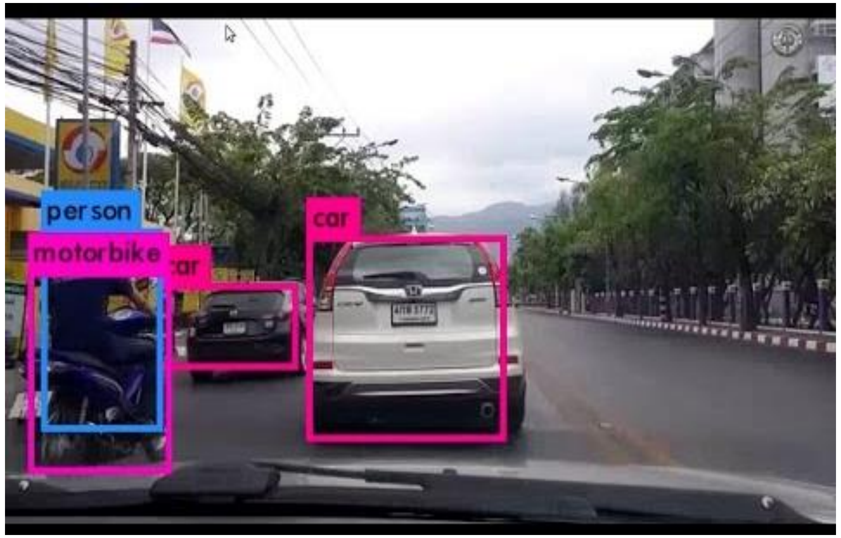
Convolutional Neural Networks

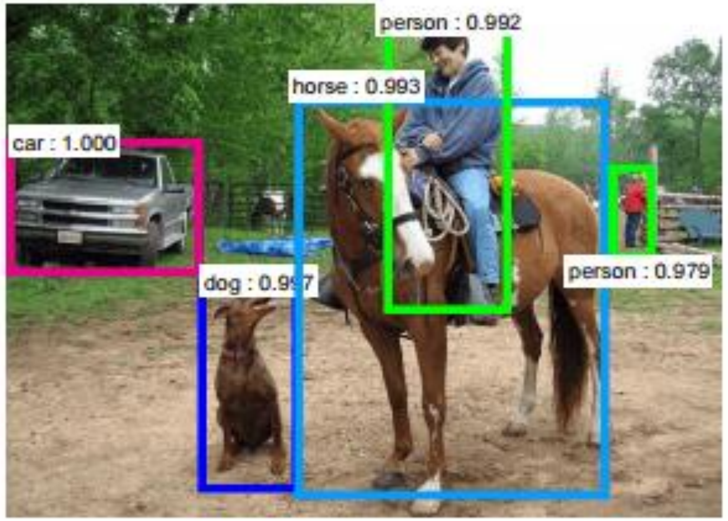
Face Recognition

Style Transferring

Image quality enhancement
Beautification



Object detection (Self driving car)



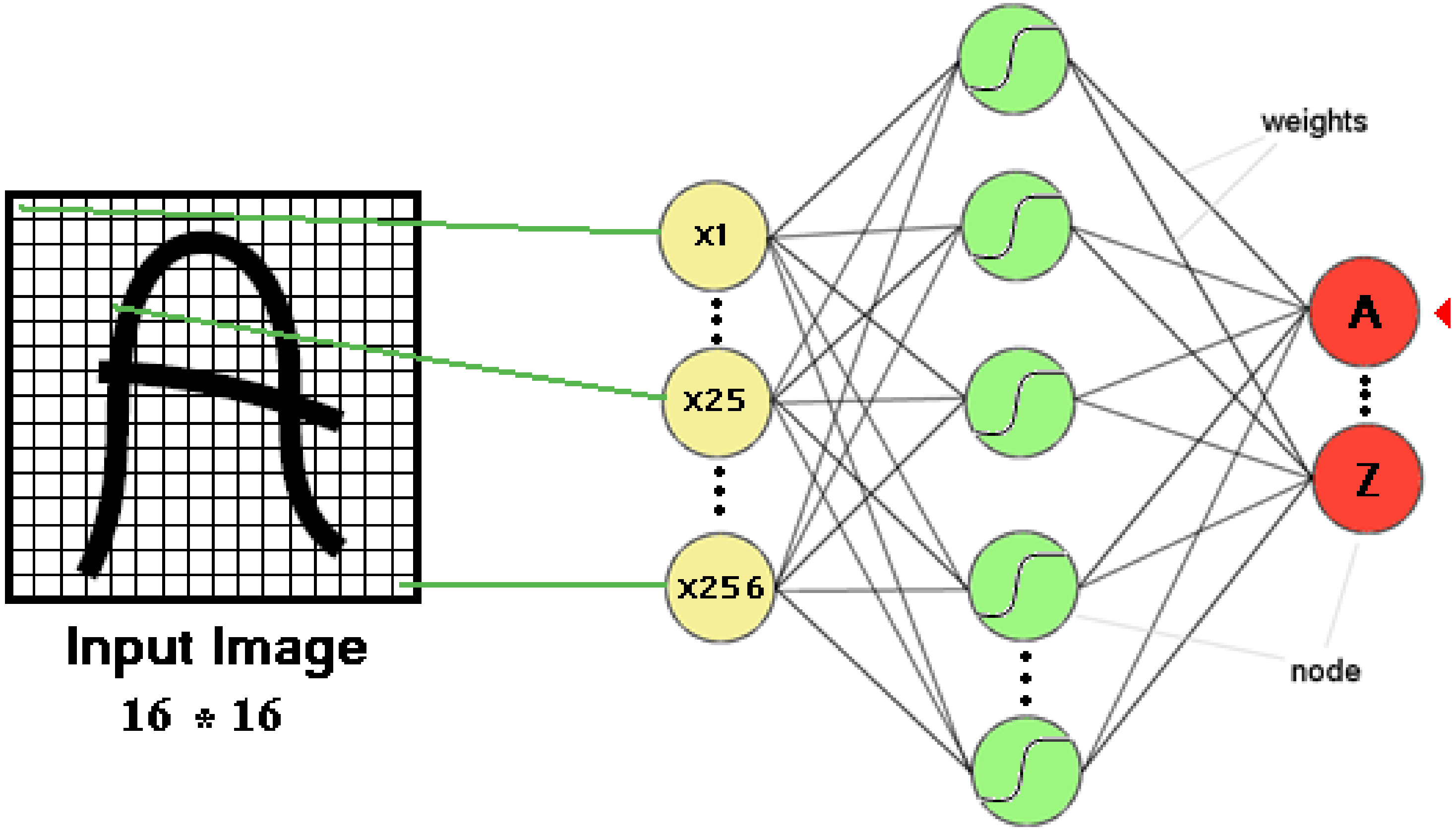
Classification

Gesture Recognition

Motivation

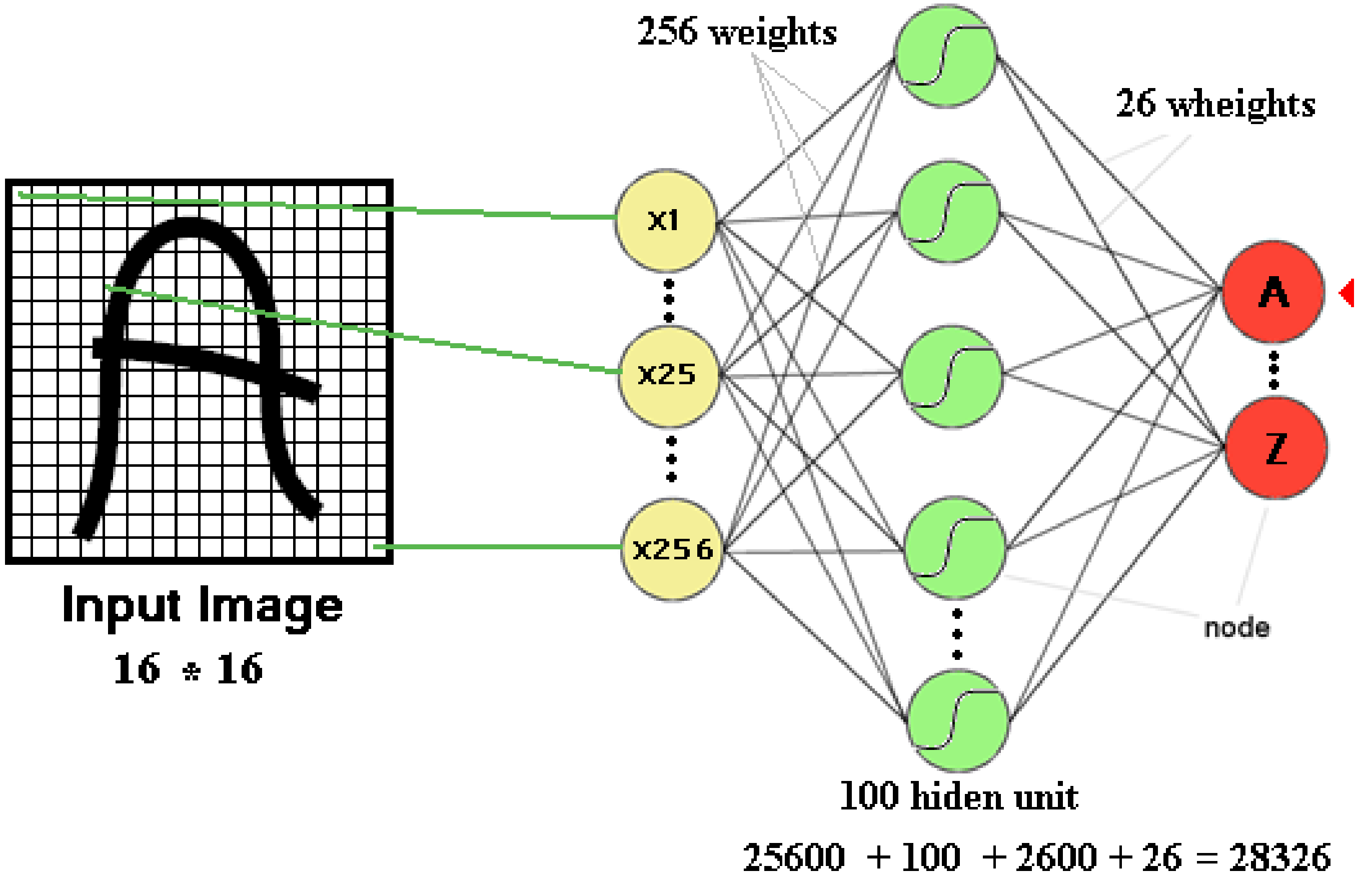
WHY WE NEED CONVOLUTIONAL NEURAL NETWROKS?

Multi-layer perceptron and image processing



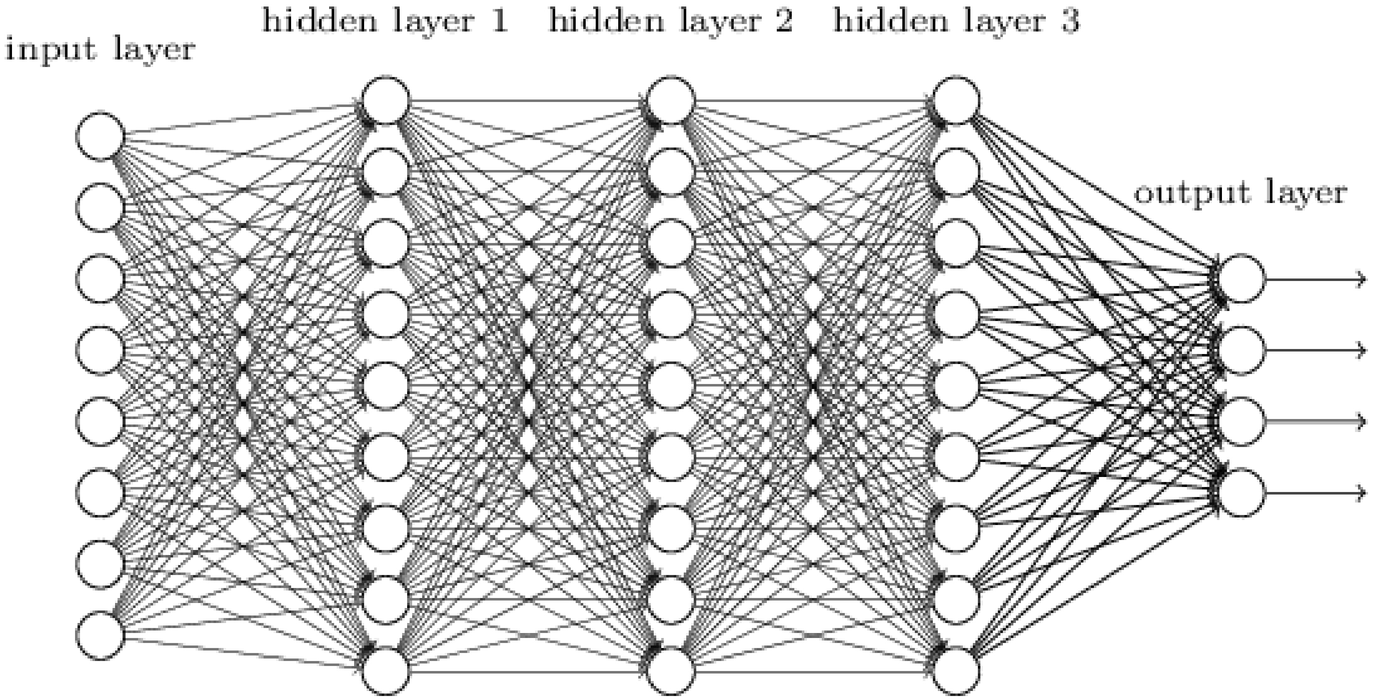
Even a simple 16x16 image

- **256x100 + 100 bias + 100x26 output neurons + 26 bias = 28236**
- the number of trainable parameters becomes extremely large

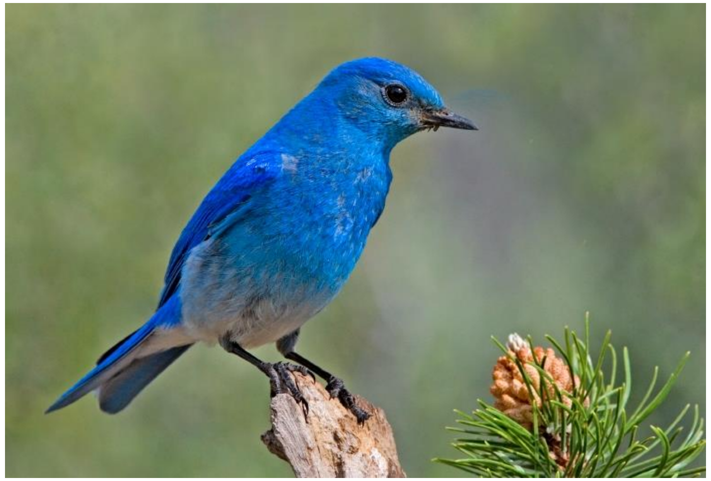


ANN: Too many parameters

- **We know it is good to learn a small model.**
- **From this fully connected model, do really need all the edges?**
- **Can some of these be shared?**



Identify



Can we do with less information?

- **How much information we can throw away and still recognize the object?**

10%

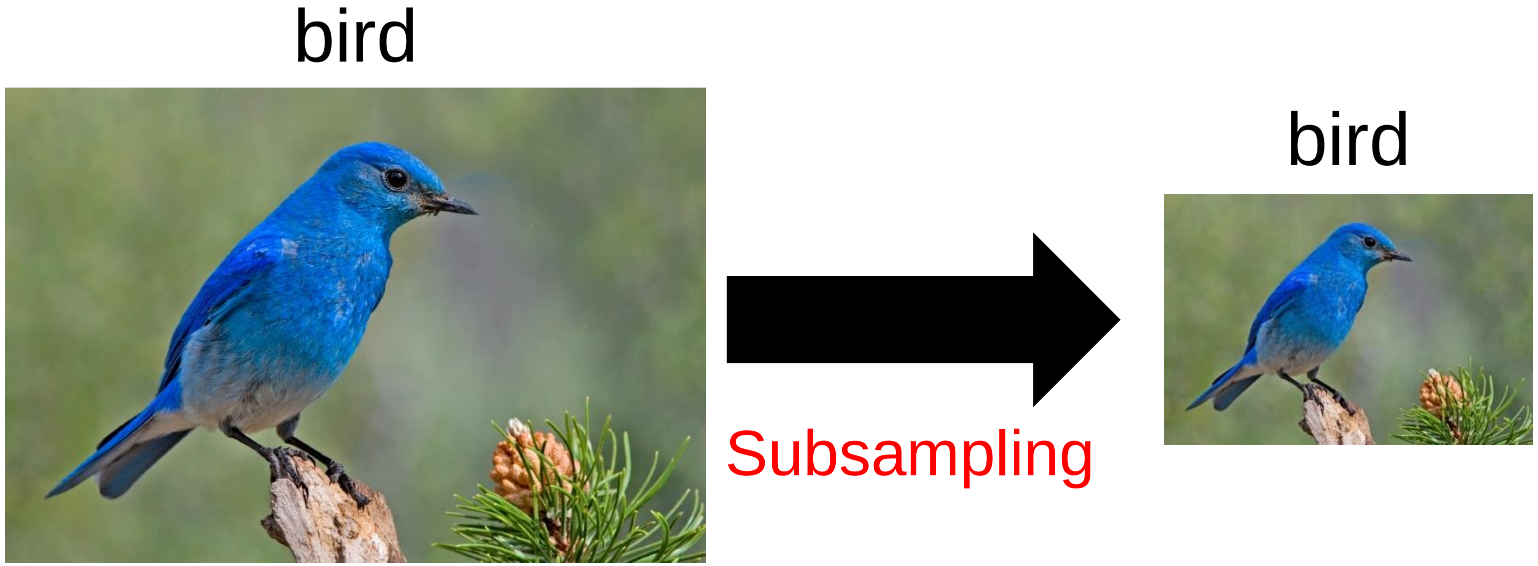
20%

Can we do with less information?

- **How much information we can throw away and still recognize the object?**

10%
20%
50%
75%

- **Subsampling pixels will not change the object**



bird

Subsampling

bird

We can subsample the pixels to make image smaller

→ **fewer parameters to characterize the image**

TIPO EXCELSIUS

CNNs Vs. ANNs

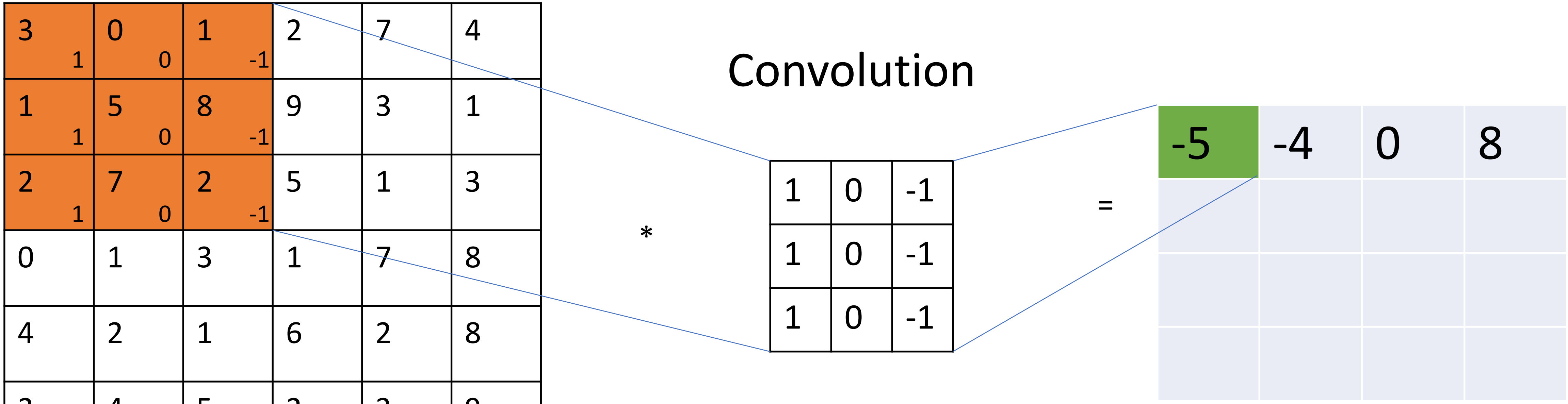
- ANNs suffer from curse of dimensionality when it comes to high resolution images
- We use filters (receptive fields) to exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers

- Parameter Sharing
- Sparsity of connection

Convolution

- Convolution is a pointwise multiplication of two functions to produce a third function.
- Primary purpose of convolution in CNN is to extract features from the input image.
- Matrix formed by sliding the filter over the image and computing the dot product is called the ‘Convolved Feature’ or ‘Activation Map’ or the ‘Feature Map‘.

Convolution Example



3 1	0 0	1 -1	2	7	4
1 1	5 0	8 -1	9	3	1
2 1	7 0	2 -1	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix (nXn)

*

Convolution

1	0	-1
1	0	-1
1	0	-1

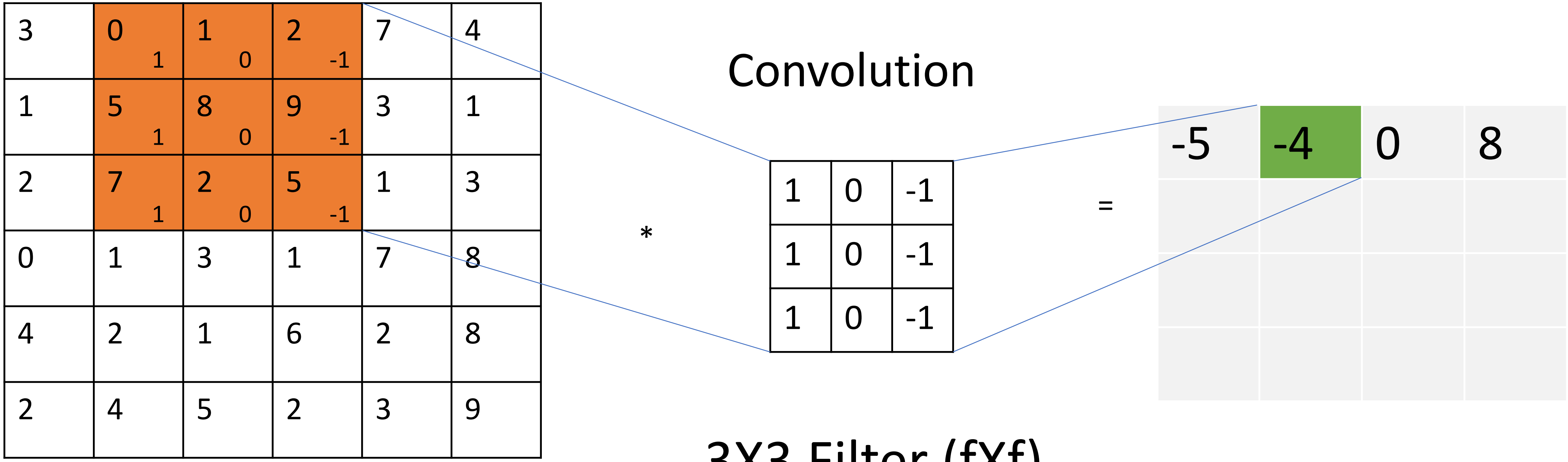
3X3 Filter (fXf)

=

-5	-4	0	8

(n-f+1)X(n-f+1)

Convolution Example



Convolution

3	0 1	1 0	2 -1	7	4
1	5 1	8 0	9 -1	3	1
2	7 1	2 0	5 -1	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix (nXn)

