Object Detection using Neural Networks

By D. Naveen Reddy

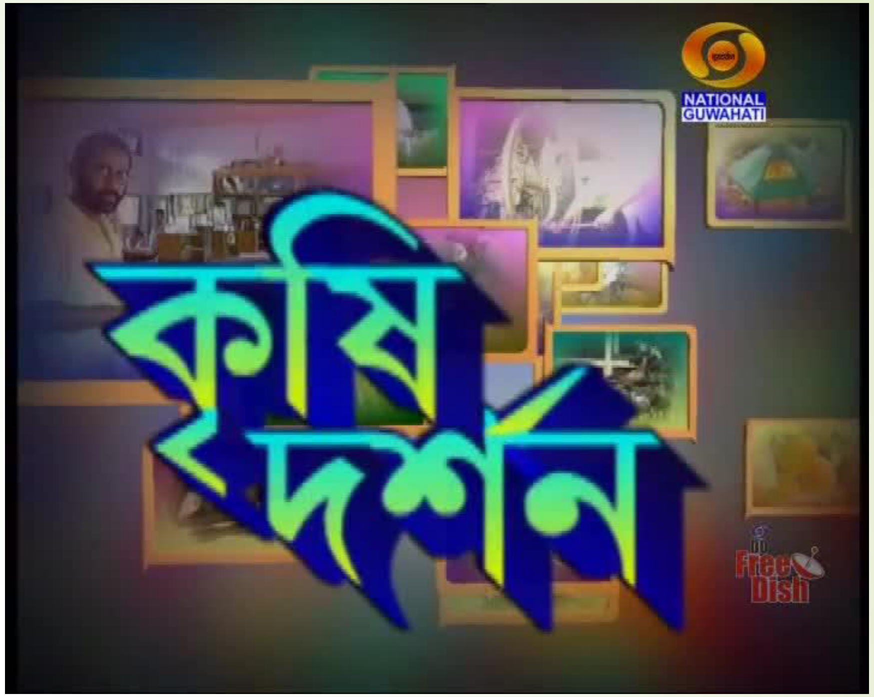
Tasks

- Detection of a Logo in a TV Feed.
- Recognition of the Logo.

Why ?

- ✓ To know which House Promo is being played and to categorize it automatically.
- ✓ This also leads us to the question “House Promo or Commercial”.

लौटेंगे : 00:22
colors
CHAKRAVARTIN
Ashoka
SAMRAT
1 HOUR SPECIAL
आज 9pm

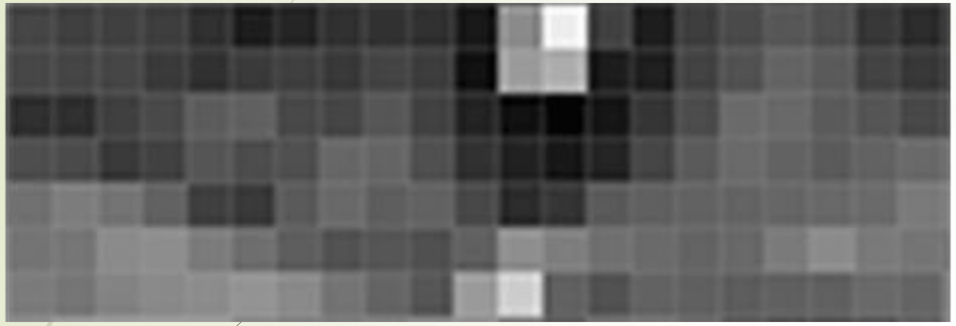


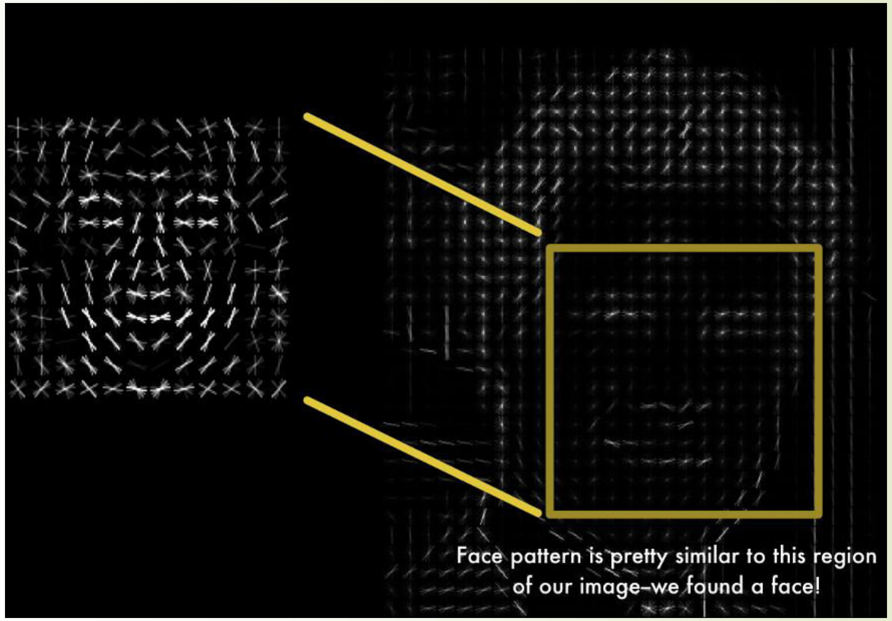
NATIONAL
GUWAHATI
কৃষি
দর্শন
Free Dish

UDAYA
COMEDY

AMRITA
അളിയൻ
VS
അളിയൻ

Initial Approach : HOG + SVM





- HOG - Histogram of Oriented Gradients

1. Get P positive samples from the training data and extract HOG descriptors from these samples.
2. Get N negative samples from a negative training set that does not contain the objects.
3. Train SVM for the positive and negative samples.

Continued..

4. Hard-negative mining.

→ Run the sliding window on every image of the negative training set.

- At each window, compute HOG descriptors and apply the classifier. If classified incorrectly :

  i) Record the feature vector associated with the FP patch.

  ii) Along with probability of the classification.

5. Take the FP samples from previous stage and sort by the confidence(or probability) and train the classifier again.

6. For the test dataset, extract HOG descriptors and apply the classifier.

- If classifier detects and object with sufficiently large probability, record the bounding box of the window.

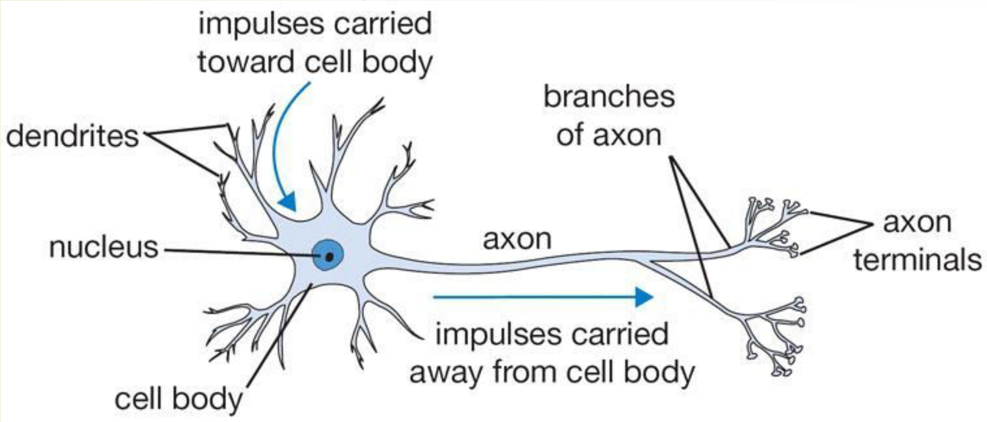
Testing for a Particular Show

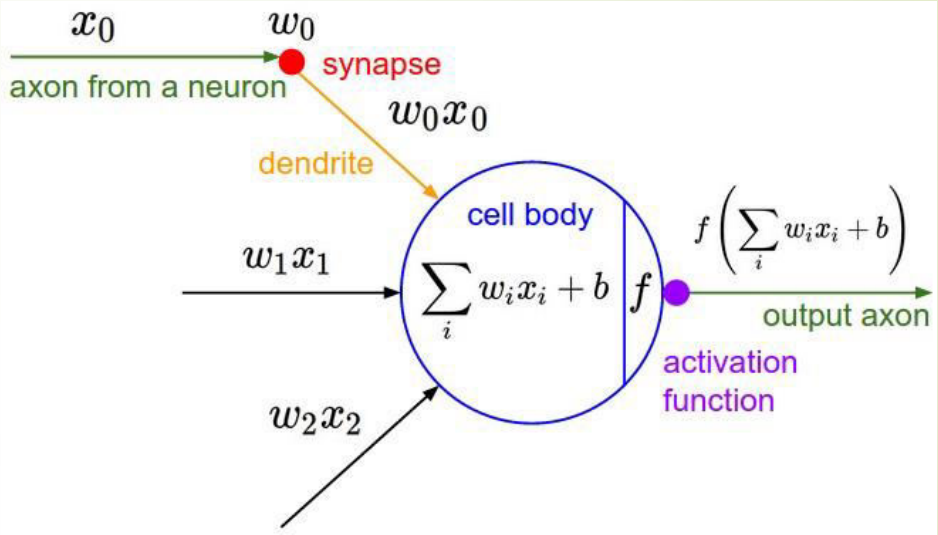
- 10 images were of the show "Dil Bole Oberoi" were given are training images and other unseen 6 images were used for testing.
- It was observed that 2 out of 6 images gave positive results and the right region of the logo was detected.

