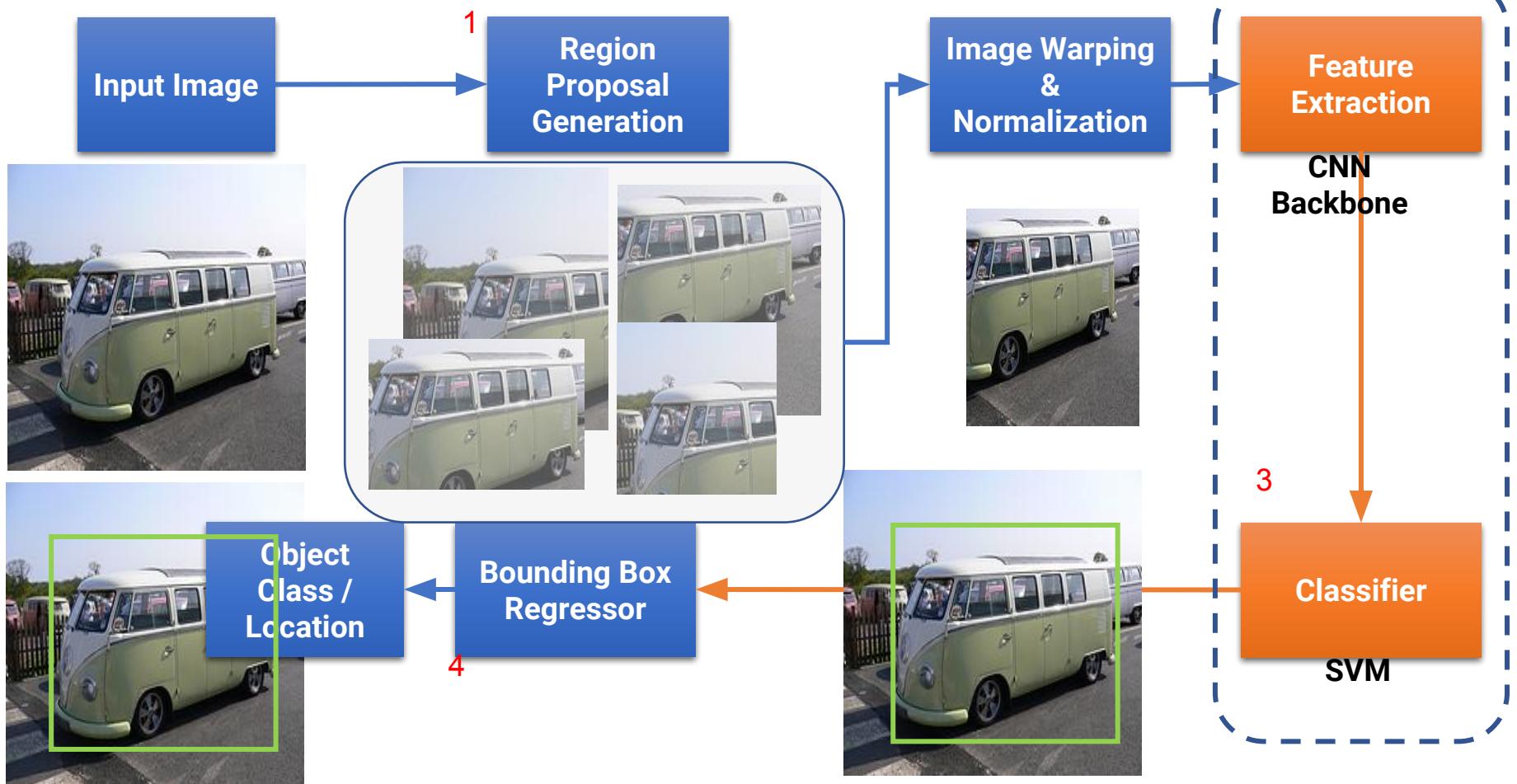
Candidate lists
with probabilities

Object Detection Comprehensive Review

1. Proposal-based detector
 - R-CNN
 - Fast-RCNN
 - Faster-RCNN
2. Single stage detector
 - YOLO
 - SSD



Regions with Convolutional Neural Network Feature (R-CNN)



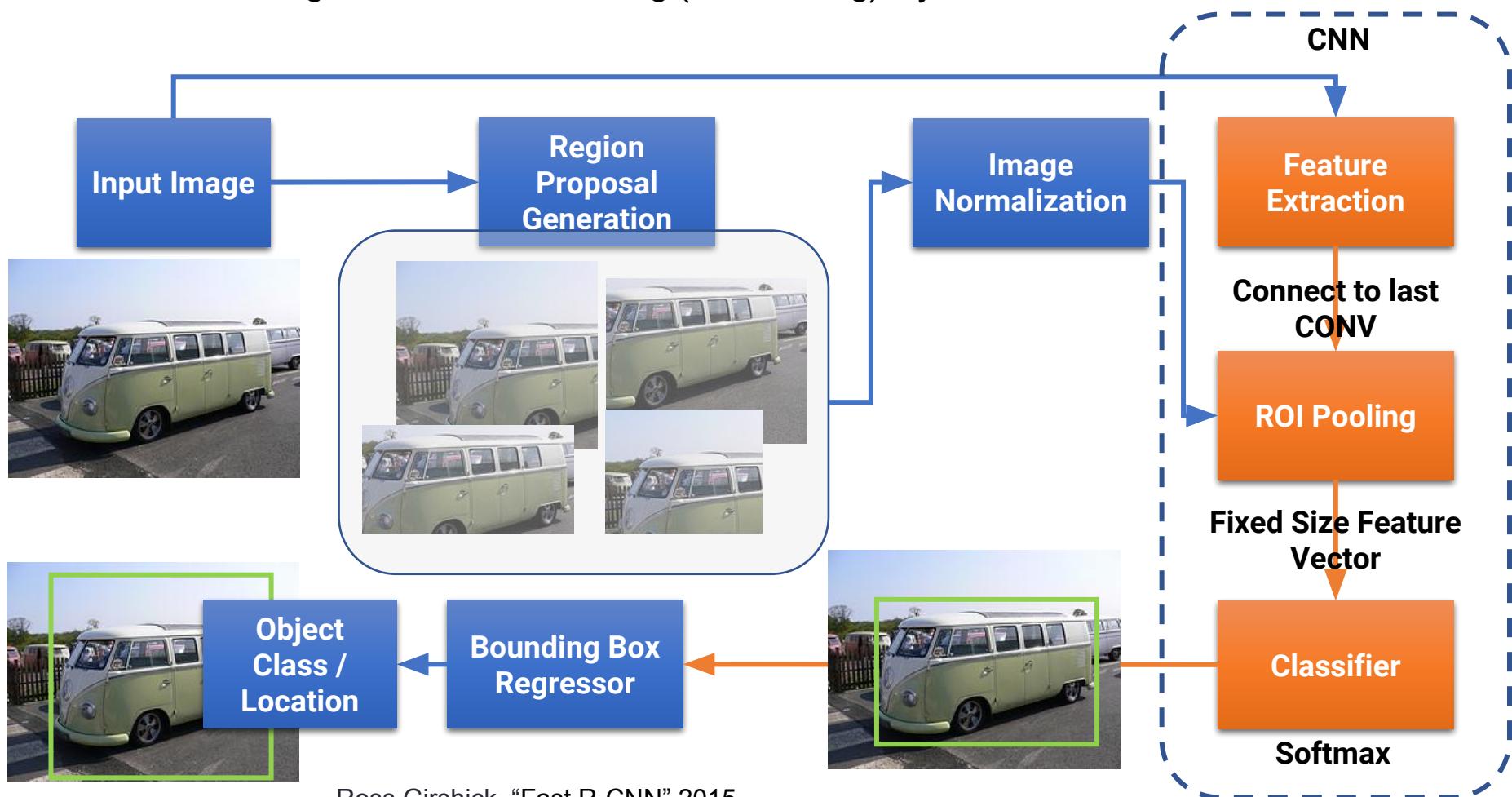
Ross Girshick, "Rich feature hierarchies for accurate object detection and semantic segmentation" 2013.

