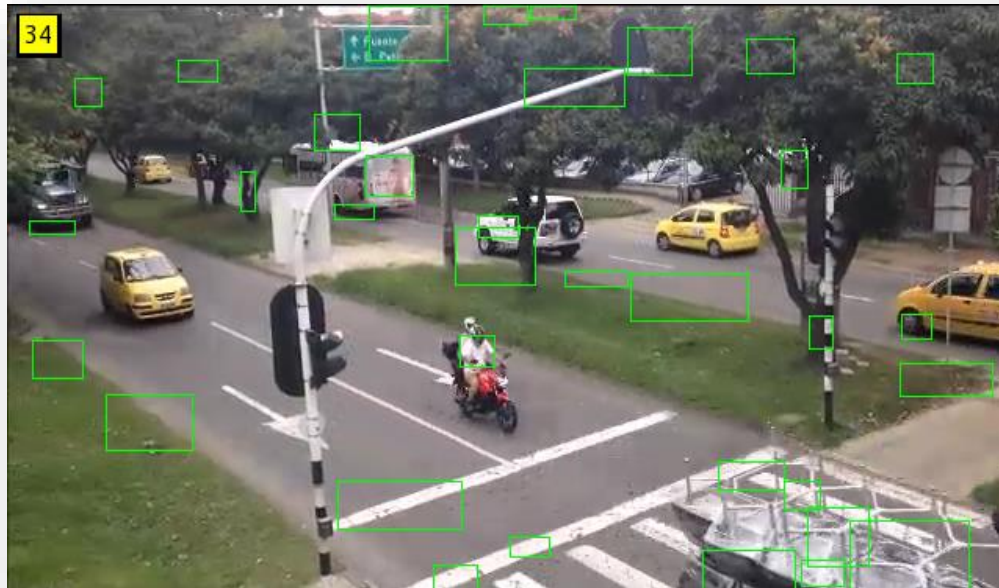


Motorcycle Classification in Urban Scenarios using Convolutional Neural Networks for Feature Extraction

