Image2Text

Prepared By

Sanju Prabhath Reddy	15114042
Sri Harsha Majeti	15114044
Harsha Vardhan Miryala	15114045

Image Captioning

A short piece of text that describes an image .

In other words explains what the object(maybe people) in it is/are doing.

Examples

A person is walking along a beach with a big dog

A black and white dog carries a tennis ball in its mouth

A soccer player takes a soccer ball in the grass

A man is doing a trick on a snowboard

A surfer dives into the ocean

A black and white dog leaps to catch a Frisbee

How we do ?

We use various techniques involved in

1. Computer Vision

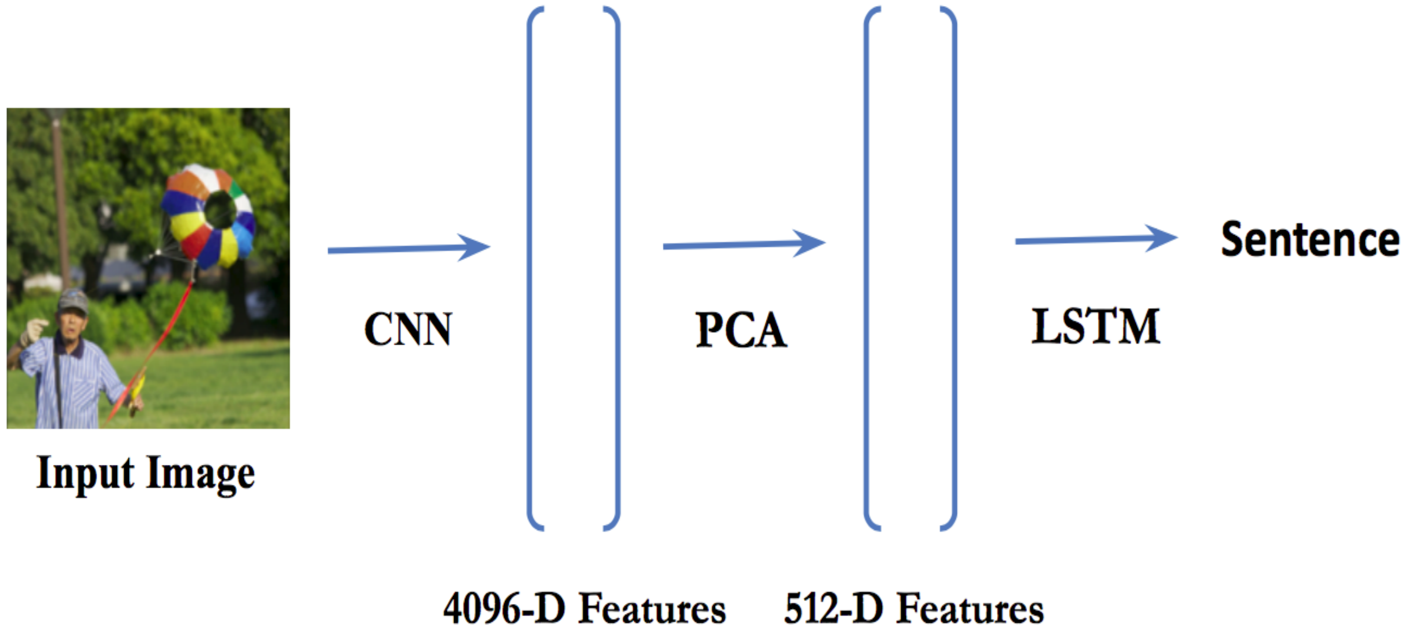
2.Natural Language Processing (NLP)

OUR GOAL:

To give an overview about the general architecture which uses both

CV and NLP

Overview


Input Image
CNN
4096-D Features
PCA
512-D Features
LSTM
Sentence

Model Architecture

Contains 3 components

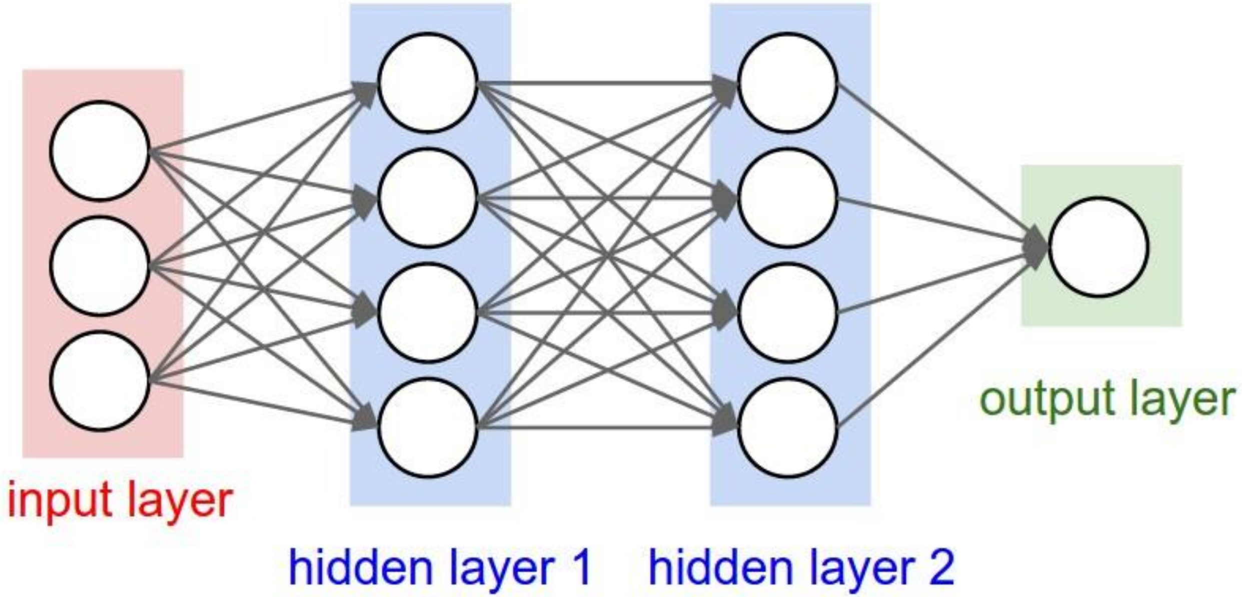
1.CNN(Convolutional neural network) based feature extraction

2.PCA (Principal Component Analysis)

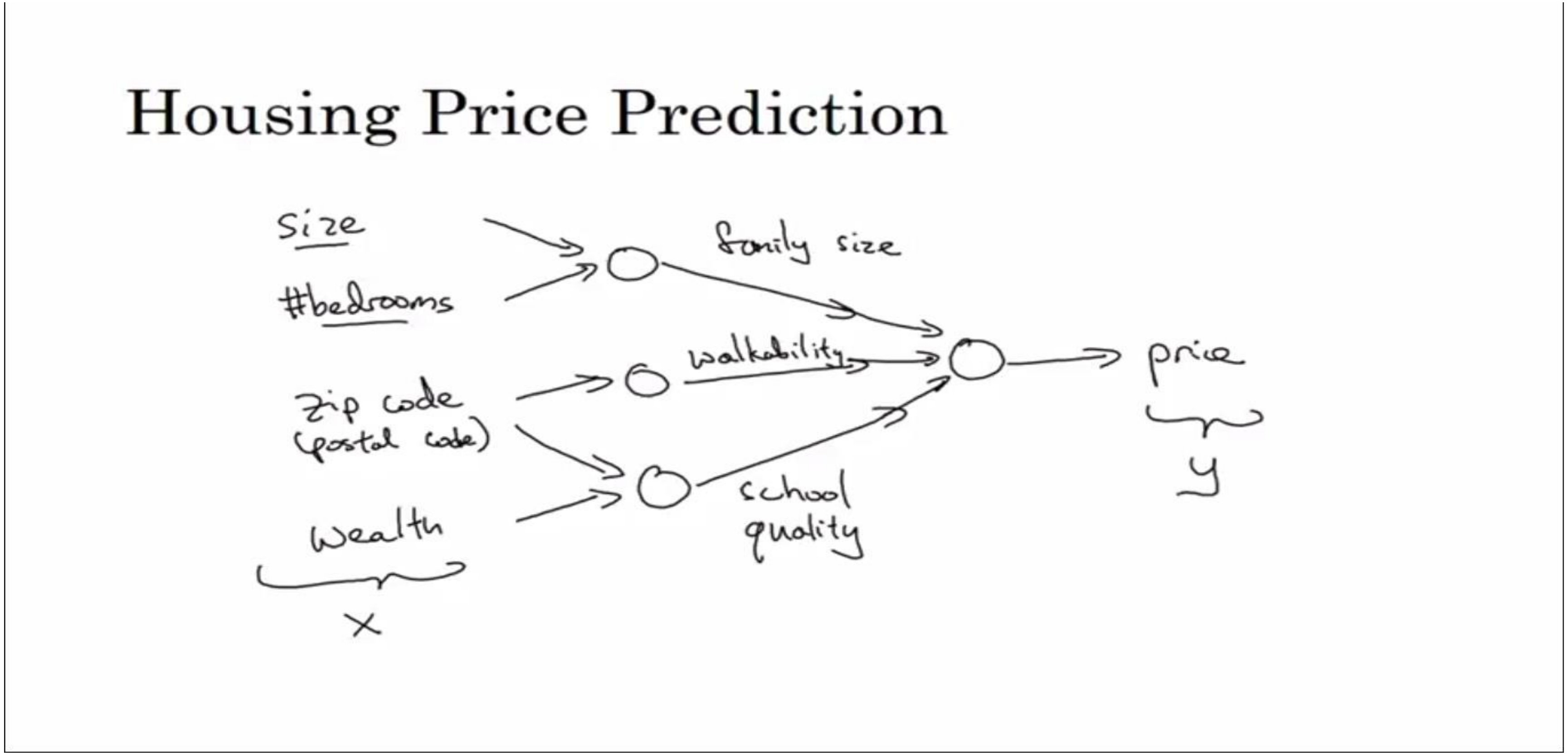
3.RNN(Recurrent neural network) based sentence generation