Next Approach :
Using Convolutional Neural Networks

→ Introduction to Neural Networks.

Introduction to Neural Networks

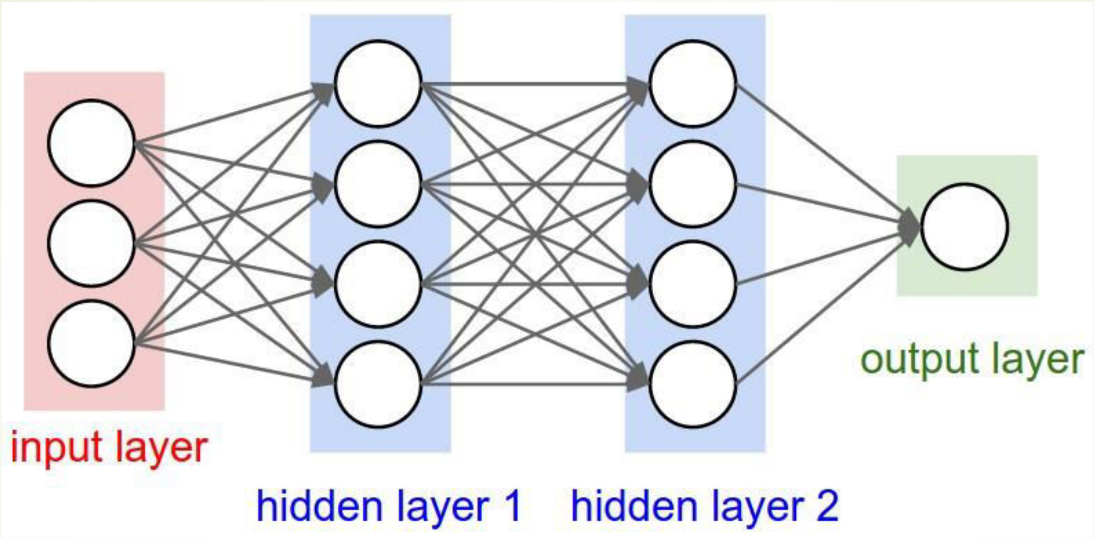




An illustration of a biological neuron (left) and its mathematical model (right).

- An Artificial Neural Network (ANN) is a computational model that is inspired by the way biological neural networks in the human brain process information.
- Neurons – Building Blocks

Networks of Neurons



- The power of neural networks come from their ability to learn the representation in your training data and how to best relate it to the output variable that you want to predict.
- In this sense neural networks learn a mapping. (Function)
- Mathematically, they are capable of learning any mapping function and have been proven to be a universal approximation algorithm.

Training Networks - Example

6,148,72,35,0,33.6,0.627,50,1
1,85,66,29,0,26.6,0.351,31,0
8,183,64,0,0,23.3,0.672,32,1
1,89,66,23,94,28.1,0.167,21,0
0,137,40,35,168,43.1,2.288,33,1

- *Patients medical record data and whether they had an onset of diabetes within five years.*

1. Number of times pregnant.
2. Plasma glucose concentration a 2 hours in an oral glucose tolerance test.
3. Diastolic blood pressure (mm Hg).
4. Triceps skin fold thickness (mm).
5. 2-Hour serum insulin (mu U/ml).
6. Body mass index.
7. Diabetes pedigree function.
8. Age (years).
9. Class, onset of diabetes within five years.

1. For this example, we use a training algorithm for neural networks is called **stochastic gradient descent**.
2. This is where one row of data is exposed to the network at a time as input. The network processes the input upward activating neurons as it goes to finally produce an output value. This is called a **forward pass** on the network.
3. The output of the network is compared to the expected output and an error is calculated. This error is then propagated back through the network, one layer at a time, and the weights are updated according to the amount that they contributed to the error. This is called the **backpropagation algorithm**.
4. The process is repeated for all of the examples in your training data.

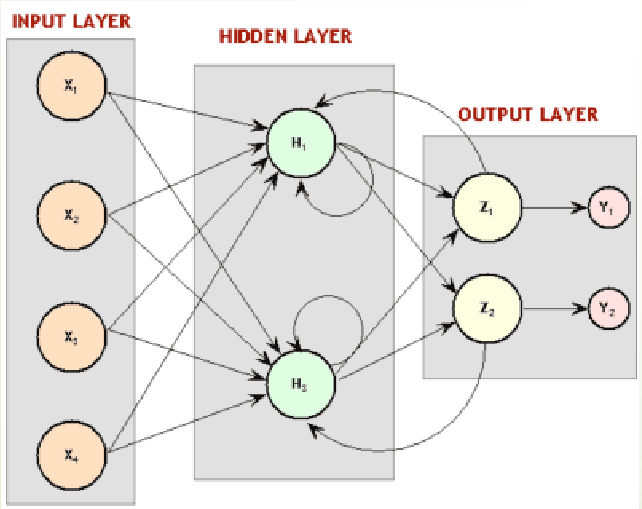
Convolutional Neutral Networks

- A **convolutional neural network** (**CNN**, or **ConvNet**) is a class of deep, feed-forward artificial neural network that have successfully been applied to **analyzing visual imagery**.

Recurrent neural network

- A **recurrent neural network** (**RNN**) is a class of artificial neural network where connections between units form a directed cycle.

→ RNNs can use their internal memory to process arbitrary sequences of inputs

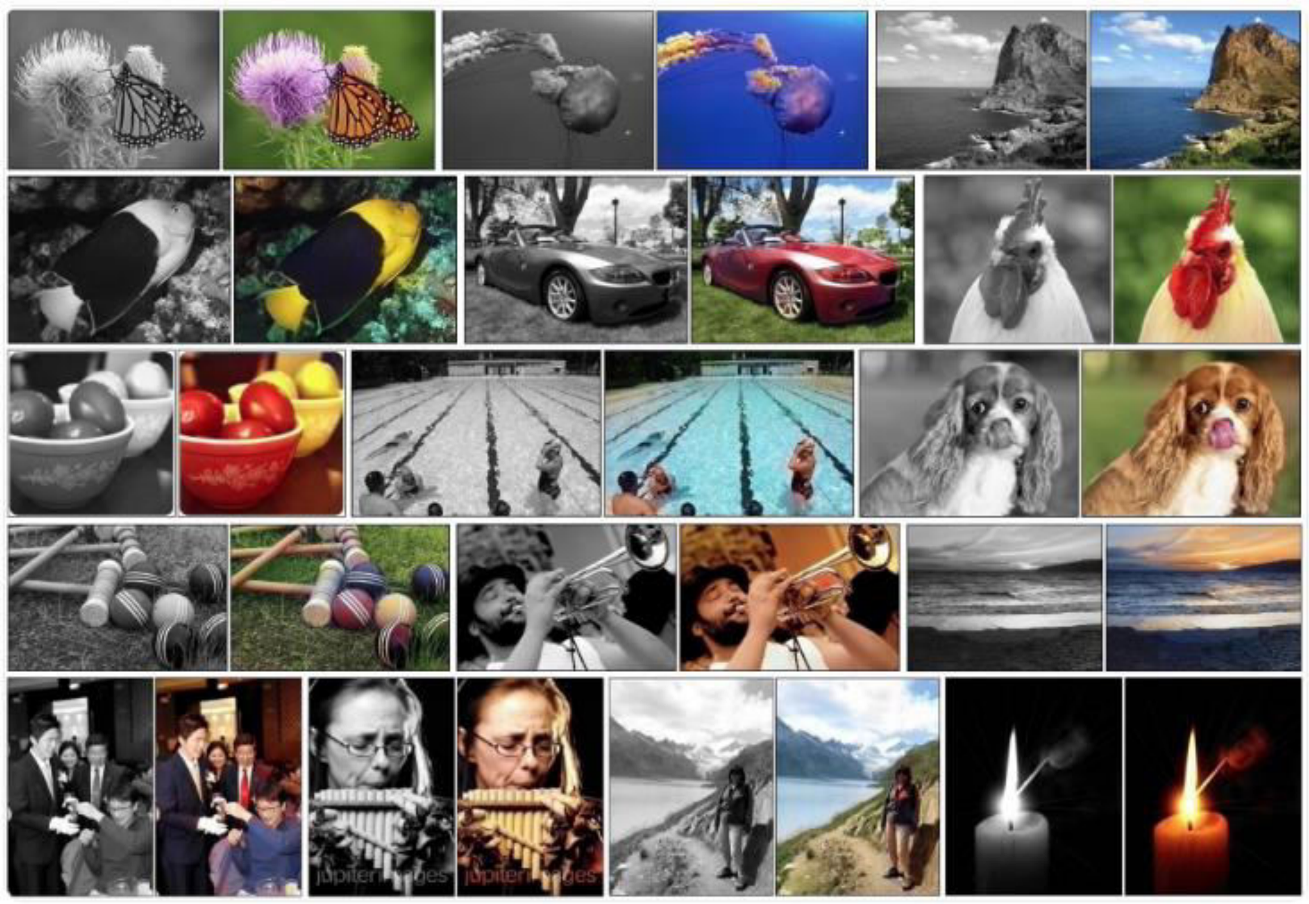


Applications :