Regions with Convolutional Neural Network Feature (R-CNN)

- Region proposal generation (selective search) is computationally expensive.
- No computation sharing between regions.
- In the training process, R-CNN suffers from the following complex multi-stages training.
 - The backbone needs to be adapted to the new task (detection and regression)
 - The SVM classifier
 - The bounding box regressor

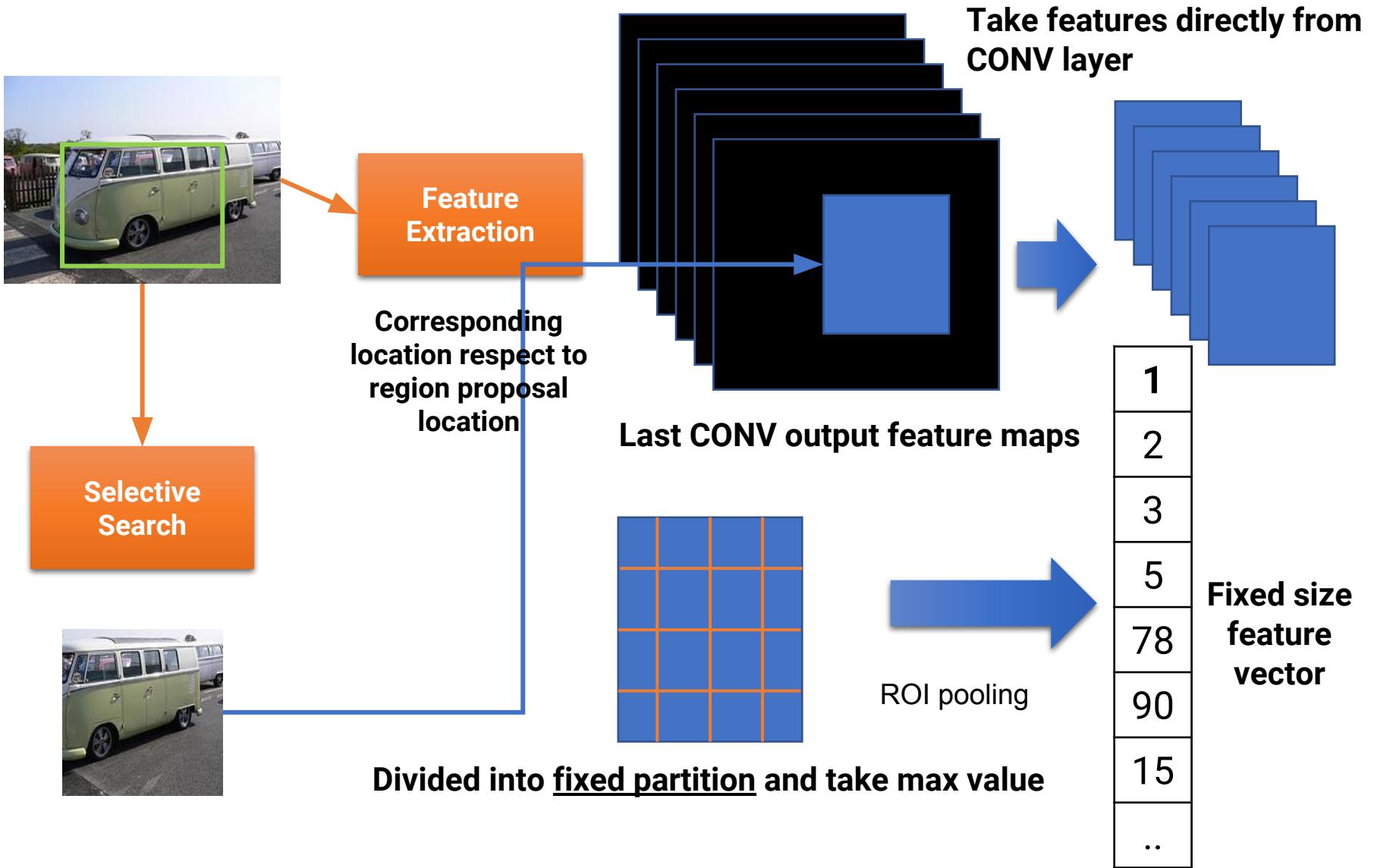
Fast-RCNN

- Computation sharing for each region proposal by computing the feature extraction once then extract features for each region.
- A novel Region Of Interest Pooling (ROI Pooling) layer.