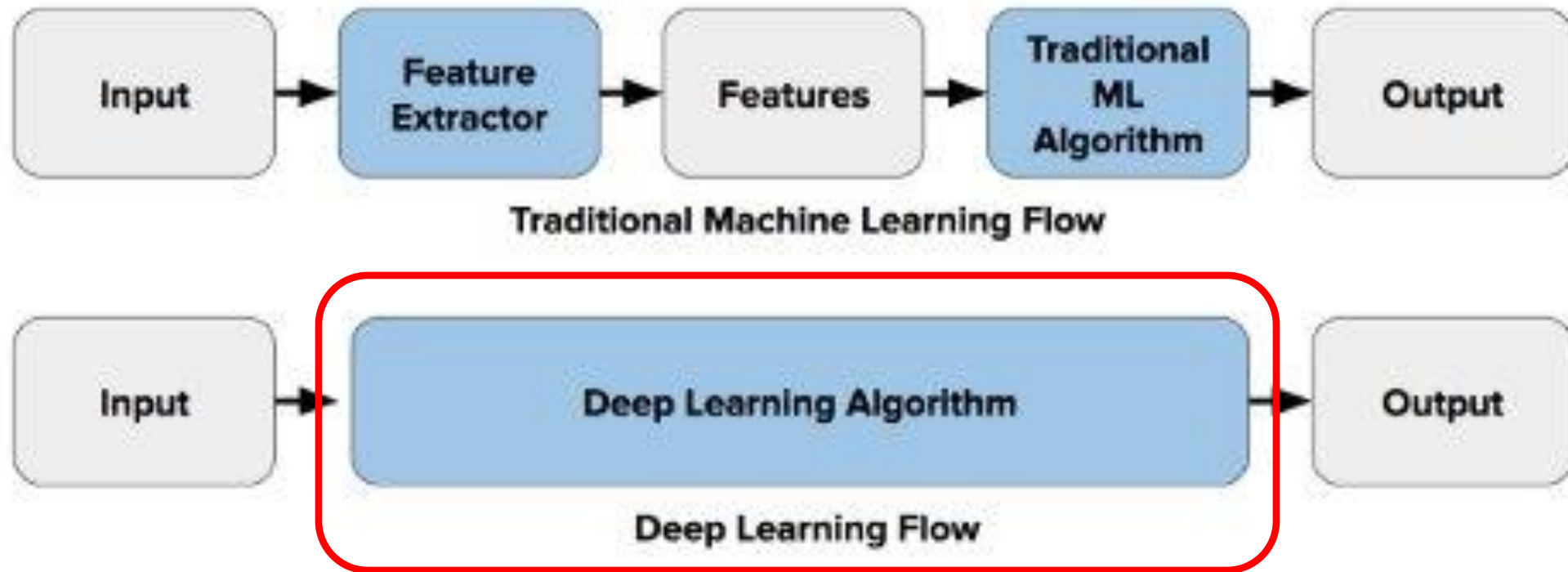


Experimentation Image



Experimentation Image

Computer Vision Workflow

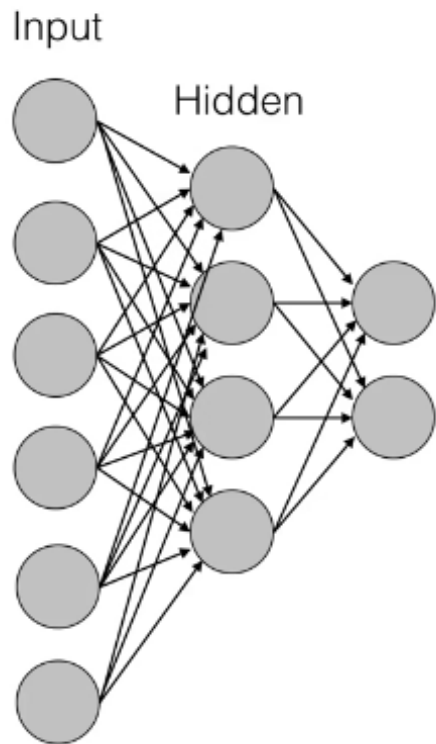


Convolutional Neural Networks

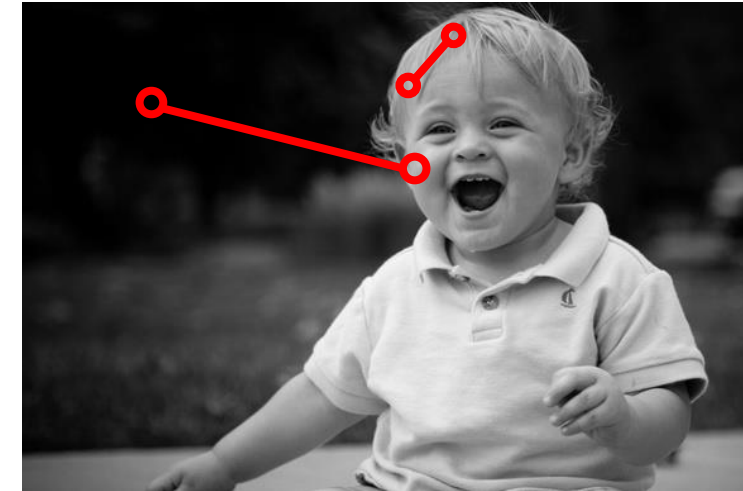
Computational Implications

MNIST dataset: 28 x 28 pixels (784 pixels)
First layer weights: ~78k parameters

Typical Image: 256 x 256 (56,000 pixels)
First layer weights: 560k parameters !



Too many parameters!!

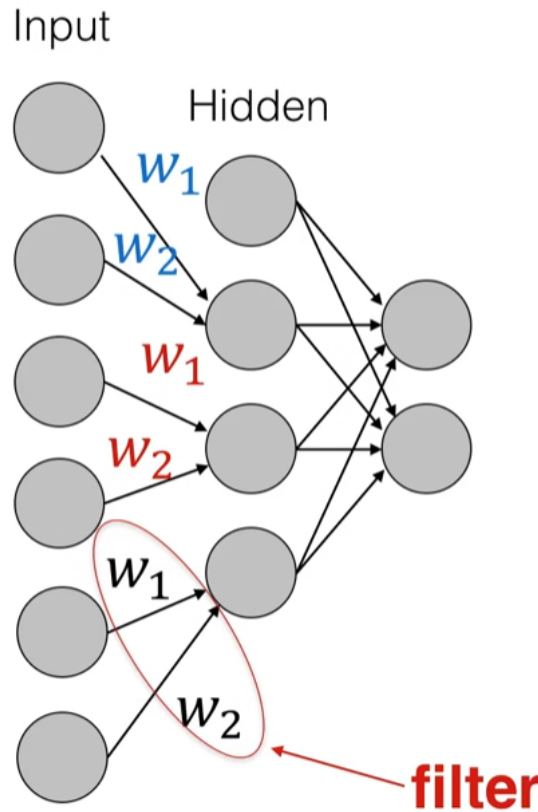


Space Matters!!

NN MLP Traditional Architecture

Convolutional Neural Networks

Why Convolution?



- Edges are filtered – Pixel with high contrast
- This operation is performed all around the image