*

1	0	-1
1	0	-1
1	0	-1

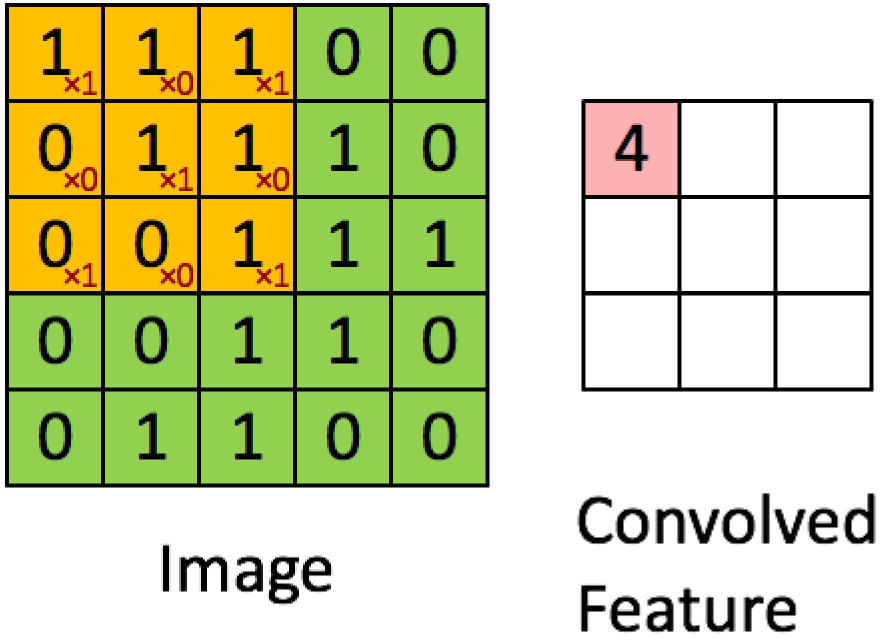
3X3 Filter (fXf)

=

-5	-4	0	8

(n-f+1)X(n-f+1)

Convolution Example



1 ×1	1 ×0	1 ×1	0	0
0 ×0	1 ×1	1 ×0	1	0
0 ×1	0 ×0	1 ×1	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved Feature

*http://ufldl.stanford.edu/tutorial/supervised/FeatureExtractionUsingConvolution/

Detecting Vertical edges

6*6 = 36

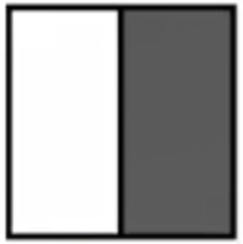
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

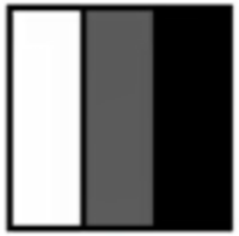
*

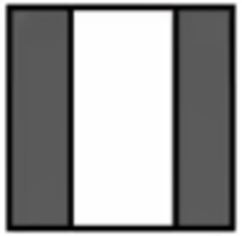
1	0	-1
1	0	-1
1	0	-1

4*4=16

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0







- In case of ANN # parameter to train = 36*16 = 576
- In case of CNN # parameter to train = 9

Filter Weights

1	1	1
0	0	0
-1	-1	-1

Horizontal Filter

1	0	-1
2	0	-2
1	0	-1

Sobel Filter

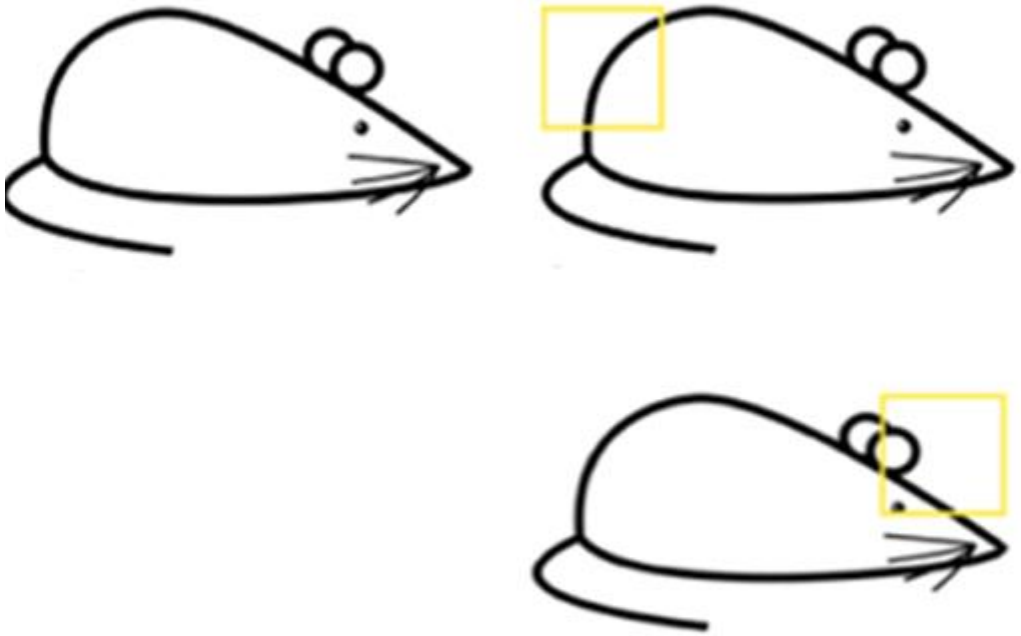
3	0	-3
10	0	-10
3	0	-3

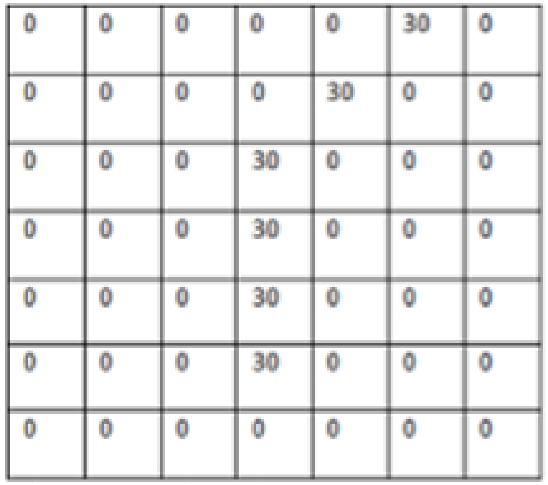
Schorr Filter

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Convolutional Neural Networks automatically estimates the weights of the filter

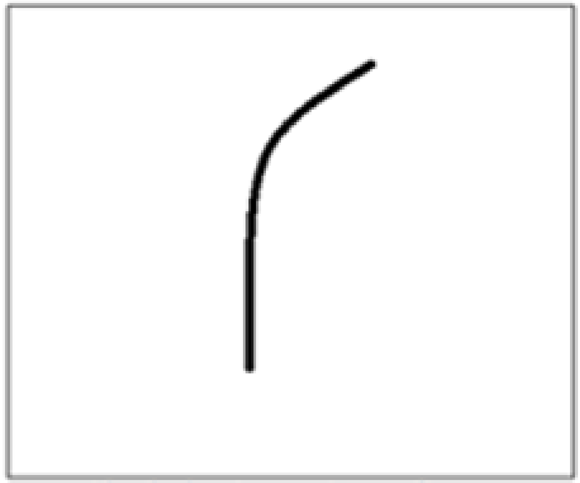
More Intuition





0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

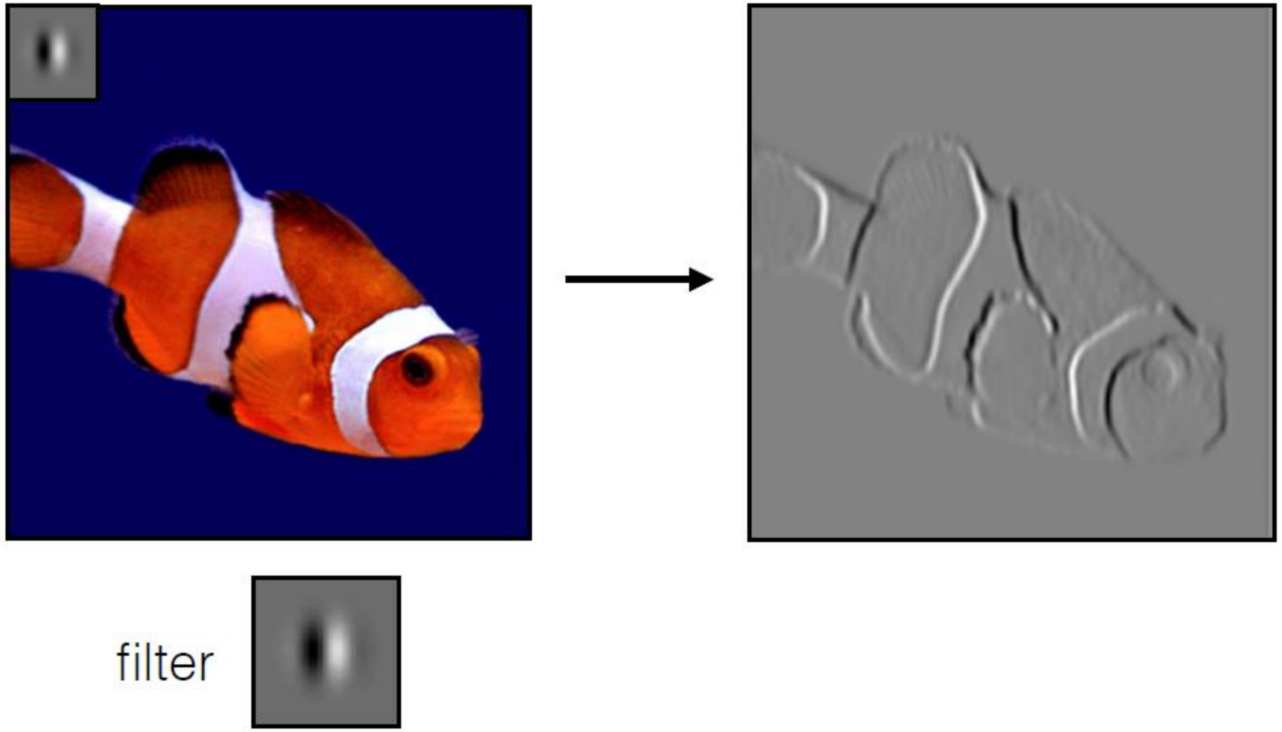


Visualization of a curve detector filter

Interpretation

Convolution is just another way of computing W^TX

In CNN, **input** is **image**, **kernel** is **convolution filter** to be learned, **response** is the **feature map**

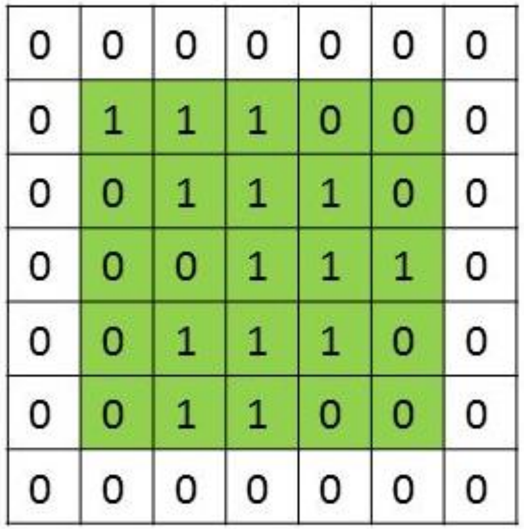


Padding

Padding is used to preserve the original dimensions of the input

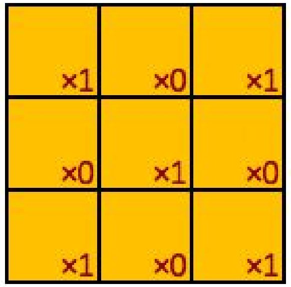
Zeros are added to outside of the input

Number of zero layers depend upon the size of the kernel

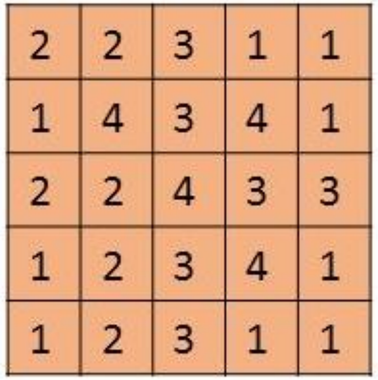


0	0	0	0	0	0	0
0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0

5X5 (with padding)



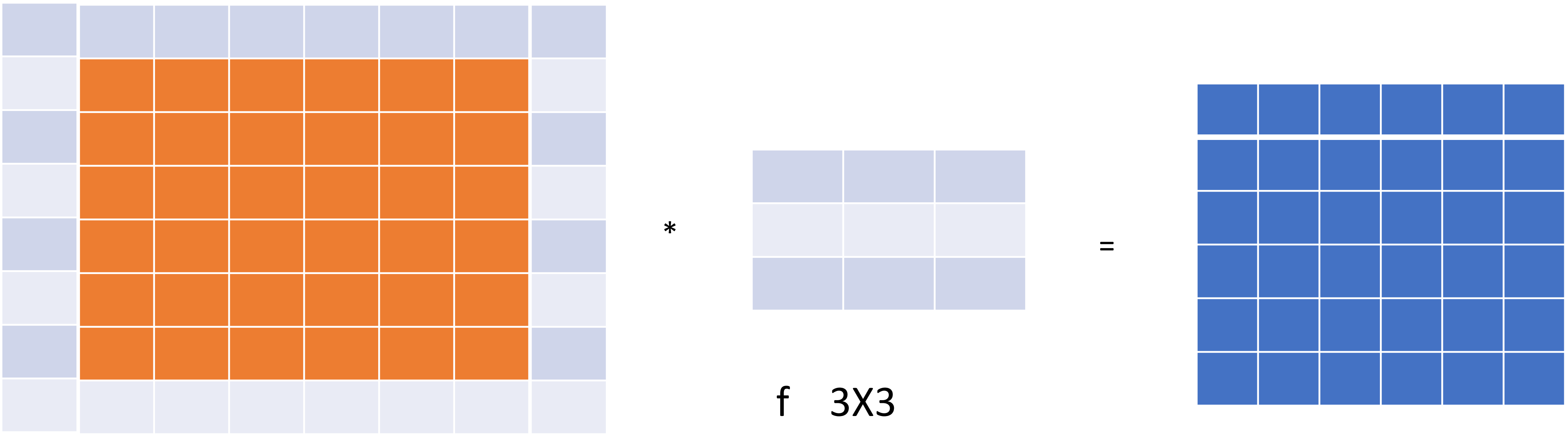
×1	×0	×1
×0	×1	×0
×1	×0	×1



2	2	3	1	1
1	4	3	4	1
2	2	4	3	3
1	2	3	4	1
1	2	3	1	1

5X5

Padding



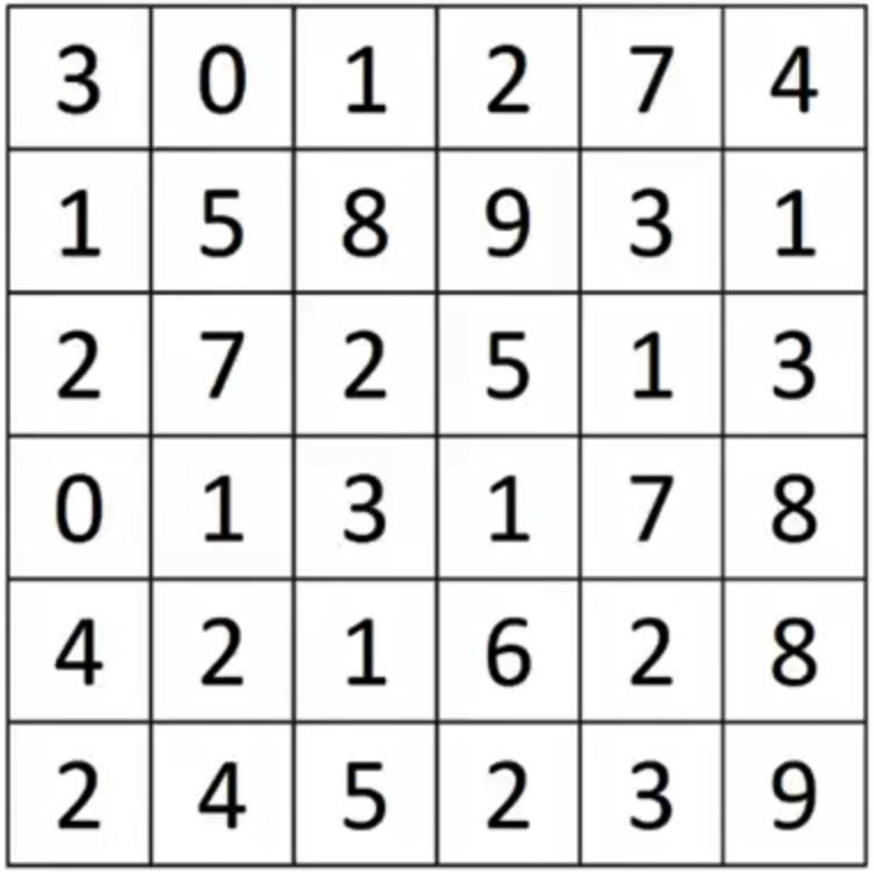
nXn 6X6 to 8X8 Padding=1

(n-f+1)X(n-f+1) to (n+2p-f+1)X(n+2p-f+1)
Valid to same

Stride=s

Floor$(\frac{n+2p-f}{s}+1)$ XFloor$(\frac{n+2p-f}{s}+1)$

Stride



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix

1	0	-1
1	0	-1
1	0	-1

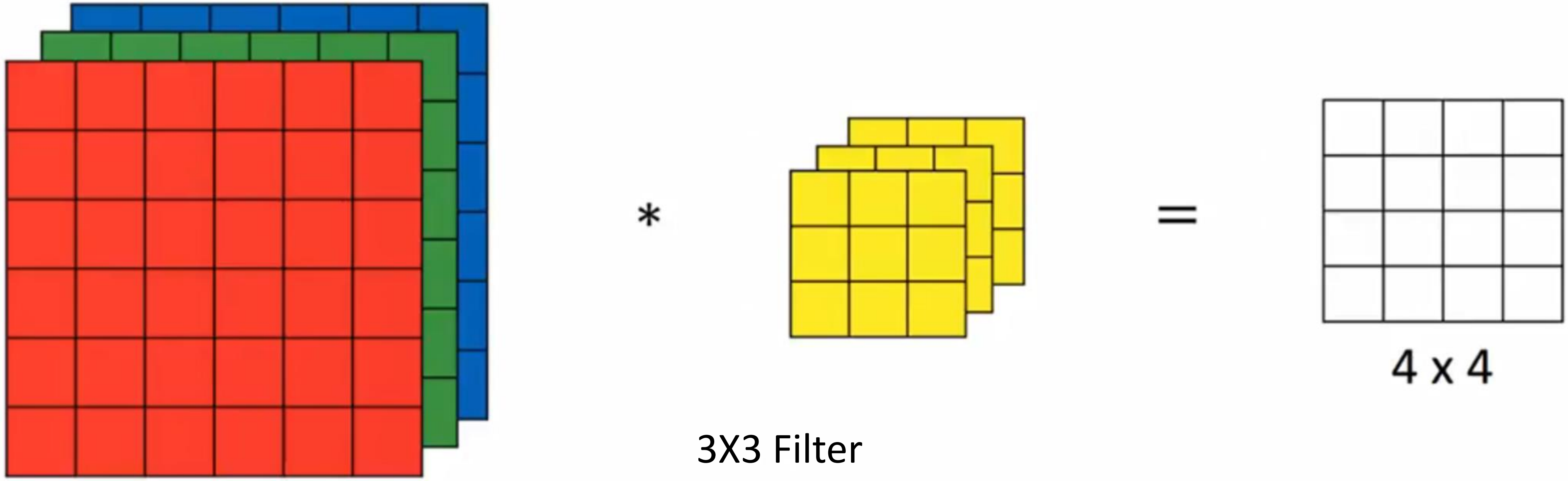
3X3 Filter

-5	8
-3	-16

2X2

Stride=s (3 Here) Floor($\frac{n+2p-f}{s}+1$) XFloor($\frac{n+2p-f}{s}+1$)

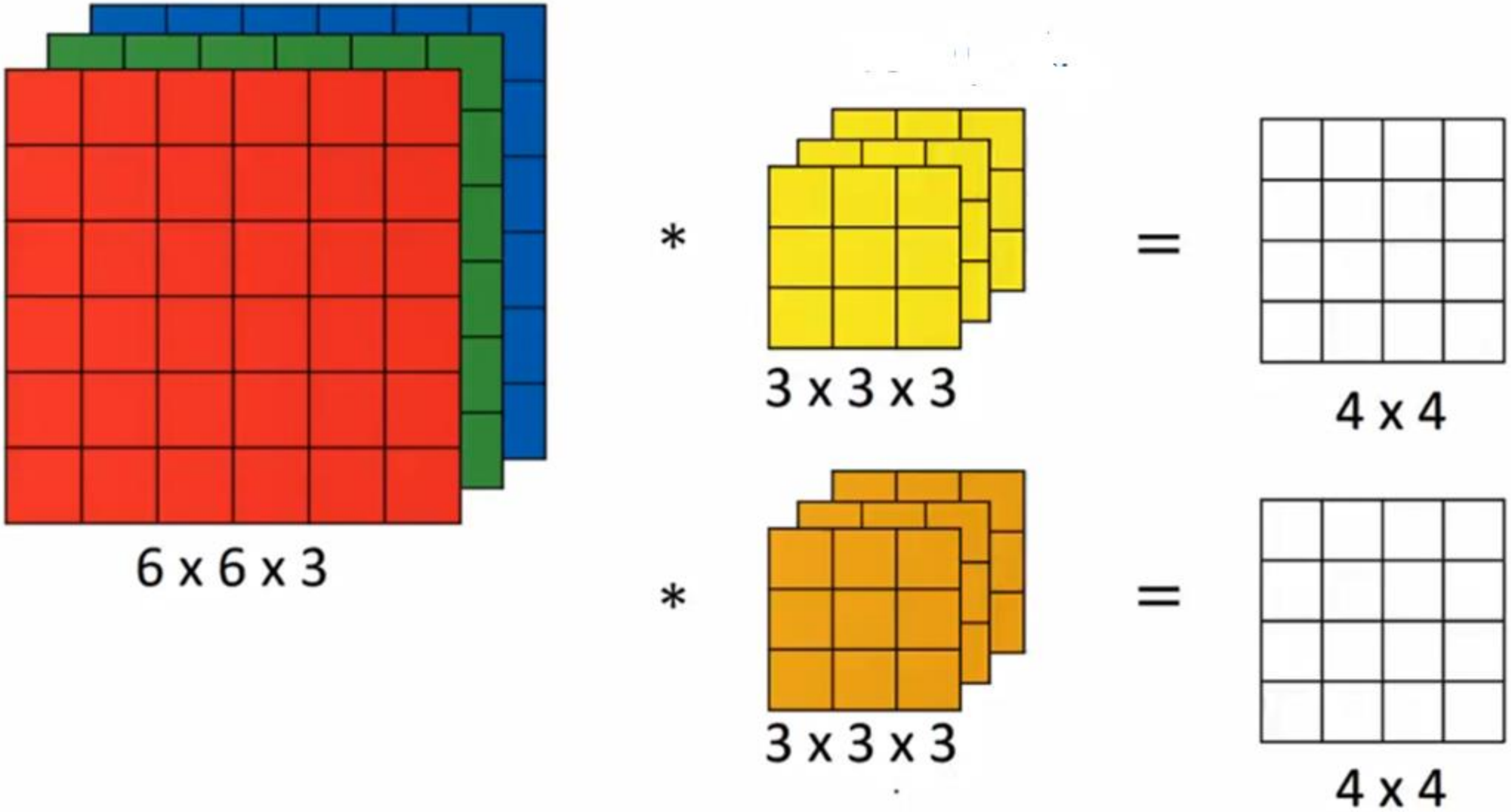
Channels



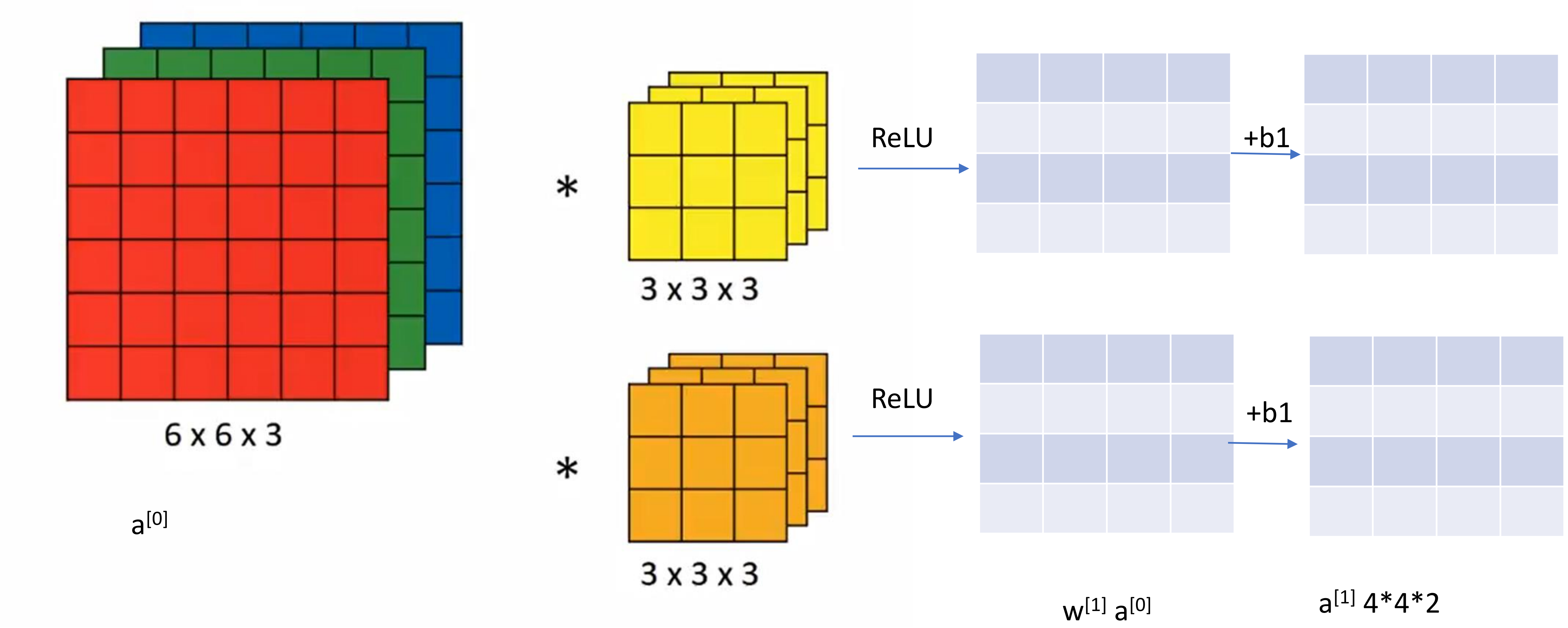
3X3 Filter

6X6 Matrix

Padding =0 Stride=1



$nXnXn_c$ $fXfXn_c$ $n-f+1Xn-f+1Xn_{c'}$ c'=no of filters



$w^{[1]}$ 2 filters means two units here

Pooling

1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

Max Pooling : One example of pooling layer

8	9
9	4

f=2

s=2   4X4 converted to 2X2

Function of Pooling is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network.