Ross Girshick, "Fast R-CNN" 2015

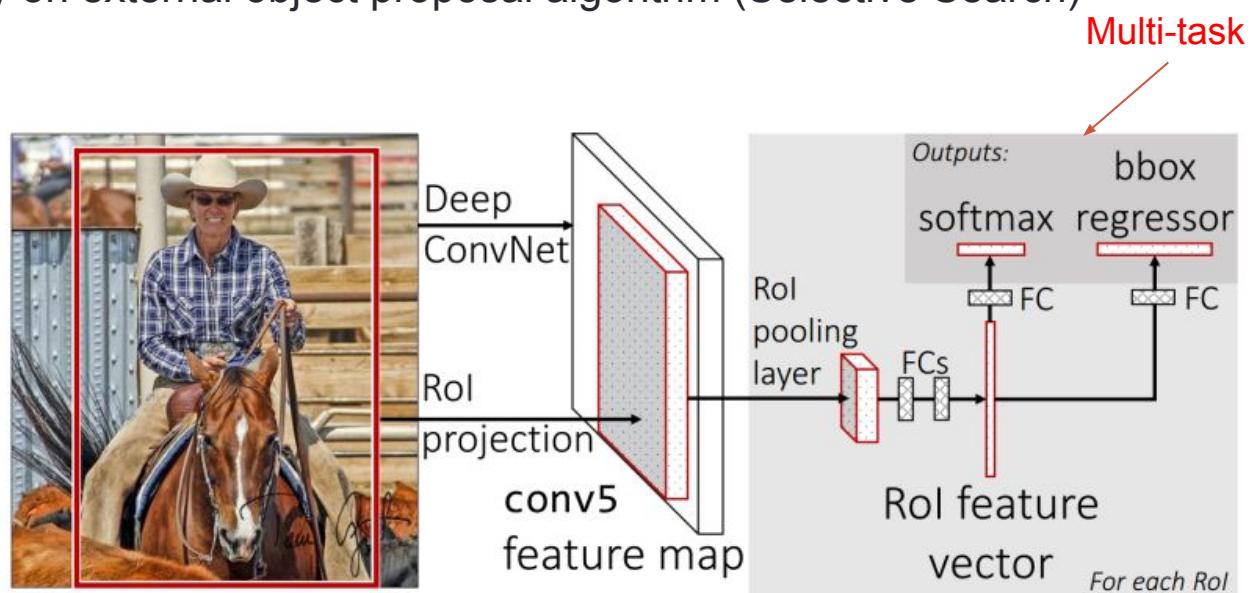
Fast-RCNN



Fast-RCNN

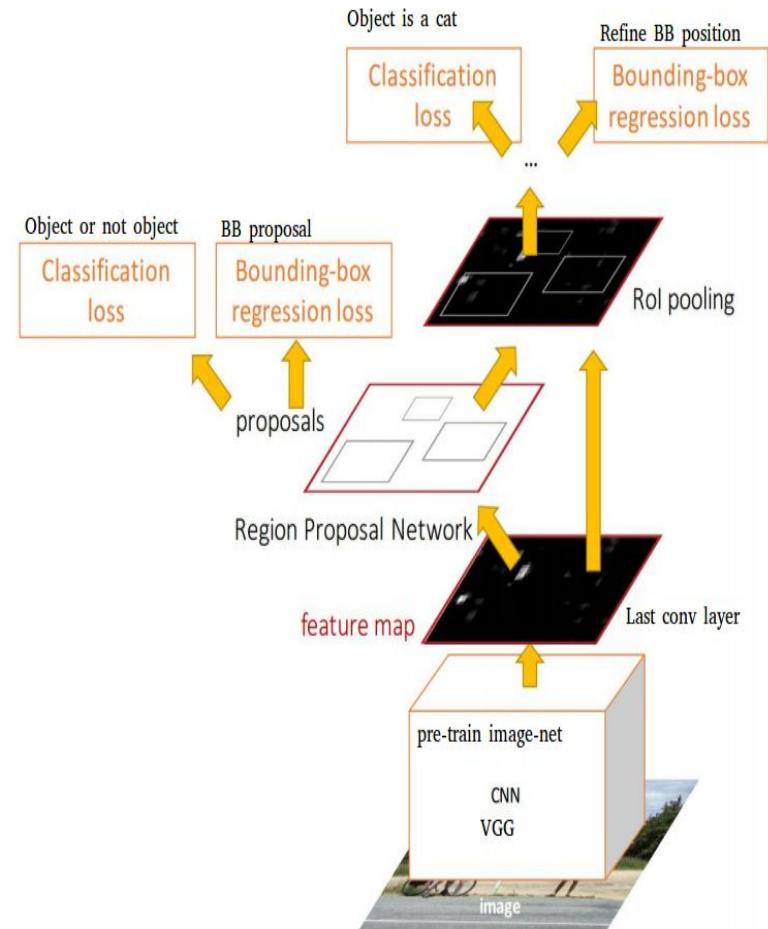
- Fast R-CNN still rely on external object proposal algorithm (Selective Search)

