Machine Learning vs Deep learning on Image recognition

Ekin Yilmaz, Leonardo Ghiani, Mohamed Alie Kamara

Instructor: Guido Borghi
Email: guido.borghi@unibo.it

Keywords
Computer vision; machine learning; deep learning; feature descriptor; hog; cifar-10; Mnist; classification; scikit-image; tqdm; cv2; convolutional neural networks; SVM.

1. Introduction

Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos, and other visual inputs - and take actions or make recommendations based on that information. Computer vision enables machines to see, observe and understand, while AI enables computers to think [1].

In this paper we will do a comparative analysis of image recognition between machine learning using feature descriptions and support vector machine (svm) Classifier vs deep learning using convolutional neural networks (CNN), on CIFAR-10 dataset. The comparison will be based on the following aspects: data, accuracy, training time, hardware, features, interpretability.

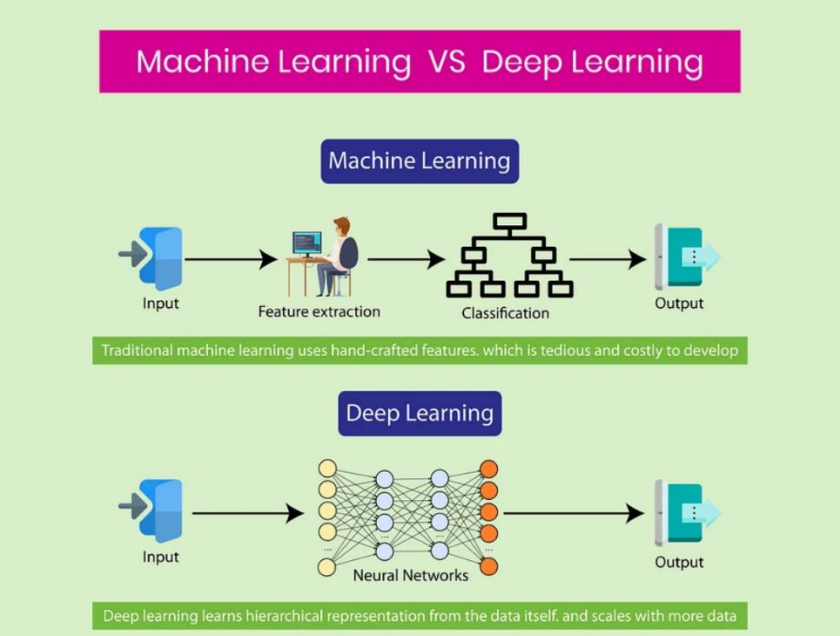


Figure 1: Machine learning vs deep learning source: projectpro

a. CIFAR-10 Description

The **CIFAR-10** dataset (Canadian Institute for Advanced Research, 10 classes) is a subset of the Tiny Images dataset and consists of 60000 32x32 color images. The images are labelled with one of 10 mutually exclusive classes: airplane, automobile (but not truck or pickup truck), bird, cat, deer, dog, frog, horse, ship, and truck (but not pickup truck). There are 6000 images per class with 5000 training and 1000 testing images per class [2].

The data provided by the university of Toronto is already pre-processed and stored in batch files

[3] however we will use raw images rather than already pre-processed in our analysis which we downloaded from Kaggle [4].

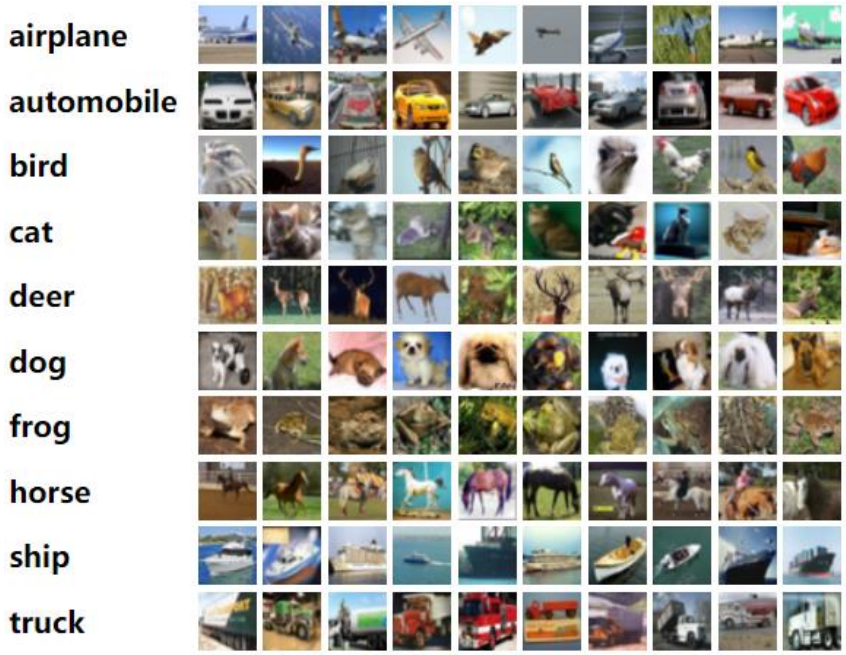


Figure 2: Cifar-10 images source: CIFAR-10

2. Related work

a. A Comparison of Traditional Machine Learning and Deep Learning in Image Recognition by Yunfei Lai. In Lai's paper he made a comparative analysis between old traditional machine learning technique using Vapnik's svm (RBF (Radial Basis Function)) 'kernel trick' on Yann LeCun's famous Mnist dataset, which is a large dataset black and white images (one channel) of handwritten digits that is commonly employed as training and testing set in the field of machine learning, and deep learning consisting of three-layer convolution neural shown in figure 3.