$$y = w_1x_1 + w_2x_2$$

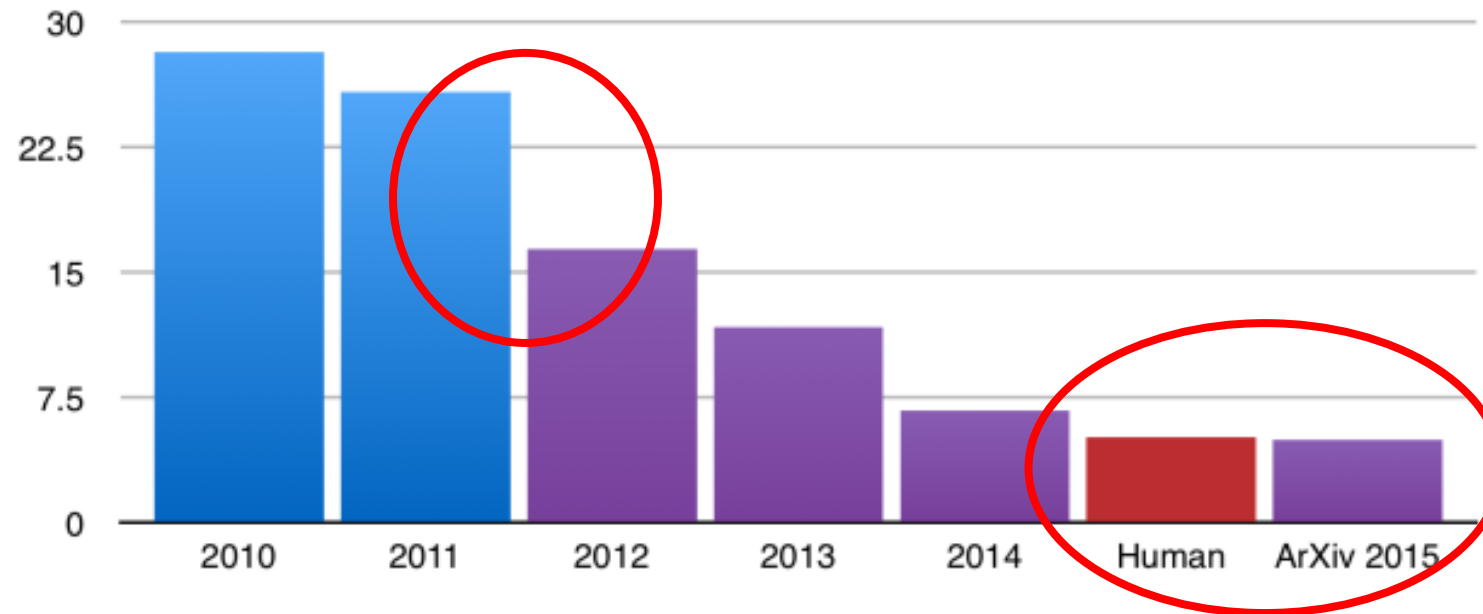
$$\text{If } (w_1, w_2) = (1, -1): y = x_1 - x_2$$

$$y \text{ maximal when } (x_1, x_2) = (1, 0)$$

Deep Learning for Vehicle Classification

IMAGENET Image Large Scale Visual Recognition Challenge

ILSVRC top-5 error on ImageNet

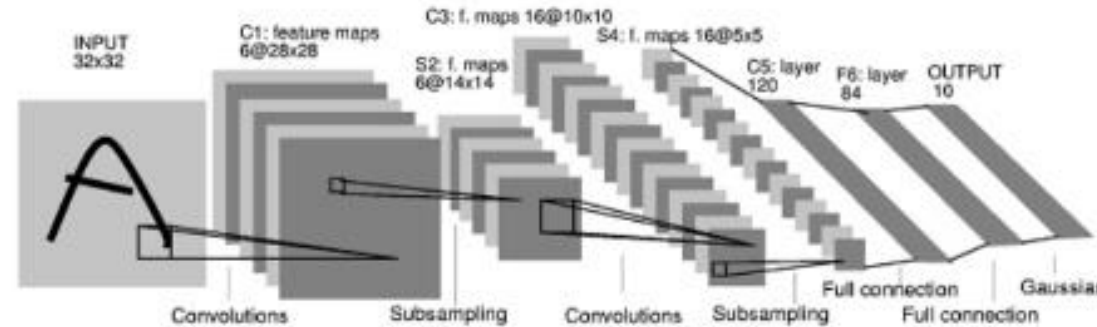


Source: <http://image-net.org/>

Deep Learning for Vehicle Classification

1998

LeCun et al.



of transistors



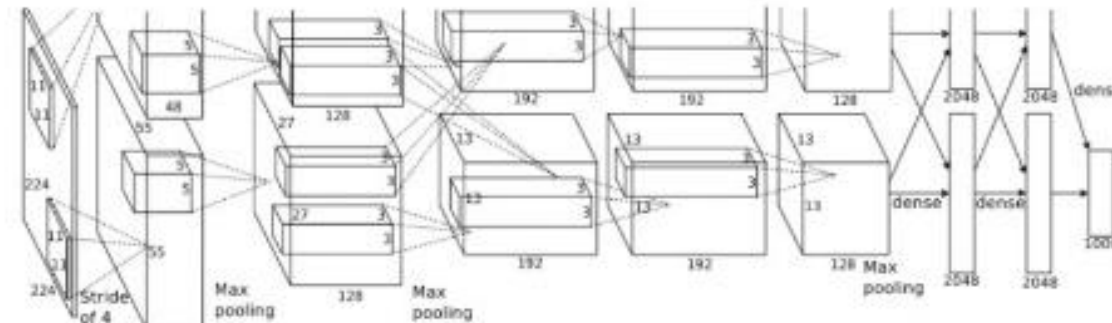
10^6

of pixels used in training

10^7 NIST

2012

Krizhevsky et al.