1. Unsegmented, connected handwriting recognition
2. Speech recognition

Interesting Applications

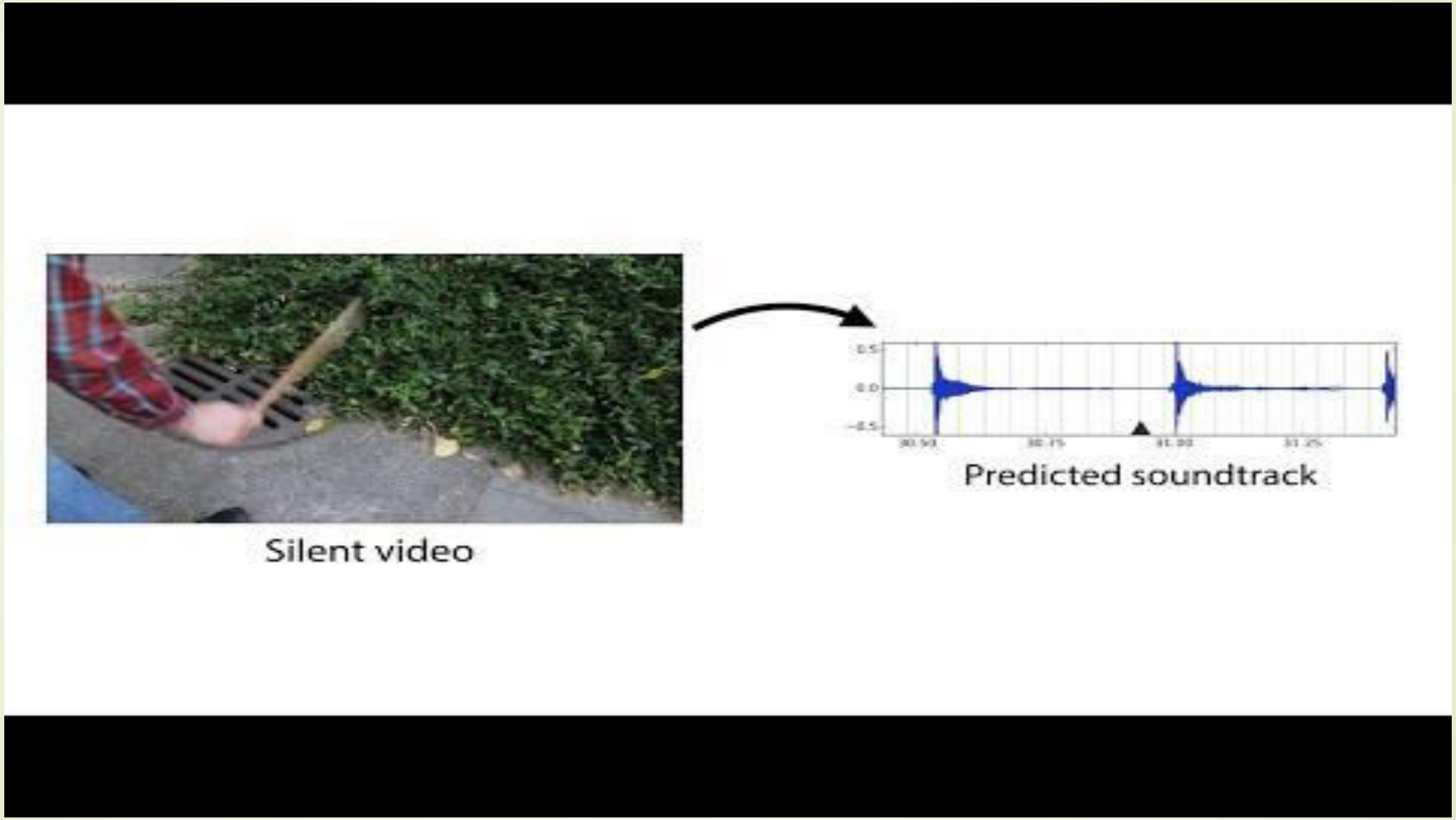
- **1. Automatic Colorization of Black and White Images**



Using very large convolutional neural networks

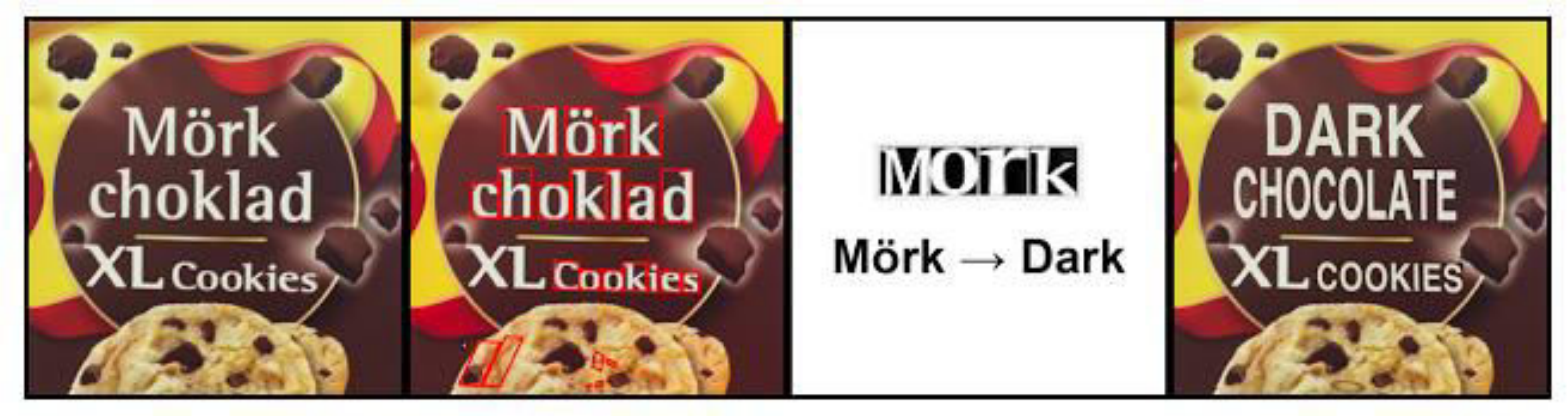
Automatically Adding Sounds To Silent Movies

- *Using both convolutional neural networks and LSTM recurrent neural networks.*



Automatic Machine Translation

- Automatic Translation of Text.
- Automatic Translation of Images.

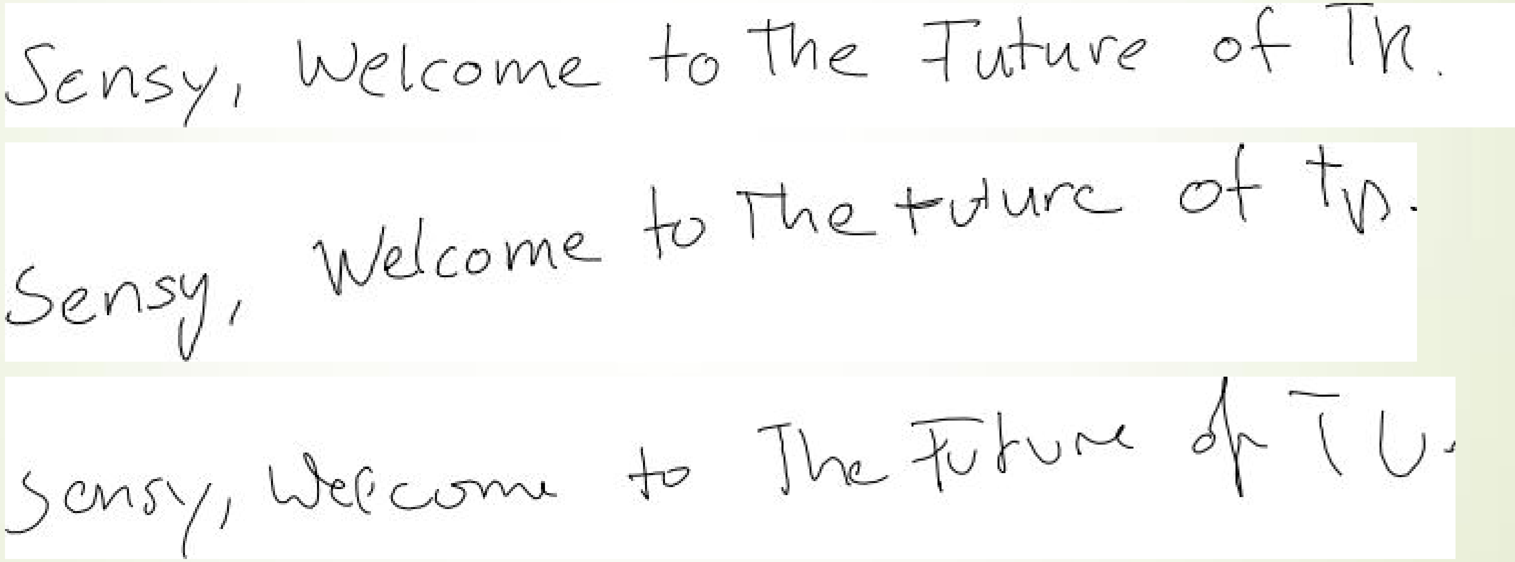


CNNS + Stacked networks of large LSTM recurrent neural networks

→Instant Visual Translation

Automatic Handwriting Generation

- *Using Recurrent Neural Networks*



Interactive Demo :
http://www.cs.toronto.edu/~graves/handwriting.html

Automatic Image Caption Generation

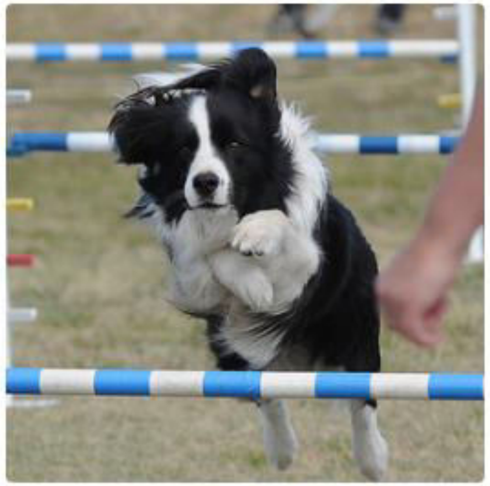
CNNS + Recurrent Neural Networks

"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

"two young girls are playing with lego toy."