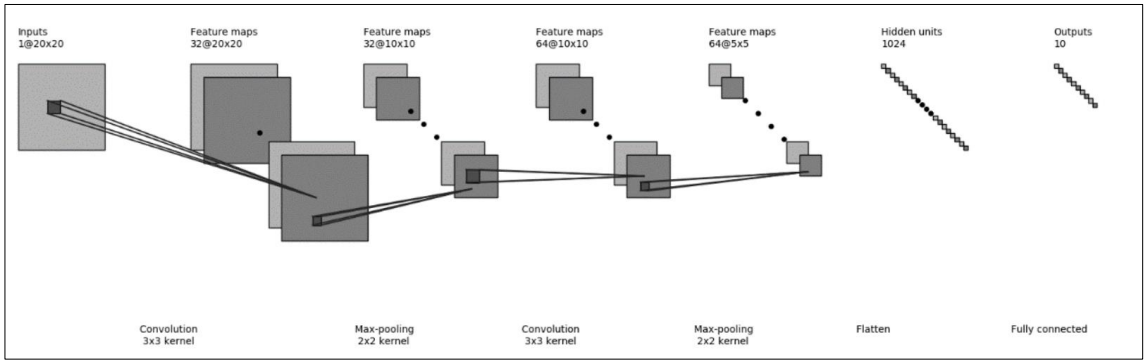


Figure 3: Yunfei Lai CNN network source: iopscience

The overall result achieved in the testing set using SVM is 93.92% total accuracy and 98.85% using CNN, clearly demonstrating that deep learning is more suitable for modelling of image data because of its ability to extract features from two-dimensional image data automatically, unlike traditional machine learning methods which need subjective feature extraction to convert binary vectors into one-dimensional vectors.

However, Yunfei also take into consideration that more training time and resources are need for deep learning, but better prediction accuracy and generalization can be achieved [5].

Figure 4 shows detailed summary of Yunfei Lai's experiment.

Methods / Comparison	SVM	Deep Learning
Operating Time	46m54s	11h50m41.2s
Accuracy in Training Set	94.09%	100%
Accuracy in Testing Set	93.92%	98.85%
Means for Extracting Features	Manually and Subjective	Automatically and Objective
Means for Processing Data	Turn Images into Vector	Directly Using Images

Figure 4: Yunfei's SVM and CNNs comparison source:iopscience

3. Proposed Method

a. SVM and feature descriptors

(i) The dataset (CIFAR-10) used in our analysis is divided into training and testing, however we further split the training set into trainset and validation set with a ratio of 70% to 30% respectively.

(ii). Feature descriptors: is a method of feature extraction for transforming raw data into numerical features that can be processed while preserving the information in the original data set. It can be accomplished manually or automatically using deep networks. In our analysis we will extract features manually using the following python libraries *cv2* and *scikit-image* for image processing, *tqdm* that allows you to output a smart progress bar that shows elapsed and estimated time remaining, Histogram of Oriented Gradients normally refers to 'hog'. (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image [6].

(iii). SVM (Support Vector Machines) algorithm: Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression, and outlier detection [7]. It was proposed by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963 but was only able to deal with linear classification problems, because of this it was later improved to a method called 'kernel trick', which was able to solve non – linear classification problems.

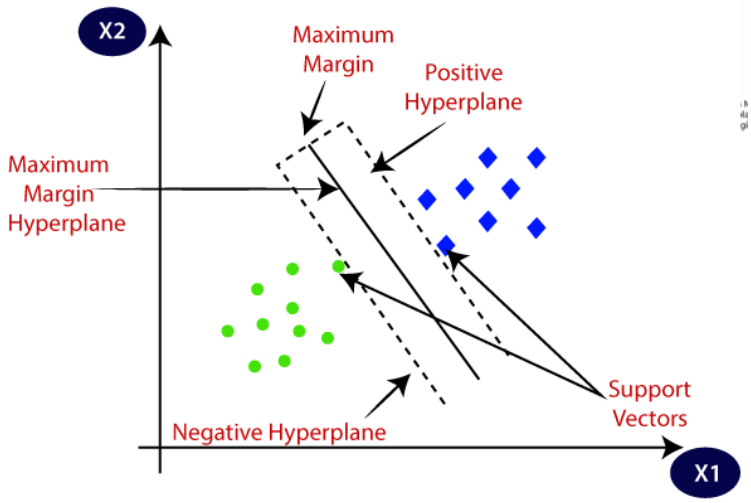


Figure 5: The intuition of SVM Algorithm

There are so many different kernel tricks and for our analysis we will use RBF (Radial Basis Function) because of its similarity to k-nearest neighbor algorithm.