CNN

Traditional Neural Network



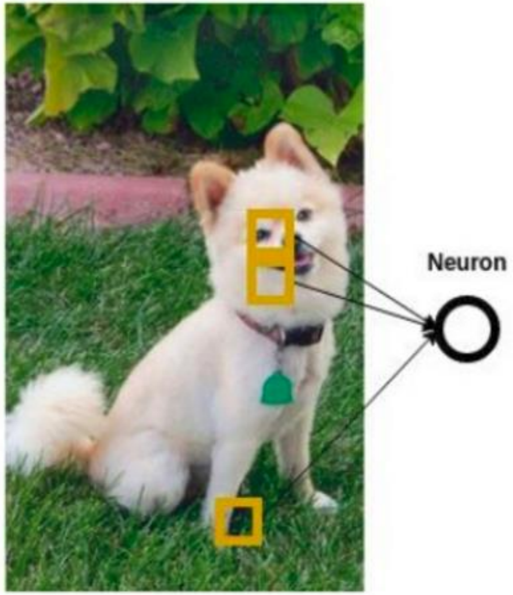
Example


Housing Price Prediction
size
#bedrooms
zip code (postal code)
wealth
x
family size
walkability
school quality
price
y

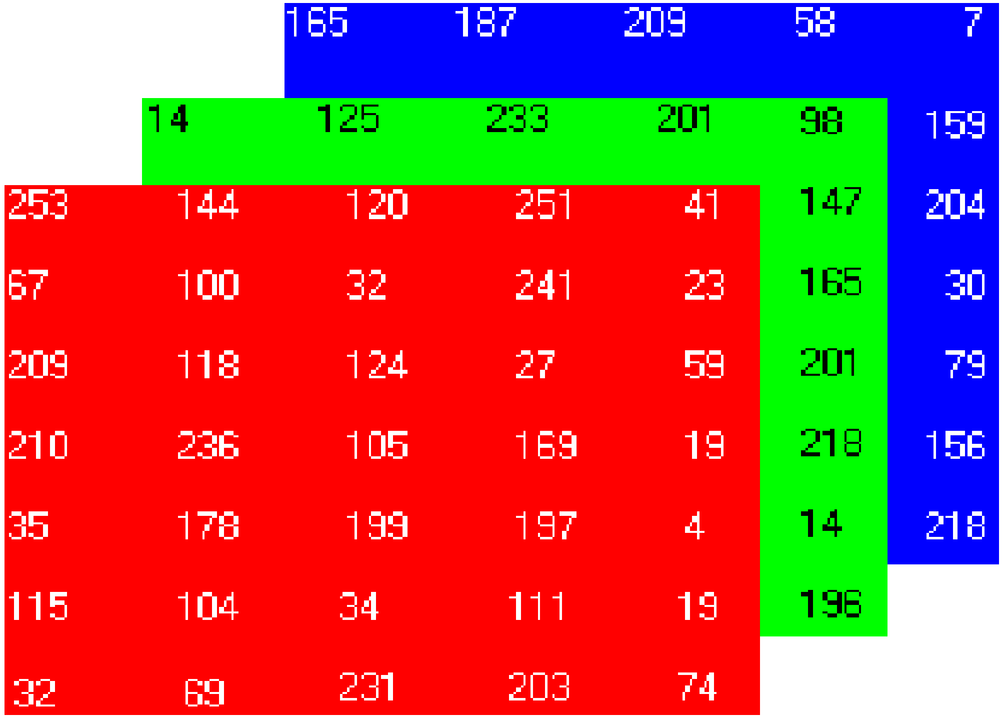
Clip slide

Why Convolutional Neural Networks?

- NN not always best representation eg. images
  - Adjacent pixels are connected similarly to those far apart
  - Cannot detect features in different areas of image
- How to represent this concept?
  - Convolutional Neural Networks aka CNN!
  - Use subset of inputs mapped to hidden layers

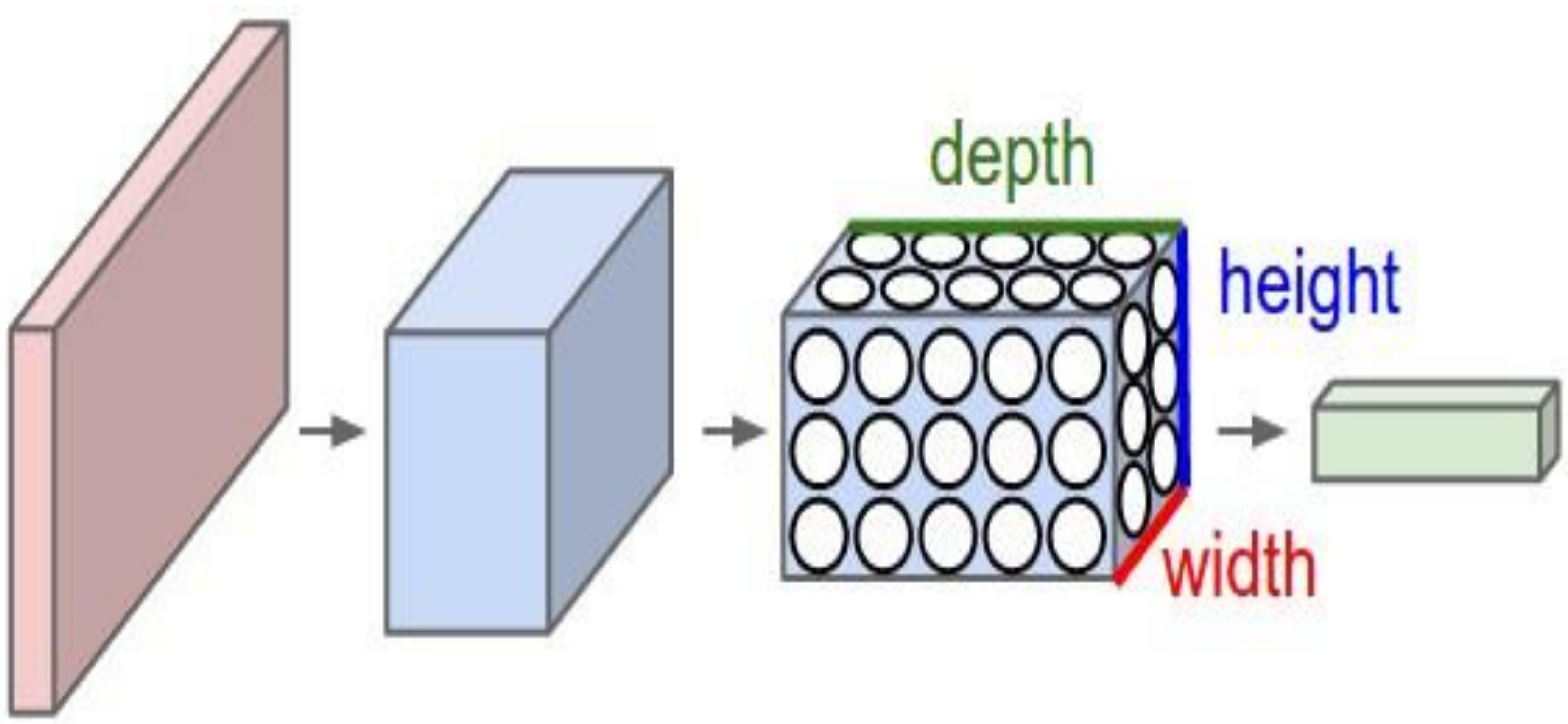


How a Computer sees an Image !



253	144	120	251	41
67	100	32	241	23
209	118	124	27	59
210	236	105	169	19
35	178	199	197	4
115	104	34	111	19
32	69	231	203	74

General Representation Of a CNN



Layers in CNN

Generally there are 3 layers in a CNN

1.Convolutional

2.Pooling

3.Fully Connected

What is convolution?

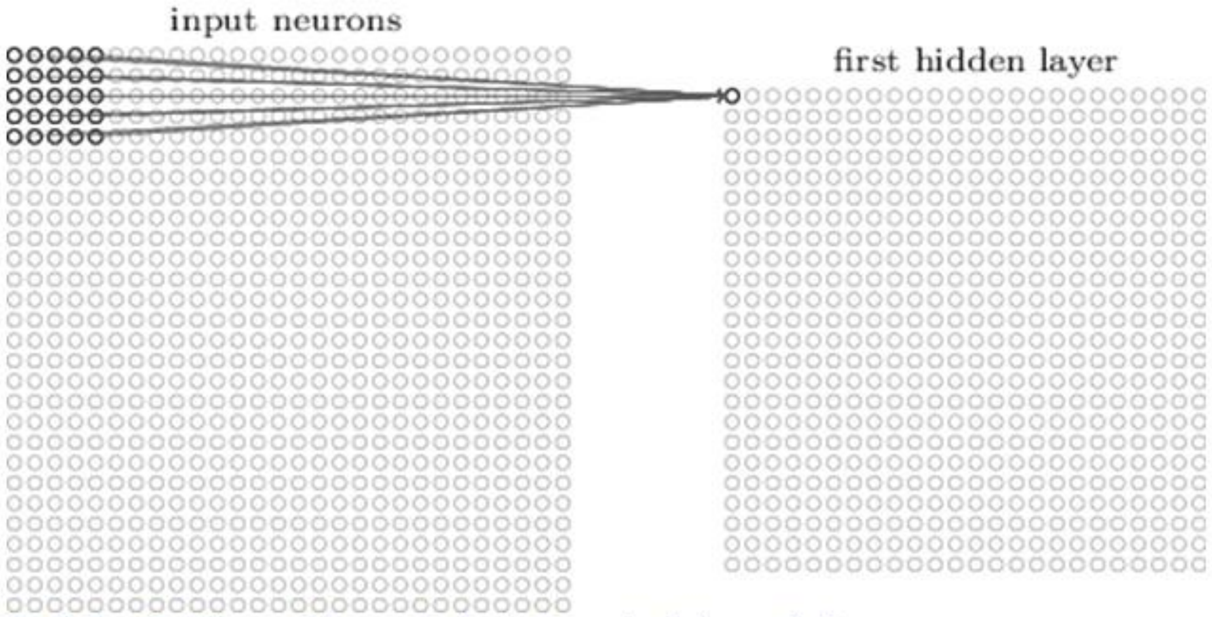
Example : Let’s take input image of 32*32*3

Let’s take filters of 5*5*3 size.Consider filter as a feature identifier(interpretation).