"girl in pink dress is jumping in air."



"black and white dog jumps over bar."

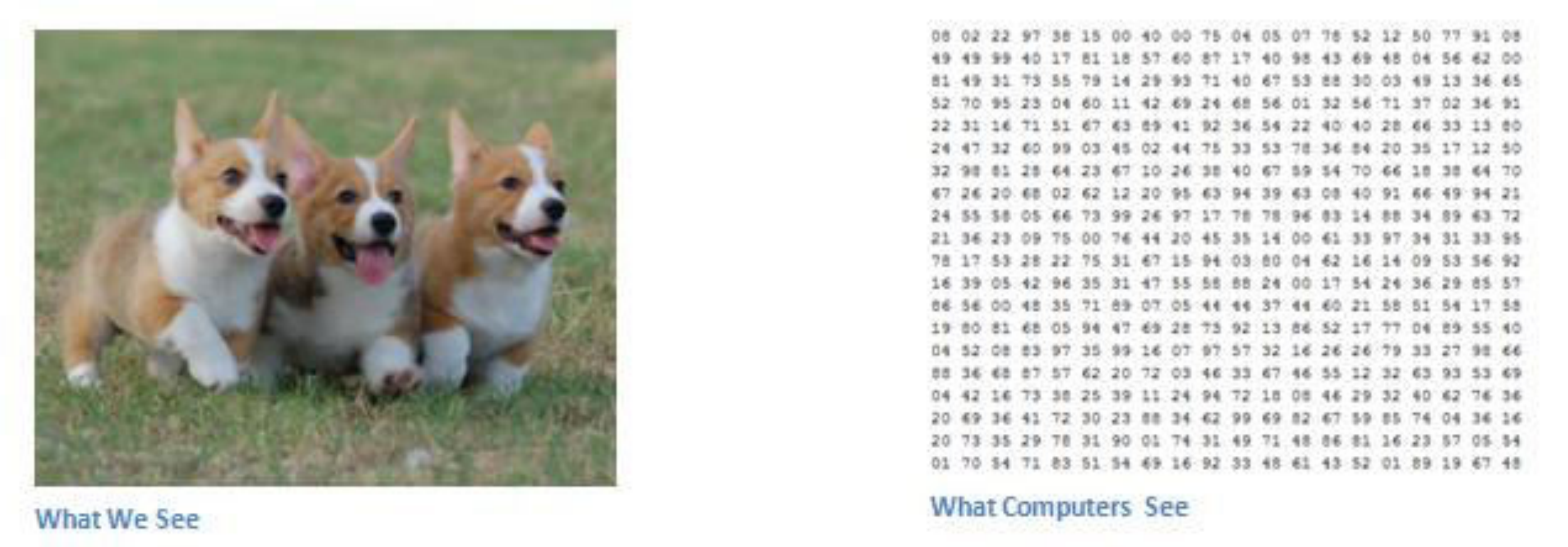
"young girl in pink shirt is swinging on swing."

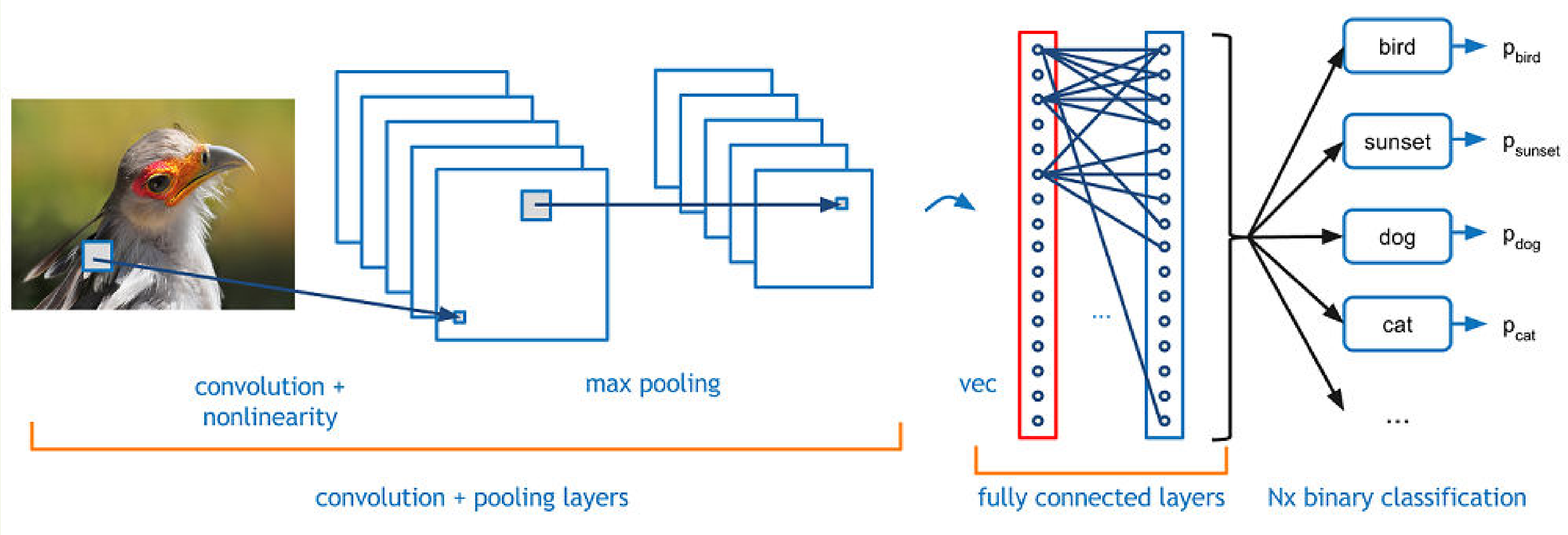
Others.

- Automatic Text Generation
- Automatic Game Playing
- Object Classification and Detection in Photographs

1. **Facebook** uses neural nets for their automatic tagging algorithms,
2. **Google** for their photo search,
3. **Amazon** for their product recommendations,
4. **Pinterest** for their home feed personalization,
5. **Instagram** for their search infrastructure.