of transistors



10^9

GPUs



of pixels used in training

10^{14} IMAGENET

Deep Learning for Vehicle Classification

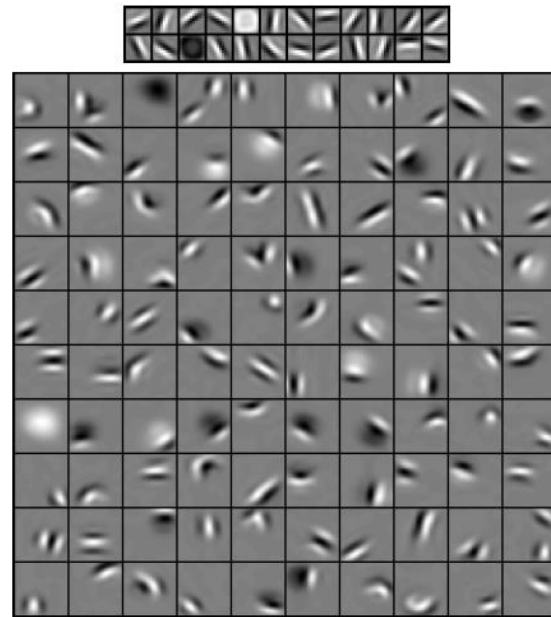
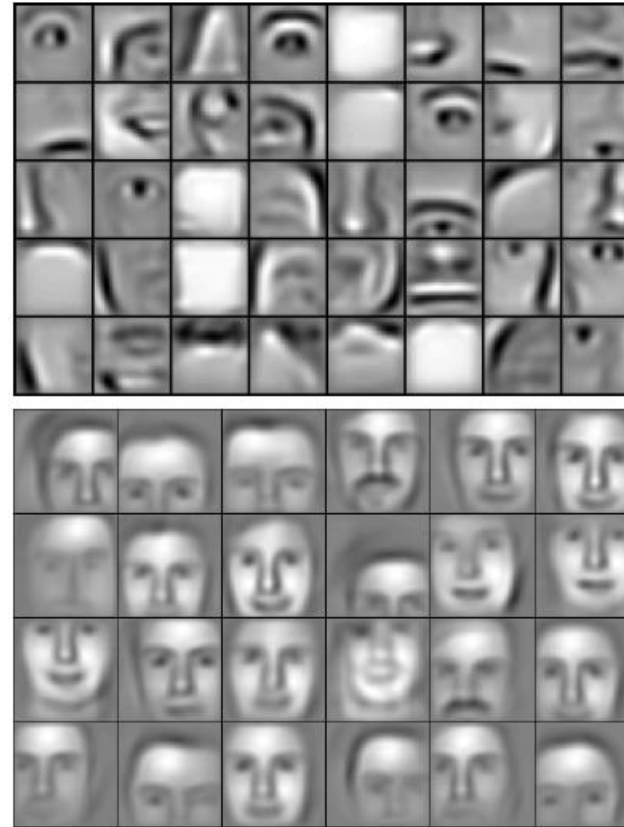
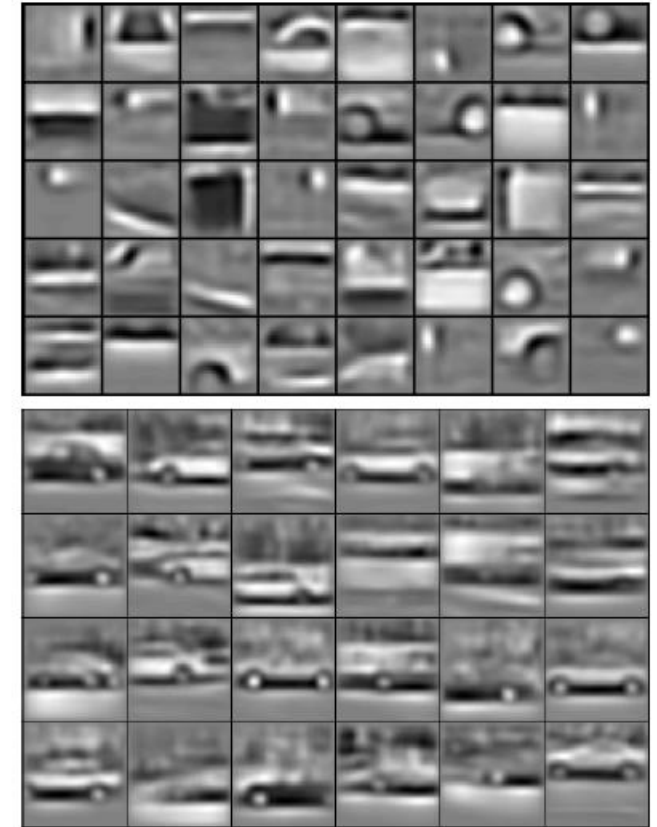


Figure 2. The first layer bases (top) and the second layer bases (bottom) learned from natural images. Each second layer basis (filter) was visualized as a weighted linear combination of the first layer bases.