Selective
Search

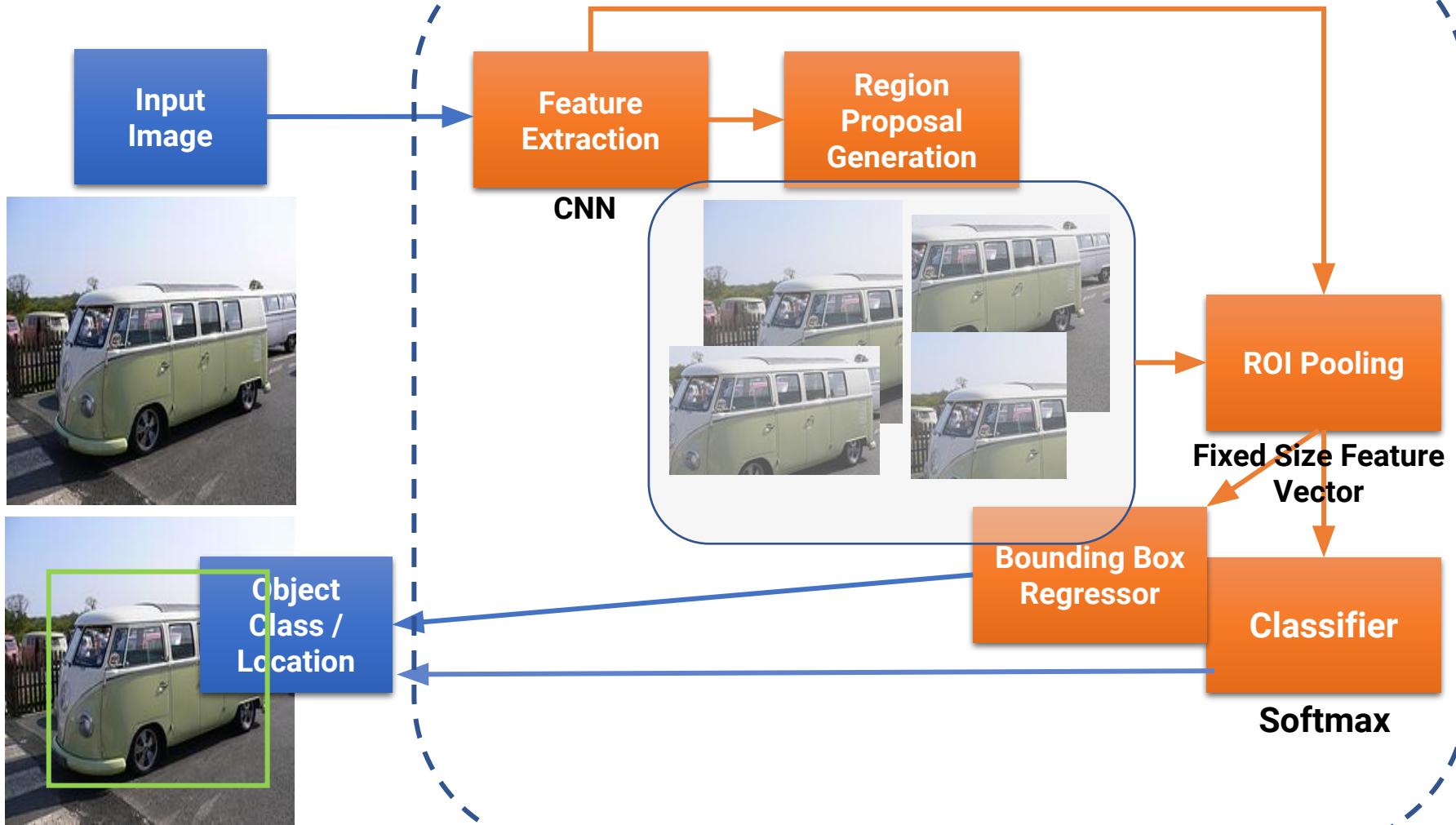


Faster-RCNN

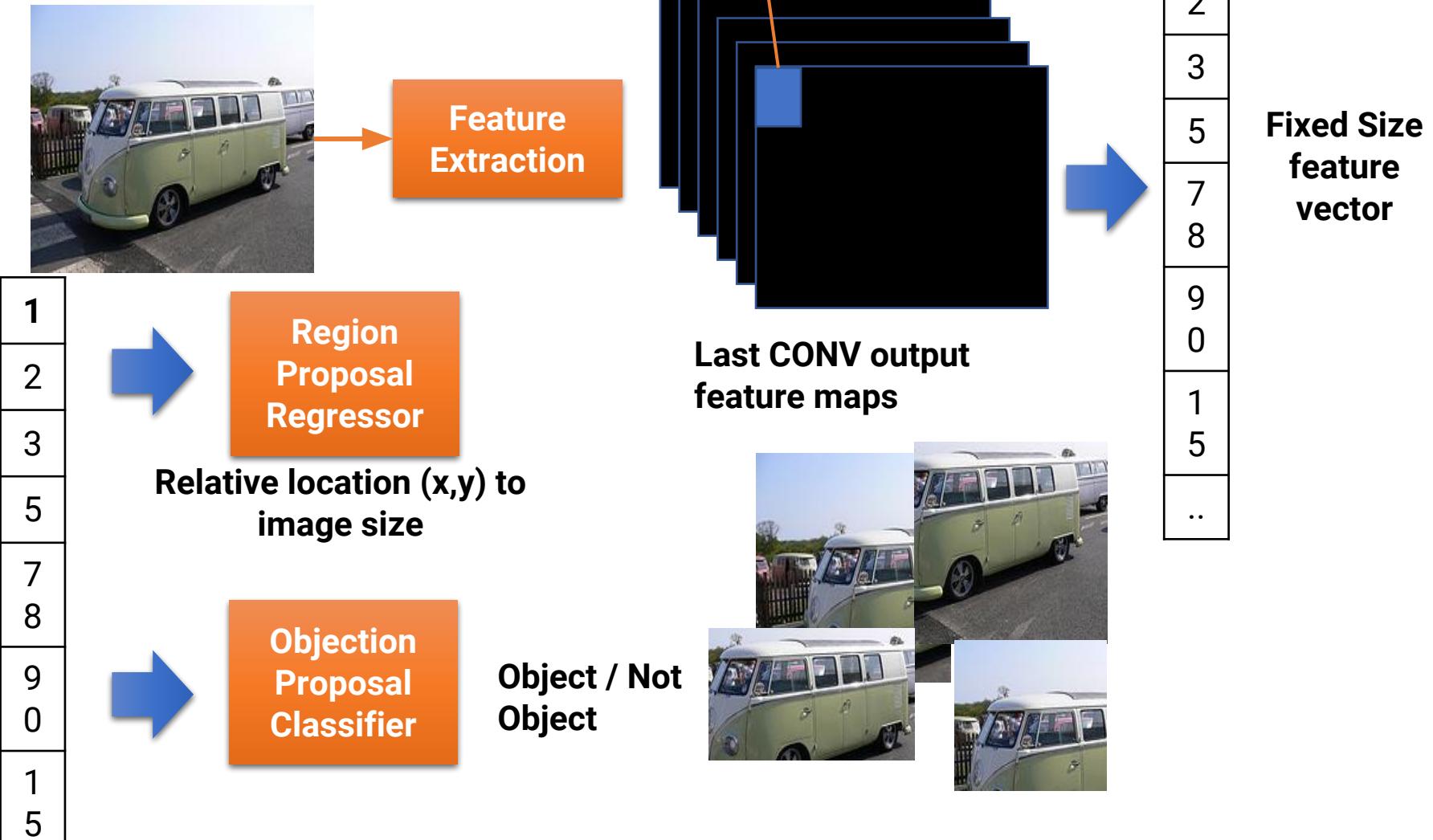
- Combine the region proposal and classification into a single network
- End-to-End learning from region proposal to classification stage.



Faster-RCNN



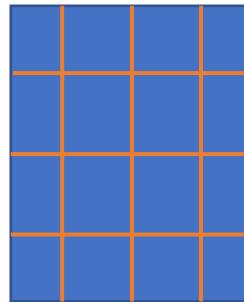
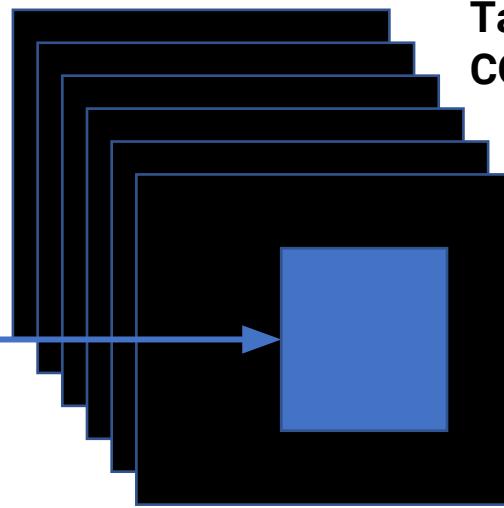
Region proposal