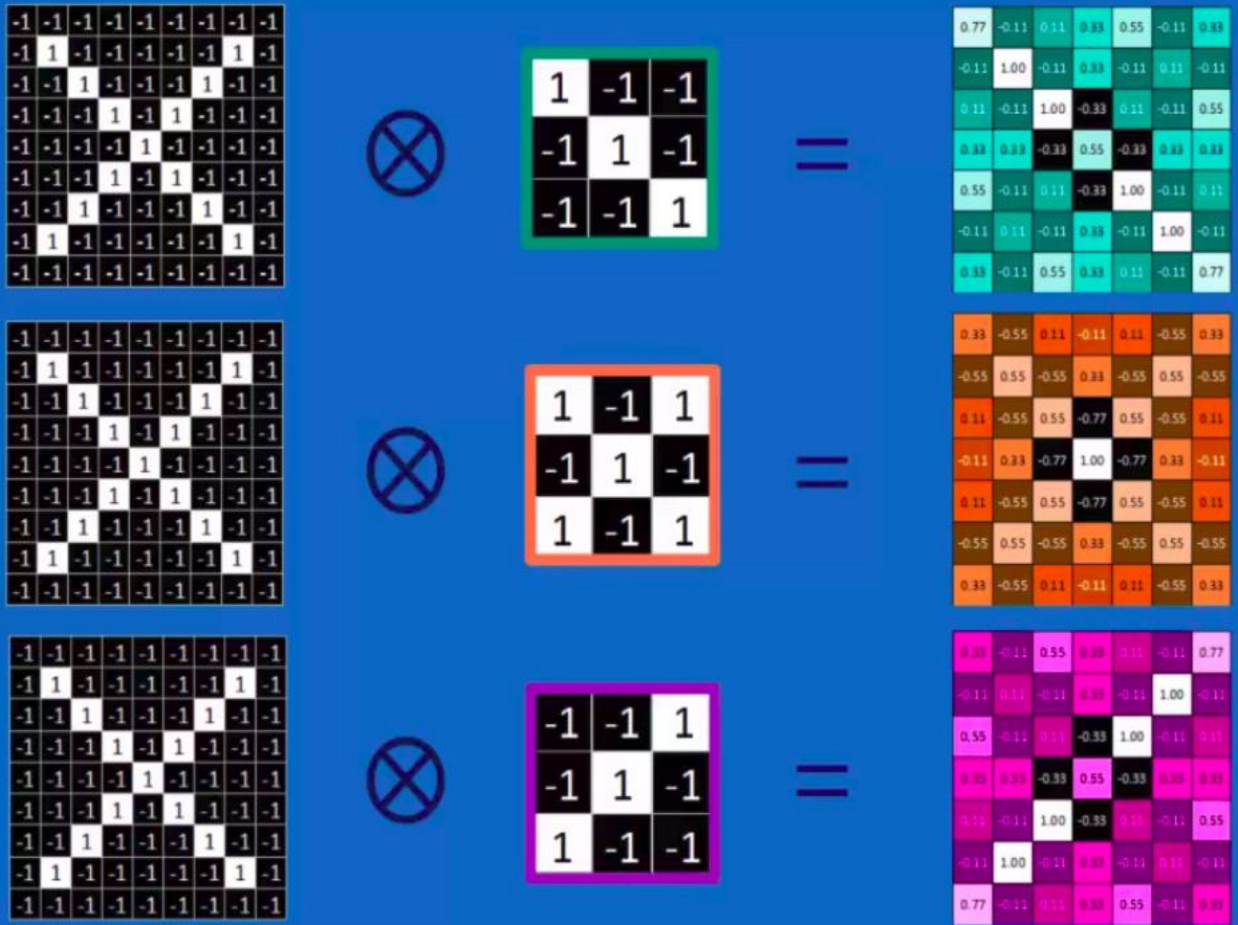
Slide this filter with stride 1 and find the sum of multiplication corresponding pixel values with feature values in the filter.

We get a matrix of 28*28*x if we take x such filters.

This process is called convolution..



Visualization of 5 x 5 filter convolving around an input volume and producing an activation map



$$\begin{bmatrix} -1&-1&-1&-1&-1&-1&-1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&-1&-1&1&-1&-1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&-1&-1&-1&-1&-1&-1&-1 \end{bmatrix} \otimes \begin{bmatrix} 1&-1&-1\\ -1&1&-1\\ -1&-1&1 \end{bmatrix} = \begin{bmatrix} 0.77&-0.11&0.11&0.33&0.55&-0.11&0.33\\ -0.11&1.00&-0.11&0.33&-0.11&0.11&-0.11\\ 0.11&-0.11&1.00&-0.33&0.11&-0.11&0.55\\ 0.33&0.33&-0.33&0.55&-0.33&0.33&0.33\\ 0.55&-0.11&0.11&-0.33&1.00&-0.11&0.11\\ -0.11&0.11&-0.11&0.33&-0.11&1.00&-0.11\\ 0.33&-0.11&0.55&0.33&0.11&-0.11&0.77 \end{bmatrix}$$

$$\begin{bmatrix} -1&-1&-1&-1&-1&-1&-1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&-1&-1&1&-1&-1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&-1&-1&-1&-1&-1&-1&-1 \end{bmatrix} \otimes \begin{bmatrix} 1&-1&1\\ -1&1&-1\\ 1&-1&1 \end{bmatrix} = \begin{bmatrix} 0.33&-0.55&0.11&-0.11&0.11&-0.55&0.33\\ -0.55&0.55&-0.55&0.33&-0.55&0.55&-0.55\\ 0.11&-0.55&0.55&-0.77&0.55&-0.55&0.11\\ -0.11&0.33&-0.77&1.00&-0.77&0.33&-0.11\\ 0.11&-0.55&0.55&-0.77&0.55&-0.55&0.11\\ -0.55&0.55&-0.55&0.33&-0.55&0.55&-0.55\\ 0.33&-0.55&0.11&-0.11&0.11&-0.55&0.33 \end{bmatrix}$$

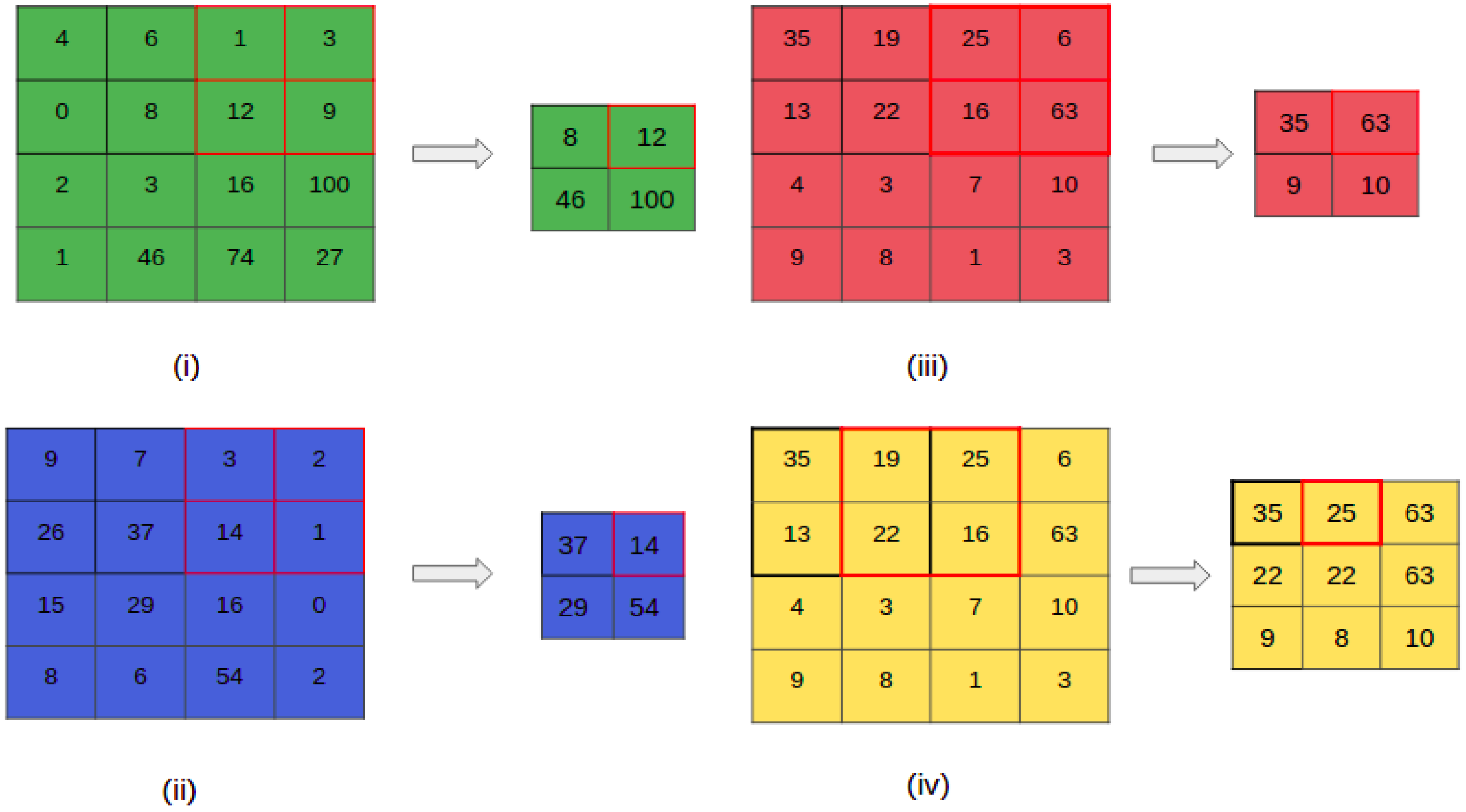
$$\begin{bmatrix} -1&-1&-1&-1&-1&-1&-1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&-1&-1&1&-1&-1&-1&-1\\ -1&-1&-1&1&-1&1&-1&-1&-1\\ -1&-1&1&-1&-1&-1&1&-1&-1\\ -1&1&-1&-1&-1&-1&-1&1&-1\\ -1&-1&-1&-1&-1&-1&-1&-1&-1 \end{bmatrix} \otimes \begin{bmatrix} -1&-1&1\\ -1&1&-1\\ 1&-1&-1 \end{bmatrix} = \begin{bmatrix} 0.33&-0.11&0.55&0.33&0.11&-0.11&0.77\\ -0.11&0.11&-0.11&0.33&-0.11&1.00&-0.11\\ 0.55&-0.11&0.11&-0.33&1.00&-0.11&0.11\\ 0.33&0.33&-0.33&0.55&-0.33&0.33&0.33\\ 0.11&-0.11&1.00&-0.33&0.11&-0.11&0.55\\ -0.11&1.00&-0.11&0.33&-0.11&0.11&-0.11\\ 0.77&-0.11&0.11&0.33&0.55&-0.11&0.33 \end{bmatrix}$$