faces



cars



[8] H. Lee, R. Grosse, R. Ranganath, y A. Y. Ng,

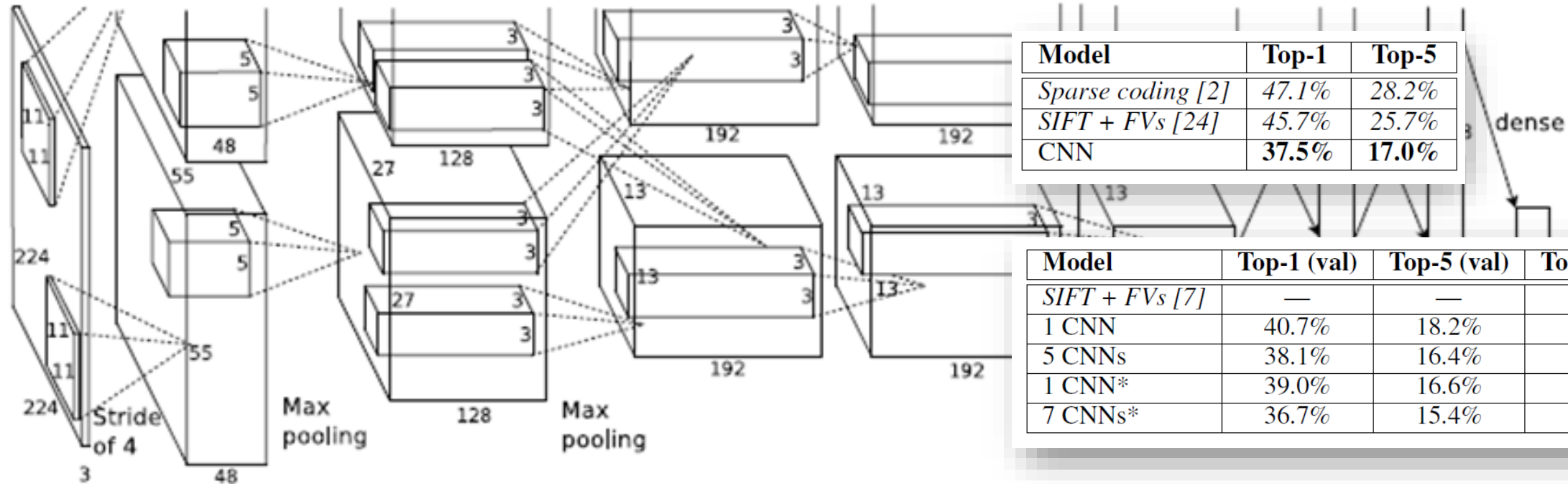
Pretraining CNN

- AlexNet (ImageNet [2]) took almost a **entire week** for training , running in two GPUs GTX 580 3GB, **ImageNet** dataset contains more than 15 millions of high res images, distributed in around 22 thousand categories and labelled and classified on 1000 categories.
- CNN can be pretrained for two purposes:
 - **Feature extraction:** Feature extraction: where a CNN is used to extract features from data (in this case images) and then use the learned features to train a different classifier, e.g., a support vector machine (SVM).
 - **Transfer learning:** Where a network already trained on a big dataset is retrained in the last few layers on a more compact data set.

This can be verified in Razavian et al. [10], where a generic descriptor is generated from a CNN and then it is used in the net OverFeat[11] to perform task of object recognition and classification.

Deep Learning for Vehicle Classification

AlexNet



Model	Top-1	Top-5
<i>Sparse coding</i> [2]	47.1%	28.2%
<i>SIFT + FVs</i> [24]	45.7%	25.7%
CNN	37.5%	17.0%

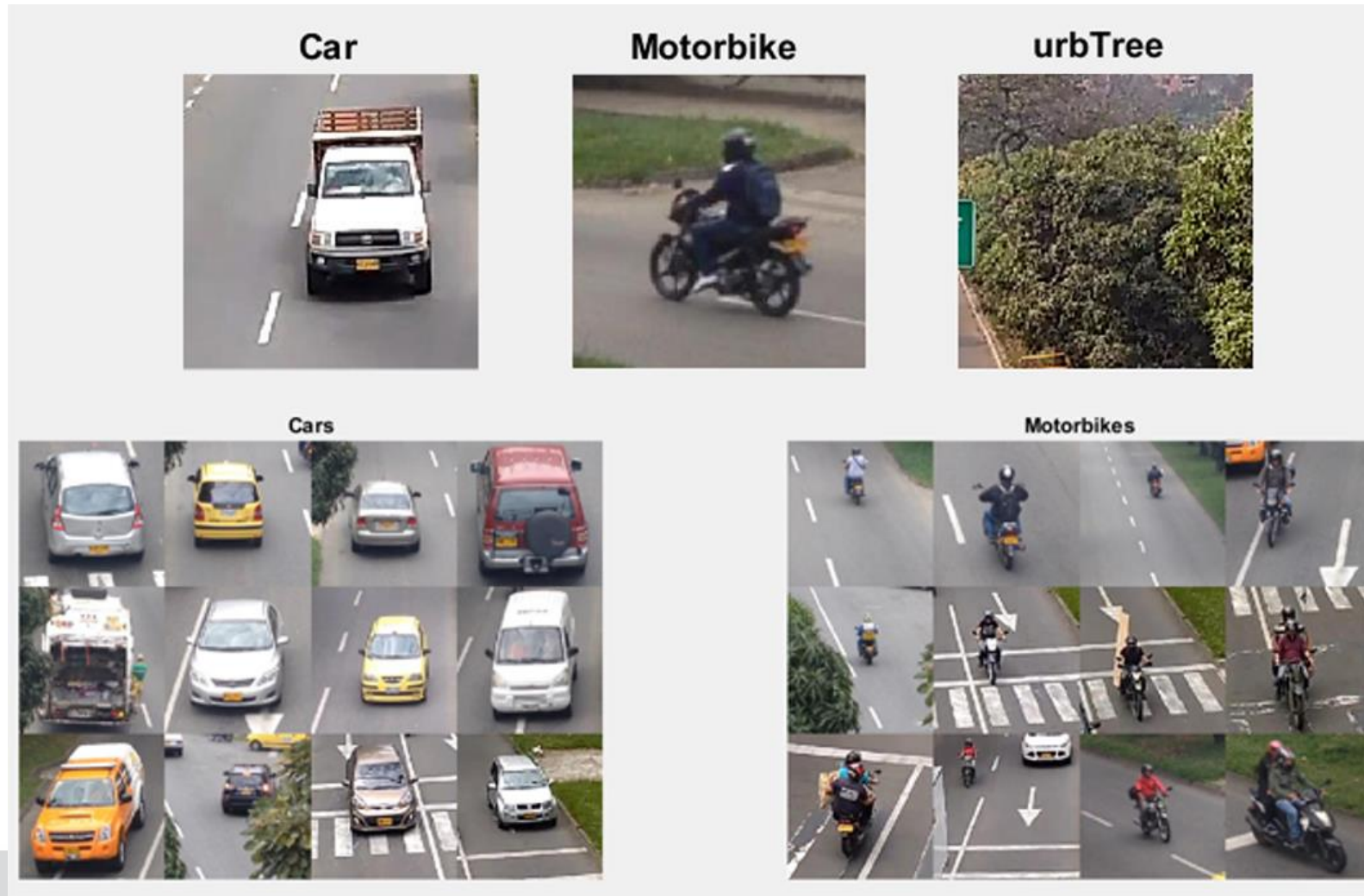
Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
<i>SIFT + FVs</i> [7]	—	—	26.2%
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	16.4%
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	15.3%