Final output



Corresponding
location respect to
region proposal
location



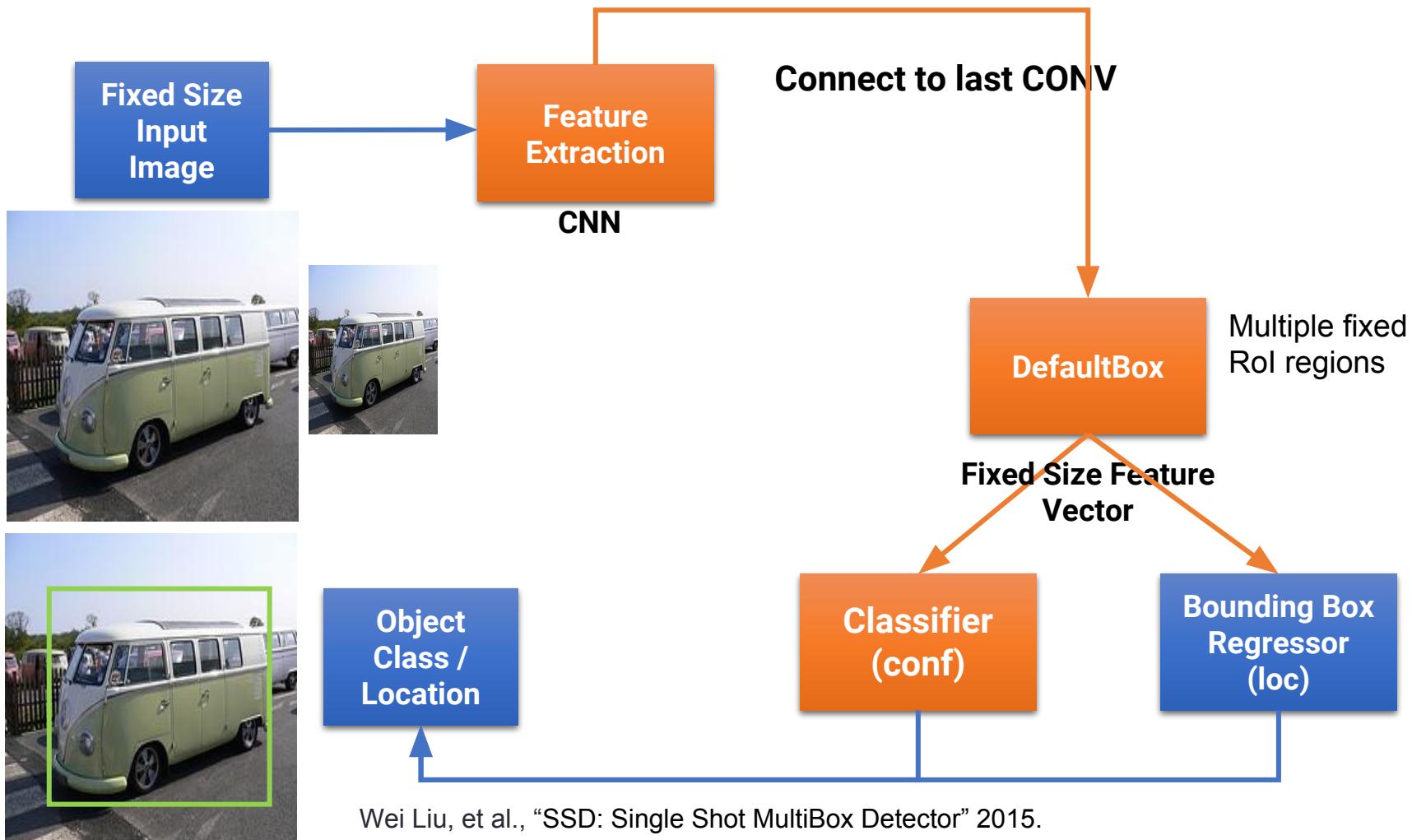
Take feature directly from
CONV layer

| |
|----|
| 1 |
| 2 |
| 3 |
| 5 |
| 78 |
| 90 |
| 15 |
| .. |

Fixed size
feature
vector

SSD : Single Shot Multibox Detector

- Trade off accuracy for computational speed => Realtime detector

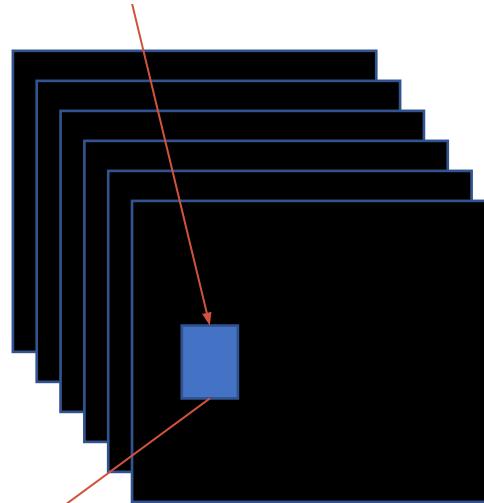


SSD : Default Box Idea

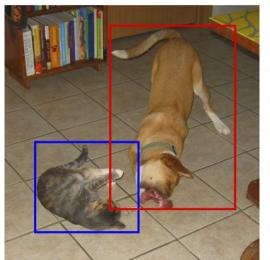


Feature Extraction

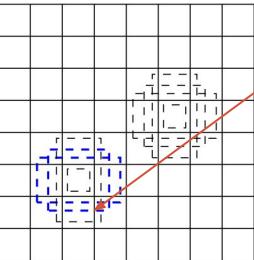
3x3 conv



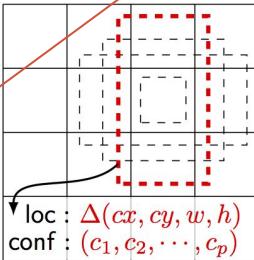
Last CONV output
feature maps



(a) Image with GT boxes



(b) 8 × 8 feature map

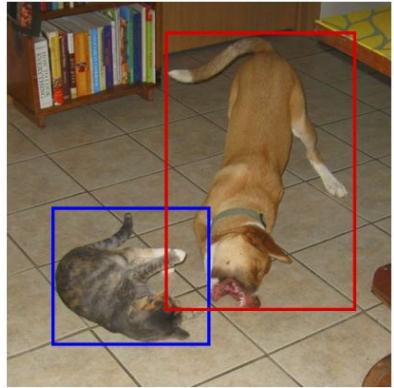


(c) 4 × 4 feature map

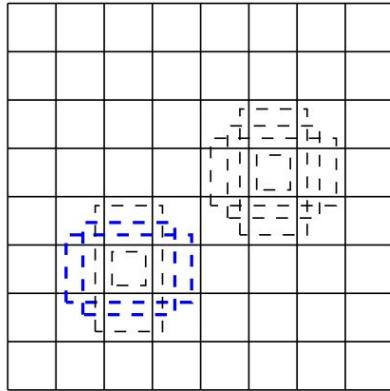
Every feature map cell is associated with a set of default bounding boxes of different dimensions and aspect ratios. Represents “default box” in xywh coordinate system (loc) and class output (conf)

In training process, SSD will optimize loc,conf which corresponds to the object's real position based on IOU criterion.

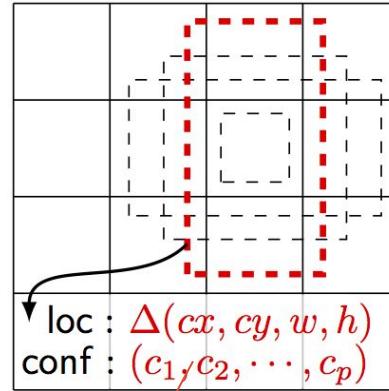
SSD



(a) Image with GT boxes



(b) 8×8 feature map



(c) 4×4 feature map

loc : $\Delta(cx, cy, w, h)$
conf : (c_1, c_2, \dots, c_p)

$$\text{MultiBoxLoss}(x, c, l, g) = L_{\text{conf}}(\text{CrossEntropyLoss}(x, c)) + L_{\text{loc}}(\text{SmoothL}_1\text{Loss}(x, l, g))$$