Pooling

2 kinds

1.Max Pooling

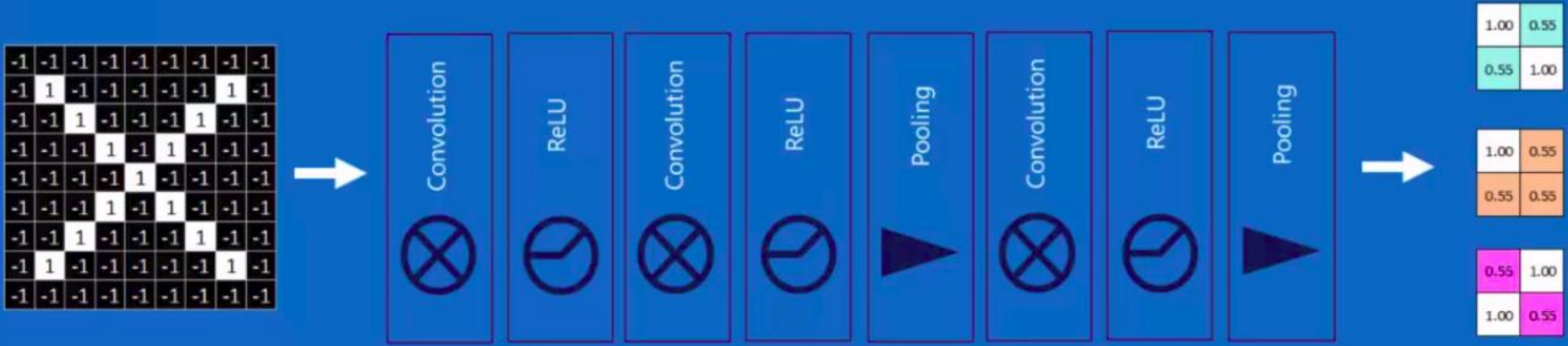
2.Average Pooling.



4 6 1 3
0 8 12 9
2 3 16 100
1 46 74 27
8 12
46 100
(i)
9 7 3 2
26 37 14 1
15 29 16 0
8 6 54 2
37 14
29 54
(ii)
35 19 25 6
13 22 16 63
4 3 7 10
9 8 1 3
35 63
9 10
(iii)
35 19 25 6
13 22 16 63
4 3 7 10
9 8 1 3
35 25 63
22 22 63
9 8 10
(iv)

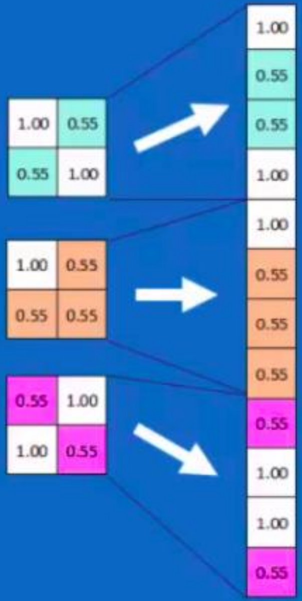
Deep stacking

Layers can be repeated several (or many) times.



Fully connected layer

Every value gets a vote



Various Architectures

AlexNet

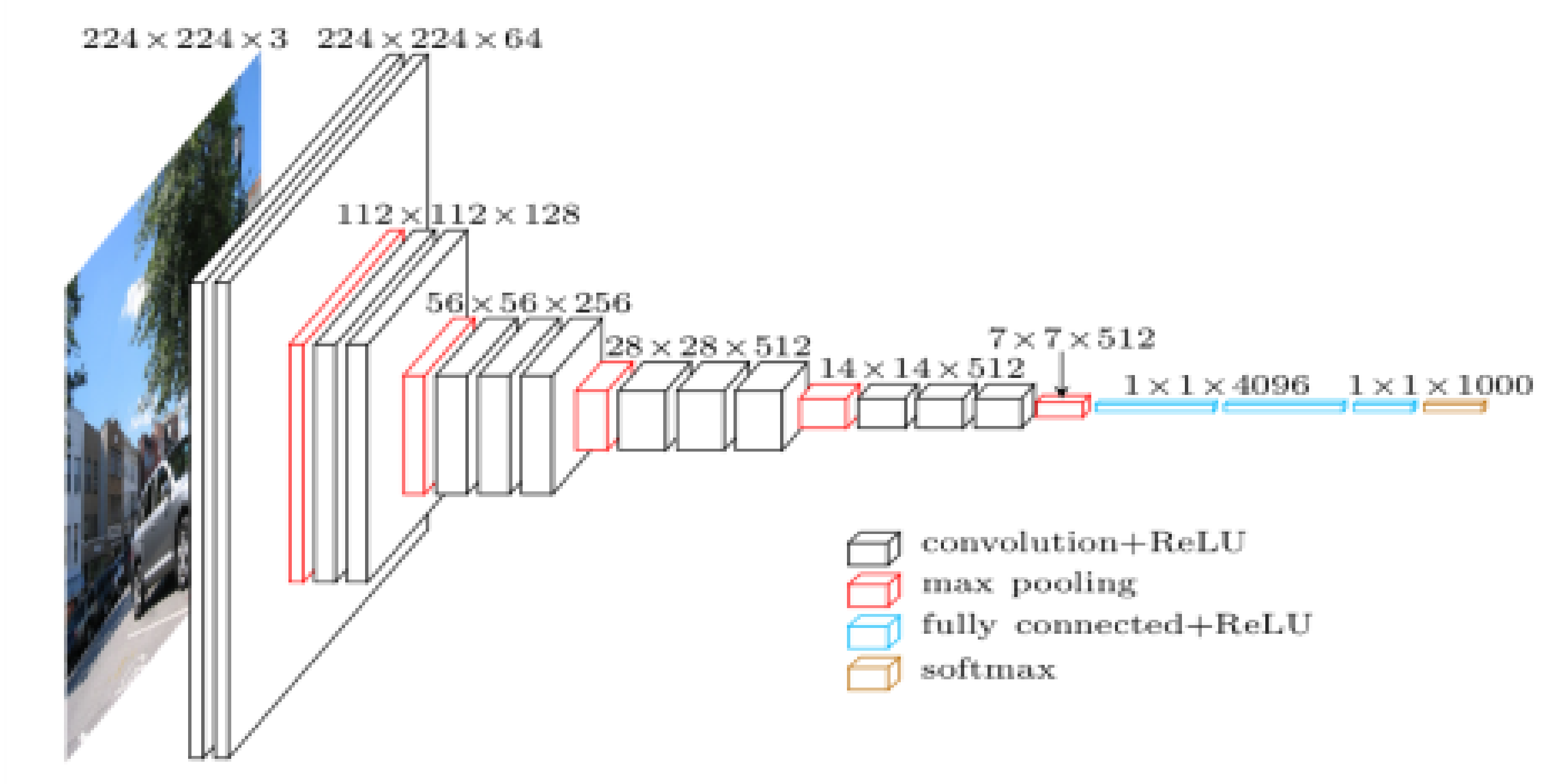
GoogLeNet

ZFNet

VGGNet

For our problem we used a VGGNet which gives a fully connected layer of 4096 features.

VGGNet



224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

PCA

Principal Component Analysis(PCA)

Earlier we got a feature vector of 4096*1

We reduce this to 512*1

Why?

Due to the limitations of computational power.

To speed up our process before we pass it to LSTM or RNN

What is PCA??