Pooling

1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

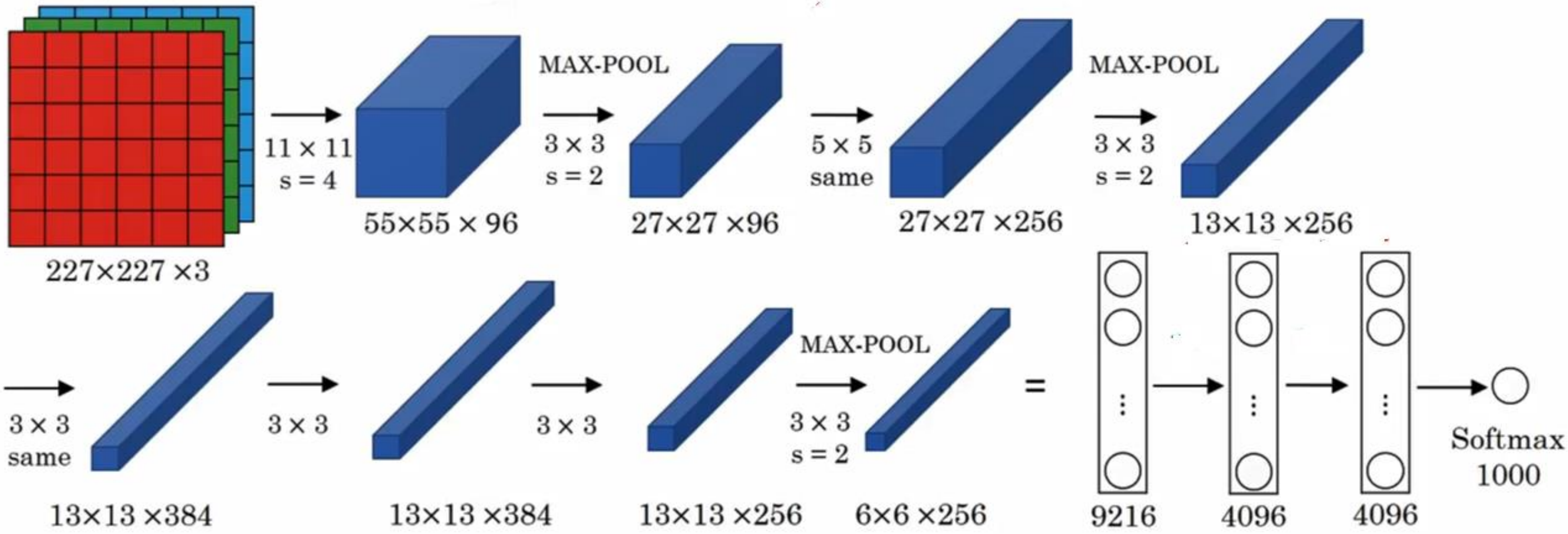
3.5	6.25
4.5	2.5

Average Pooling : Another example of pooling layer

f=2

s=2   4X4 converted to 2X2

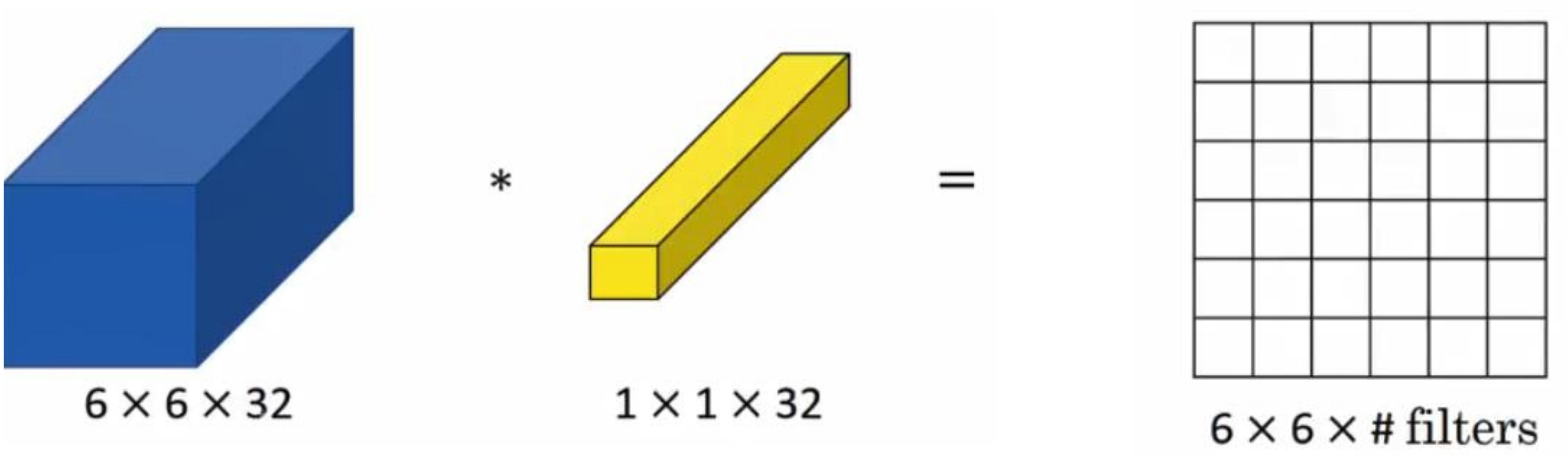
AlexNet


227×227 ×3
11 × 11
s = 4
55×55 × 96
MAX-POOL
3 × 3
s = 2
27×27 ×96
5 × 5
same
27×27 ×256
MAX-POOL
3 × 3
s = 2
13×13 ×256
3 × 3
same
13×13 ×384
3 × 3
13×13 ×384
3 × 3
13×13 ×256
MAX-POOL
3 × 3
s = 2
6×6 ×256
=
9216
4096
4096
Softmax
1000

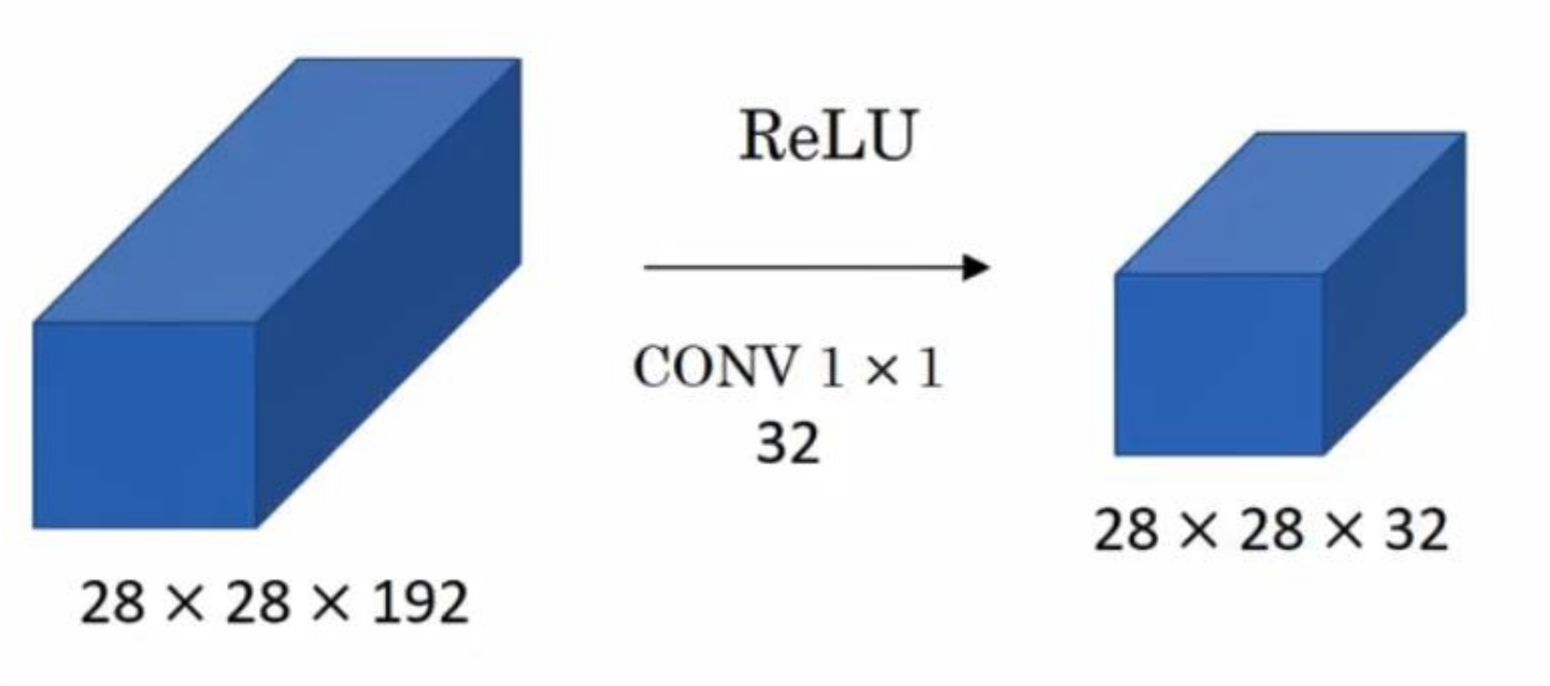
Famous CNN Models

- LeNet - 1990
- AlexNet - 2012
- ZFNet - 2013
- GoogLeNet - 2014
- VGGNet - 2014
- ResNet - 2015
- Inception v3 - 2016
- MobileNet – 2017
- Squeezenet - 2017

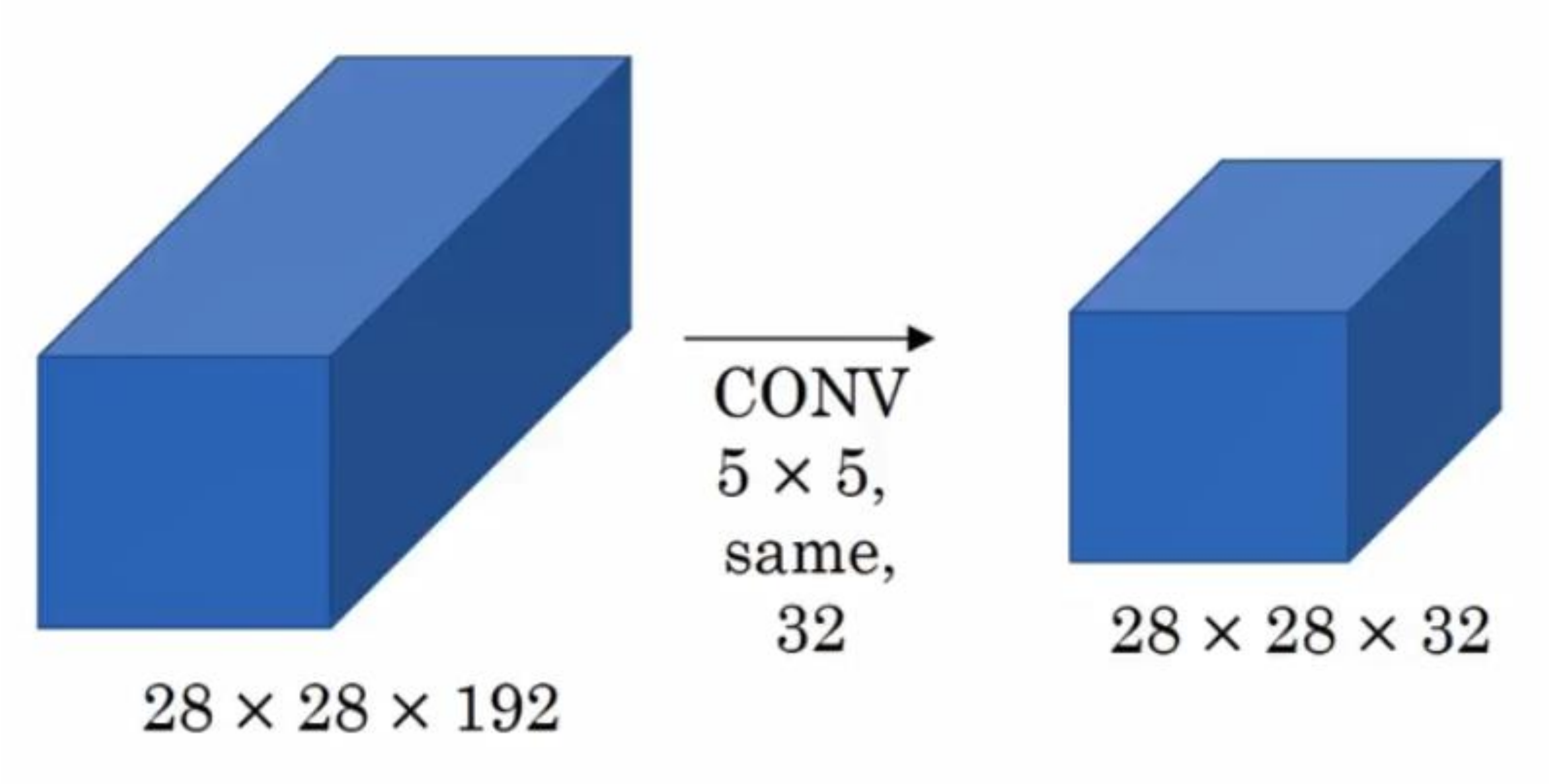
Innovative use of 1X1 Convolution Filter



Shrinking of Channels

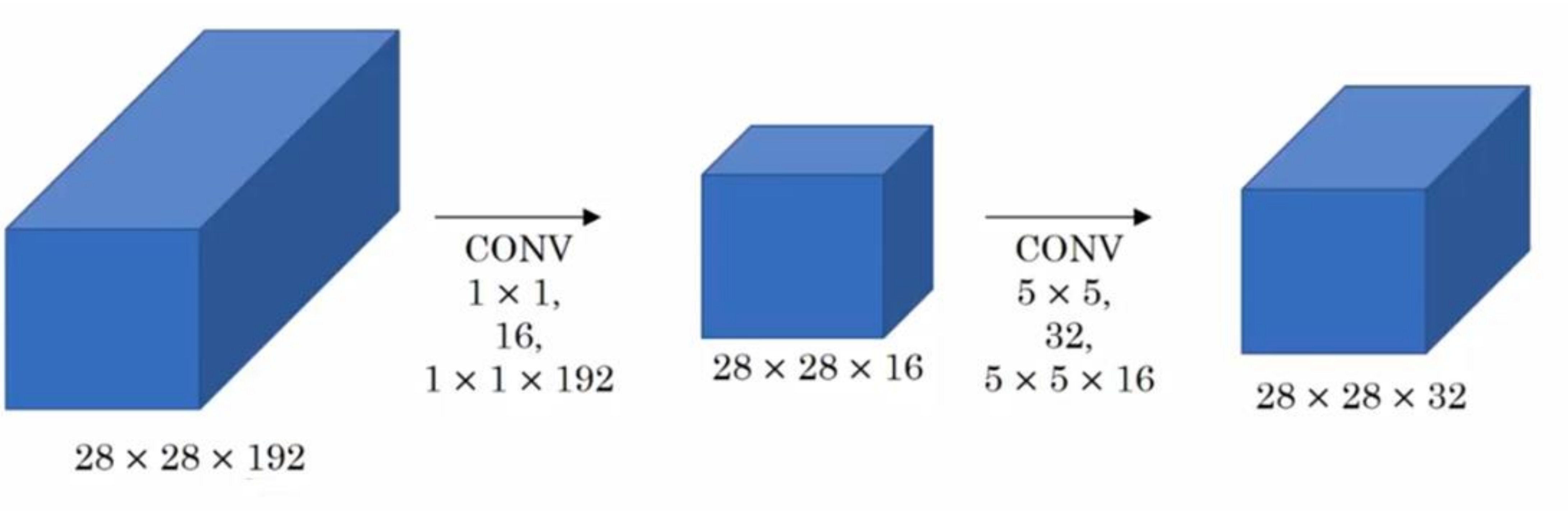


Problem of Computational Cost



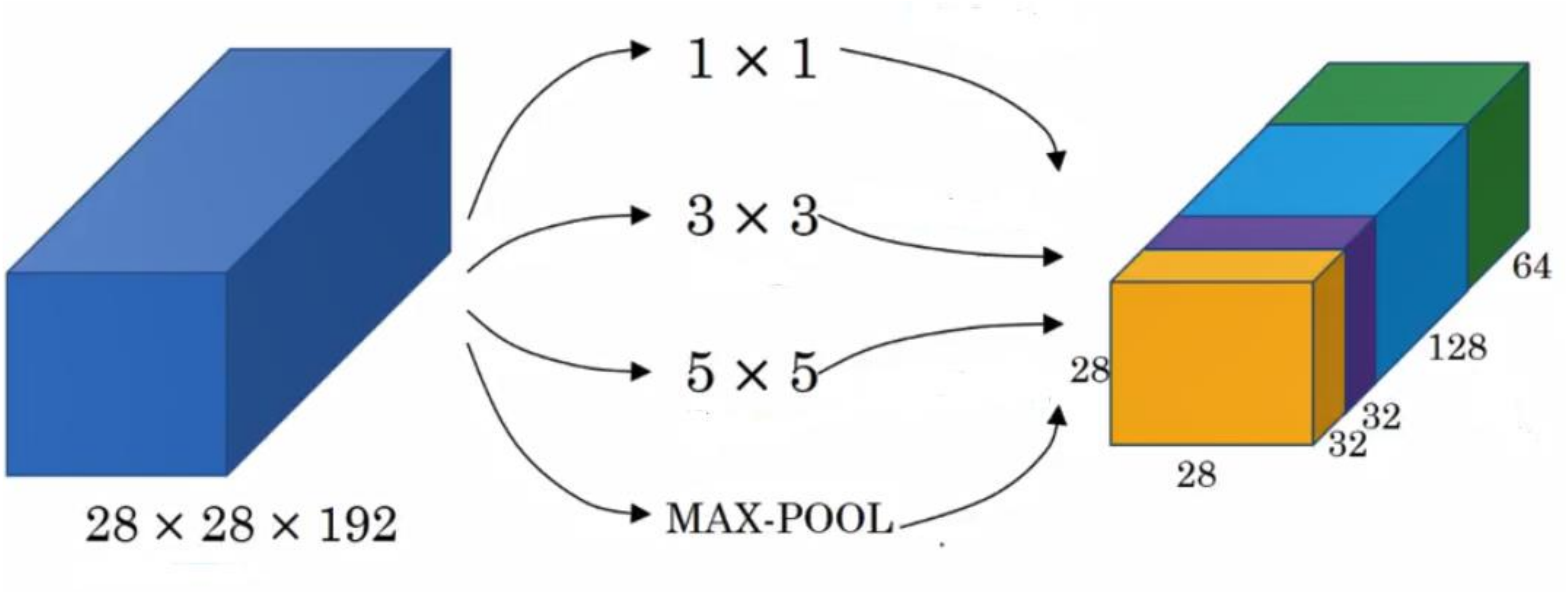
28X28X32X5X5X192=120 Million Calculations