8 Layers network, first 5 of Convolution last three Fully Connected. The output of the last fully-connected layer is fed to a 1000-way softmax which produces a distribution over the 1000 class labels.

The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax

AlexNet - A. Krizhevsky, I. Sutskever, y G. E. Hinton, [2]

Deep Learning for Vehicle Classification

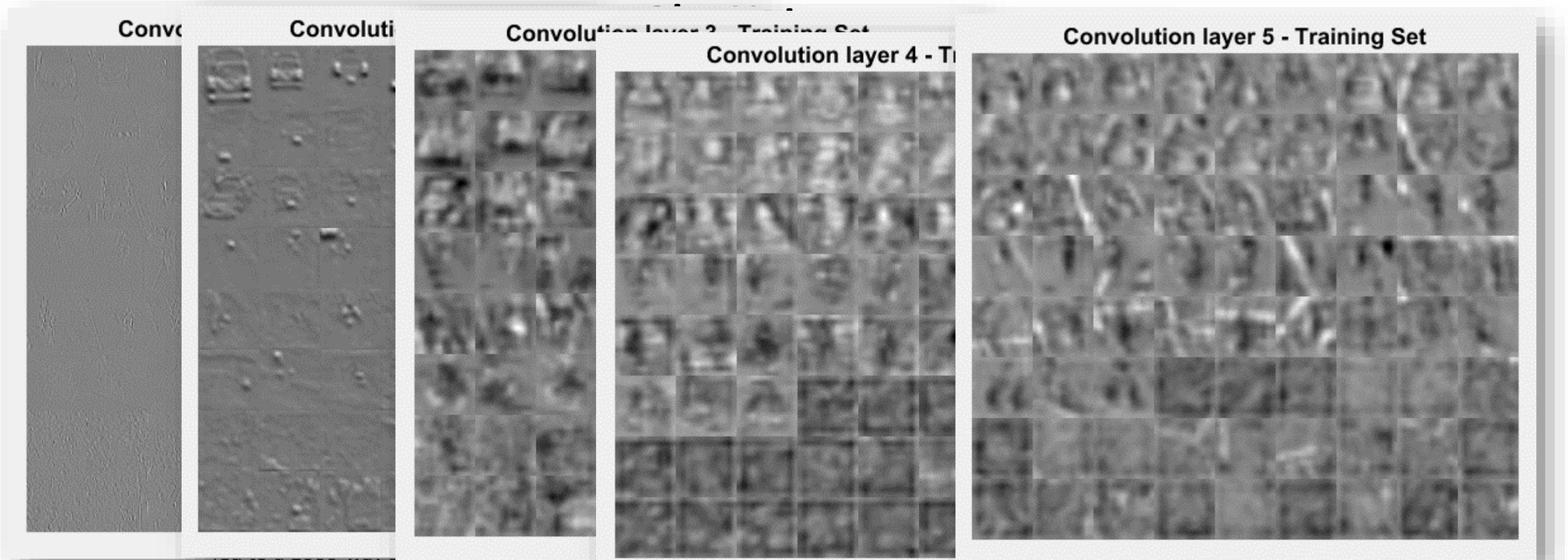


8th International Conference of Pattern Recognition Systems

The three categories for Dataset created for classification
Only 80 Examples per Category = 240 Total