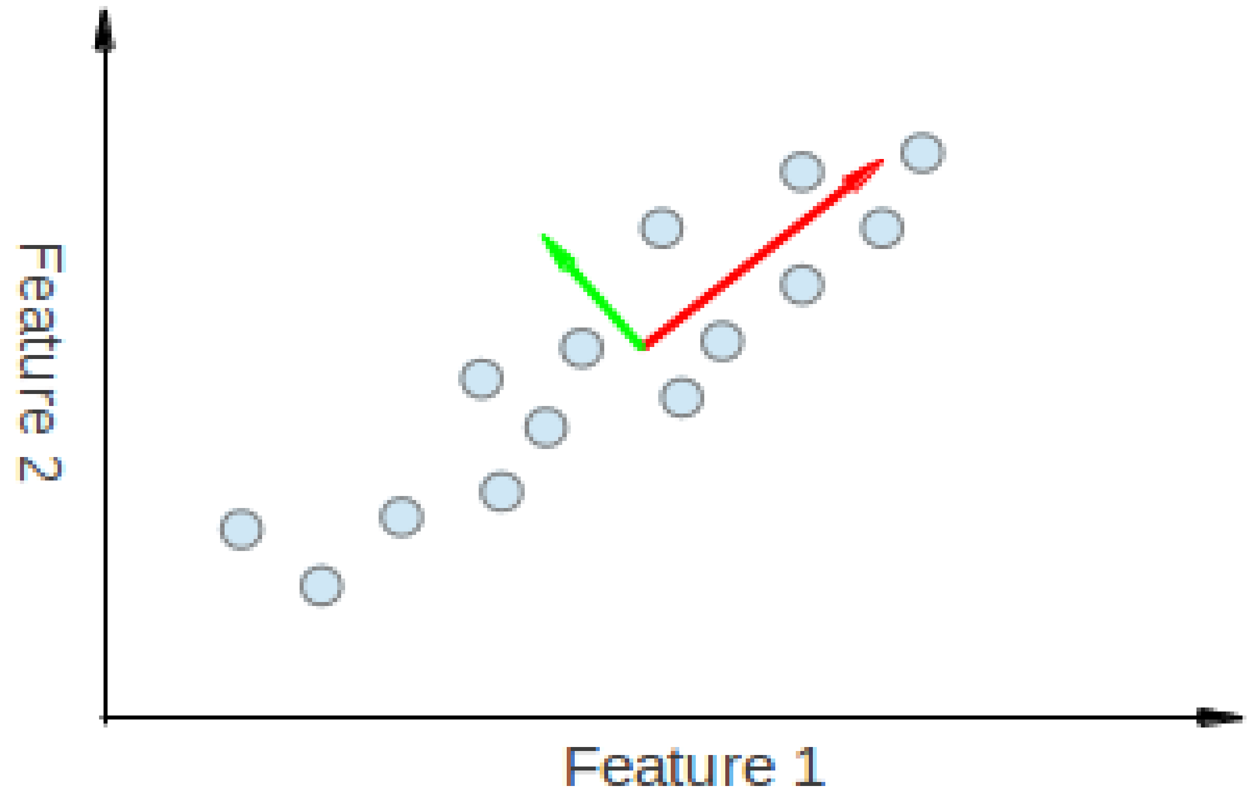
PCA is a dimensionality reduction algorithm that can be used to significantly speed up our feature learning algorithm.

What does PCA do?

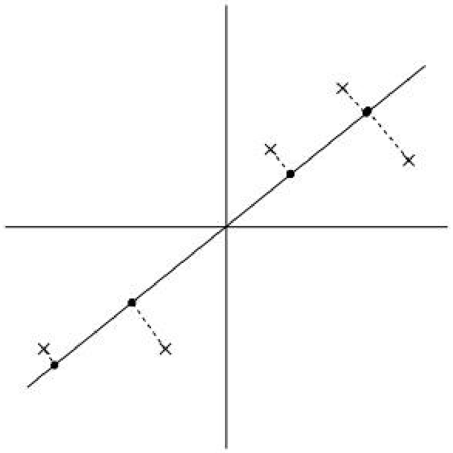
Identifies patterns in data

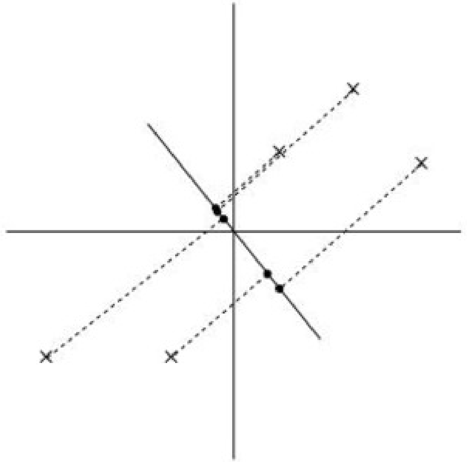
Finally expresses the data in a way that highlights similarities and differences

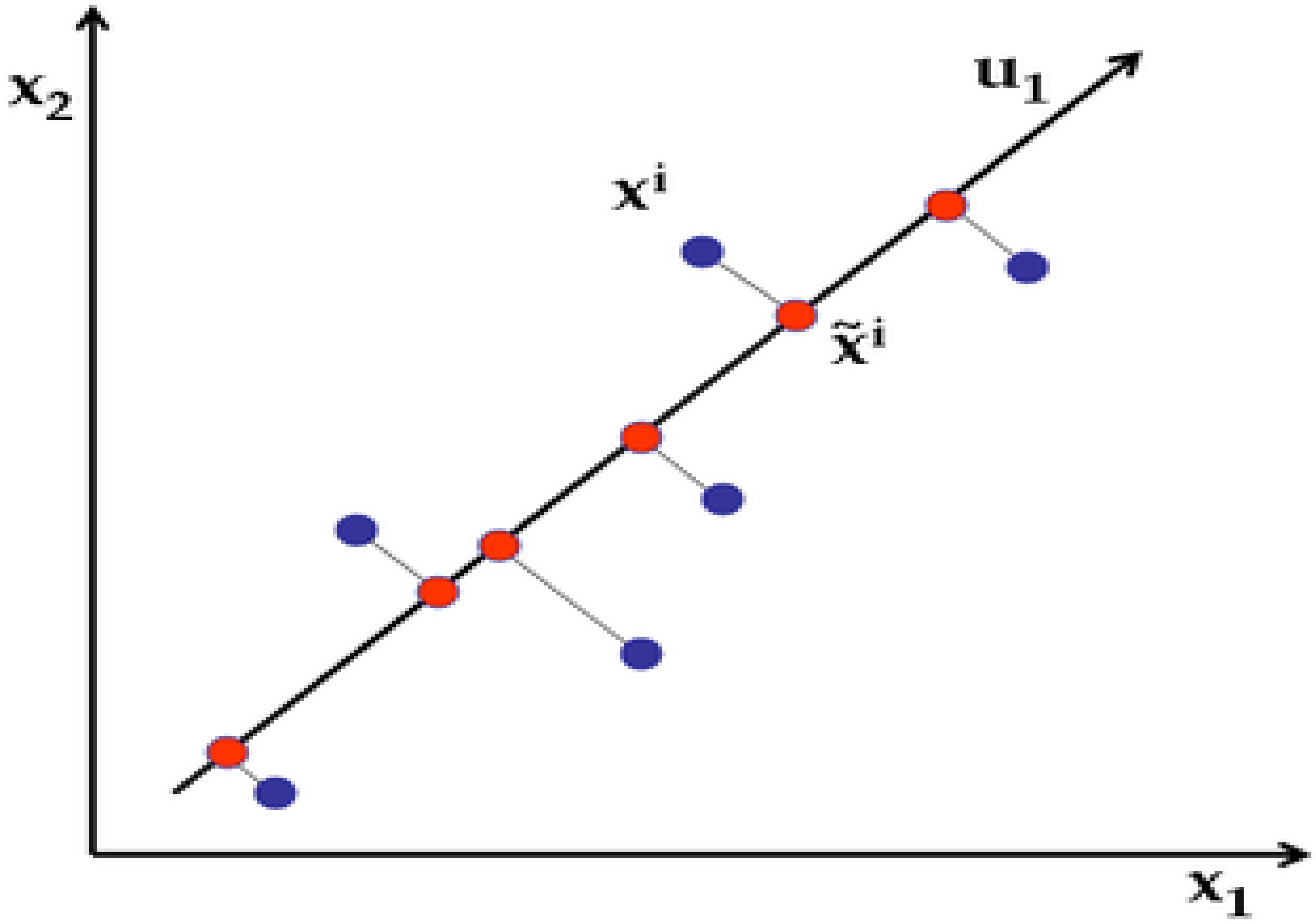
Given Example



Minimize Projection Error






x_2
u_1
x^i
$\tilde{x}^i$
x_1

Data Preprocessing

Normalisation of mean and variance

1. Replace each x (i) with x (i) − μ
2. Replace each x (i) j with x (i) j /σj

   where,

   μ - mean

   σj - standard deviation of the input training examples

Variance

PCA tries to retain most of the variance even after heavily reducing dimensions

PCA did not manipulate data. It just made sure that we use feature vectors that approximately represent the original data by retaining more than 95 or 99% originality of our actual input data

Word Embeddings

Word Embeddings

A **Word Embedding** : word(W)➔vector(V) is a parameterized function mapping words in some language to high-dimensional vectors.