Using 1X1 Convolution to reduce the cost

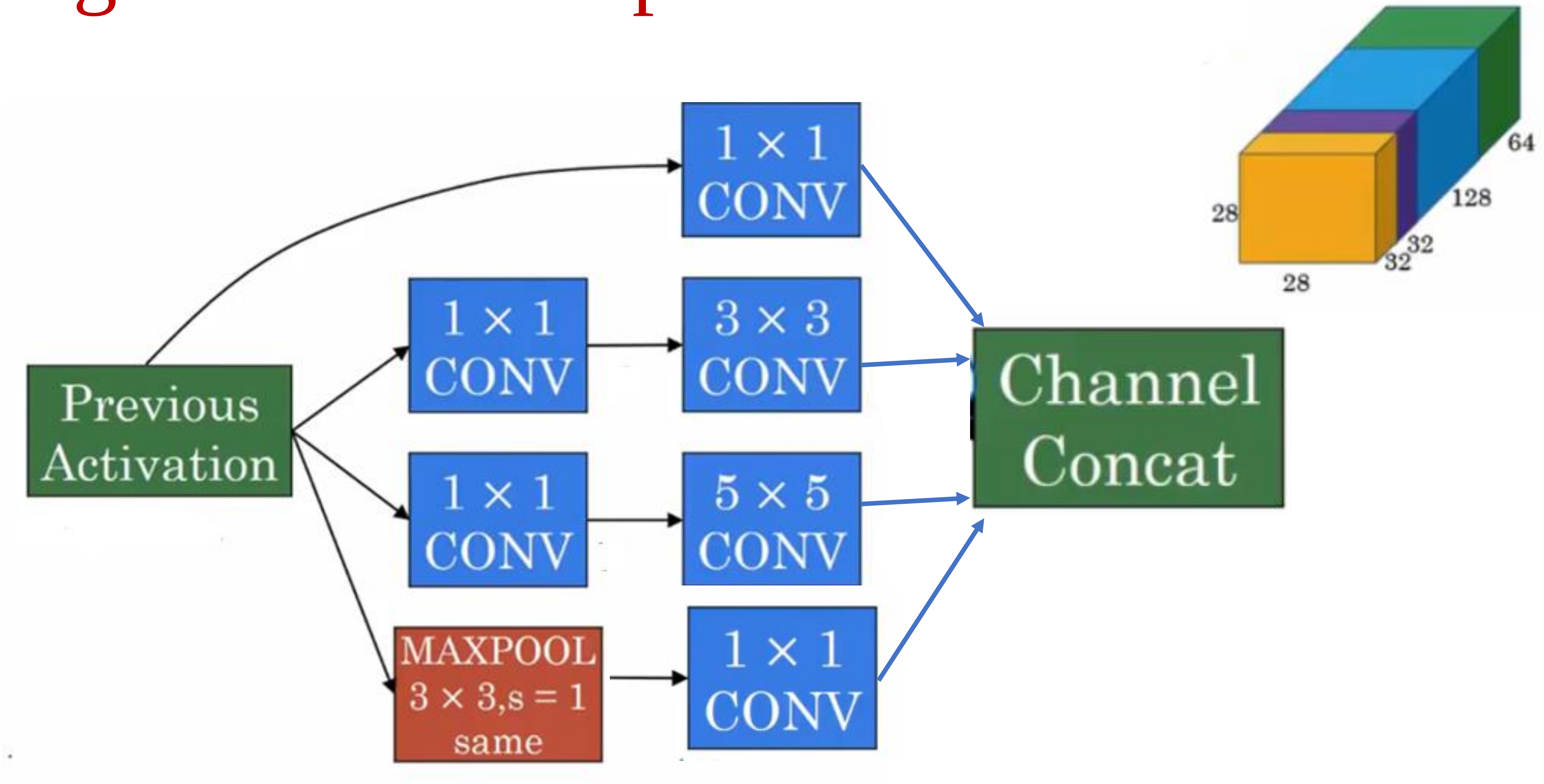


28X28X16X192 + 28X28X32X5X5X16 = 12.4 Million Calculations

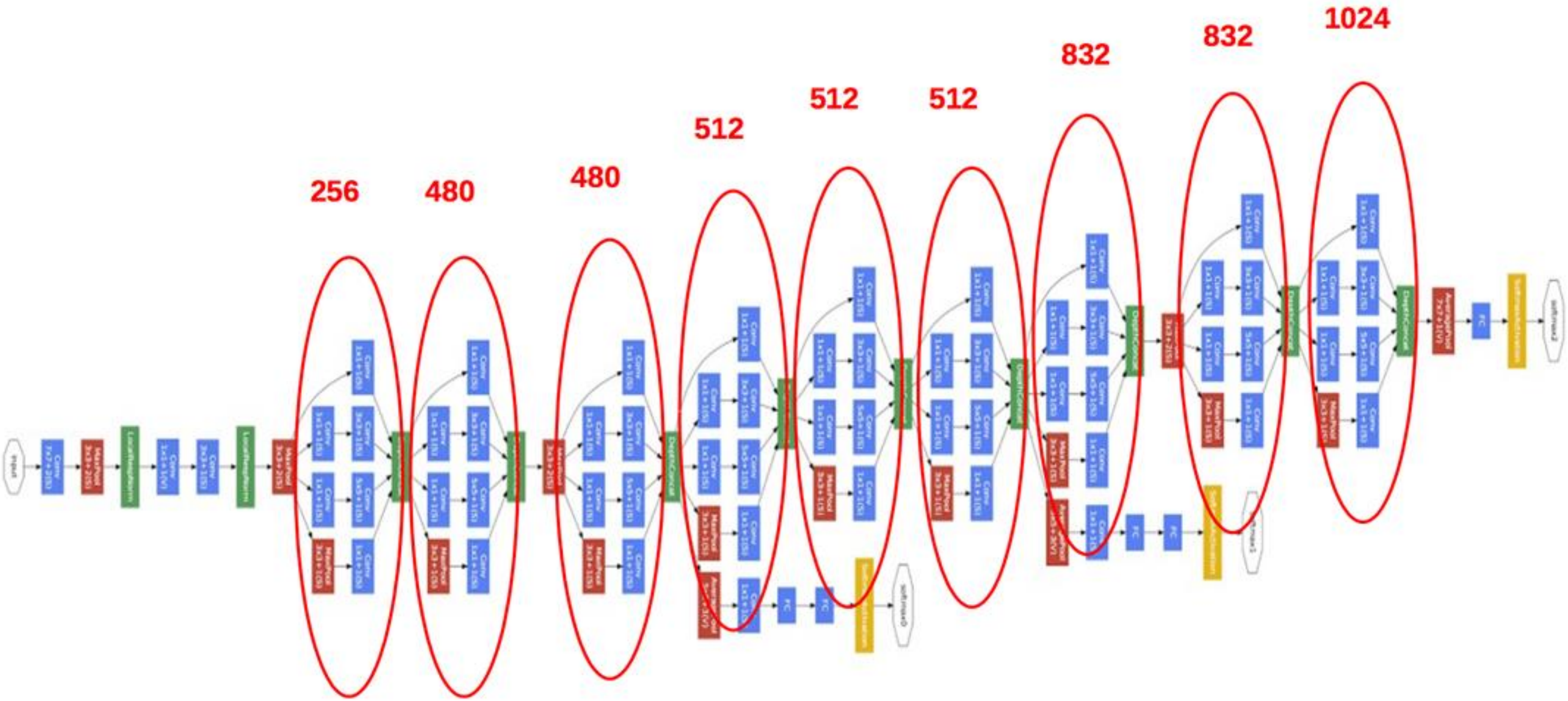
Inception Network



Single Unit of Inception Network



Inception Network and Inception Block



Open Source Implementations

-Developers has spent months in developing some good open source networks

-Many of them have access to high-speed GPUs and have the capacity to do lot of experimentation for months

-It is always good to start with a known open source implementation

-Search <name of the network> Github

-you will get few links pointing you to Github profiles of individuals who have put these network models along with the learned weights online

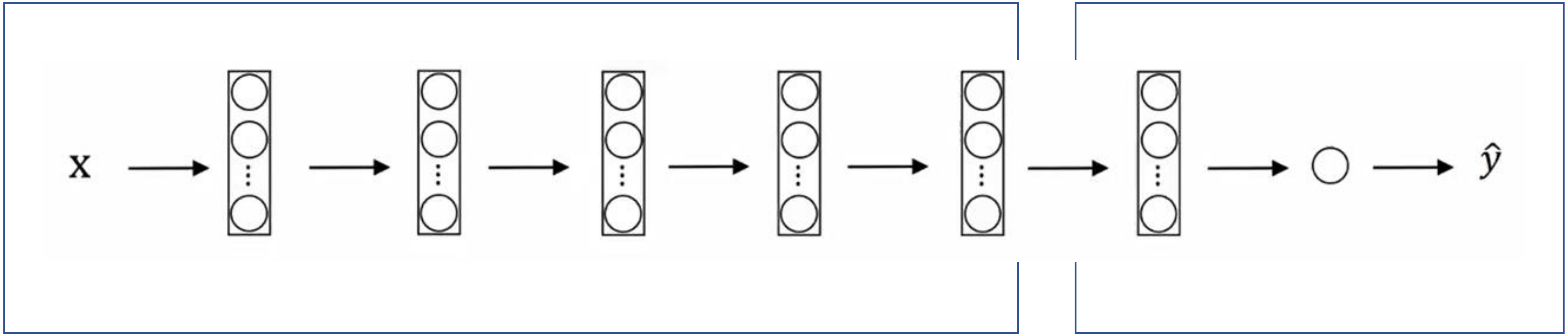
-Download or clone these network depending upon whether you are working on the your machine or remotely on a notebook

-Use this trained model fully or partially depending upon your requirements

Transfer Learning

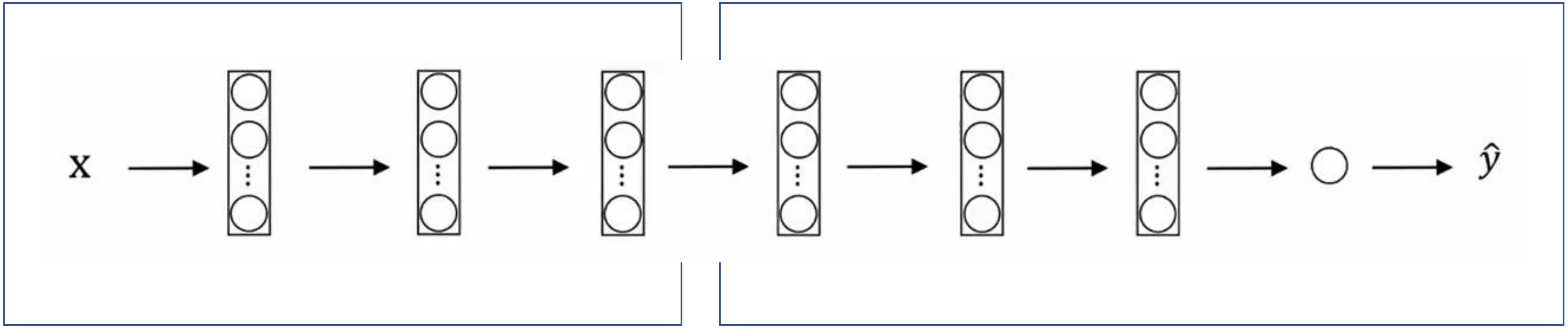
- Sharing the Knowledge gained solving one problem and applying to a different but related problem
- Transfer Learning is next popular driver of deep learning after supervised learning
- The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages.
- The marginal probability distributions of source and target domain are different, e.g. the documents discuss different topics.
- Simulation training is becoming a hot area within the sphere of Deep Learning. Few labs have also started using AR/VR Technologies to be integrated for making advance learning models for some of the critical problems area

Transfer Learning – Small Data



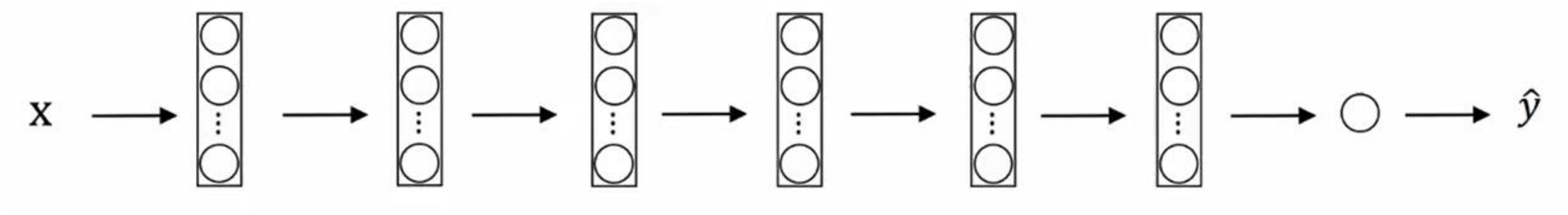
*Finding a Bengal Cat or British Cat where you have small dataset of these categories of cats
*You can download a cat classifier and train last few layers on your data specifically
*In Popular Deep Learning platforms, now you have good support for transfer learning
* Functions like Trainable_Parameters and Freezing specific layers are available

Transfer Learning-Mid Size Data



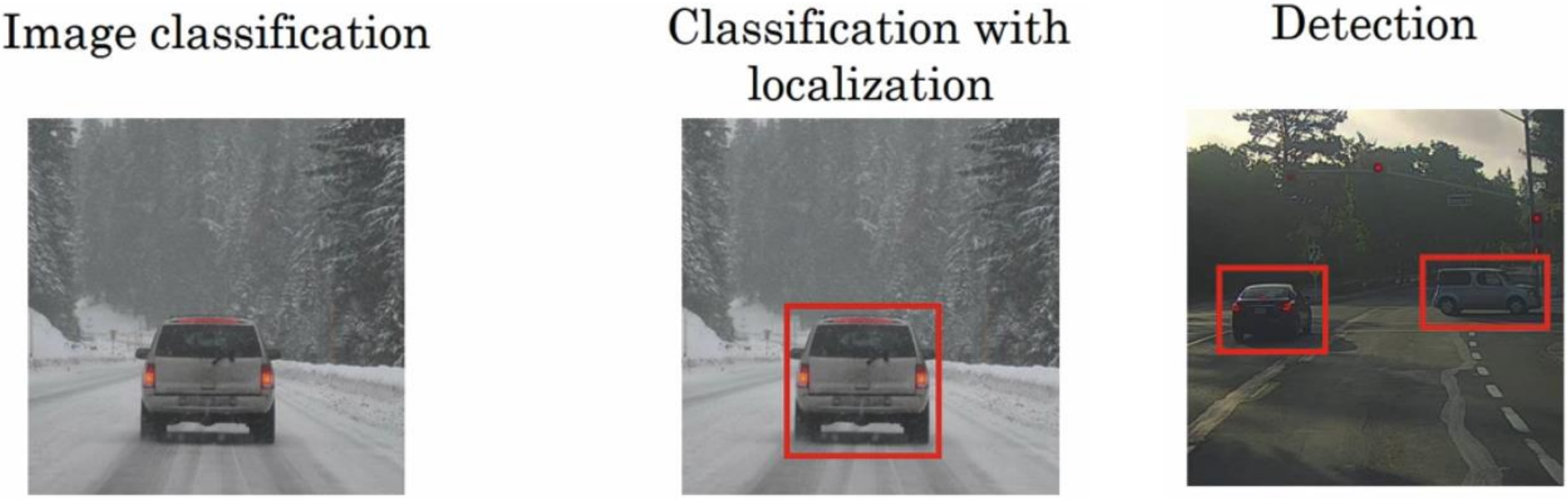
In this category we use initial set of layers from the open source model and use the trained weight values. For the remaining layers there are two ways to handle: a) either we can train the layers from start or b) we can start the weight optimization from the existing set of weights that we have received from the open source model

Transfer Learning- Enough Data



In scenarios, where we have enough data to train our new model, open source model may still be useful. We can use the pre-trained weights of the model from the same domain and consider that as a starting point for our model. It may be much easier and faster to adapt these model for new set of data that we want to train upon.

Object Localization and detection



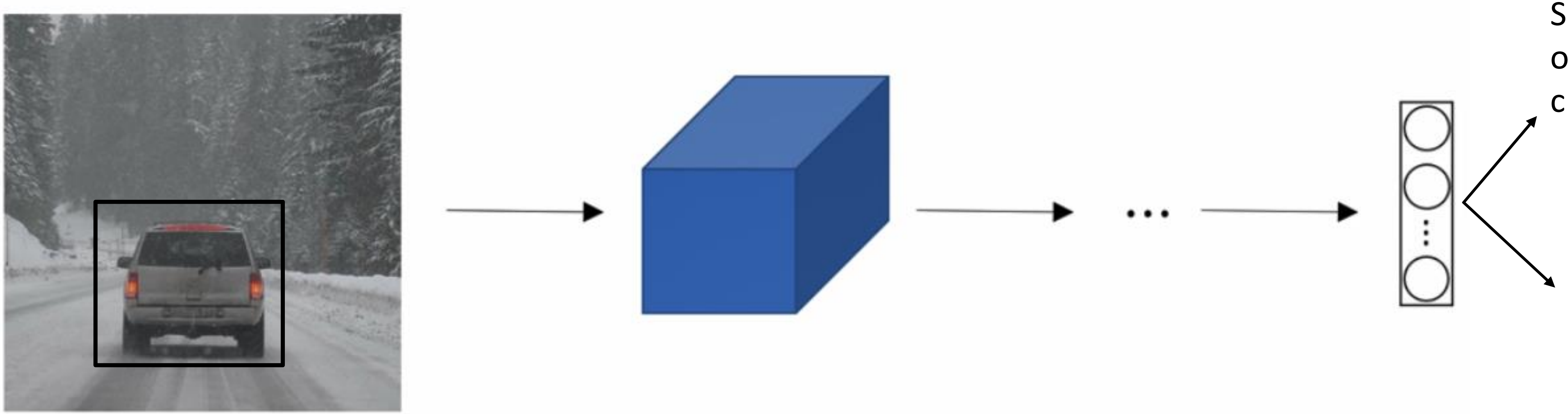
Car or Not

Usually Single Object

Multiple Objects

Classification with localization

Pedestrian
Car
Motorcycle
background



Softmax of four classifiers

bx,by, bh,bw

Defining the target label

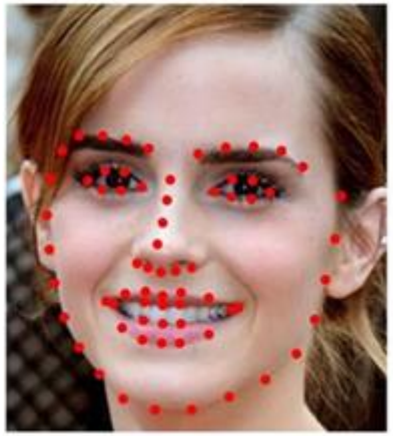
Presence of an object(Y/N)	1	0
b_x	b_x	-
b_y	b_y	-
b_h	b_h	-
b_w	b_w	-
c_1	0	-
c_2	1	-
c_3	0	-

$$L(\hat{y},y)=(\hat{y}_1-y_1)^2+(\hat{y}_2-y_2)^2+(\hat{y}_3-y_3)^2+(\hat{y}_4-y_4)^2$$
$$+(\hat{y}_5-y_5)^2+(\hat{y}_6-y_6)^2+(\hat{y}_7-y_7)^2+(\hat{y}_8-y_8)^2$$