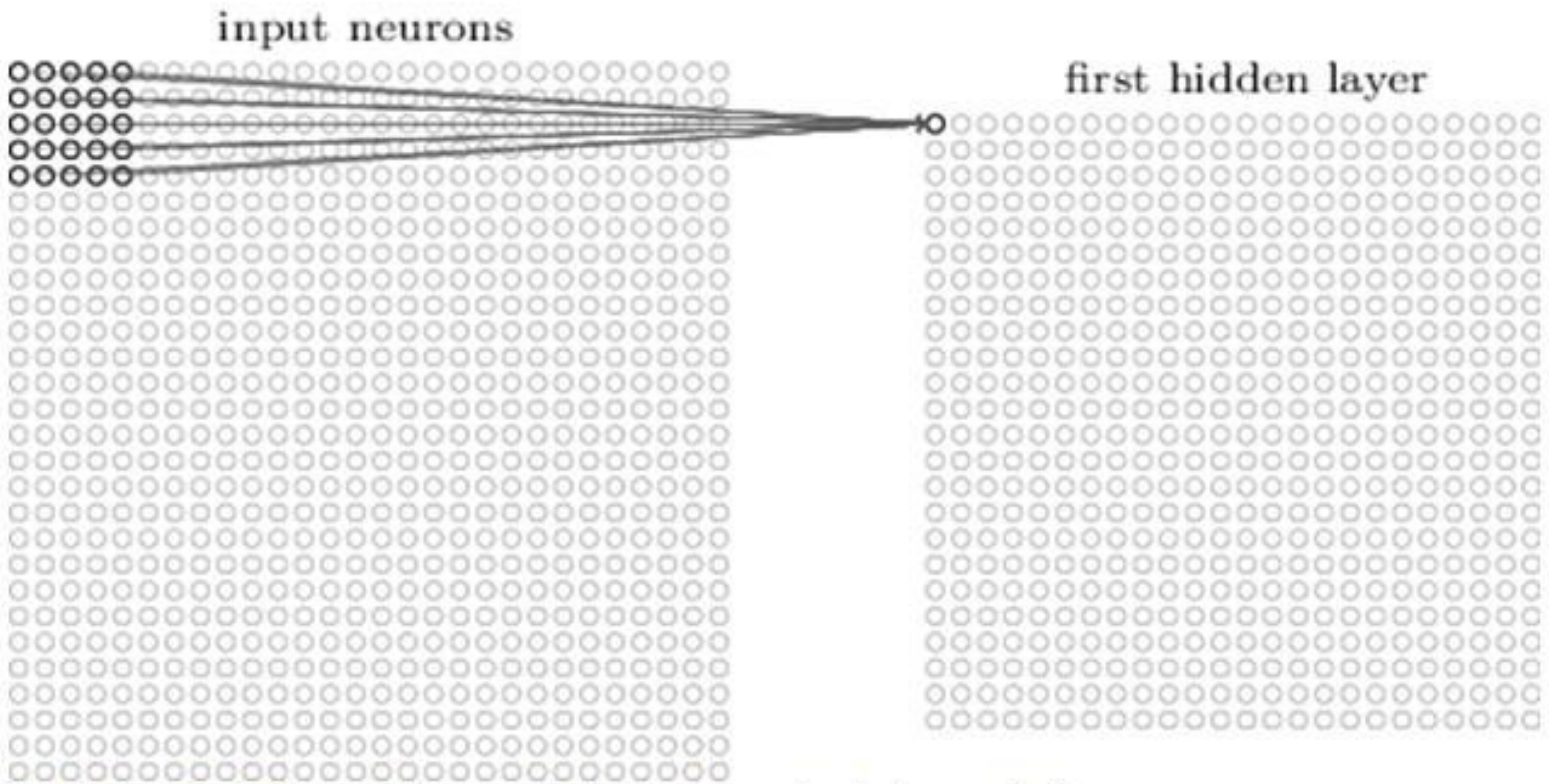
CNNs


What We See
What Computers See


convolution + nonlinearity
max pooling
vec
bird
sunset
dog
cat
...
p_{bird}
p_{sunset}
p_{dog}
p_{cat}
convolution + pooling layers
fully connected layers
Nx binary classification

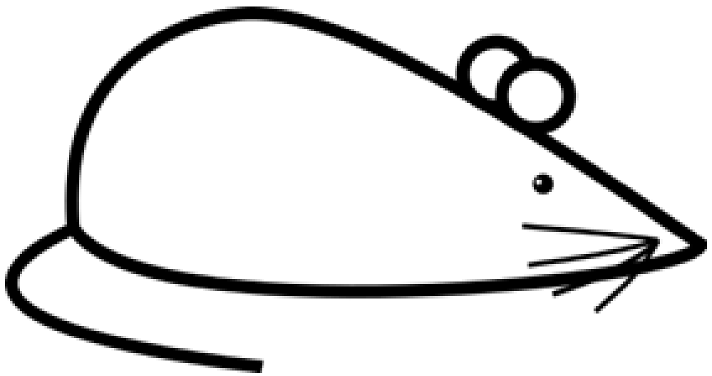
Layers

- Convolution Layers

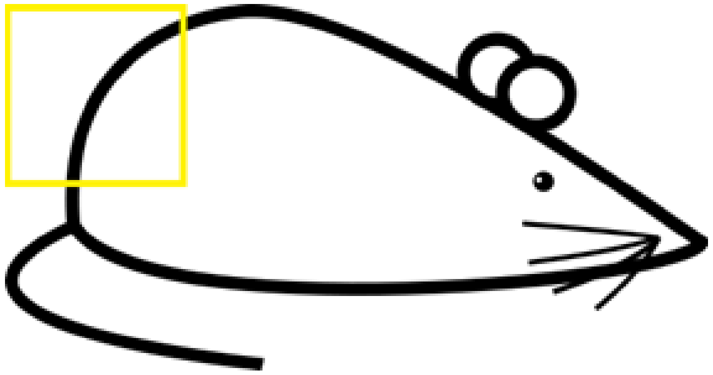


Visualization of 5 x 5 filter convolving around an input volume and producing an activation map

By using more filters, we are able to preserve the spatial dimensions better.



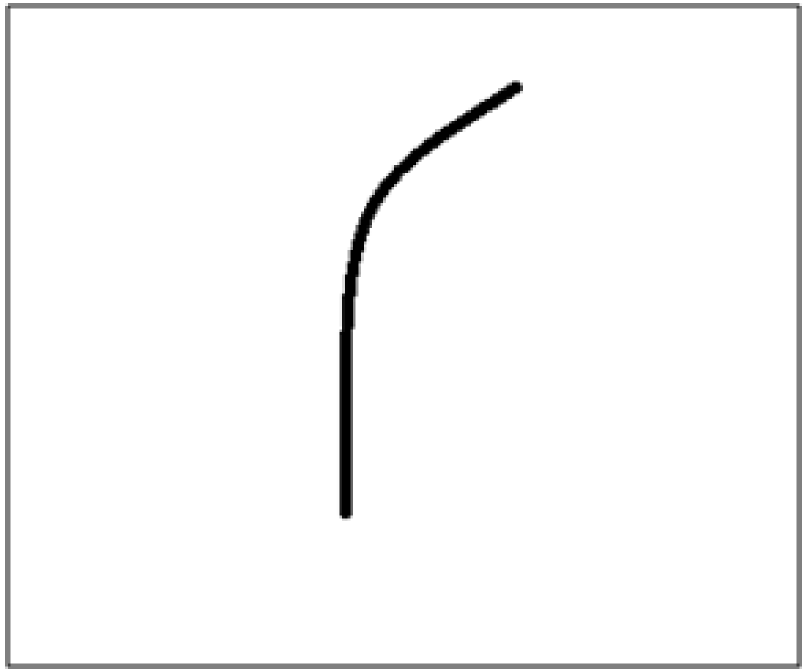
Original image



Visualization of the filter on the image

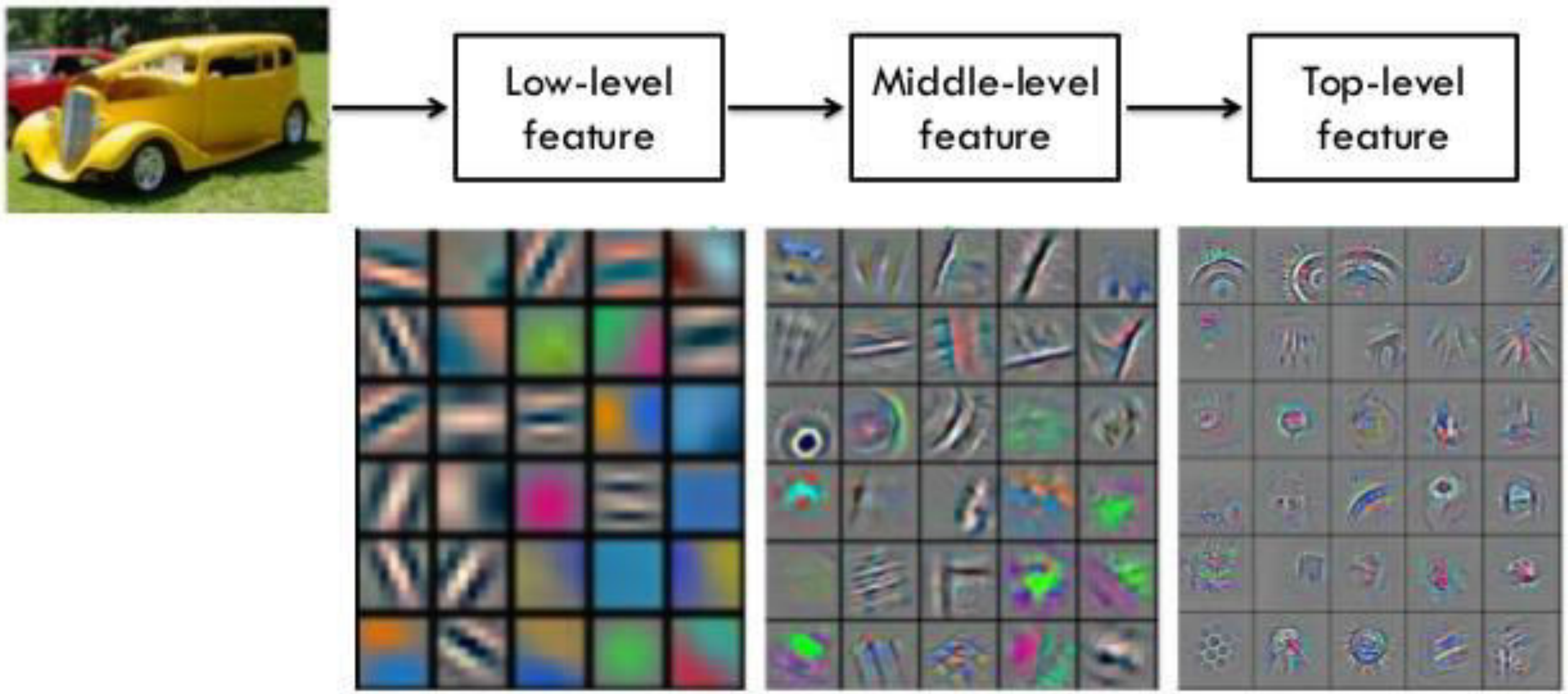
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter



Visualization of a curve detector filter

Hierarchy of trained representations



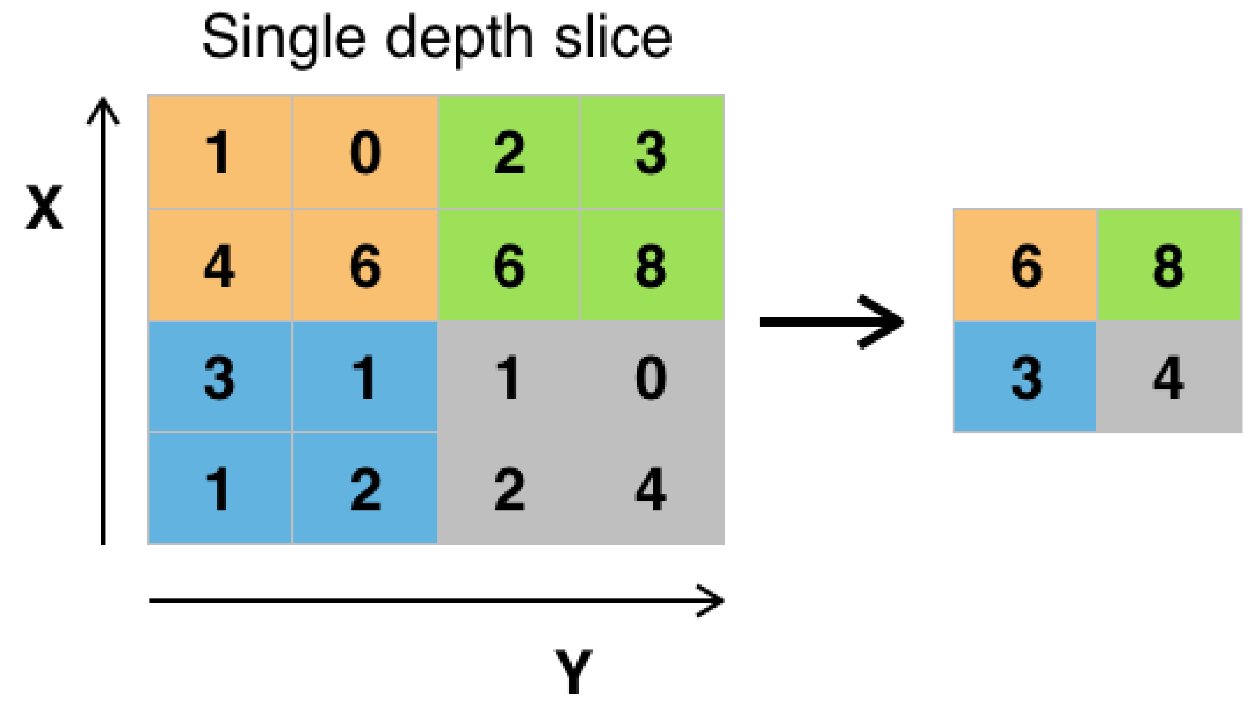
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

Fully Connected Layer

- Now that we can detect these high level features, we attach a **fully connected layer** to the end of the network.
- This layer basically takes an input volume (whatever the output is of the conv or ReLU or pool layer preceding it) and outputs an N dimensional vector where N is the number of classes that the program has to choose from.
- For example, if you wanted a digit classification program, N would be 10 since there are 10 digits.
- Each number in this N dimensional vector represents the probability of a certain class.

Max Pooling

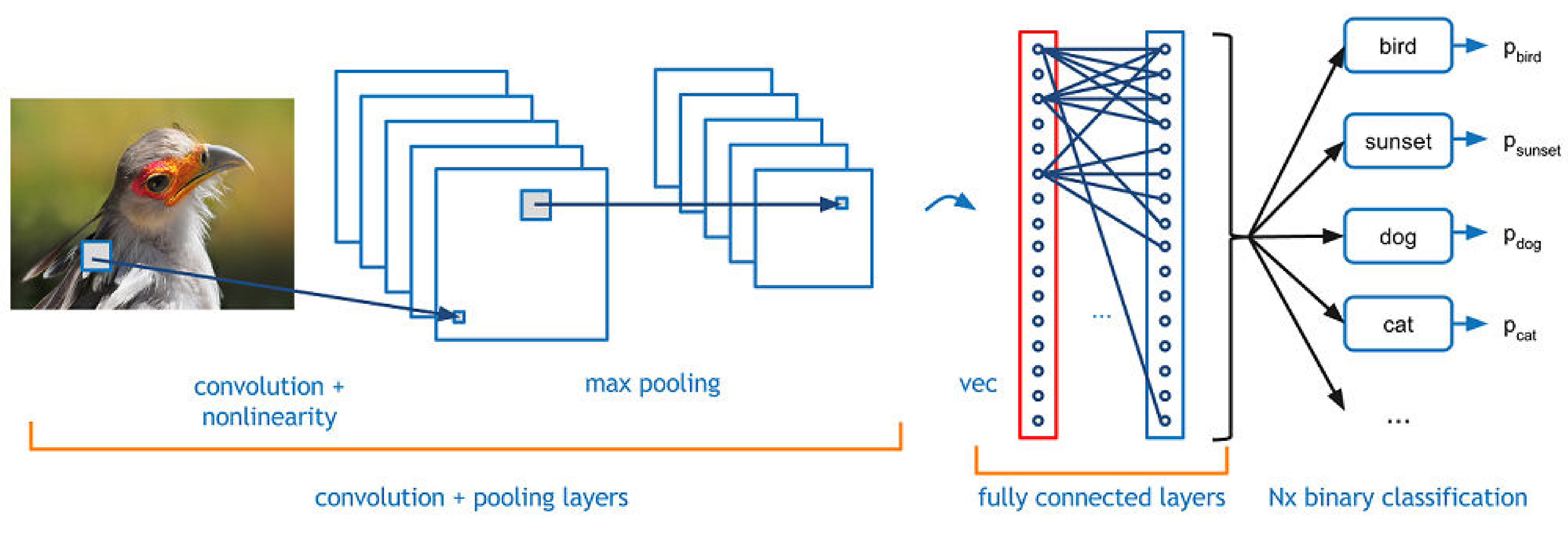
- A **pooling layer** is also referred to as a downsampling layer.



Example of Maxpool with a 2x2 filter and a stride of 2

Basic Architecture

Input -> Conv -> ReLU -> Conv -> ReLU -> Pool -> ReLU -> Conv -> ReLU -> Pool ->Fully Connected



Concept Of Transfer Learning

- **Transfer learning** is the process of taking a pre-trained model (the weights and parameters of a network that has been trained on a large dataset by somebody else) and "fine-tuning" the model with your own dataset.
- Pre-trained model :
  - ImageNet is a dataset that contains 14 million images with over 1,000 classes.
  - ✓ The idea is that this pre-trained model will act as a feature extractor.
  - ✓ You will remove the last layer of the network and replace it with your own classifier.Pre-trained model :

❑ Rather than training the whole network through a **random initialization of weights**, we can use the weights of the pre-trained model (and freeze them) and focus on the more important layers (ones that are higher up) for training.

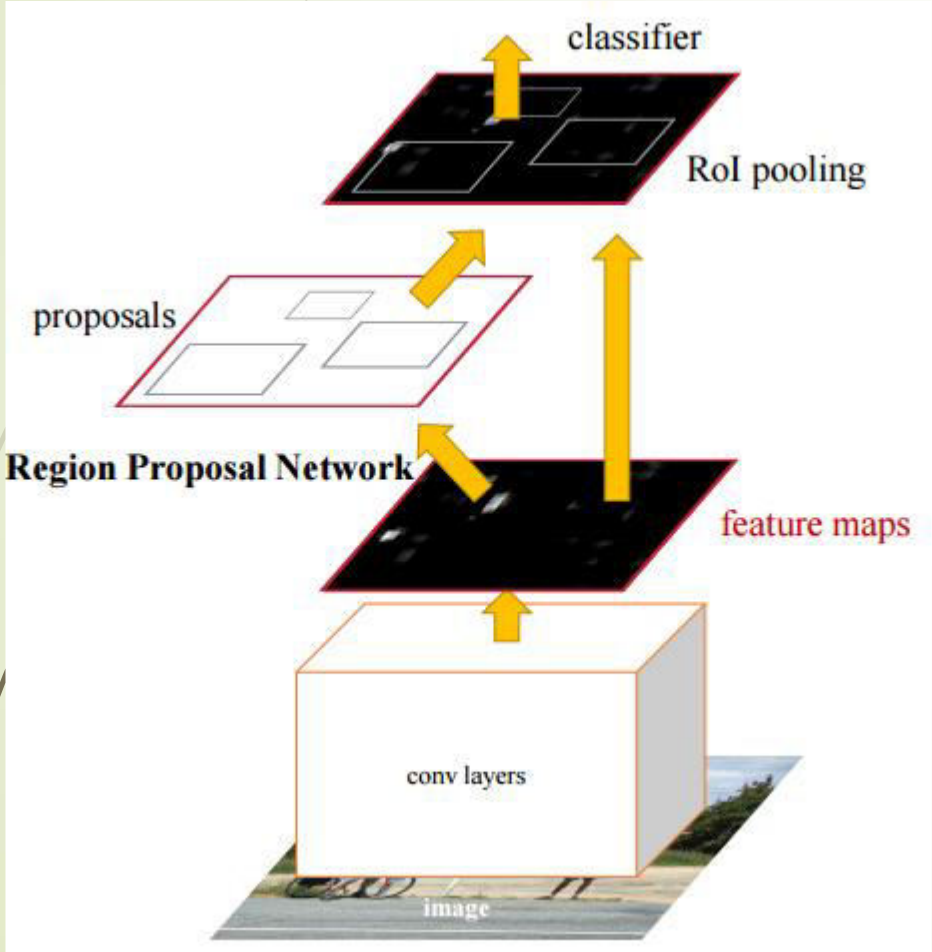
CNN Models for Object Detection

- Current state of the art Object detection models are :
  - ✓ Faster R-CNN (Microsoft)
  - ✓ YOLO (You Only Look Once) –> Real-time Object Detection.