For example, we might find:

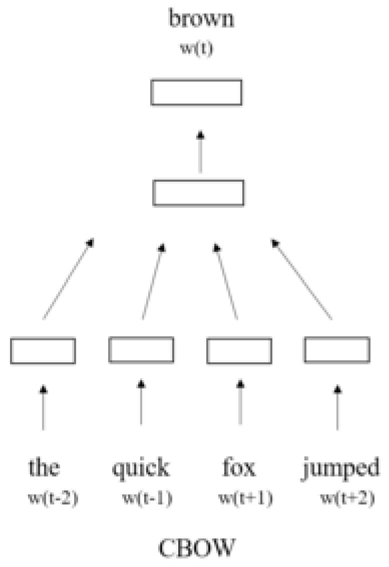
W(“cat")=(0.2, -0.4, 0.7, ...) a 300 dimensional vector

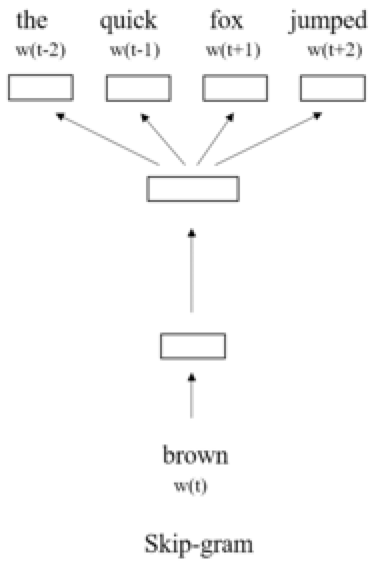
W(“mat")=(0.0, 0.6, -0.1, ...) a 300 dimensional vector

Now how do we obtain this representation ?

Learning Representations

Two good ways,

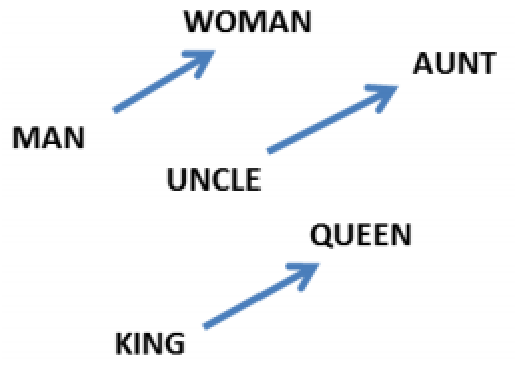




Nearest Words

FRANCE	JESUS	XBOX	REDDISH	SCRATCHED	MEGABITS
AUSTRIA	GOD	AMIGA	GREENISH	NAILED	OCTETS
BELGIUM	SATI	PLAYSTATION	BLUISH	SMASHED	MB/S
GERMANY	CHRIST	MSX	PINKISH	PUNCHED	BIT/S
ITALY	SATAN	IPOD	PURPLISH	POPPED	BAUD
GREECE	KALI	SEGA	BROWNISH	CRIMPED	CARATS
SWEDEN	INDRA	PSNUMBER	GREYISH	SCRAPED	KBIT/S
NORWAY	VISHNU	HD	GRAYISH	SCREWED	MEGAHERTZ
EUROPE	ANANDA	DREAMCAST	WHITISH	SECTIONED	MEGAPIXELS
HUNGARY	PARVATI	GEFORCE	SILVERY	SLASHED	GBIT/S
SWITZERLAND	GRACE	CAPCOM	YELLOWISH	RIPPED	AMPERES

Capturing Relations

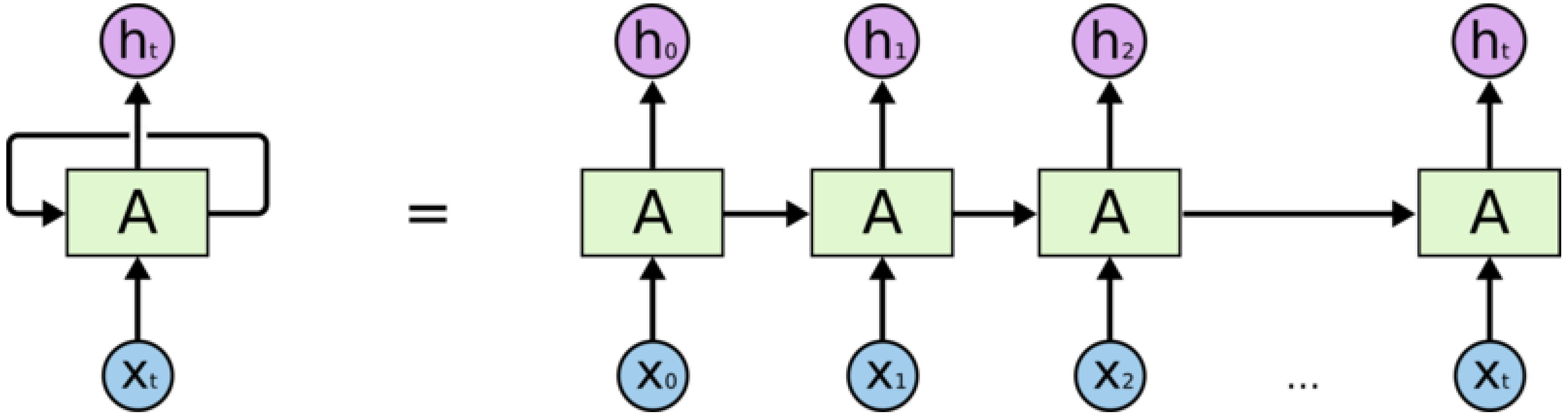


W(“woman")–W(“man") ≃ W(“aunt")–W(“uncle")

W(“queen")–W(“king") ≃ W(“woman")–W(“man")

LSTM

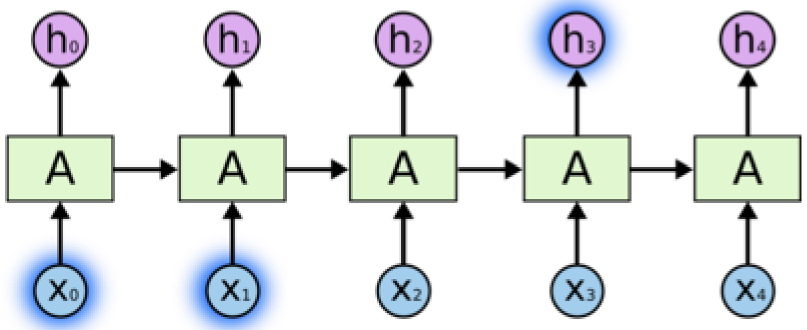
Simple RNN



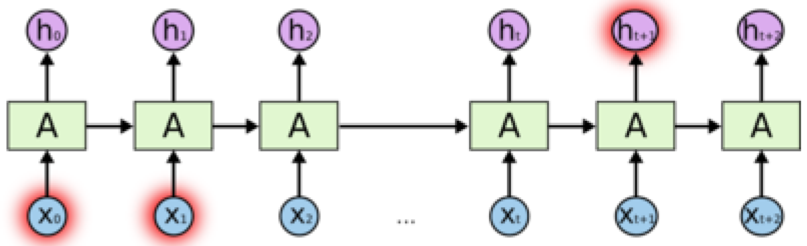
Works well with short term dependencies but not with long term dependencies

Problem with Simple RNN

Short-Term



Clouds are in sky.