else

$$(\hat{y}_1-y_1)^2$$

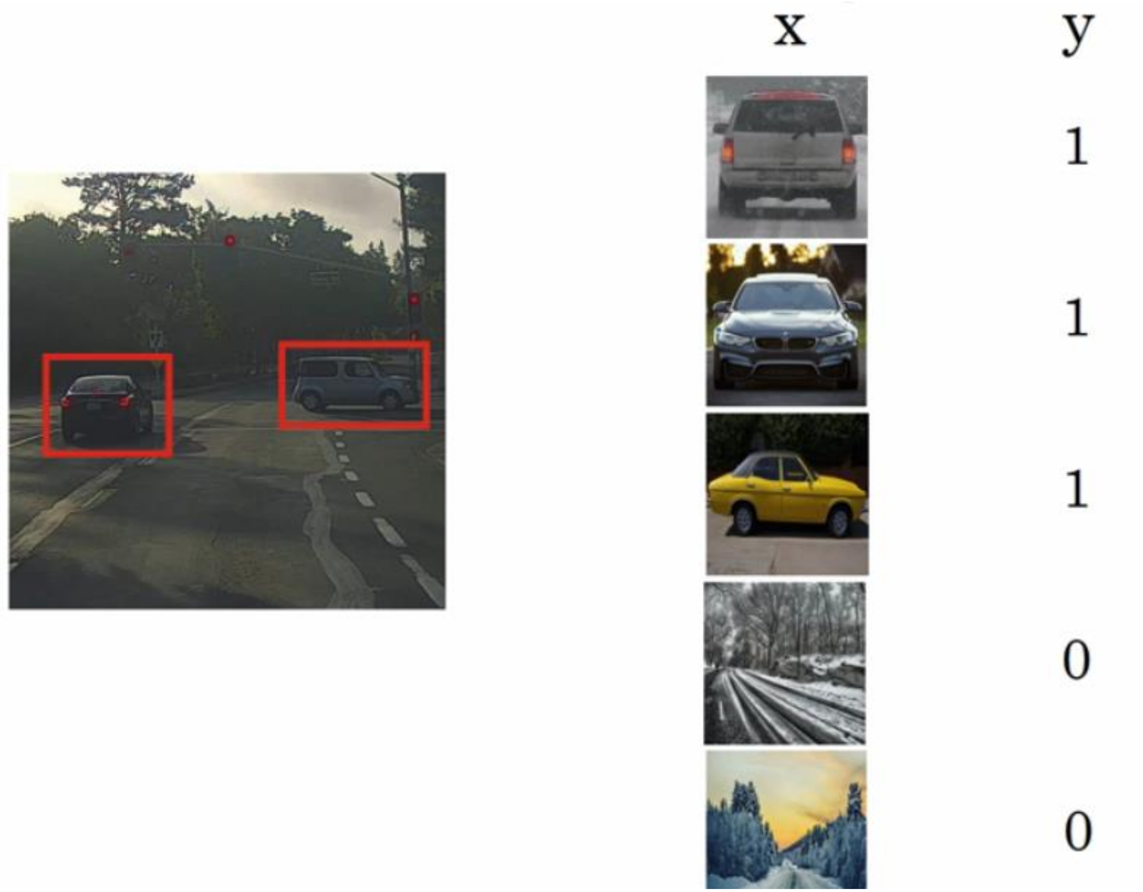
Landmark Detection



Order/Sequence of the landmarks is predefined

Face? (Y/N)
$L1_{X,}L1_{Y}$
$L2_{X,}L2_{Y}$ $L3_{X,}L3_{Y}$:
:
$L64_{X,}L64_{Y}$

Car Detection

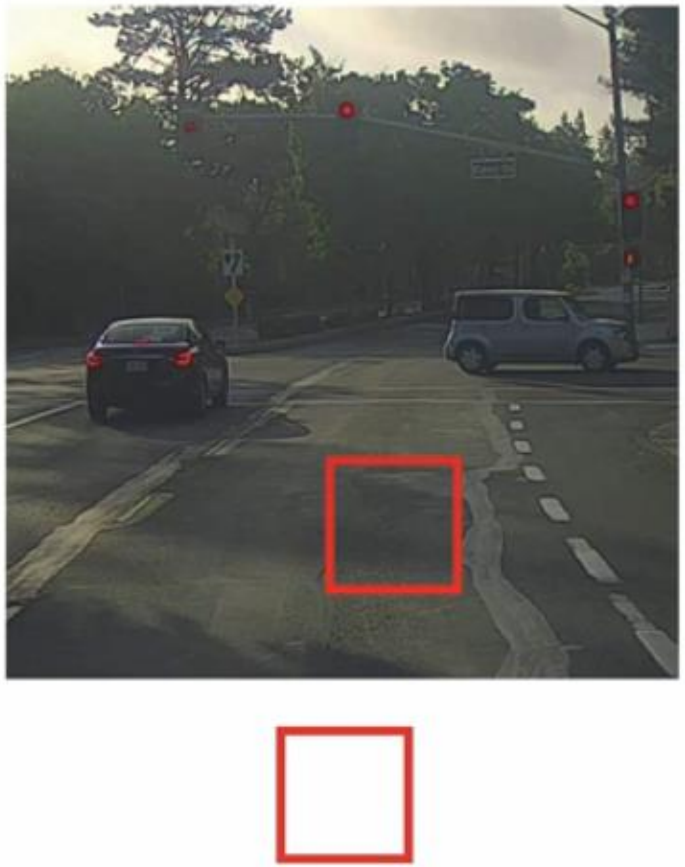


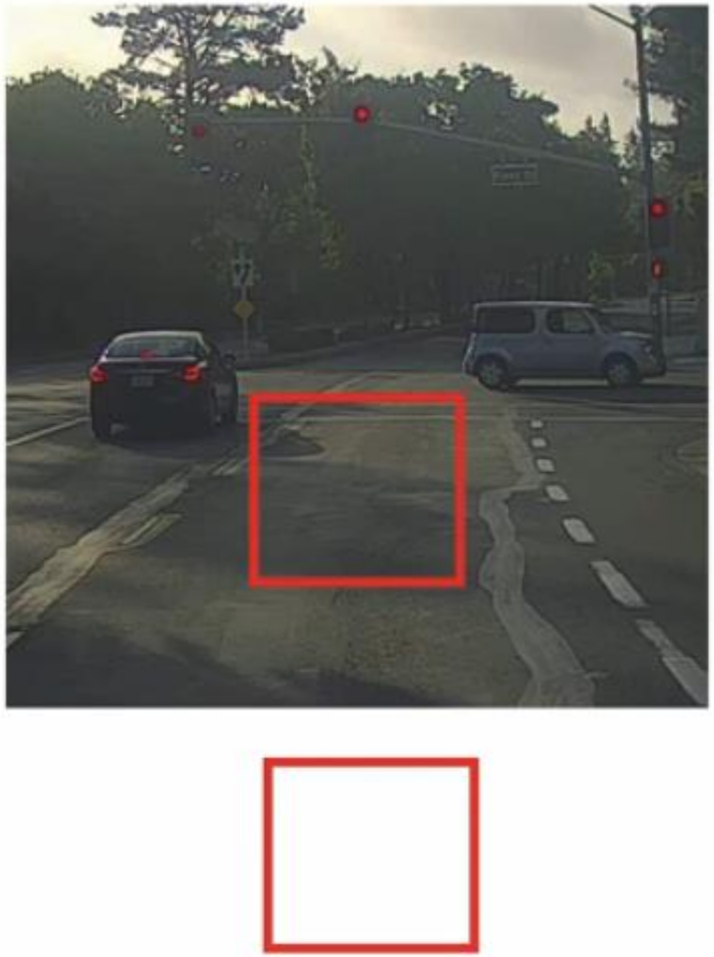
Conv Net

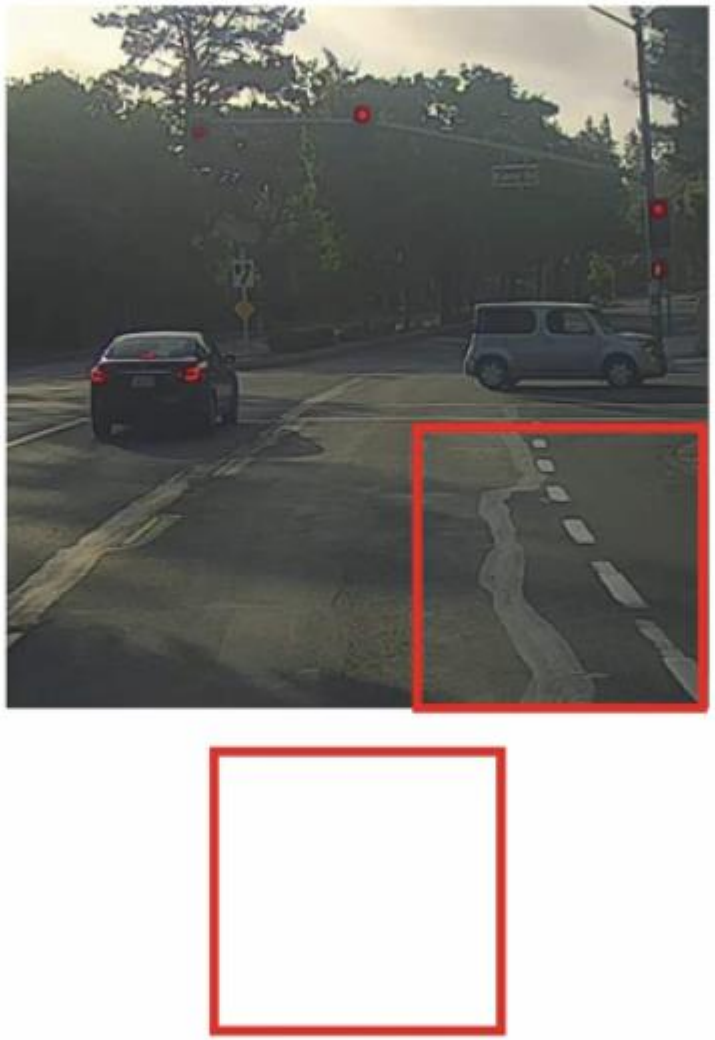
Y

Having Closely Cropped Images to train the System

Sliding Windows Detection







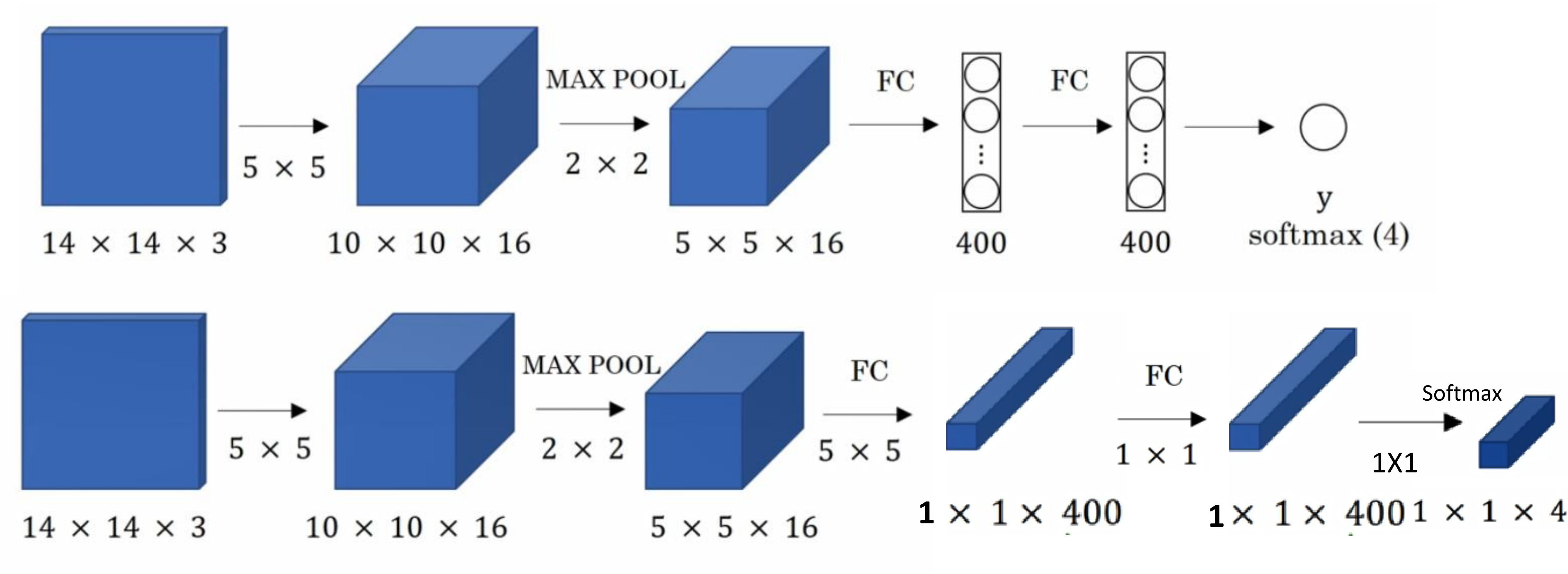
Sliding Window Detection

Each Iteration for each of the Sliding Window Box needs a convnet to classify and localize the object.

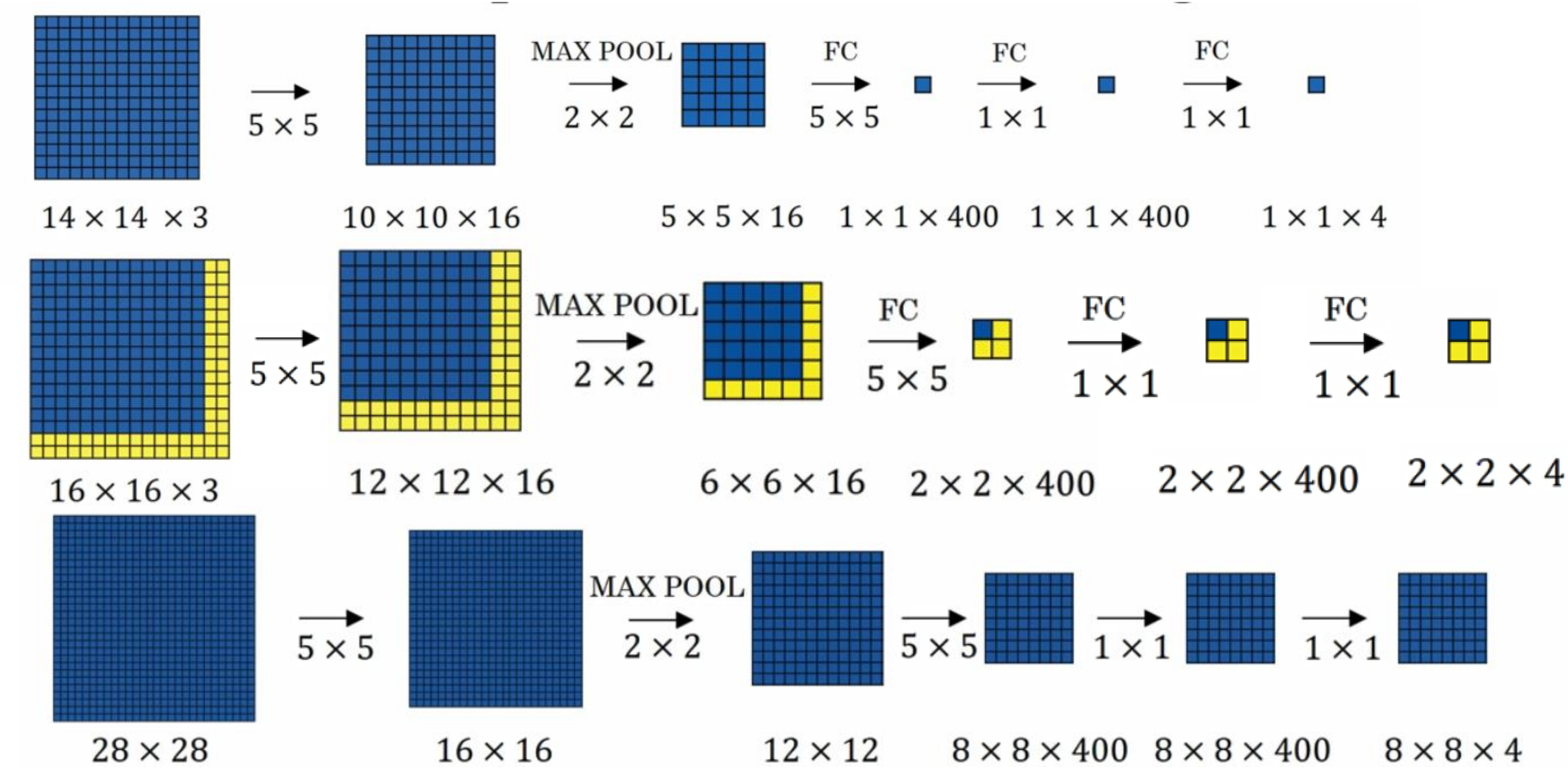
Computation Cost is Huge.

We have a solution

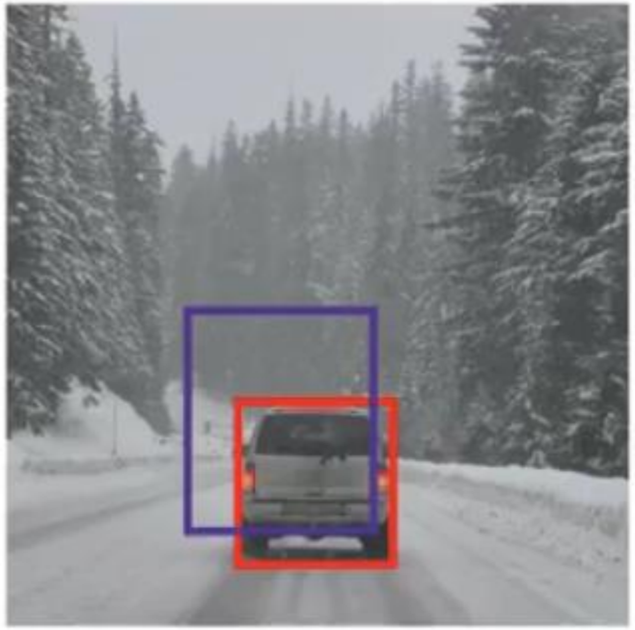
Turning Fully Connected Layer to Convolution Layers

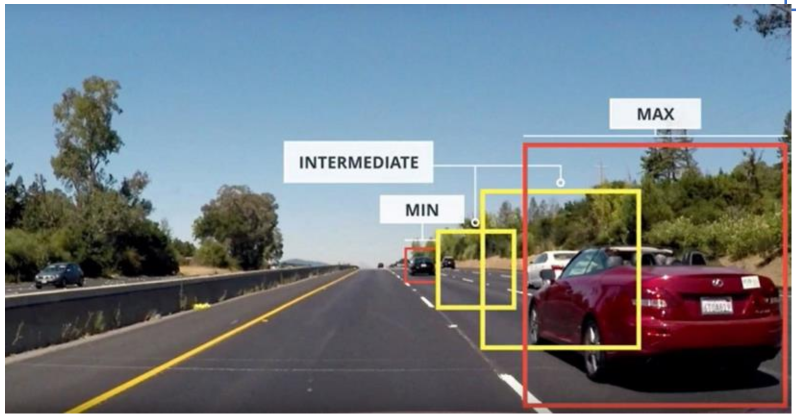


Convolutional Implementation of Sliding Windows



Intersection Over Union



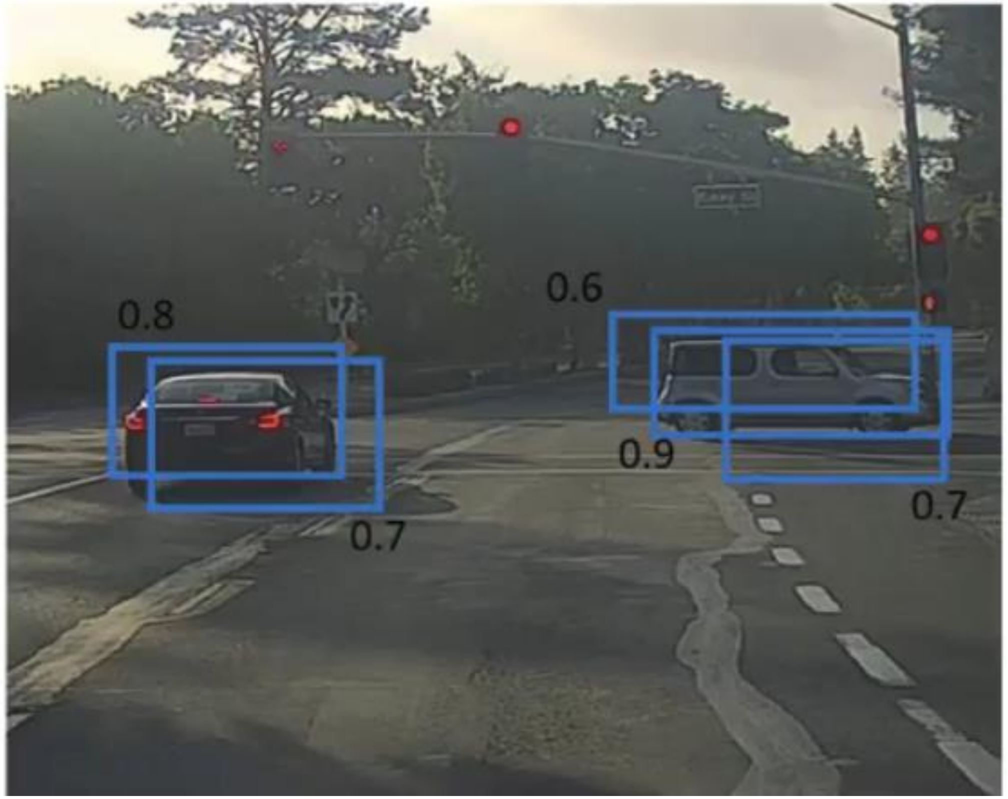


Intersection Over Union

Size of Intersect Area/Size of Union Area
May be Correct if IoU ≥ 0.5

So, it is a measure of the overlap between two bounding boxes

Non-Max Suppression



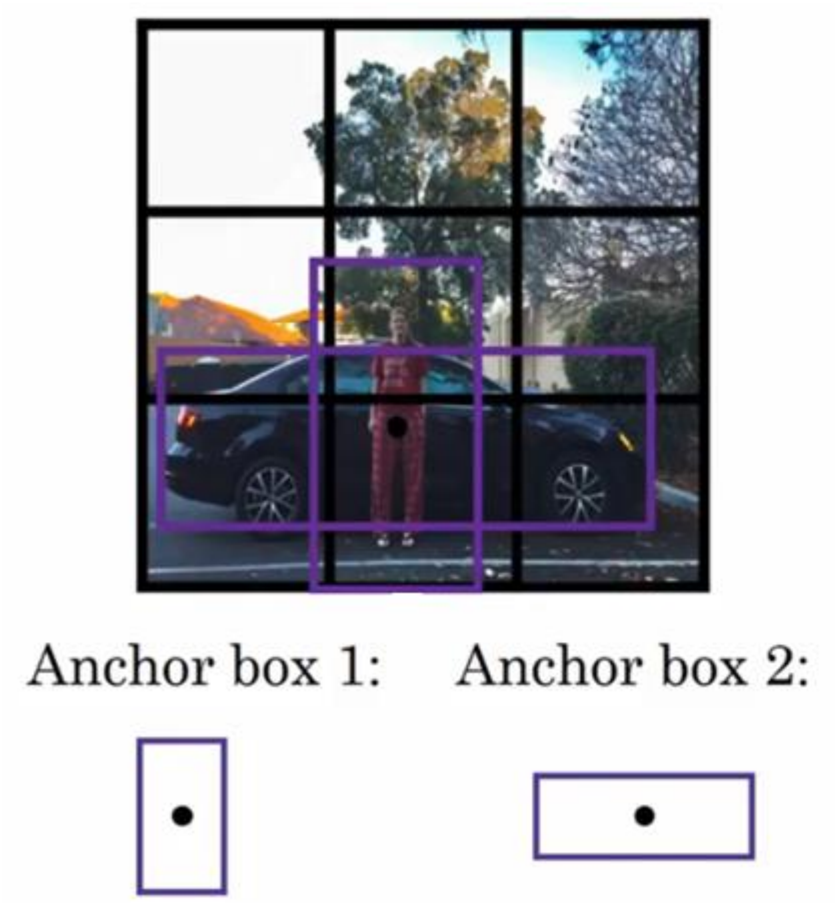
It Cleans up multiple bounding boxes around one object and gives us one bounding box per object.

Each Output (Probabilities associated with multiple bounding boxes is Pc)

```
Discard all boxes with Pc ≤ 0.6
While There are any remaining boxes
        Pick the box with the largest Pc
        Discard any remaining box
```

For Multiple objects we can repeat Non-max suppression algorithm for each of the object we are trying to find out like pedestrian, motorcycle.

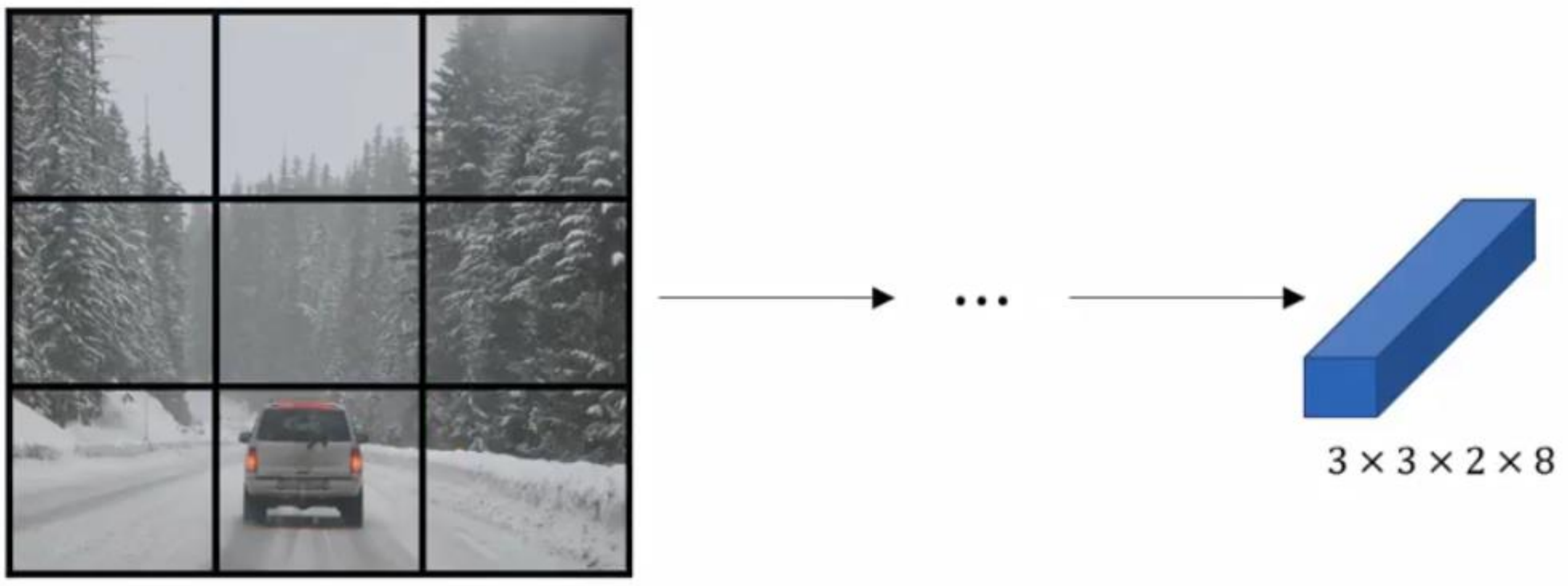
Anchor Box for overlapping objects



Each object in the training image is assigned to grid cell that contains objects midpoint and anchor box for the grid cell with Highest IoU

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Yolo Algorithm (combines all the pieces together)



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Each Grid Cell will produce a 16 size output vector

2 is the Number of Anchor boxes and 8 is the no of output points in the given vector for each object to be detected

Face Recognition Vs. Verification

Verification

Input an image and also the Name/ID of the person

Output whether the input image is of the claimed person

Validation

Input an Image

Output whether the image is any of the K persons in the database

One Shot learning

Where you have only one image for training the system (learning from one example to recognize the person again)

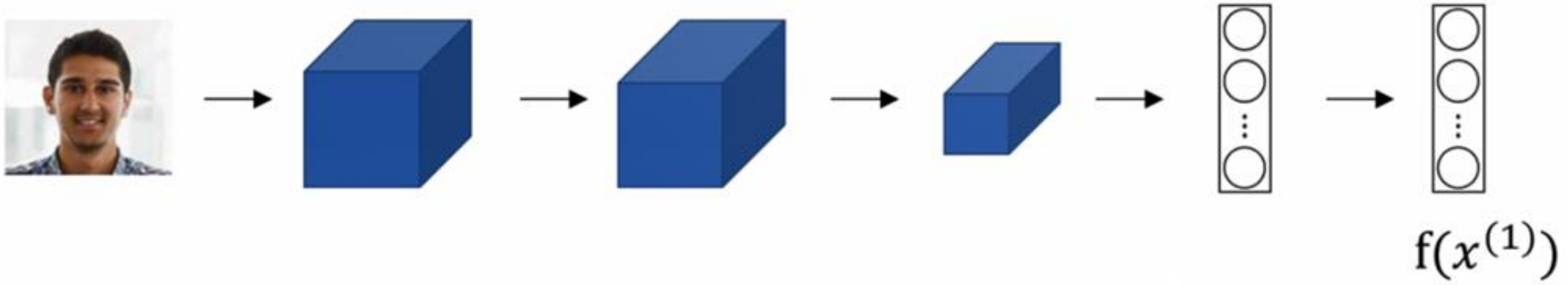
For that You need to learn a similarity function

d(img1, img2) = degree of difference between images

if $d(\text{img1},\text{img2}) \leq \tau$ (Match)

Siamese Network for Single Shot learning

Siamese Network means containing two or more identical networks and they are good in finding similarity between two comparable things

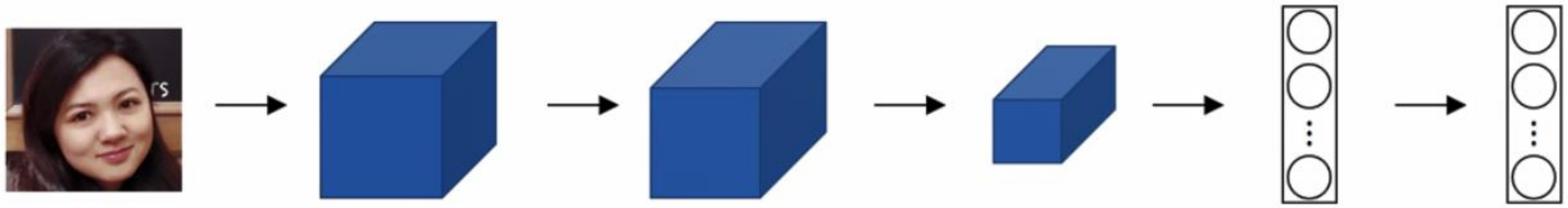


Parameters of NN define an encoding $f(x^{(i)})$

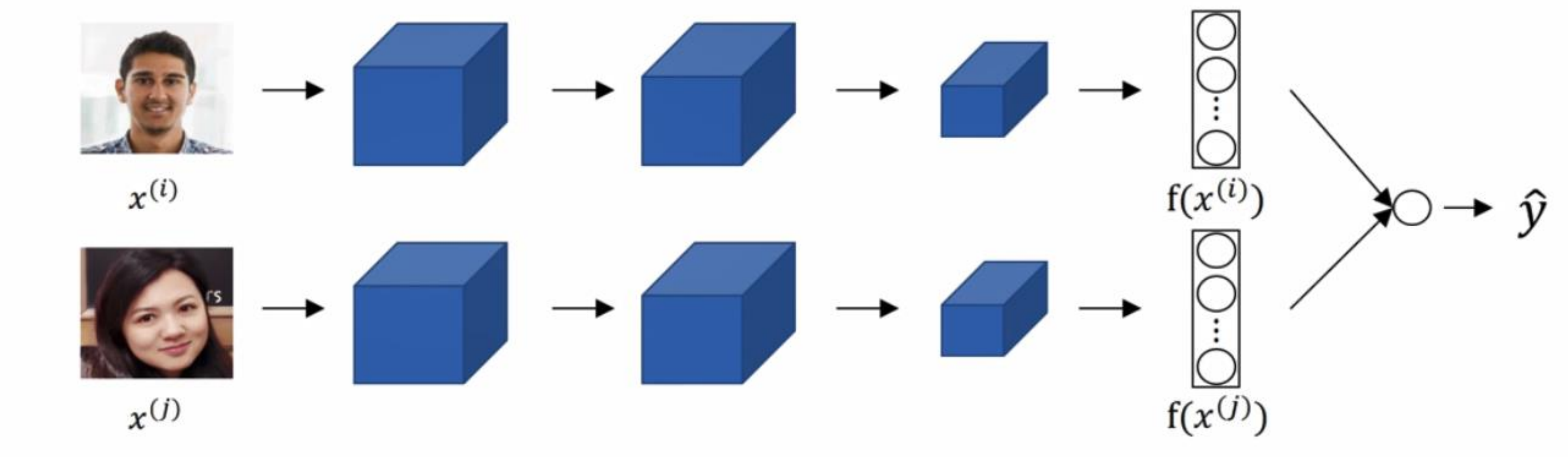
Learn parameters so that:

If $x^{(i)}, x^{(j)}$ are the same person, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is small.