Deep Learning for Vehicle Classification

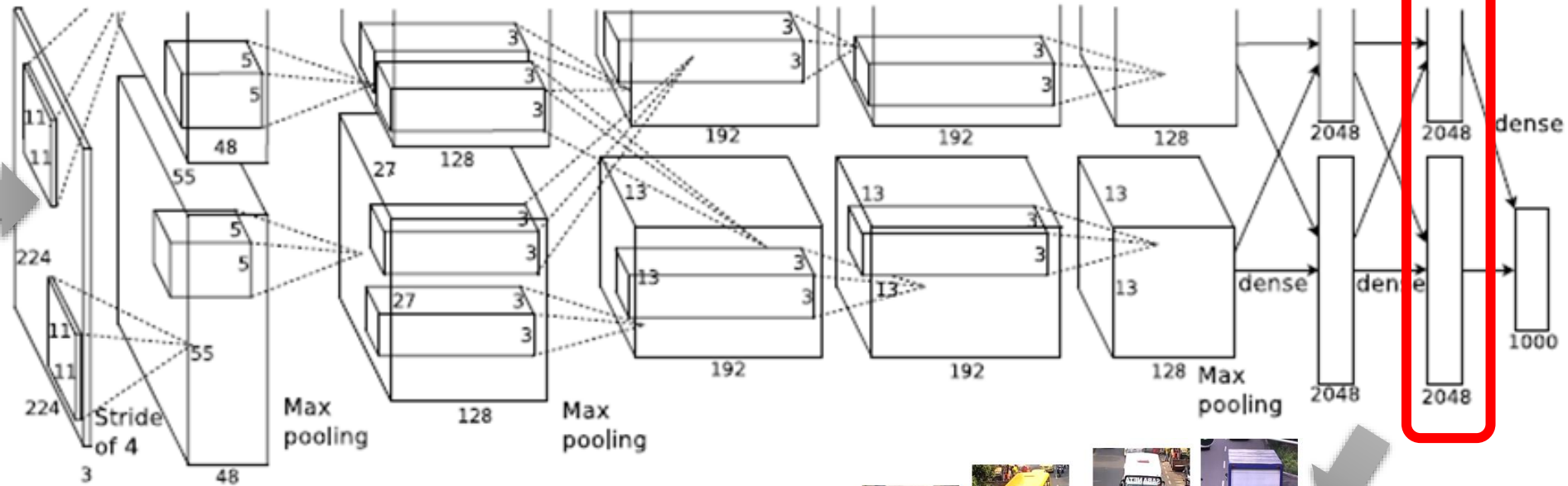


The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax

AlexNet - A. Krizhevsky, I. Sutskever, y G. E. Hinton, [2]

AlexNet for Vehicle Classification

AlexNet



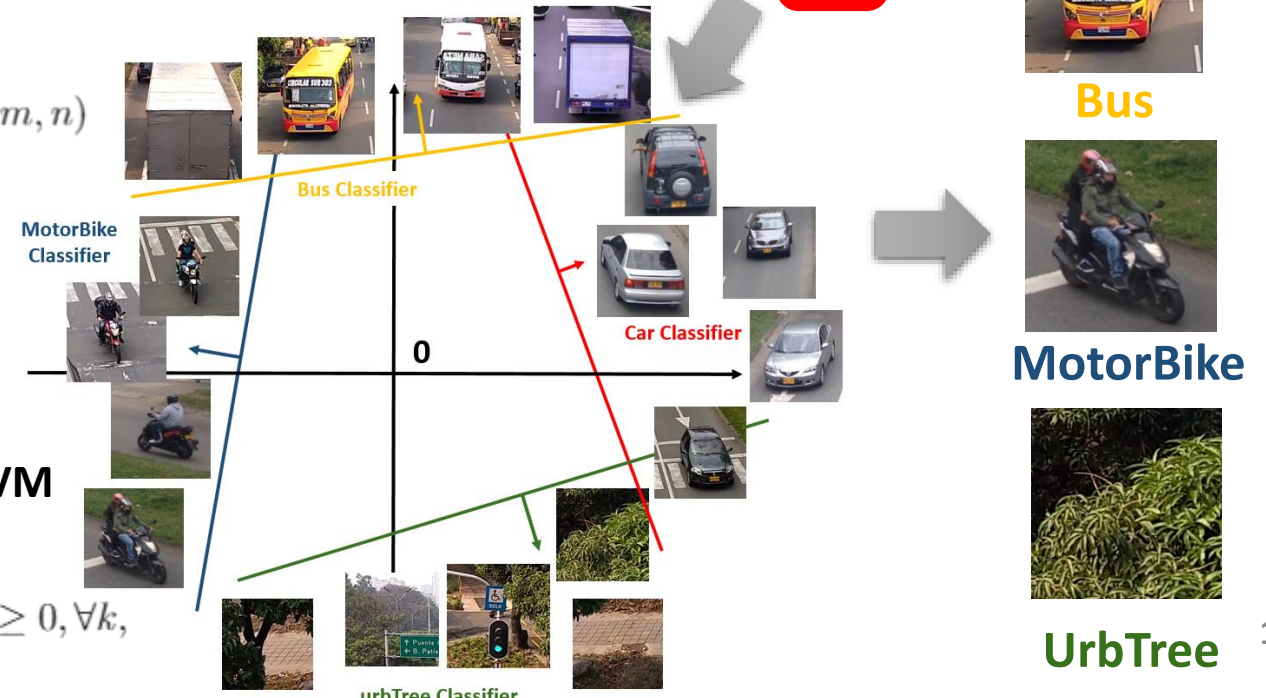
$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n)$$

$$x_j^l = f \left(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l \right)$$

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + C \sum_{k=1}^M \xi_k$$

$$s.t. \quad y_k(w^T \phi(x_i, y) + b) \geq 1 - \xi_k, \xi_k \geq 0, \forall k,$$