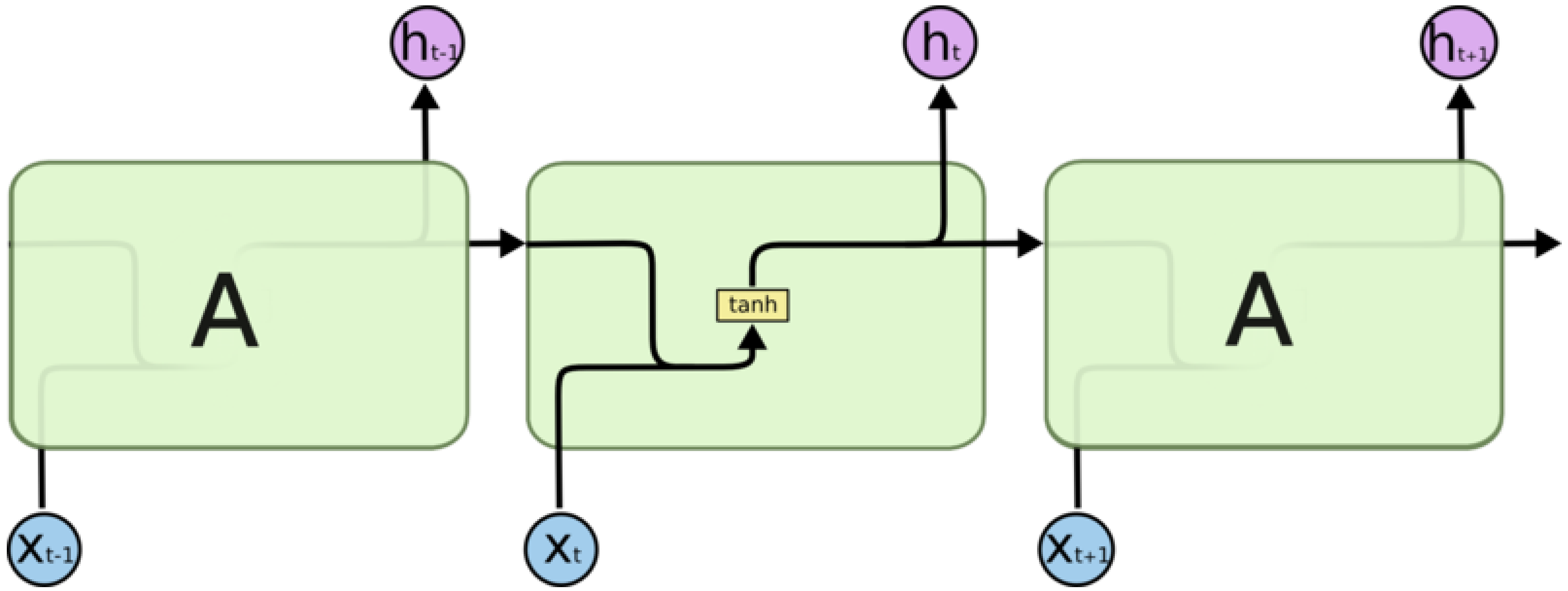


Long-Term

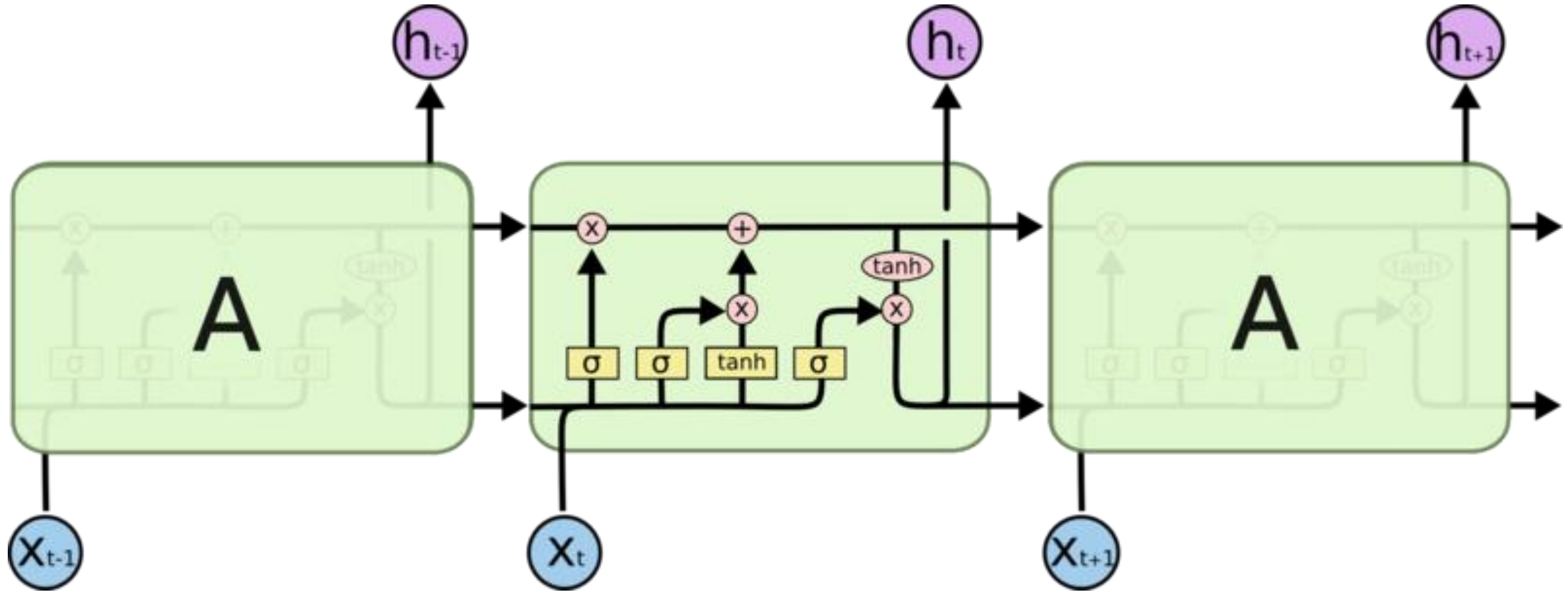
I live in France. ....

I am fluent in ______.

Vanilla RNN



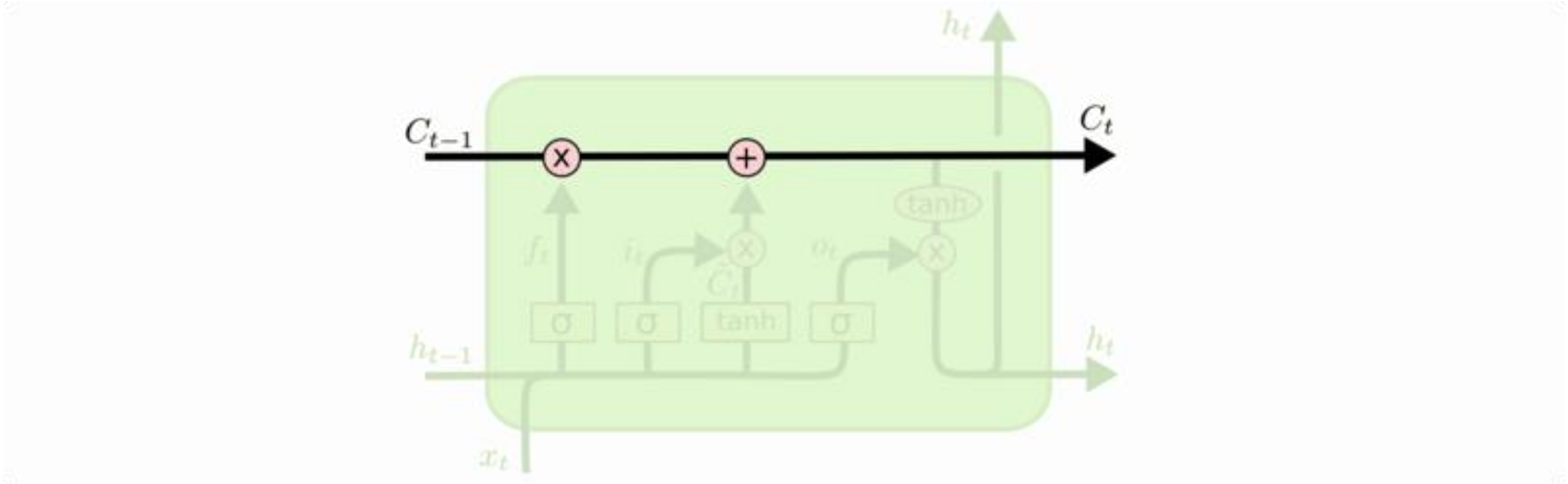
LSTM



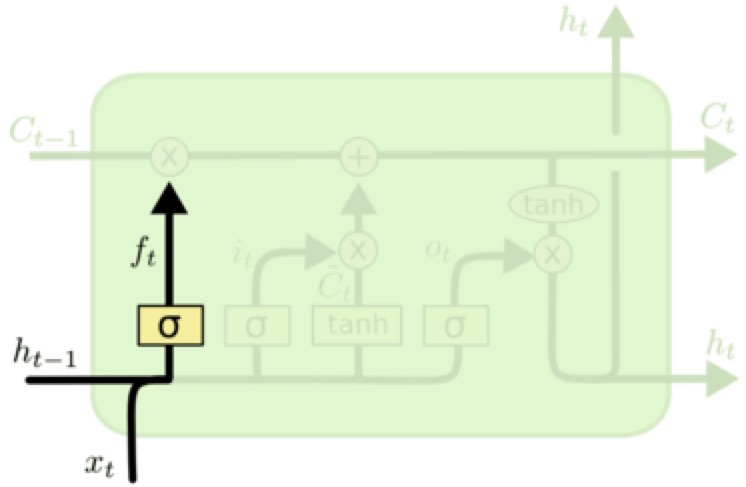
Why Long-Term ?

The key to LSTMs is the cell state, the horizontal line running through the top of the diagram.

It’s very easy for information to just flow along it unchanged

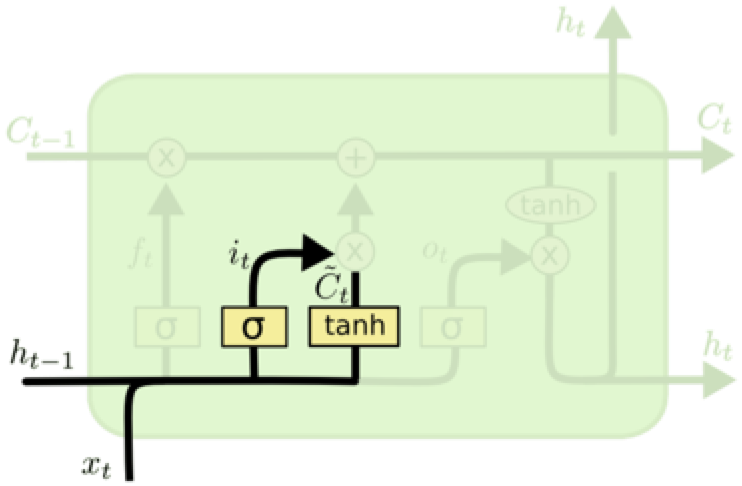


Forget Part



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

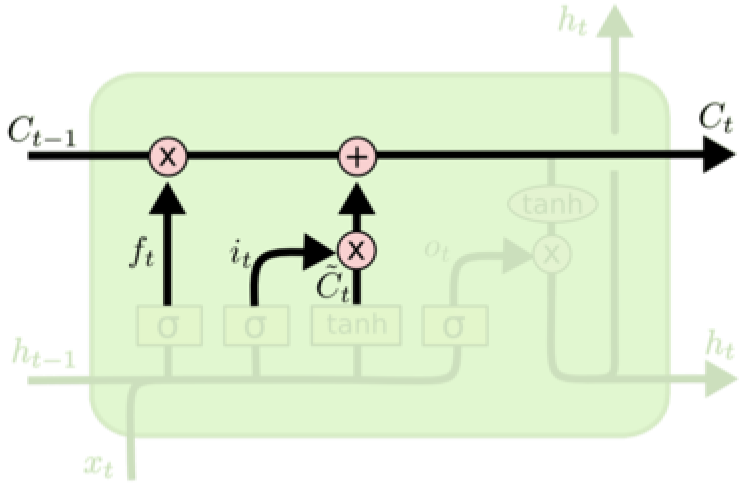
New Info Part



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

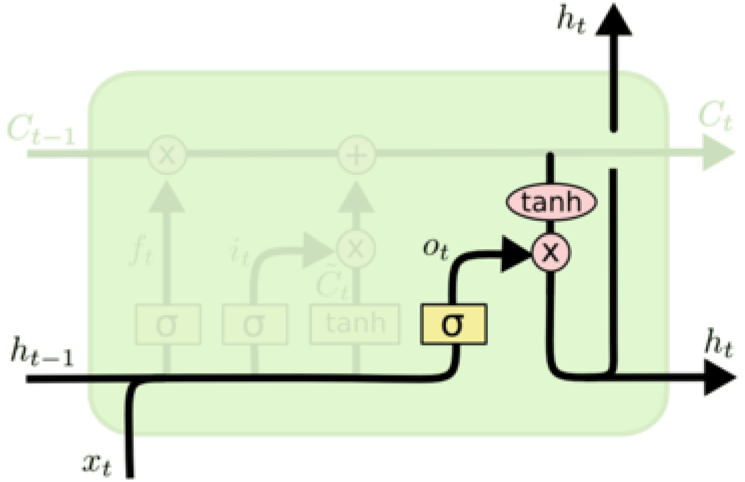
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Adding the Two