If $x^{(i)}, x^{(j)}$ are different persons, $\|f(x^{(i)}) - f(x^{(j)})\|^2$ is large.



Face Verification using Siamese Network



To calculate the Sigmoid classification we can take the differences of the $f(x^{(i)})$ and $f(x^{(j)})$ Vectors and can predict whether the two persons are same or not.

Also, to reduce your computations, One of the images which comes from your database can have a precomputed network and may not require to be computed every time.

RNN

Recurrent Neural Networks

Sequence Application Variation

- Audio Signal to Sequence - Speech Recognition
- Nothing to Sequence or Single Parameter to Sequence - Music Generation
- Sequence to Single Output - Sentiment Classification
- Sequence to Sequence - Machine Translation
- Video Frame Sequence to Output - Activity Recognition
- Sub-Sequence from a Sequence - Finding Specific Protein from a DNA Sequence
- Outlining Specific parts of a sequence - Name Entity Recognition

Notation Understanding

X: Rama Conquered Ravana to install the virtue of dharma

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ .... $x^{<t>}$ .... $x^{<9>}$

$T_x = 9$ (Length of training sequence: 9)

$x^{i<t>}$: t^{th} word of i^{th} training sequence

Y: 1 0 1 0 …… 0 0 0 0

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ .... $y^{<t>}$ .... $y^{<9>}$

$T_y = 9$ (Length of output sequence: 9)

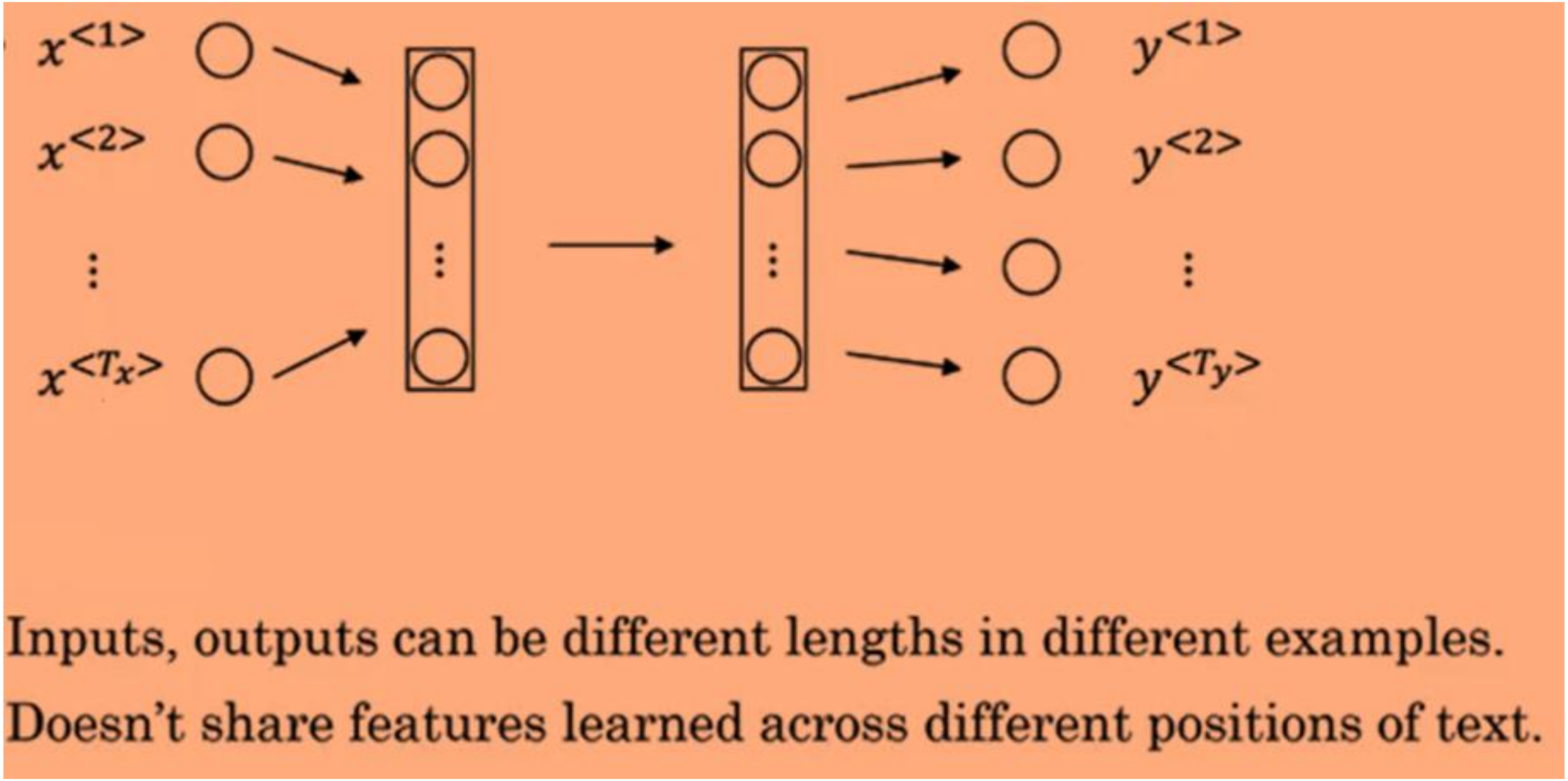
$y^{i<t>}$: t^{th} word of i^{th} output sequence

Representing words and one-hot encoding

X: Rama Conquered Ravana to install the virtue of dharma

A	1	Rama	Ravana
:		0	0
:		0	0
Conquered	329	0	0
:		0	0
:		0	0
Install	4521	:	:
:		:	:
:		:	:
Rama	7689	1 -7689	:
:		:	1-7900
Ravana	7900	:	:
:		0	0
ZZZ	10000	0	0

Standard Neural Network Does not works out to give a good application for sequence models



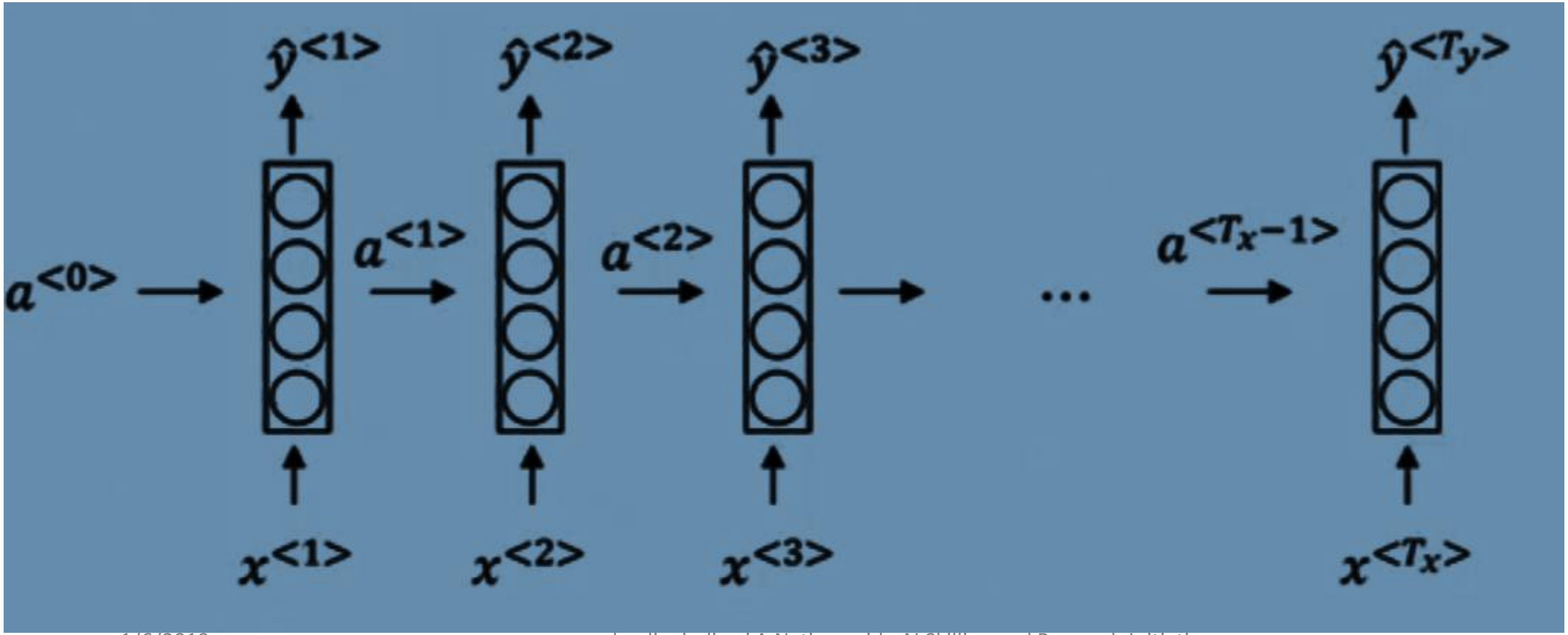
Inputs, outputs can be different lengths in different examples.

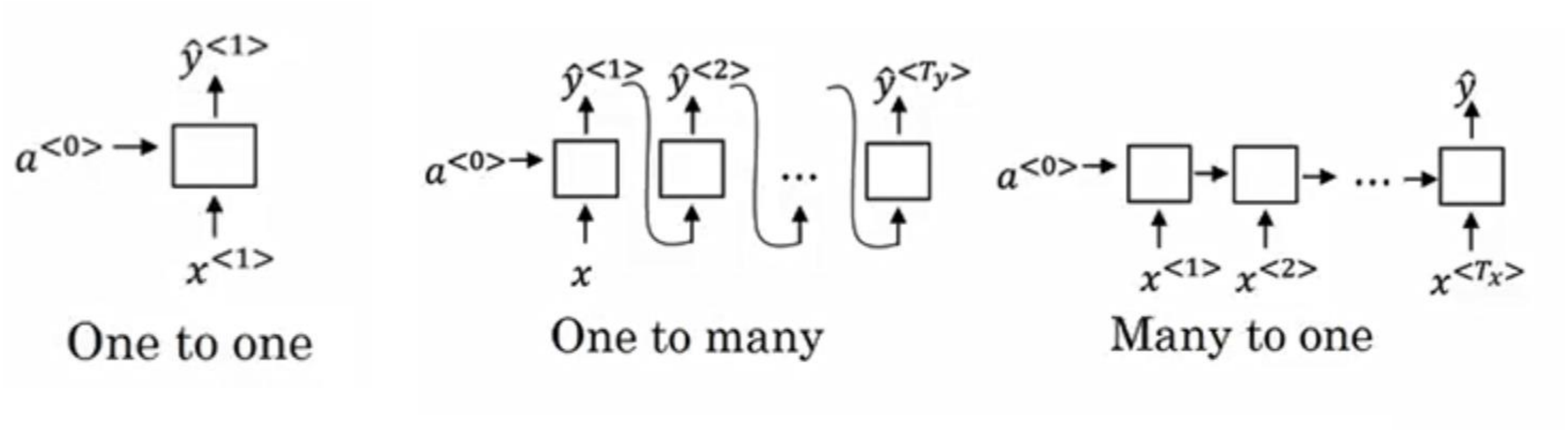
Doesn't share features learned across different positions of text.

Forward Propagation

$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

$$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$$



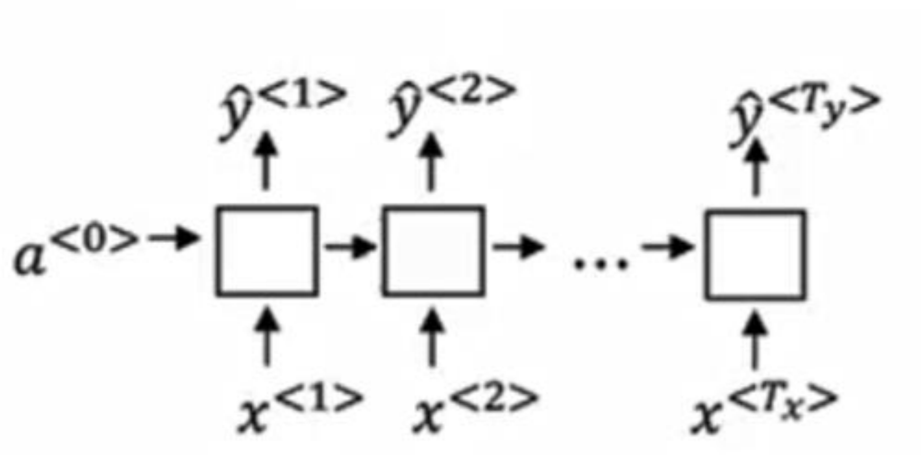


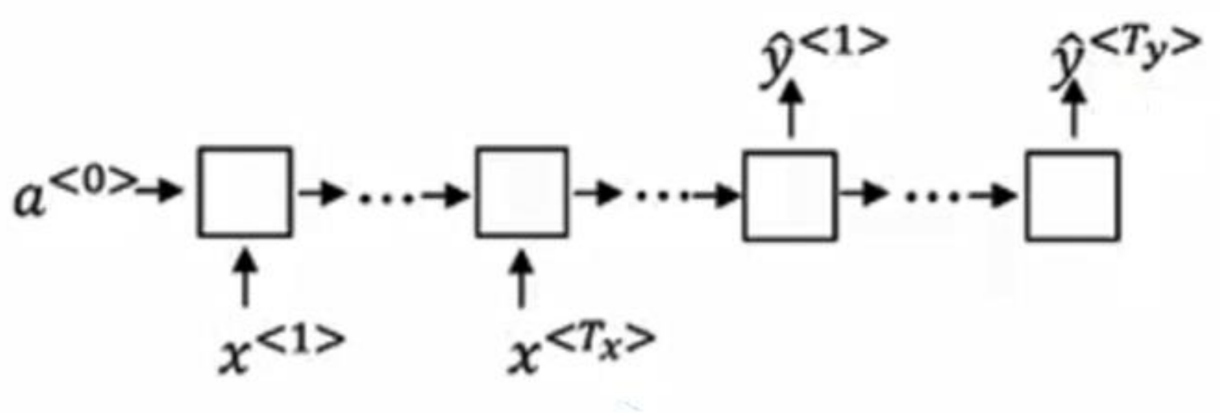
One to one

One to many

Many to one

Different Type of RNN Architectures





Many to Many

Many to Many

Word Level Language Model

Train your Language model on a large data.

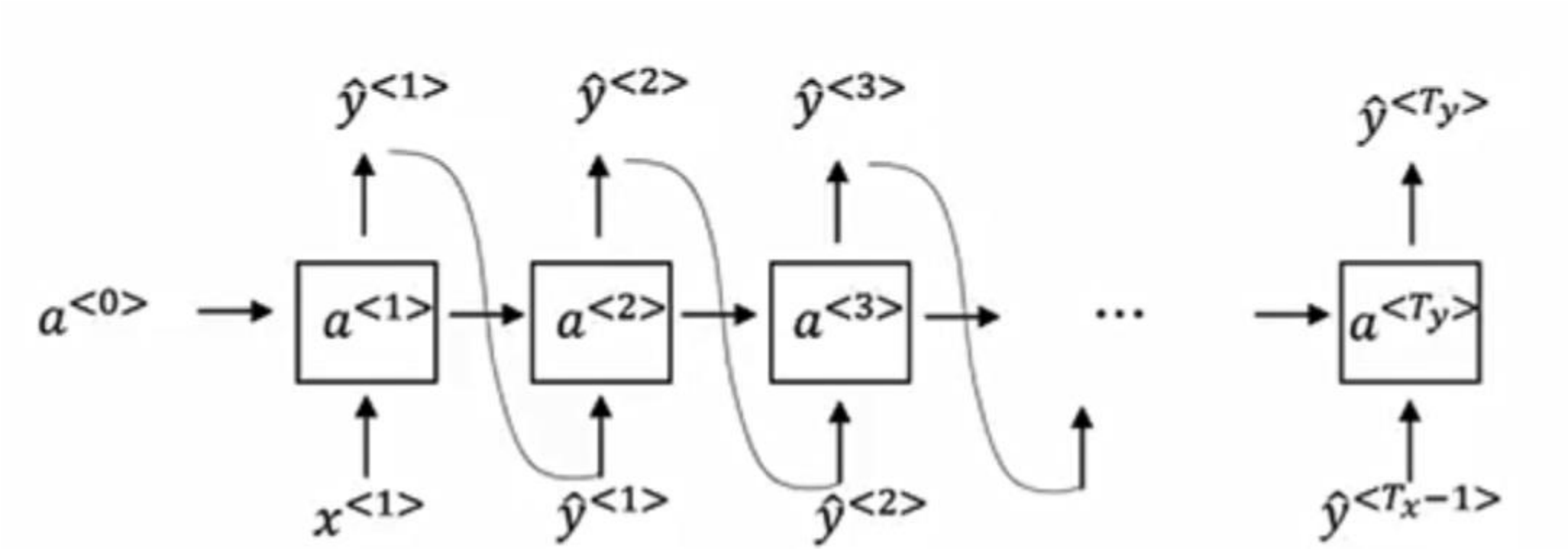
Then You can build on it different kinds of NLP applications as discussed.

Also, You will be able to generate new sentences or paragraphs etc as per the requirements of your application.

Difference between Word Level and Character Level Language Model

- Word Level Language models are more common due to their better performance as of now.
- Word level language models have <EOS> and <UNK> also as tokens in the corpus.
- Character level corpus is very small as compared to word level corpus
- In most cases word level corpus are of size 30-50k but in some cases can be upto 1 million
- In character level language model it becomes hard to predict and relate the relationship between far off characters as the distance becomes very large as compared to word language models.

Word level and character level language model



Sampling a novel sequence

Once you have a trained model on a corpus you can also have a RNN that can sample new sequences for you.

In that case you initialize with a zero and your first output gives a probability in terms of softmax function of the size of the no of categories equal to the size of your corpus.

You Choose a random word as the first output and then that word acts as the input for the second input and so on.

If you get a <UNK> then you can reject that token and continue with the next guess. It can go on until you get a <EOS> token.

Vanishing Gradients Issue

Regular RNNs are mostly influenced by local variations.

The cat , which already ate a lot while enjoying the party, was full.

The cats , which already ate a lot while enjoying the party, were full.

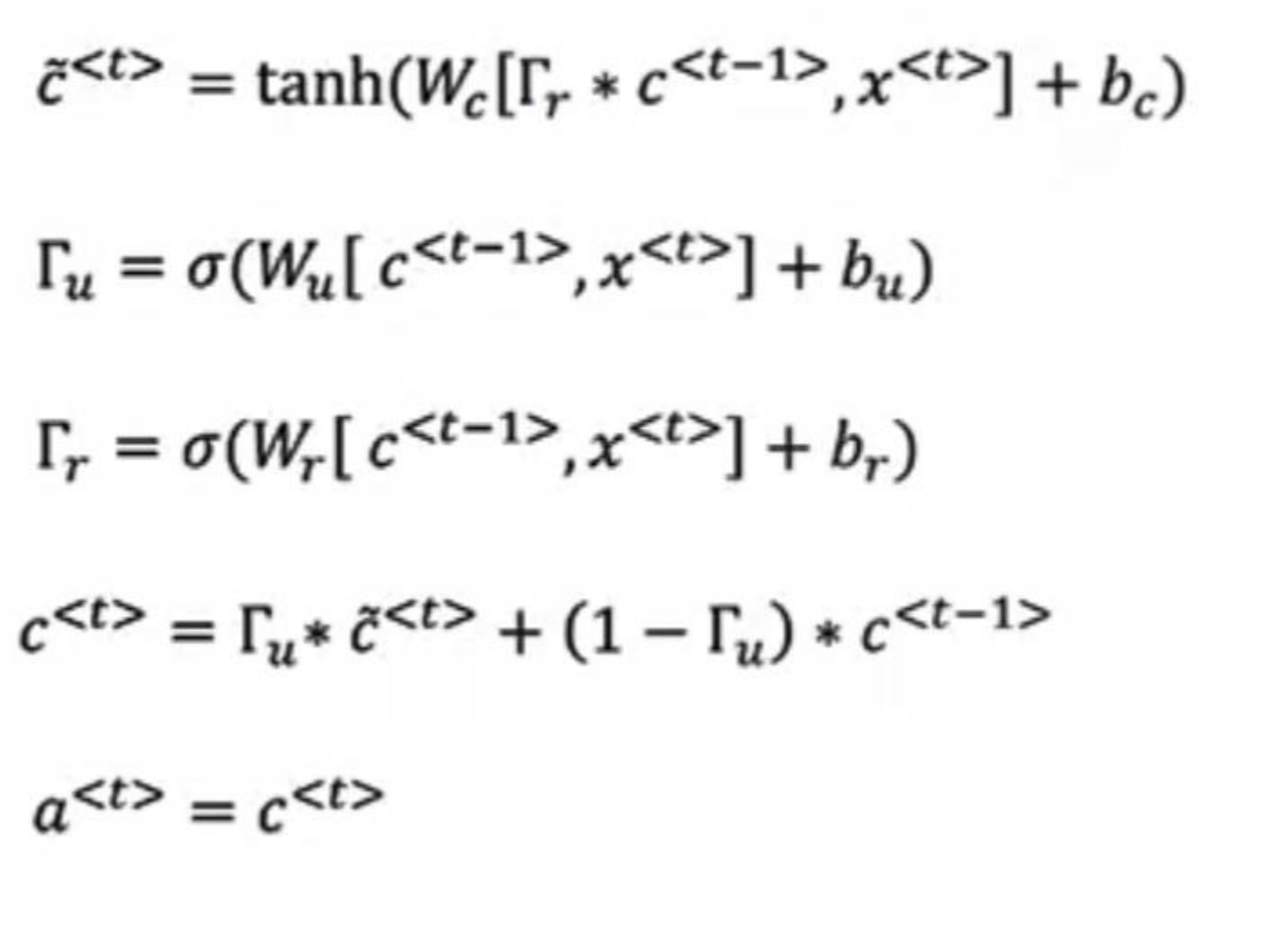
If the RNNs are not able to take care of long term dependencies, then the performance will be below expectations.

For Feed forward RNNs and also for back propagation, it is very difficult to reflect/relay the long term dependencies.