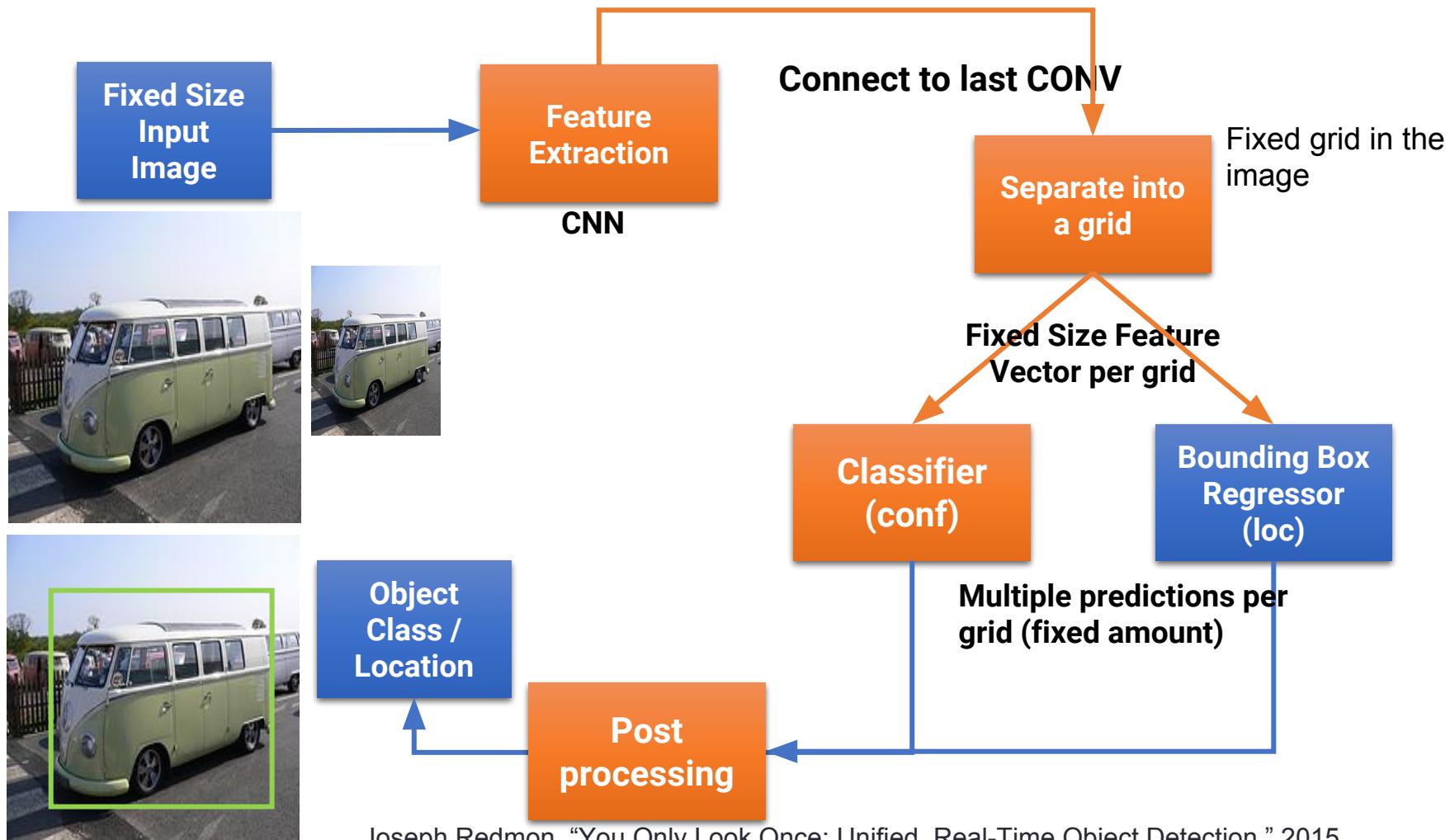
IOU > 0.5
(Match with O1)

IOU > 0.5
(Match with O2)

SSDLoss = Classification loss + Localization Loss

YOLO: You Only Look Once

- Similar concept as SSD, fixed locations to look for things

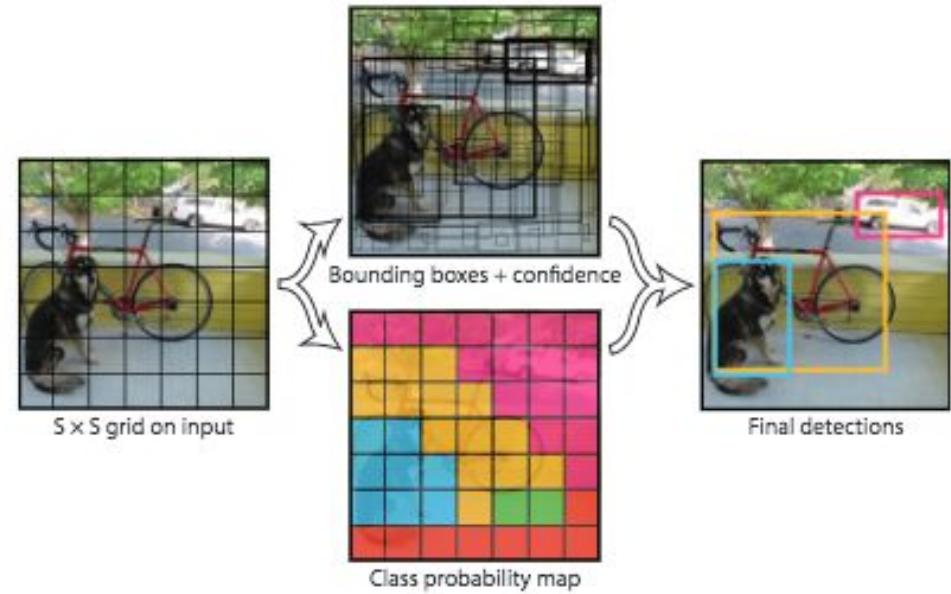


YOLO (You Only Look Once)

SSD have multiple boxes per location (multi-resolution), YOLO has one box per location

YOLO makes multiple predictions per box at once.

Limitation of YOLO: cannot predict things that are close together.



YOLO v2 (YOLO9000)-v3

Batch norm

Anchor boxes: fixed set of bounding boxes per grid

Handles multi-resolution by concatenating features from multiple parts of the backbone