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Output



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Model Working

Final Model

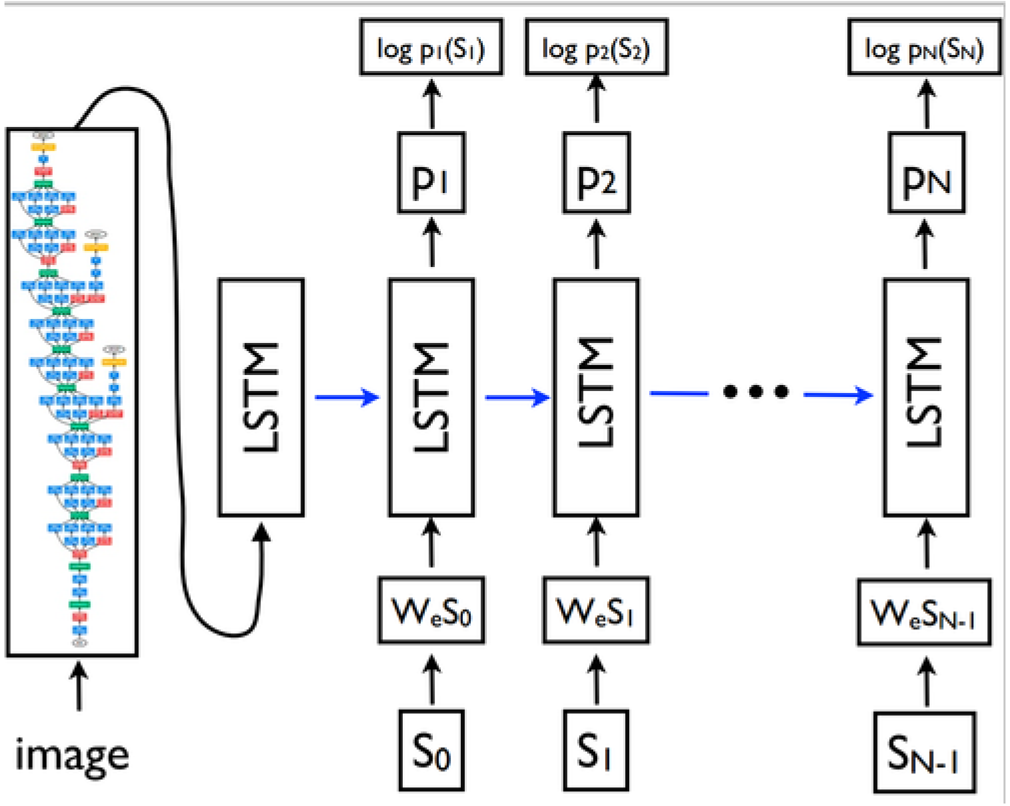
S_0 ➔ START word

S_n ➔ END word

$x_{-1} = \text{CNN}(I)$

$x_t = W_e S_t, \quad t \in \{0 \ldots N-1\}$

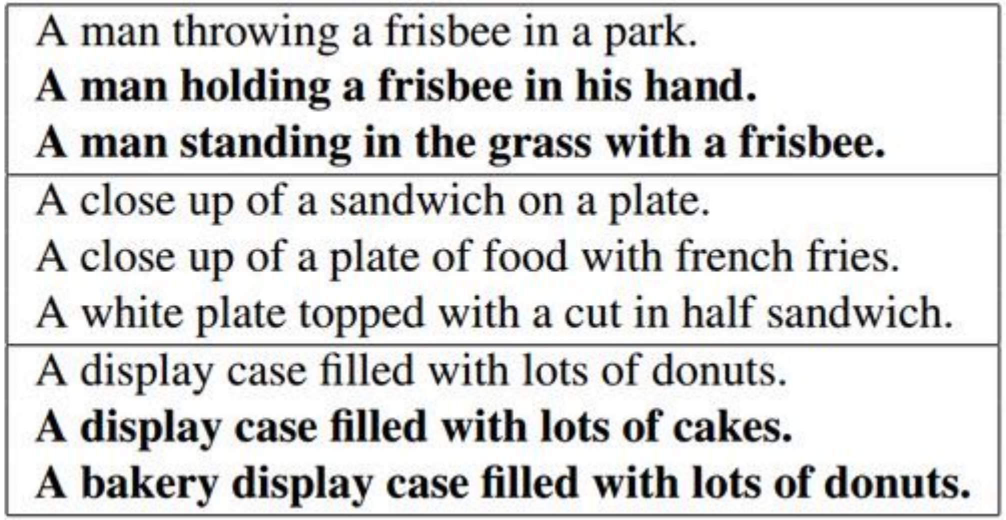
$p_{t+1} = \text{LSTM}(x_t), t \in \{0 \ldots N-1\}$



Training and Evaluation

Since all sentences mean the same thing we can’t have evaluation function which checks word by word.

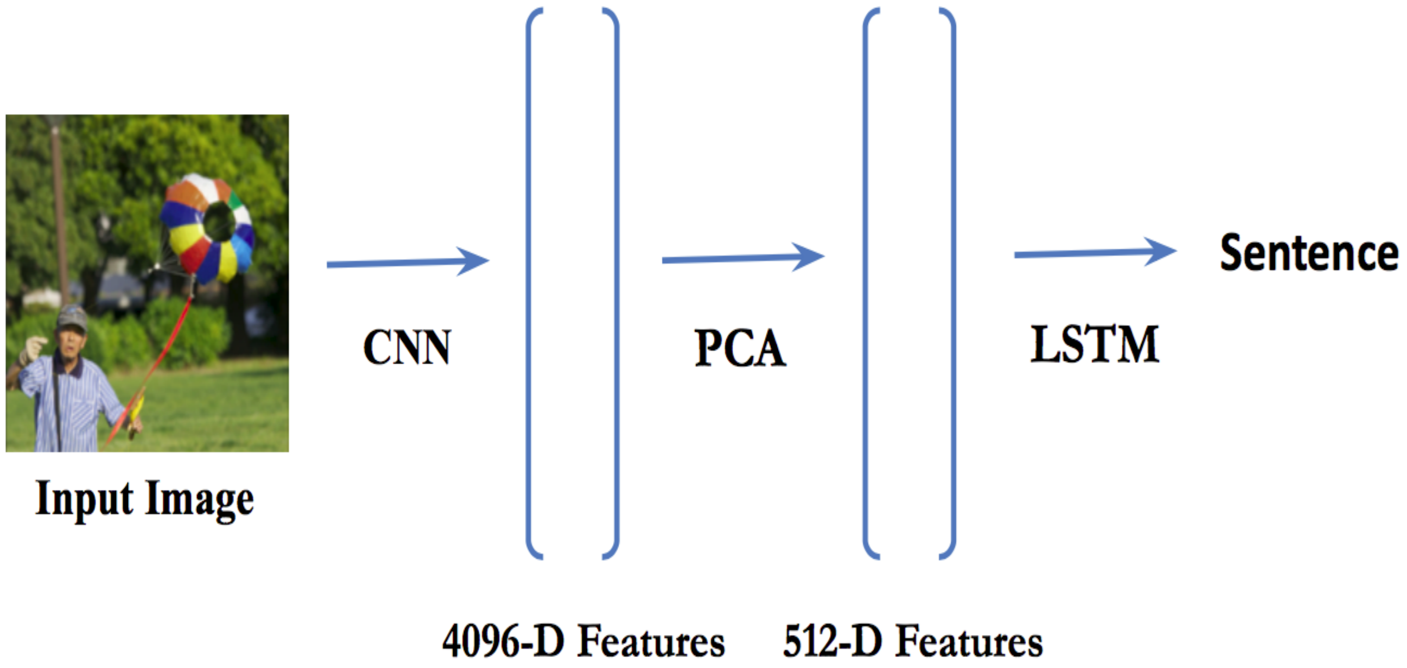
Hence, a metric called **BLEU score** will be used which checks presence or absence of particular words as well as ordering, or we could ask human evaluators to assign a score based on appropriate captioning.



A man throwing a frisbee in a park.
A man holding a frisbee in his hand.
A man standing in the grass with a frisbee.
A close up of a sandwich on a plate.
A close up of a plate of food with french fries.
A white plate topped with a cut in half sandwich.
A display case filled with lots of donuts.
A display case filled with lots of cakes.
A bakery display case filled with lots of donuts.

Summary

Conclusion


Input Image
CNN
4096-D Features
PCA
512-D Features
LSTM
Sentence

Conclusion ..contd.

We have presented a deep learning model that automatically generates image captions. Our described model is based on a CNN that encodes an image into a compact representation, followed by a RNN that generates corresponding sentences based on the learned image features. The generated captions are highly descriptive of the objects and scenes depicted on the images.

Applications

1. Smartphones made it possible for the visually impaired to take images of their surroundings. These images can be used to generate captions that can be read out loud to give visually impaired people a better understanding of their surroundings.

2. Social media platforms like Facebook can infer directly from the image, where you are (beach, cafe etc), what you wear (color) and what you’re doing also (in a way) when you upload a image.

3. Can be extended to Video Captioning with some extra work.

The End