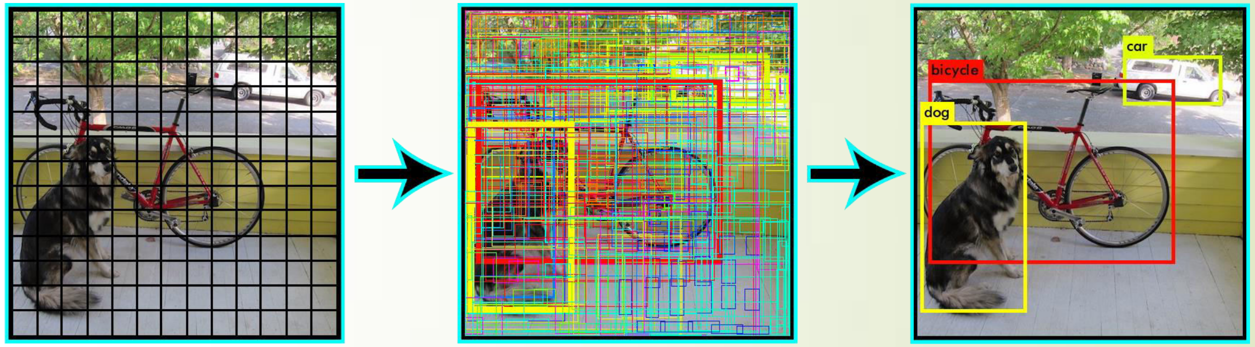
Why we chose YOLO ?

Uses a single neural network for the task of object detection.

- Whereas RCNN had different networks for each task and was slower compared to Yolo.

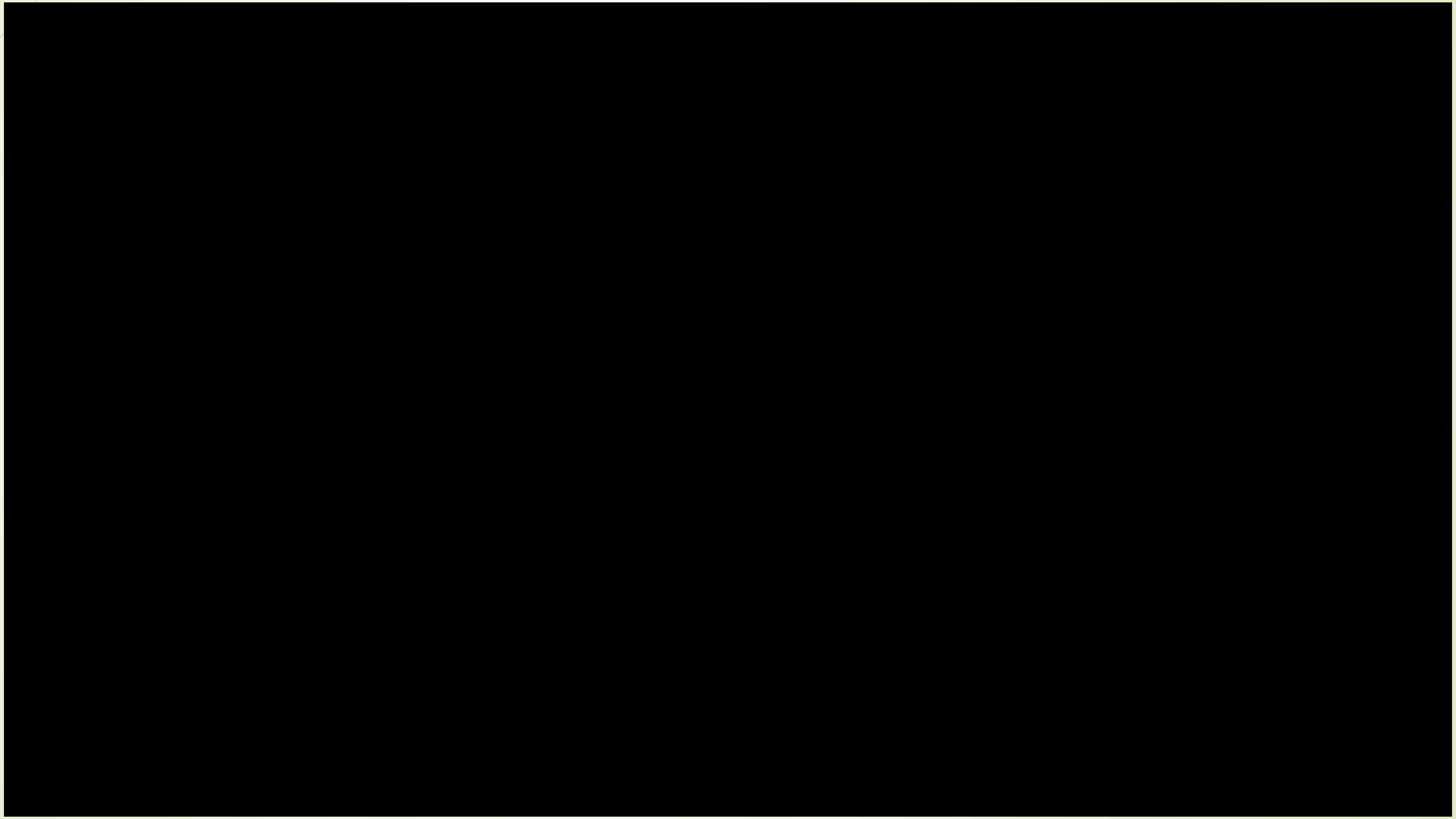


Faster RCNN



YOLO v2

Demo of Yolo V2.

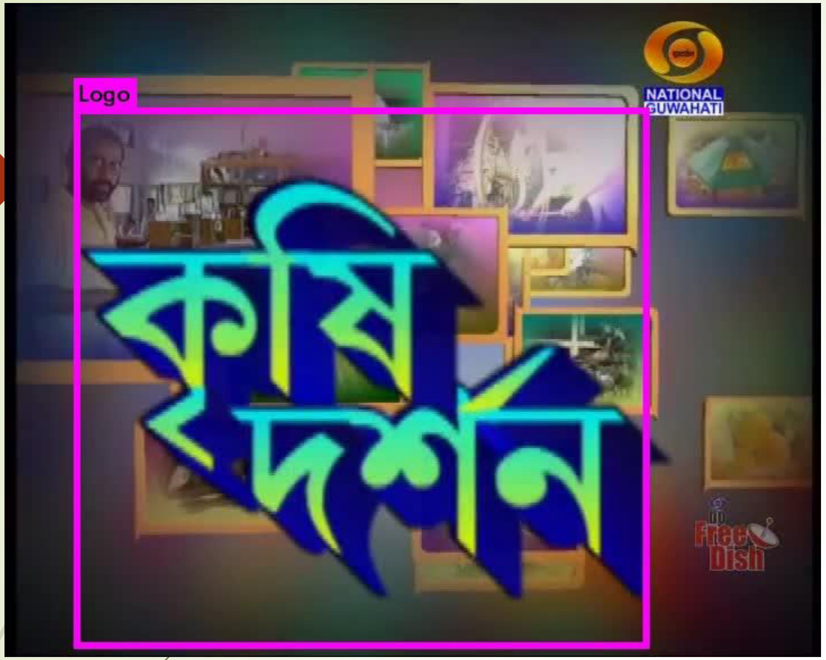


How we trained YOLO for Logo ?

- We wanted to see if this would work for Logo's by considering **Logo as the object.**
- Yolo architecture was a deep network with about 24 Layers.
- **Initial Model** : Trained only for Hindi Channel Logos. (About 600 Logos)
  - Training Data contained the Logo image with a text file containing the bounding coordinates of the logo which were manually tagged.
  - Model was trained on **Nvidia GeForce 920M** for about **10-12 hours** for 1000 iterations.

Results

- The results were amazing and Yolo detected Logos of other language channels as well.


NATIONAL
GUWAHATI
Logo
কৃষি দর্শন
Free Dish


STAR PLUS
Logo
vivo
Dil Hai Hindustani
कल रात से
शनि-रवि | 8.00pm


&tv
भाबीजी घर पर हैं !
Logo
अग्निफेरा

Detection Time
→ Accuracy vs Speed TradeOff

Yolo (More Accurate):

- On GPU : About 0.4 seconds per image.
- On CPU (or Server) : About 10 seconds per image.

We wanted to see how a smaller version of Yolo called **Tiny-Yolo** which was about **9 layers** (comparatively faster) would work.

In a similar fashion, it was trained on GPU for about 5000 iterations.