Gated Recurrence Unit

- Helps a Lot in Long Term Connection and also helps a lot in vanishing gradient issue .
- Main difference is that bring in the change in the hidden unit calculations and we introduce a memory cell
- We introduce an update gate u which will have value 0 or 1
- When the value of Gate is 0 that means it will keep the previous value and will not update else it will update the previous value with the new value
- We also introduce a gate r which means relevance of previous memory cell with the current cell

GRU



$$\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, c^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_r = \sigma(W_r[\, c^{<t-1>}, x^{<t>}] + b_r)$$

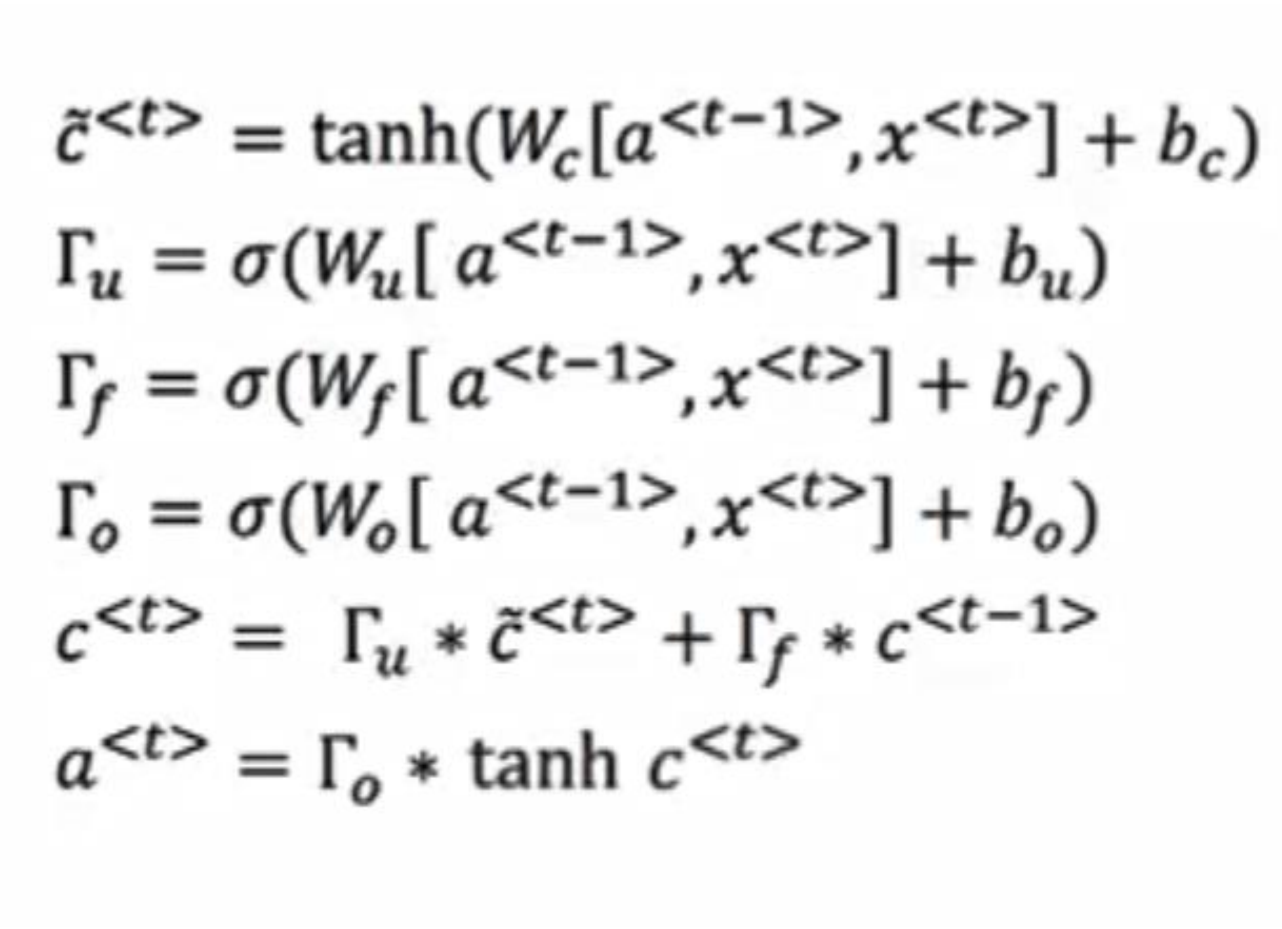
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

LSTM (Long Short Term Memory)

- In this we don’t have memory Cell value equivalent to activation value.
- We also introduce an additional gate called forget gate. It gives us the option to keep the previous values and also to add/update this gate with additional value from update gate.
- LSTM is most popular now for dealing with long term dependencies
- We also have a output gate.
- LSTM is more robust than GRU, but people use both of them based on applications.

LSTM



$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[\, a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[\, a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh c^{<t>}$$

Featurized Representation: Word Embeddings

	Man 5391	Woman 9853	King 4914	Queen 7157	Apple 456	Orange 6257
Gender	1	1	0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0
Age	0.03	0.02	0.7	0.69	0.03	0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
Size						
Cost						
Alive						

If we have 300 such properties and 10000 Words then it will be a 300x10000 Matrix and is denoted as Embedding Matrix (E) and O_{man} is One hot Vector for Man Word and e_{man} can be embedding vector for Man word.

I Want A glass of orange __Juice
I want a glass of Apple

Transfer Learning and Word Embeddings

- Learn Word Embeddings from Large Text Corpus (1-100 Billion Words)

- Pre-trained embeddings are available online

- Transfer Embedding to a new task with smaller training set

- Continue to finetune word embeddings with new data

Analogies using Word Vectors

As Man-> Woman King->?

As INR->India Dollar->?

As Delhi->India Kathmandu->?

As Tall->taller Big->?

As Man->Woman Boy->?

……

e_{man}-e_{woman} $\approx$ e_{king}-e_{w}

Find a word w : Maximize similarity(e_{w} , e_{king}-e_{man}+e_{woman})

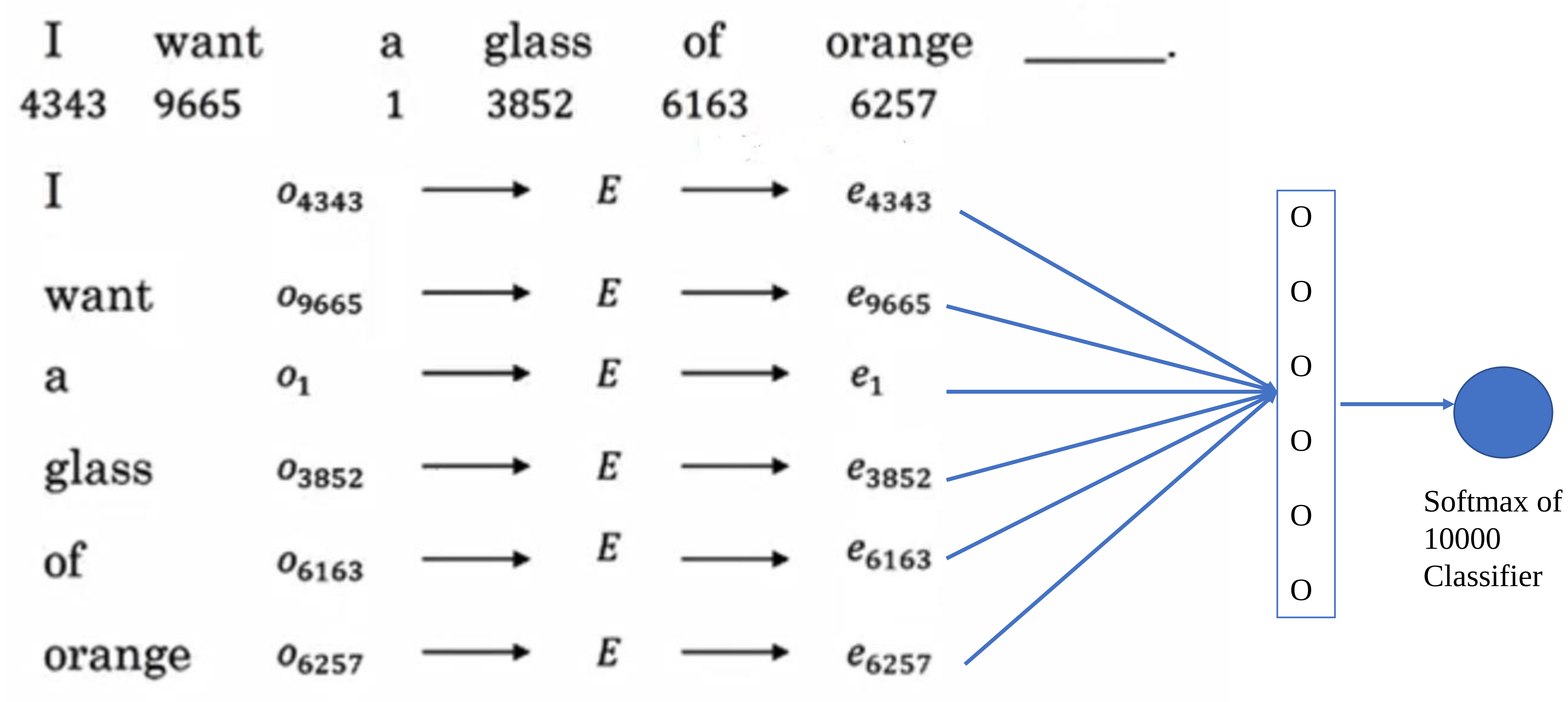
Cosine similarity

Sim(u,v)= $u^{T}v$ / $||u||\ ||v||$

Contexts which can help to learn

- Last 4 words
- Last x words and next x words
- Last one word
- One nearby word
- Randomly pick a word to be a context word and randomly pick a target word with in a window of x of the context word – Skip Gram Model

Neural Language Model



I	want	a	glass	of	orange	______.
4343	9665	1	3852	6163	6257	

I $o_{4343} \longrightarrow E \longrightarrow e_{4343}$

want $o_{9665} \longrightarrow E \longrightarrow e_{9665}$

a $o_{1} \longrightarrow E \longrightarrow e_{1}$

glass $o_{3852} \longrightarrow E \longrightarrow e_{3852}$

of $o_{6163} \longrightarrow E \longrightarrow e_{6163}$

orange $o_{6257} \longrightarrow E \longrightarrow e_{6257}$

Softmax of 10000 Classifier

Skip Gram Model

I want a glass of orange juice to go along with my cereal

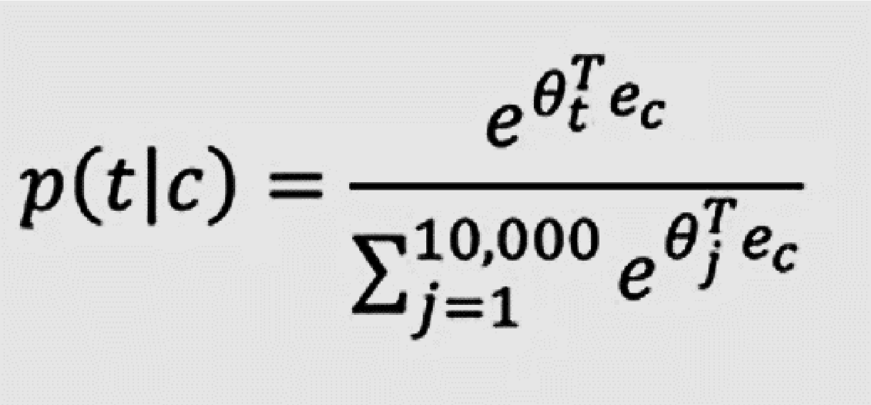
Context: Orange   Target: Juice

Context: Orange   Target: Glass

Context: Orange   Target: my

O_c->E->e_c------→Softmax Classifier→ŷ

Θ_t is the parameter associated with output t i.e chance of being the label



$$p(t|c) = \frac{e^{\theta_t^T e_c}}{\sum_{j=1}^{10,000} e^{\theta_j^T e_c}}$$

Negative Sampling Algorithm

Defining a new learning problem

I want a glass of orange juice to go along with my cereal

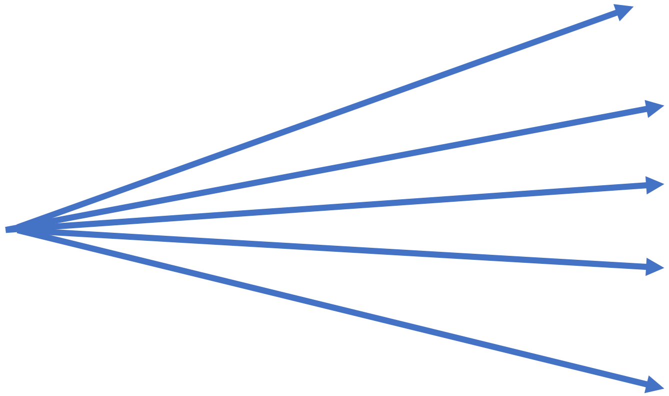
We will use of pair of words and then find out whether the second word can be a target word for the first context word

Context	Word	Target	
Orange	Juice	1	(We choose First example as positive example)
Orange	King	0	
Orange	Table	0	
Orange	to	0	
Orange	Pencil	0	

Generally we choose 5-20 such pairs for smaller datasets and 2-4 words for bigger data sets

Converting Softmax to Logistic Classifier

- O_c->E->e_c------→10000 Binary Logistic Classifier→ŷ

- O_c->E->e_c 

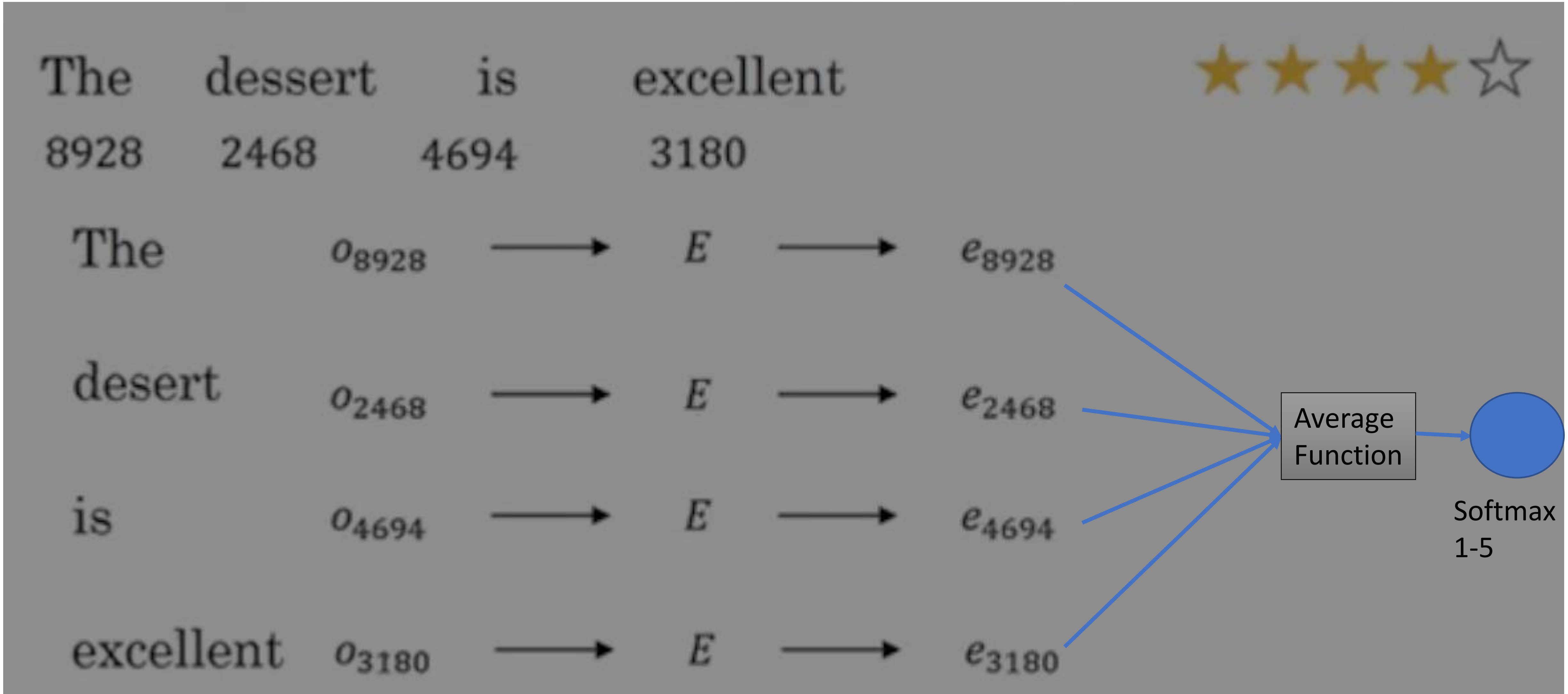
- We have 10000 binary classifiers but we only train 5 random pair words in every iteration

Removing Biases in NLP

- It related to Gender and ethnicity biases and we need to be very careful about this
- Man: Computer_Programmer as Women Homemaker
- Father: Doctor Mother: Nurse

- Biases will be picked from the text it has been trained upon
- First step is to identify Bias Direction e.g. male to female
- Next is to Neutralize the bias for all the non-definitional word for example Father, Mother, He, She are definitional word for Gender and should not get changes due to this. However, Non-definitional word like soldier, doctor, Manager, Programmer etc should be neutralized for bias
- Last step is to equalize pairs like niece, nephew; grandmother, grandfather and they should be equidistant from words like babysitter etc.

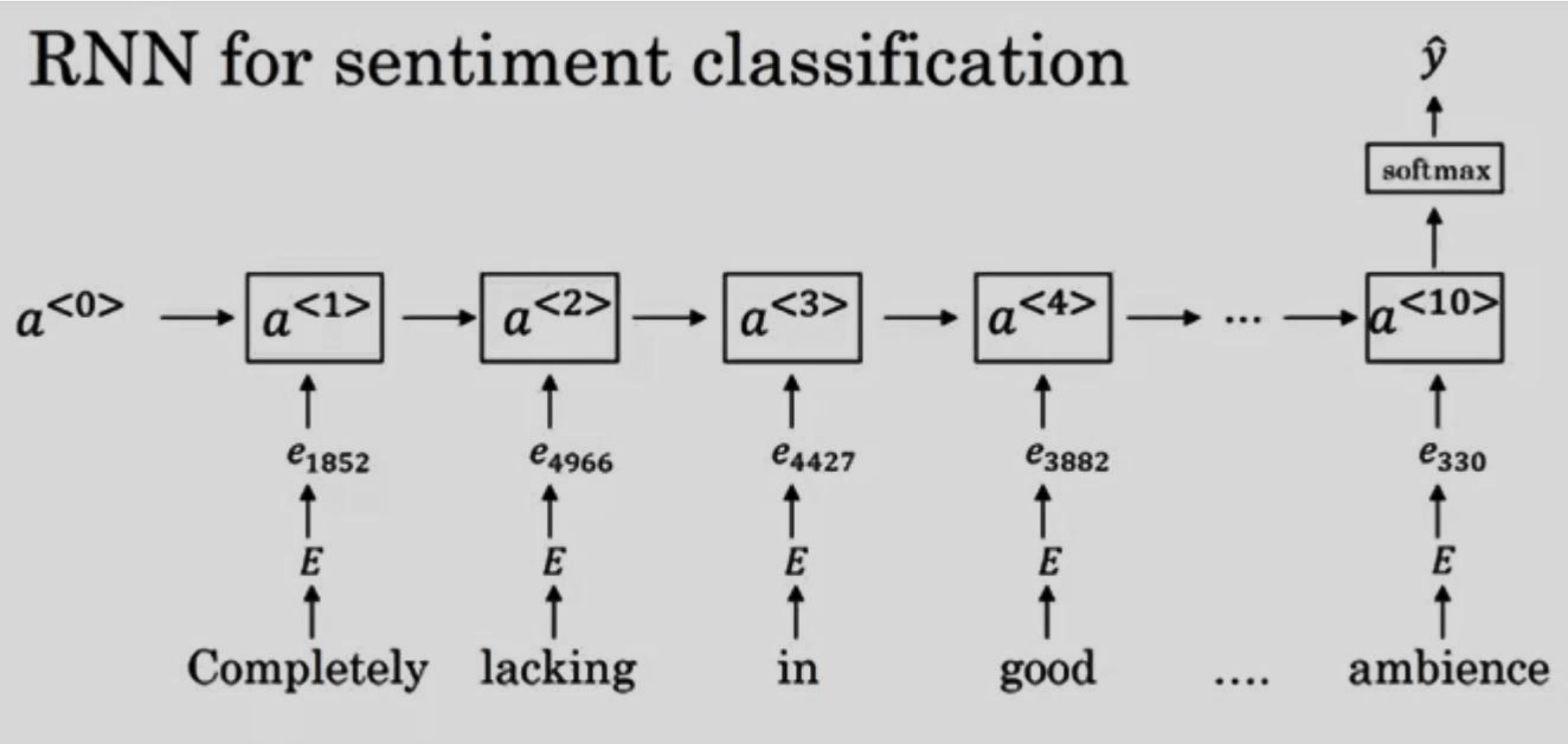
Simple Sentiment Classification Model



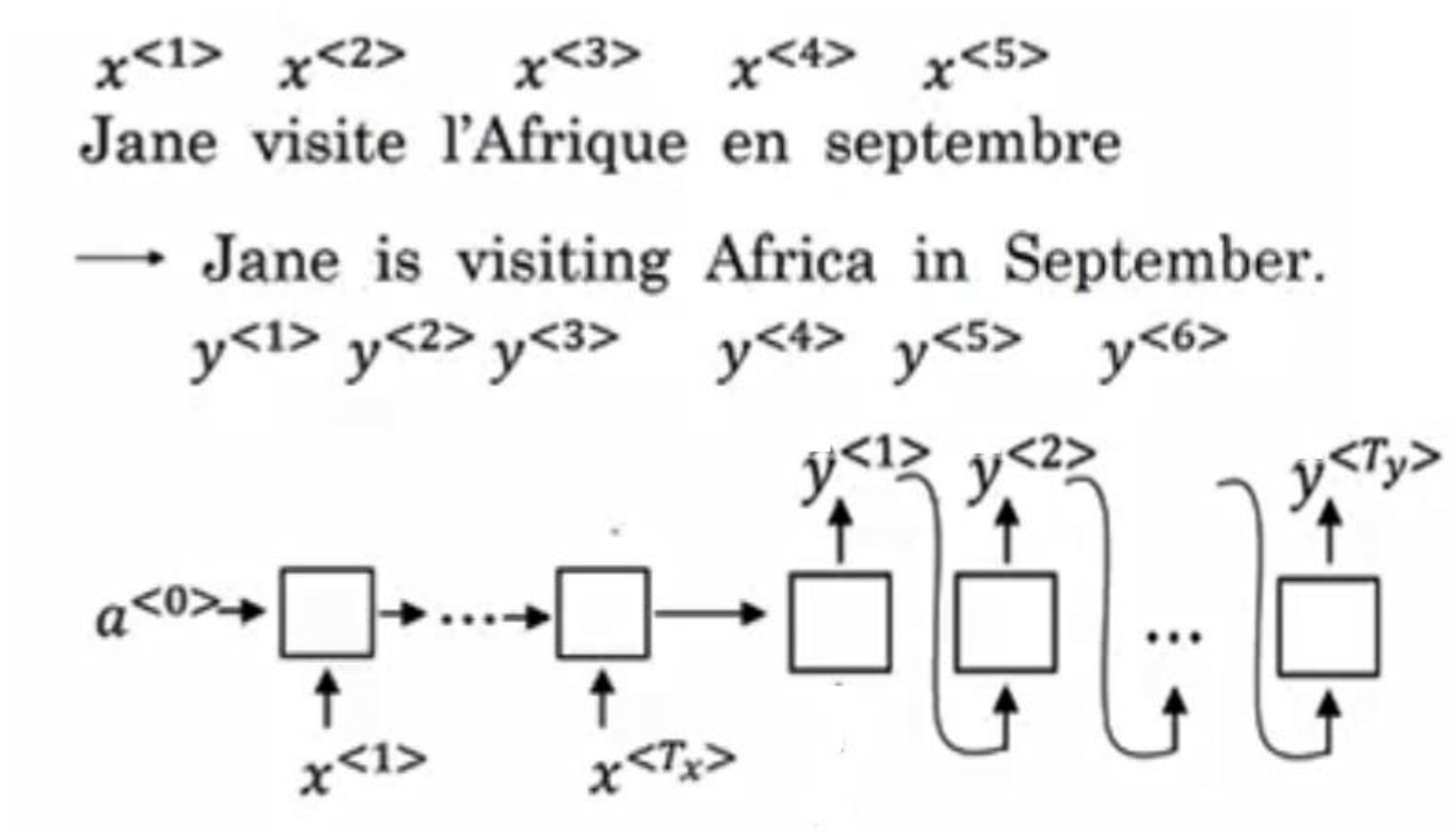
The dessert is excellent
★★★★☆
8928 2468 4694 3180
The o_{8928} $\longrightarrow$ E $\longrightarrow$ e_{8928}
desert o_{2468} $\longrightarrow$ E $\longrightarrow$ e_{2468}
is o_{4694} $\longrightarrow$ E $\longrightarrow$ e_{4694}
excellent o_{3180} $\longrightarrow$ E $\longrightarrow$ e_{3180}
Average Function
Softmax 1-5