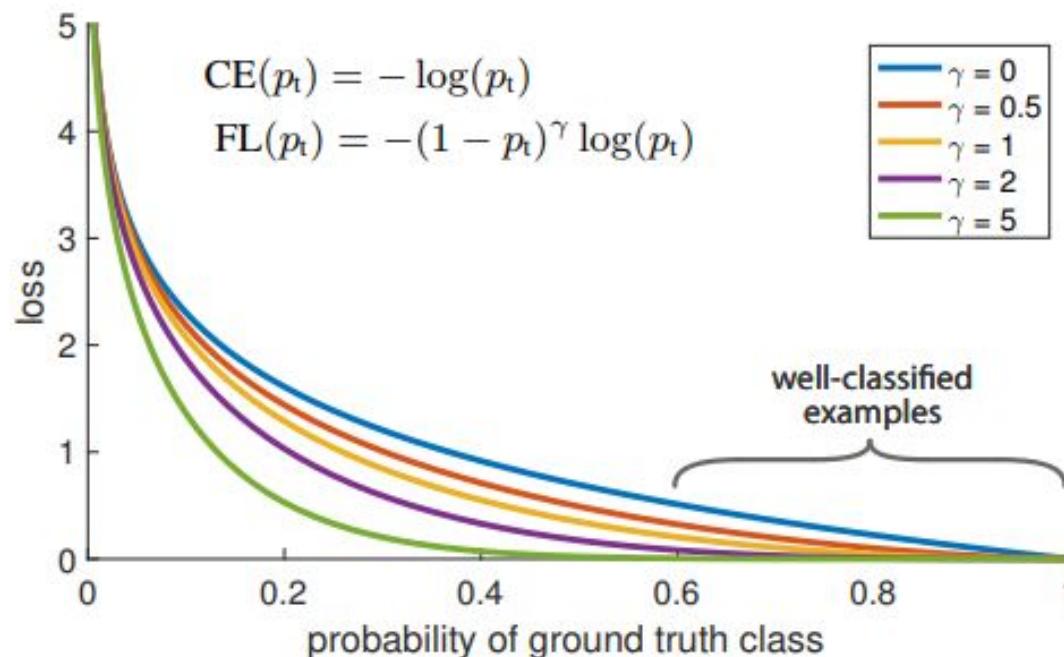
Focal Loss

Object detectors suffer from class imbalance

Most boxes have no objects

Some training examples are also easy to classify

Cross entropy put too much focus on easy examples



Class weights to handle imbalance

$$\text{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1 - p) & \text{otherwise.} \end{cases} \quad P \text{ probability of predicting class 1}$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases} \quad P_t \text{ Probability of being correct}$$

$$\text{CE}(p, y) = \text{CE}(p_t) = -\log(p_t)$$

Add α which put emphasis on different class. It can be the inverse of number of training data for that class. (Class weights)

$$\text{CE}(p_t) = -\alpha_t \log(p_t).$$

Help reduce class imbalance problem.

Focal loss

$$\text{CE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1 - p) & \text{otherwise.} \end{cases} \quad P \text{ probability of predicting class 1}$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases} \quad P_t \text{ Probability of being correct}$$

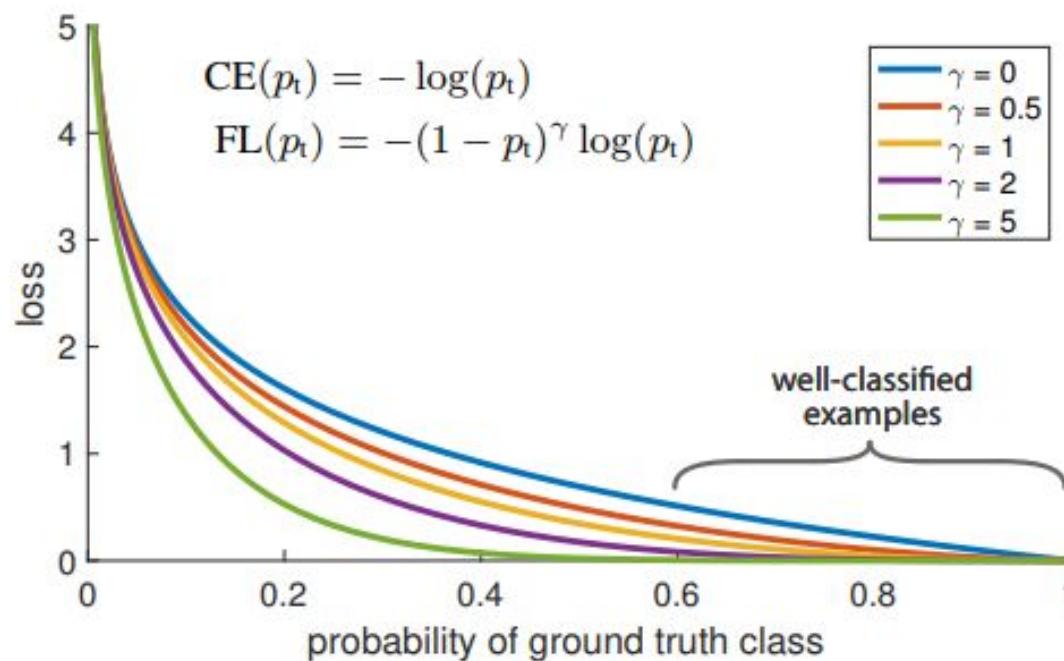
$$\text{CE}(p, y) = \text{CE}(p_t) = -\log(p_t) \quad \text{CE}(p_t) = -\alpha_t \log(p_t).$$

$$\text{FL}(p_t) = -(1 - p_t)^\gamma \log(p_t) \quad \text{Focal loss}$$

$$\text{FL}(p_t) = -\alpha_t (1 - p_t)^\gamma \log(p_t) \quad \text{Focal loss with class weights}$$

Focal Loss

Focal loss put higher emphasis on harder to classify examples



Good for many tasks besides CV

Misc topics

- Encoder-Decoder
- Attention modeling
- Transfer Learning, Multi-task models, Multi-models
- Object detection
- Alternatives to Backprop
 - 2nd order methods
 - Synthetic gradients
 - Genetic Algo
- Reinforcement Learning <- Next lecture!
- GAN <- Next Next lecture!
 - Adversarial inputs

2nd order methods

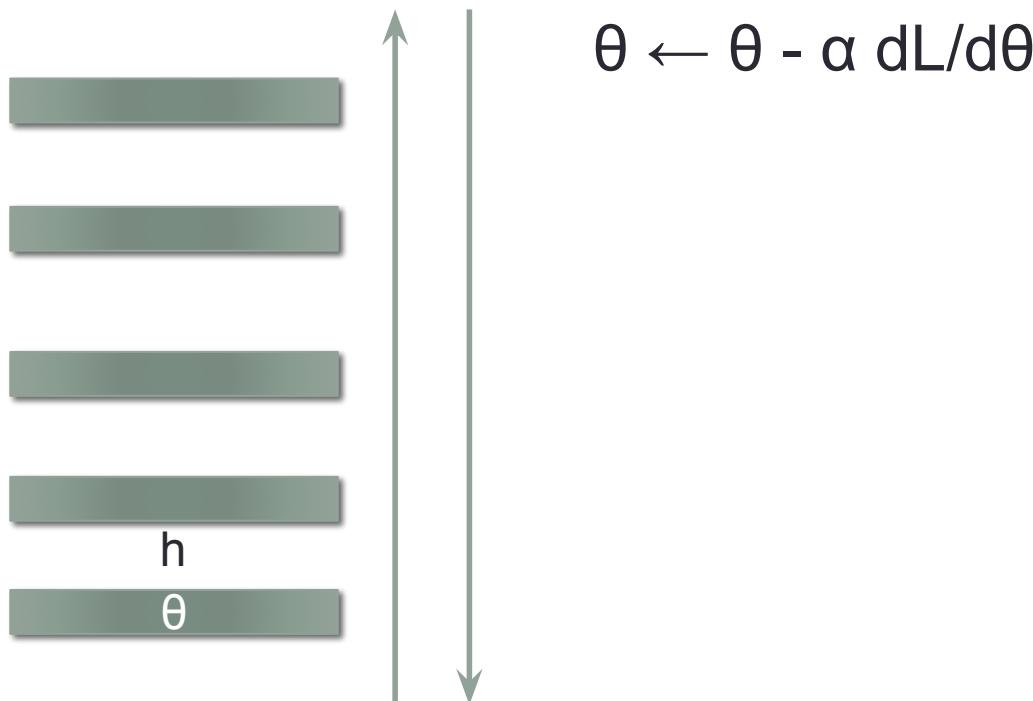
- How to minimize a function
 - Differentiation, set to 0
 - Hard for real problems
 - SGD of the differentiation
 - Needs step size because 1st derivative is local in that point
 - Use 2nd derivative information with 1st derivative.
 - 1st tells direction
 - 2nd tells how that direction would change as we move along.
 - Can be more sure about stepping and somewhat avoid local minima/inflection points
 - Other name: Newton method, Hessian method
 - Hessian – 2nd derivative matrix