Tiny-Yolo (Slightly less accurate compared to Yolo):

- On GPU : About 0.2 seconds per image.
- On CPU (or Server) : About 5-6 seconds per image.

Why are GPUs required ?

- Deep learning involves huge amount of **matrix multiplications** and other operations which can be massively parallelized and thus sped up on GPU-s.

Parallelization on Server

- The server had 32 CPUs and we wanted to see if there would be any speed up with parallelization.
- We observed that a function called GEMM (GEneral Matrix to Matrix Multiplication) took most of computation time.
- We parallelized this function using **OpenMP.**

Detection Time on Server with Parallelization:

i) Yolo : About 2 seconds per image. (Before 10 sec/image.)
ii) Tiny-Yolo : About 1.5 seconds per image. (Before 6 sec/image)

Logo Recognition

[Index our dataset] <-> [Image Descriptor] <-> [Feature Vector]

1. SIFT descriptor.
2. CNN based feature descriptor.

Finding similar sets

Query Image
[Feature Vector]

-- *Using Distance Metric*
-- *knnMatch + Ratio Test*

Indexed Dataset :
- Feature Vector 1 [Image 1]
- Feature Vector 2 [Image 2]
- Feature Vector 3 [Image 3]

.
.
.
.
.

- Feature Vector n [nth Image]

SIFT : knnMatch + Ratio Test

```
# Match the features
   bf = cv2.BFMatcher()
   matches = bf.knnMatch(queryFeatures,data[1], k=2)
   # Apply ratio test
   good = []
   for m,n in matches:
       if m.distance < 0.75*n.distance:
           good.append([m])
```

Image with **highest number of good matches** was returned as the result.

In case of CNN, we use a distance metric and return the image with the least distance.

Model failed for some cases.

No detection

Multiple Detections

Problems :
→ Edges being cut out.
→ Only part of logo detected.


Logo
சிரிப்பு
வருது
கலைஞர்

రావే మా ఇంటికి
Powered by
XXX
Special Partner
KOLORS

New Model : Trained across all Language Channels.

- This was done for both Yolo and Tiny-Yolo models.
- Model was trained for Hindi, English, Telugu, Kannada, Marathi, Tamil and Malayalam. (Total about 600 Images)
- Same training process was repeated.

Testing

- For the purpose of testing, we carefully prepared and manually picked about 110 images.
- Picked different types of logos.
- Logos with text only, logos with text and some background, logos with people.
- Logos with graphics

Goal : To see if the model works for the majority of dataset i.e. the typical kind of logos.

Images from the Testing Dataset

Summary of Results

- Old-Yolo : 105 Detections (5 images had no detection)
- Old-Tiny-Yolo : 99 Detections (11 images had no detection)
- **Yolo - 109 Detections (1 image had no detection)**
- Tiny-Yolo : 107 Detections (3 images had no detection)


Connexion
ಕನೆಕ್ಷನ್
23:08:48
Star ಸುವರ್ಣ
Dance
Dance
GRAND FINALE

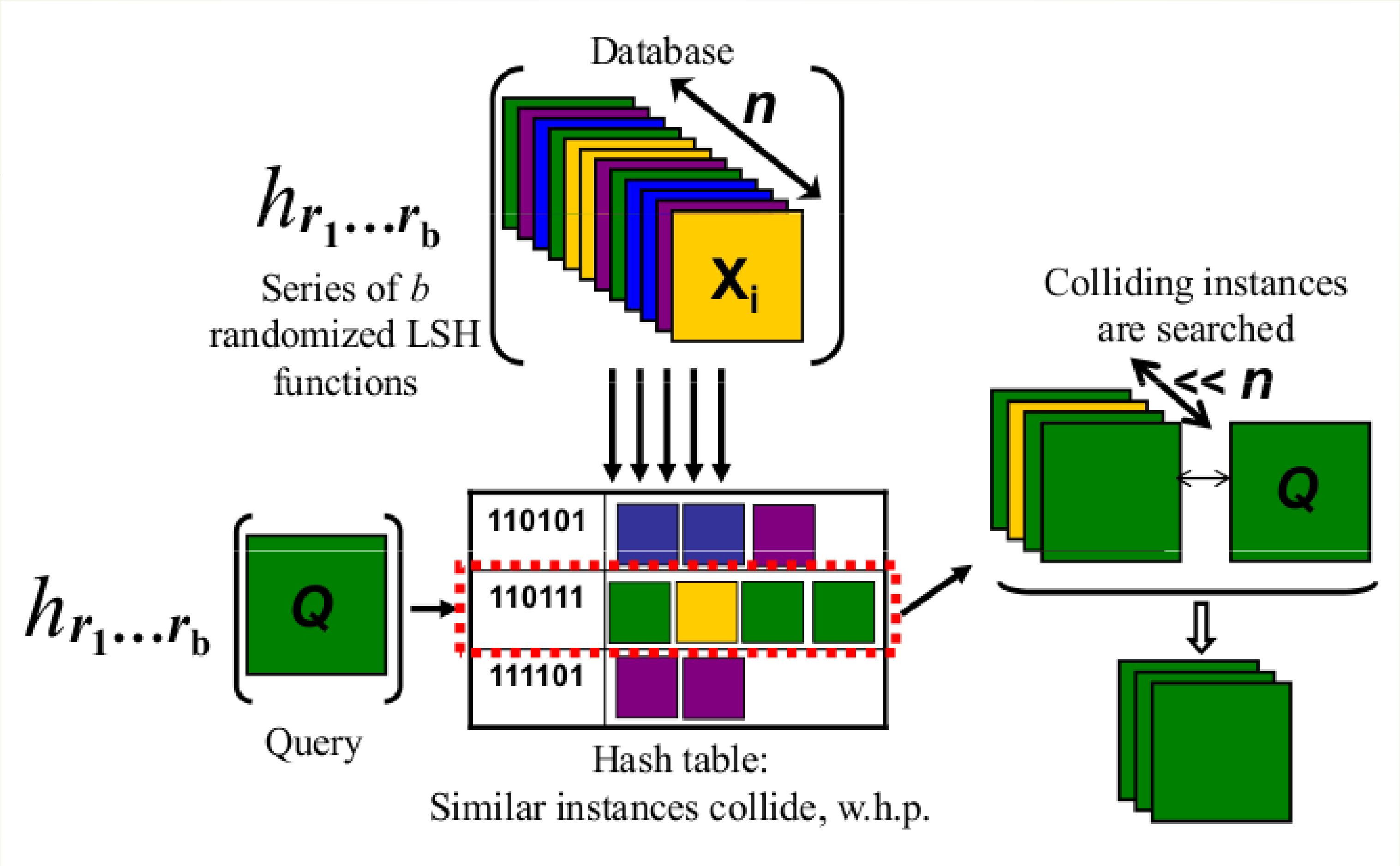

SOORYA
FESTIVAL


కామెడి
టై
GEMINI

Speeding Up The Recognition.

- **Locality Sensitive Hashing**
- Locality-sensitive hashing (LSH) reduces the dimensionality of high-dimensional data.
- LSH hashes input items so that similar items map to the same “buckets” with high probability
- LSH differs from conventional and cryptographic hash functions because it aims to maximize the probability of a “collision” for similar items.

Generate candidate pairs i.e. likely similar items and search only among them.



Generate candidate pairs i.e. likely similar items and search only among them

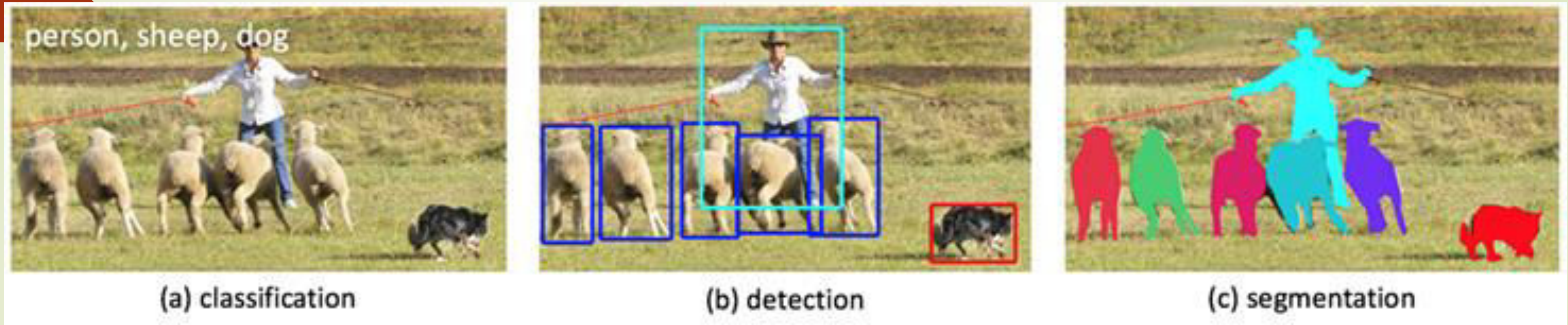
Speed Gain

- Time taken by LSH to search : less than 0.1 sec.
- CNN based brute force search : 0.4 – 0.5 seconds.

Future Work

- This can be extending for Product Detection and Recognition as well.
- For products, the search space increases so we would need to use a faster search method such as LSH.

Facebook's DeepMask+SharpMask



(a) classification (b) detection (c) segmentation

Thank You ☺