Linear SVM



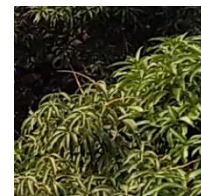
Car



Bus



MotorBike



UrbTree

Deep Learning for Vehicle Classification

Results

Confusion Matrix

Output Class \ Target Class	1	2	3	
1	55 32.7%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	56 33.3%	0 0.0%	100% 0.0%
3	1 0.6%	0 0.0%	56 33.3%	98.2% 1.8%
	98.2% 1.8%	100% 0.0%	100% 0.0%	99.4% 0.6%

1. Cars

2. Motorbikes

3. UrbTree

- Mean Accuracy: 99,40%
(Training 30% – Test 70 %)
- Cross Validated Mean Accuracy : 100%
(k=10, Training 90% – Test 10 %)
- Cross Validated Mean Accuracy : 99,31%
(k=10, Training 10% – Test 90 %)

Deep Learning for Vehicle Classification

5 Classes



Experiment Set extension. (Class 1=Cars, 2 = Motorbikes, 3 = urbTree, 4=Car side, 5=Motorbikes side (Las two from Caltech-101) [12]

- Try to evaluate if the Features obtained for Motorcycles forces the classifier to treat all motorbikes as single class, and if it also happen with cars.

AlexNet for Vehicle Classification

Classification Results

Confusion Matrix

Output Class \ Target Class	1	2	3	4	
1	53 23.7%	1 0.4%	0 0.0%	0 0.0%	98.1% 1.9%
2	3 1.3%	54 24.1%	0 0.0%	0 0.0%	94.7% 5.3%
3	0 0.0%	1 0.4%	56 25.0%	0 0.0%	98.2% 1.8%
4	0 0.0%	0 0.0%	0 0.0%	56 25.0%	100% 0.0%
	94.6% 5.4%	96.4% 3.6%	100% 0.0%	100% 0.0%	97.8% 2.2%

- Mean Accuracy: 97,80%
(Training 30% – Test 70 %)
- Cross Validated Mean Accuracy : 100%
(k=10, Training 90% – Test 10 %)
- Cross Validated Mean Accuracy : 99,31%
(k=10, Training 10% – Test 90 %)

Confusion Matrix of the experiments.

(Class 1: Buses 2: Cars 3: Motorcycles 4: urbTree)

Deep Learning for Vehicle Classification

Results



Source Internet

Cars (Top_Carro)

82 %

Motorbike (Top_Moto)

40 %

UrbTree

58 %

Deep Learning for Vehicle Classification

Results



Source Internet

Carros (Cars_top)

73, 5%

Motos (Motos_top)

62,3%

UrbTree

22 %

Deep Learning for Vehicle Classification

Results



Source Google Street View at Medellín

Cars (Top_Carro)

61 %

Motorbike (Top_Moto)

85 %

UrbTree

31 %



Deep Learning for Vehicle Classification

Results



Source Internet

Cars (Top_Carro)

34 %

Motorbike (Top_Moto)

21 %

UrbTree

32%

Conclusions

- In this research it is proposed the implementation of a motorbike classification scheme in urban scenarios using CNNs for feature extraction.
- CNNs already trained with millions of examples and able to classify 1000 categories can be used for feature extraction to train a linear SVM and then classifying for instance three different classes.
- GPUs use on CNN are critical: For instance Benchmark reports [15]: The Pascal Titan X with cuDNN is 49x to 74x faster than dual Xeon E5-2630 v3 CPUs.
- Region of Interest (ROI) can accelerate the speed of CNN analysis; this can be evaluated overall in video detection and classification analysis.

Laboratorio 6

<https://colab.research.google.com/drive/1VEvimwUdwKJnRUld4aaC5yRSrsdj2ThW?authuser=2#scrollTo=-q4saleoZbWj>



ExtractDL-Features.ipynb

Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda Se guardaron todos los cambios

+ Código + Texto

Deep Learning para Extracción de características

Este laboratorio ha sido extendido de la versión de (https://github.com/chsasank/image_features)

Busca características de imagen genéricas para

- Clasificación de imágenes
- Recuperación de imágenes
- Similitud de imagen

Con image_features, se pueden extraer características de las imágenes basadas en el aprendizaje profundo con una sola línea de código:

```
from image_features import image_features
features = image_features(['your_image_1.png', 'your_image_2.jpg'])
```

Estas características podrán ser usadas para entrenar un modelo de clasificación de scikit-learn:

```
from sklearn import linear_model
from image_features import image_features
X_train = image_features(['your_image_1.png', 'your_image_2.jpg'])
y_train = ['cat', 'dog']
clf = linear_model.LogisticRegression()
clf.fit(X_train, y_train)
```

El paquete utiliza internamente PyTorch y el modelo de aprendizaje profundo resnet50 preentrenado en imagenet. Dado que el modelo preentrenado ha visto más de un millón de imágenes durante su entrenamiento, sus características pueden generalizarse a la mayoría de las tareas de imágenes.

En primera instancia cargaremos la librería de Image Featrues (image_features)

ExtractDL-Features

RAM Disco

Editar

Mostrar todo

Gracias !!!



© Man Bouncing Question Mark Towards Doctor - Artist: [Art Glazer](#)