2nd order methods and neural networks

- In theory it should be better than SGD
- Doesn't really work...yet
 - Active area of research

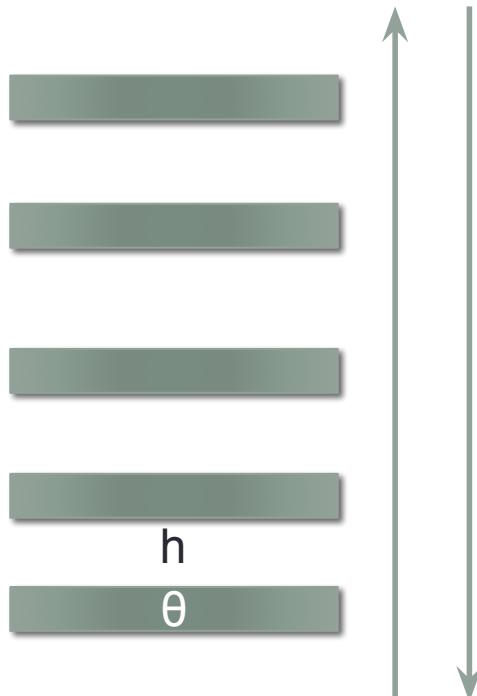
Synthetic gradients

- Deep networks can take time in doing forward and backward pass



Synthetic gradients

- To speed things up we can do early compute



$$\theta \leftarrow \theta - \alpha \frac{dL}{dh} \frac{dh}{d\theta}$$

$dh/d\theta$ we can compute right away

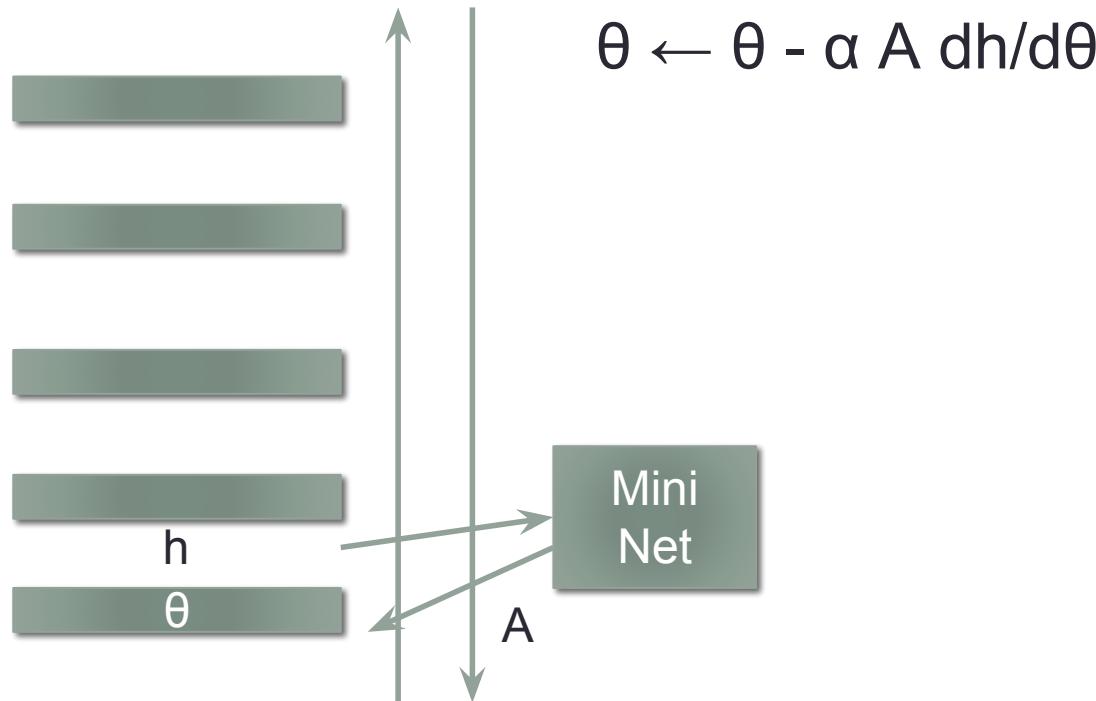
Use an estimate of $dL/dh = A$

$$\theta \leftarrow \theta - \alpha A \frac{dh}{d\theta}$$

Can compute update right away

Estimating A

- A can be a estimated small neural network
- This network gets updated as the true gradients come in.

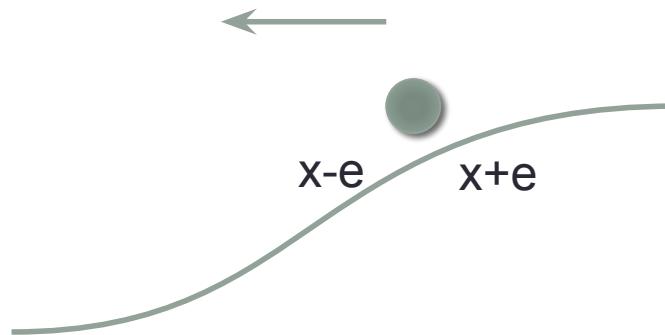


Synthetic gradients

- Probably doesn't help if you're not google/facebook/etc.
with large scale multi-gpu training
- But networks in networks are cool ☺

Genetic Algo for training neural networks

- How to minimize a function
 - Find direction
 - Find gradient by taking derivation
 - Try different directions, direction that goes down is the derivative!
 - Brute force solution



Genetic Algo for training neural networks

- Initialize network
- Spawns N networks with small perturbations
- Evaluate the loss, pick the best N
 - Update: $\text{network} \leftarrow \text{network} + r * \text{best network}$
- Works well for models that has tricky gradients
 - Reinforcement learning

Open research

- Unsupervised learning <- next next lecture
- More reasoning (understanding deep learning)
- Language understanding
- Training-test mismatch (Maximum likelihood criterion) and maybe an alternative to current training strategies

Misc topics

- Encoder-Decoder
- Attention modeling
- Transfer Learning, Multi-task models, Multi-models
- Object detection
- Alternatives to Backprop
 - 2nd order methods
 - Synthetic gradients
 - Genetic Algo
- Reinforcement Learning <- Next lecture!
- GAN <- Next Next lecture!
 - Adversarial inputs