Counter Example :Completely Lacking in Good Ambience, Good Taste, Good Service

RNN for sentiment classification



Sequence to Sequence Model

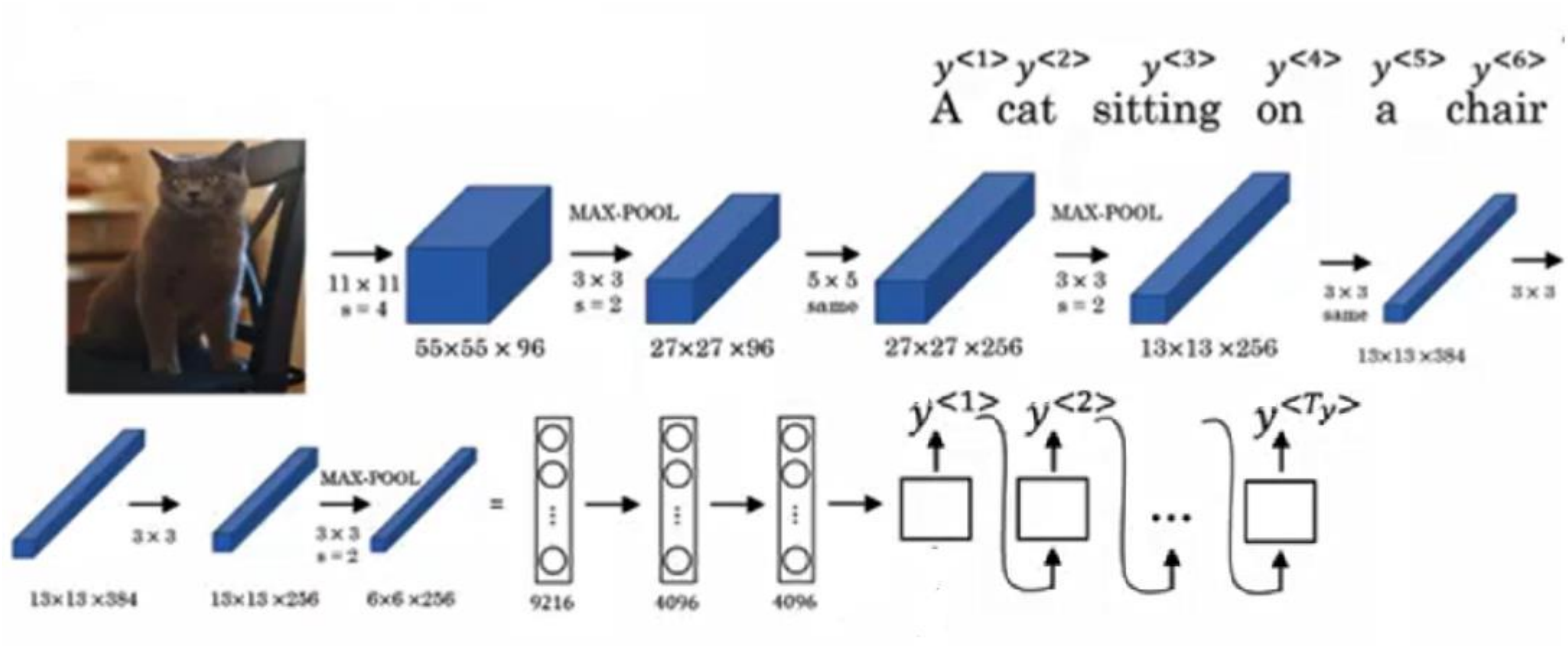


Jane visite l'Afrique en septembre

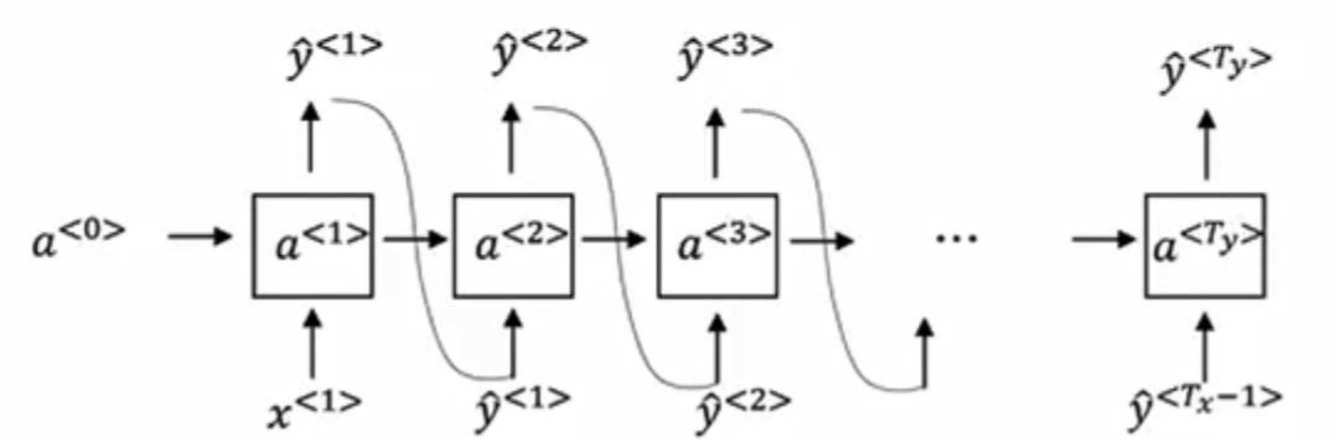
→ Jane is visiting Africa in September.

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$
$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$
$a^{<0>}$
$x^{<1>}$
$x^{<T_x>}$
$\hat{y}^{<1>}$
$\hat{y}^{<2>}$
$\hat{y}^{<T_y>}$
...

Image captioning model

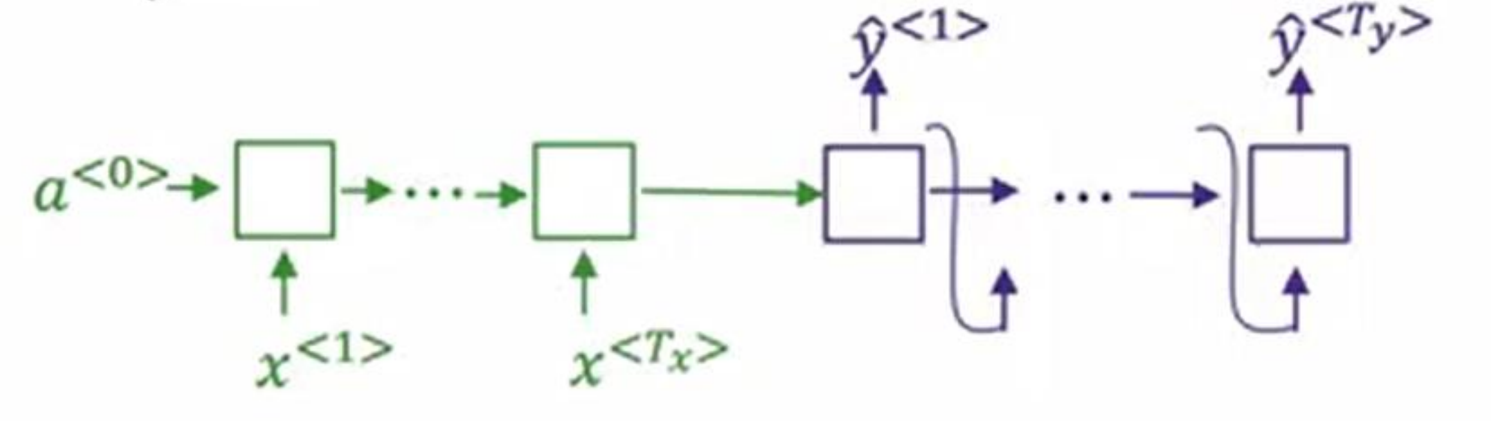


Machine translation as a conditional Language Model



Language Model

$$P(y^{<1>}, \ldots, y^{<T_y>} | x)$$



Translation Model

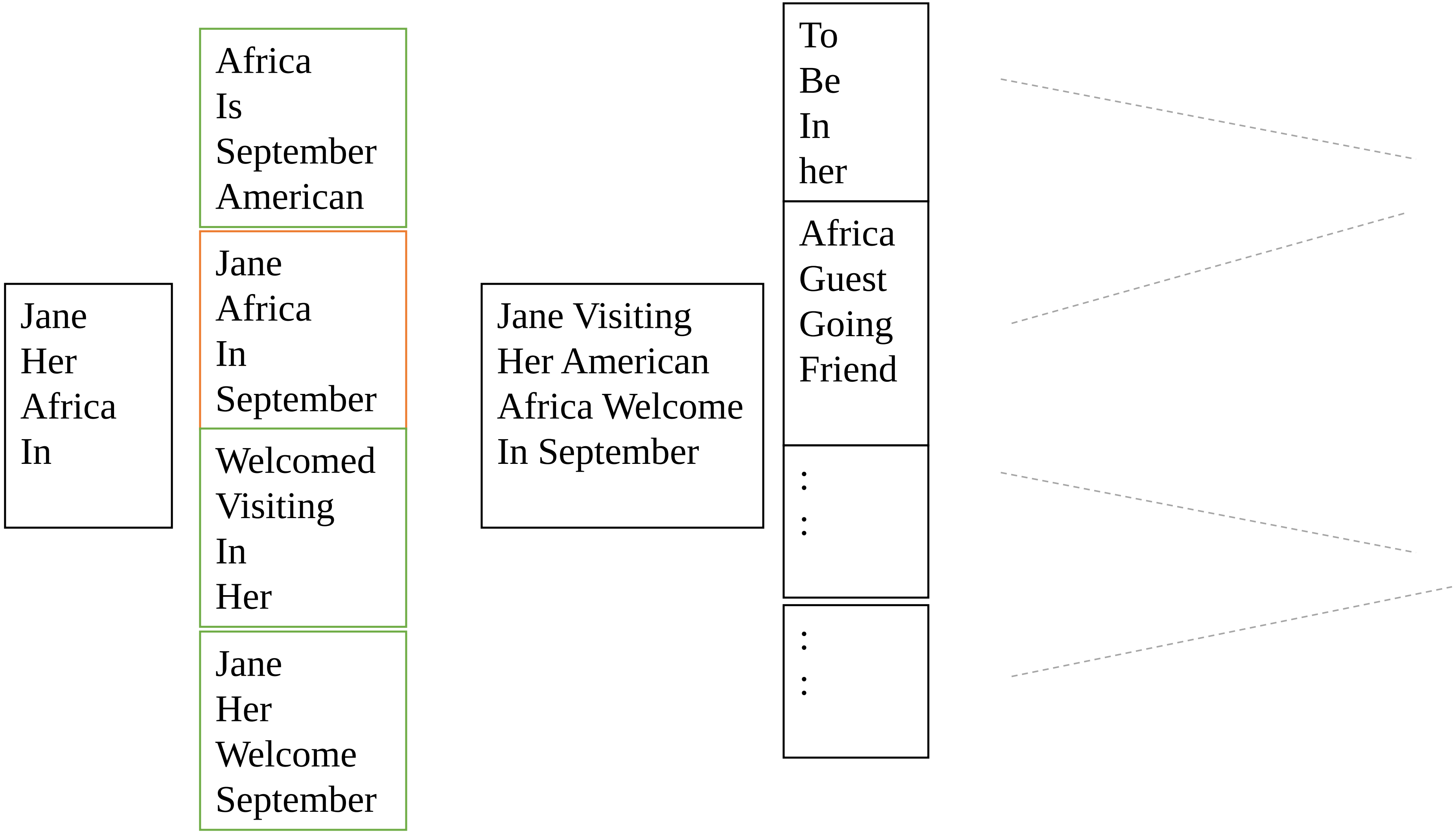
Finding the Most likely translation

Jane Visite l’afrique en septembre

- Jane is visiting Africa in September
- Jane is going to be visiting Africa in September
- In September, Jane will visit Africa
- Her American Friend Welcomed Jane in September

- If the length of the output sequence is 10 and the size of the corpus is 10000 then the possible number of permutations are 10000^{10}
- In a Greedy search algorithm we can choose the first word which is most likely to come (with maximum probability) and then the next output word of the translation with the maximum probability and so on. But, in practice this does not works and does not gives us a good translation.

Beam Search



Jane
Her
Africa
In

Africa
Is
September
American

Jane
Africa
In
September

Welcomed
Visiting
In
Her

Jane
Her
Welcome
September

Jane Visiting
Her American
Africa Welcome
In September

To
Be
In
her

Africa
Guest
Going
Friend

:
:

:
:

Beam Search

In this technique we will choose width of the Beam (Lets say 4)

It means that now we will select 4 words with highest probability given by the output of the softmax function in the last layer

Now corresponding to the each word in the first output we will choose 4 words who have max probability to come as a second word given first word.

Out of these 16 combinations of first two words, we will only choose 4 pairs with highest total probability

Now we will choose the third word corresponding to these four pairs of first two words and so on.

Beam Width

- Large beam width will result in slow performance but better results
- Smaller beam width will result in good performance with compromise in accuracy,
- Generally the acceptable length of the beam width in production systems will be in the range of 3-10 based on the applications, while in case of researchers it can even go to 100.

Bleu Score

- Bilingual Evaluation Understudy **Score**, or **BLEU**for short, is a metric for evaluating a generated sentence to a reference sentence. A perfect match results in a **score** of 1.0, whereas a perfect mismatch results in a **score** of 0.0
- Le Chat est sur le tapis
- The cat is on the mat –Ref 1
- There is a cat on the mat – Ref 2
- The the the the the the the
- Precision 7/7
- But in case of bleu score we use the clip count, which considers the maximum no of times the word appears in reference sequences. In this the appears twice in the Ref1 so clip count will be 2/7

Bleu score

It is important because there can be multiple right translations of a sentence, in that case also we need to choose one of them and it also helps in correctly assessing the error distance between the right translations and machine translation.

The cat is on the mat –Ref 1

There is a cat on the mat – Ref 2

The cat the cat on the mat – Machine Translation

The cat	2	1
Cat the	1	0
Cat on	1	1
On the	1	1
The mat	1	1

4/6 is the precision on the bigram

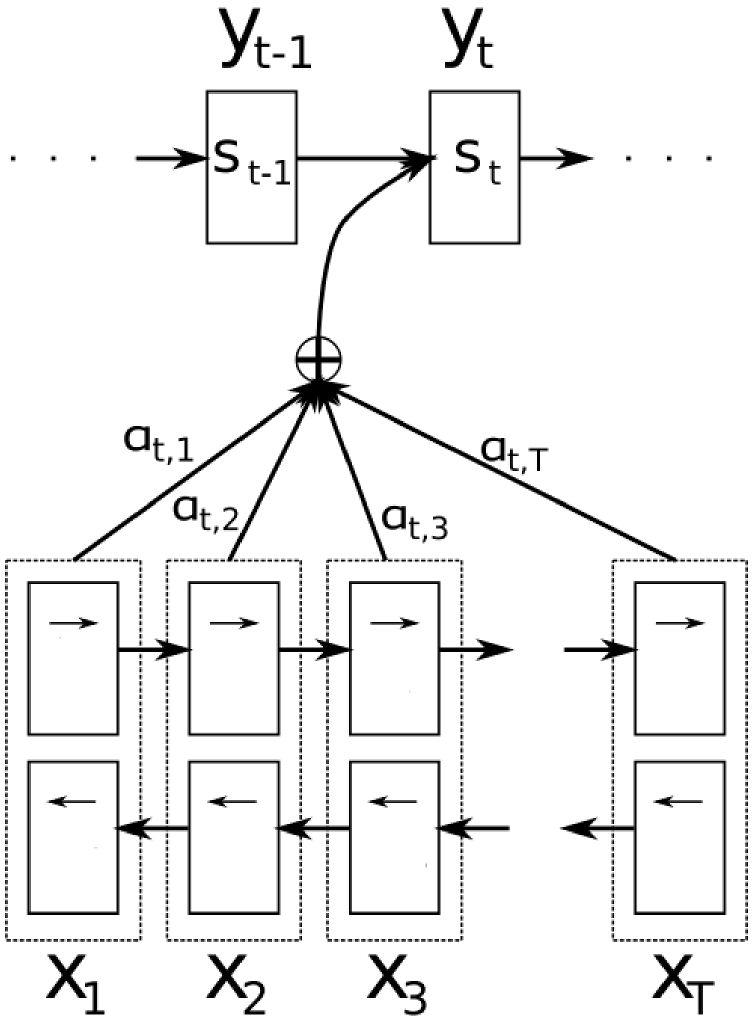
Similarly we will do on trigram and so on

It will help us to choose the right sequence

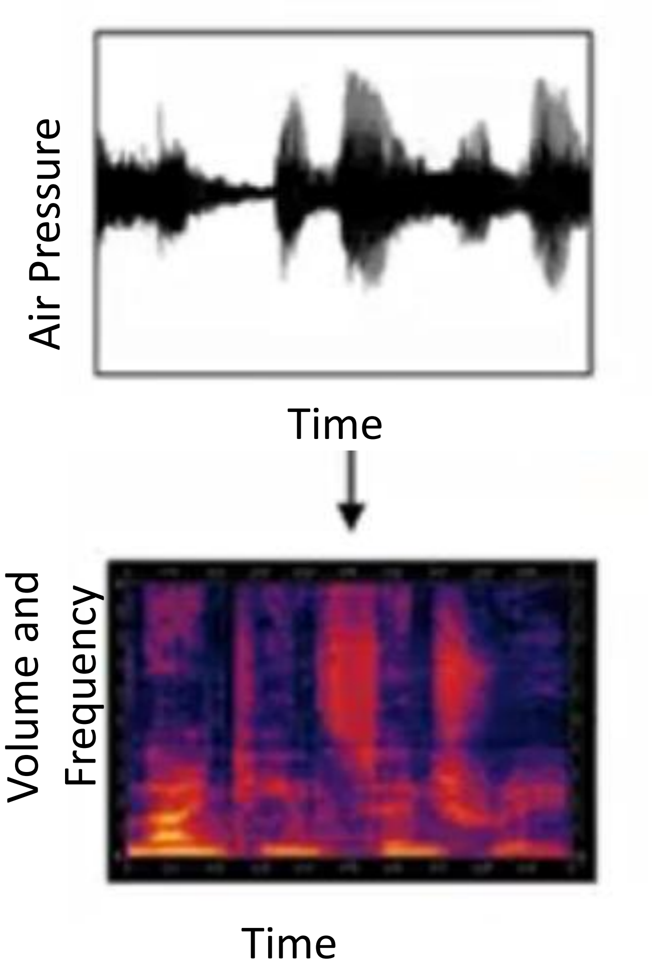
Problem of long sequences: Attention model

- When the sequences grow beyond a certain length (20 or more), then the bleu score goes down considerably and generally does not has good mapping with the goodness of the translation.
- To handle this recently Researchers came out with a new model called attention mechanism.
- It basically tells that how much attention needs to paid to each word of the source sequence for every position of the translated sequence.
- Total of attention values for any particular word should be equal to 1, so we can think of it as a softmax classifier with a small neural network for determining the probabilities of each attention value vector.

Attention Model



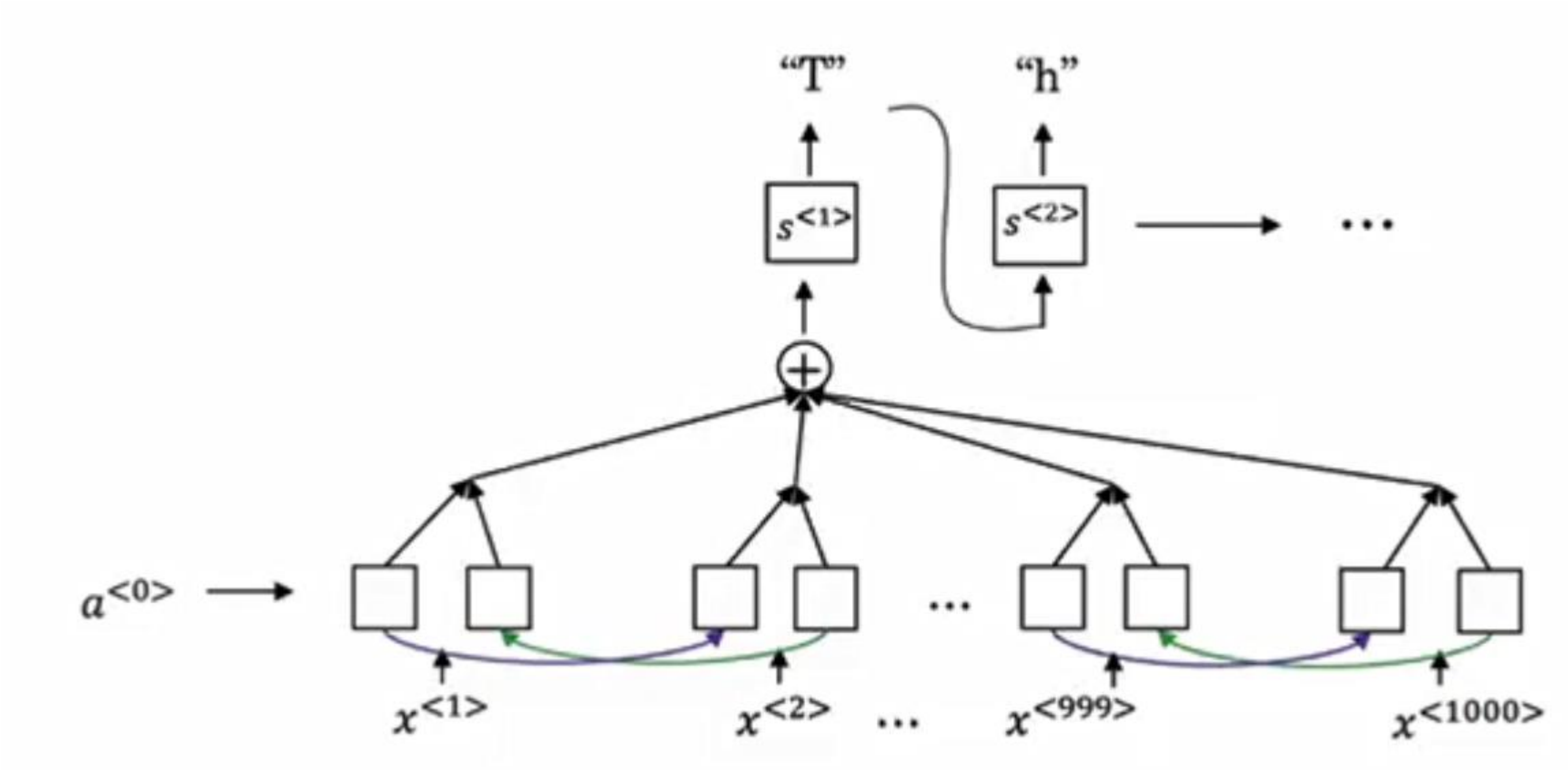
Speech Recognition



Previously we used to have Hand-engineered features consisting of different phonemes

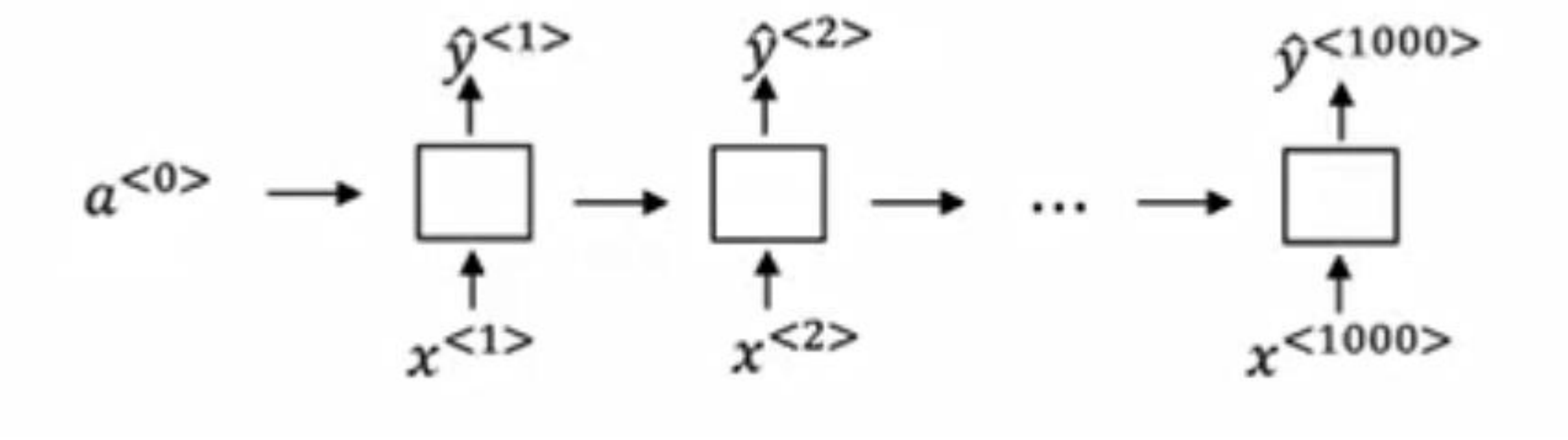
Thousands of hours of speech/audio data is used to train the data depending upon the application

Attention model for speech Systems



CTC Cost for Speech recognition

- Connectionist Temporal classification
- The Quick Brown Fox



- One of the issues in Speech Systems is that for 1000s of inputs your speech translation may actually produce only a few words. The reason is that the time frame we process to consider a individual sound unit is small as compared to the actual spoken words. So it is important to map the 1000s of outputs to a few words.
- E.g. ttt_h_eee_____ ____ ␣ _ _ _ _ _qqqq _ _ _ _
- ␣represents space and _ are blanks

Questions ?

Thanks