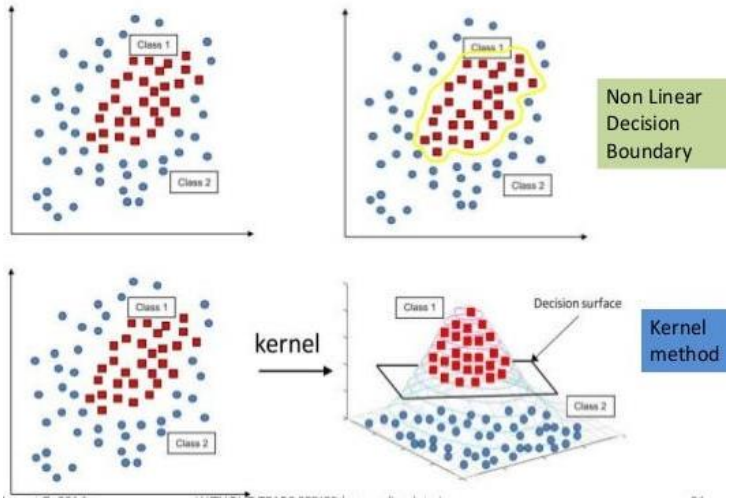


Figure 6: The svm classifier

The RBF equation:

$$K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$$

Model Architecture: SVM (RBF) + HOG + 128
Where feature type: HOG, image size: 128, model: SVM

```
# suppor vector machine (kernel svm)
clf = svm.SVC(gamma=0.001, C=100, kernel='rbf', verbose=False, random_state=0)
```

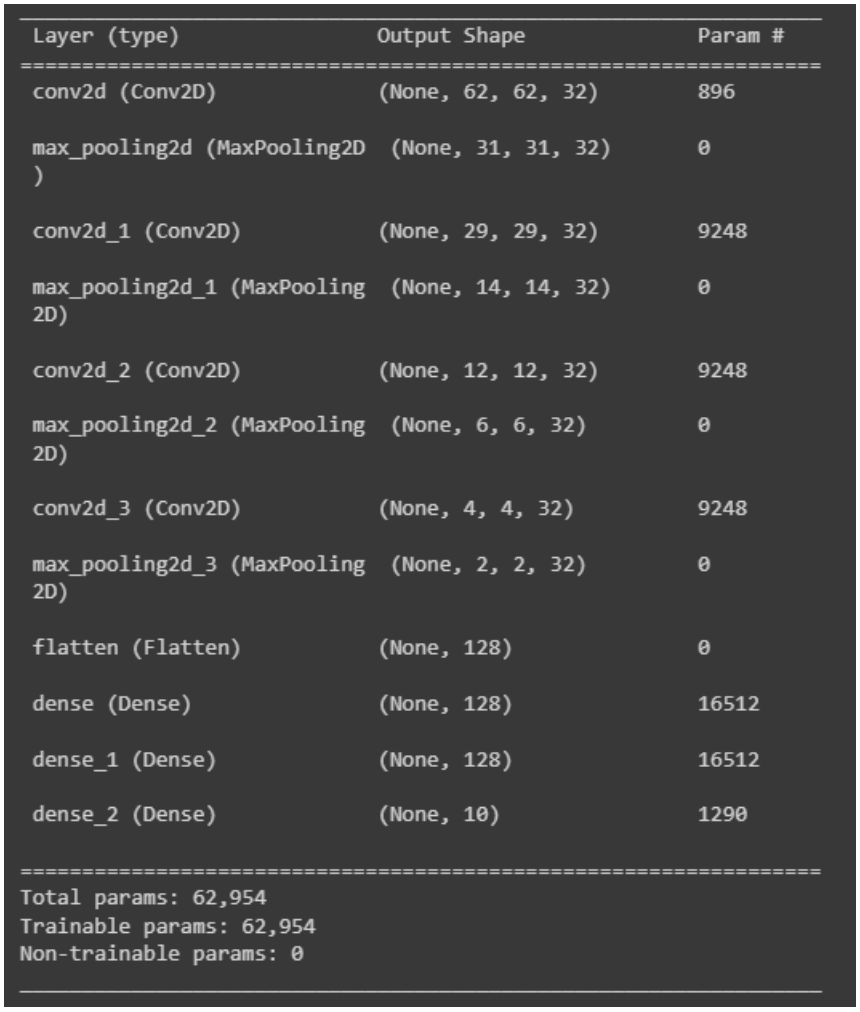
Figure 7: The svm classifier

b. Convolutional Neural Networks

First introduced by Kunihiko FuKushima in the 1979 called “Neocognitron”, a basic image recognition neural network [8]. Kunihiko’s Neocognitron laid the foundation of research around convolutional neural networks by Yann LeCun in 1980 [9]. Since then, CNNs (Convolutional Neural Networks) have been widely adopted for image recognition and object detection. Some known CNN architectures are the classic LeNet-5, AlexNet, VGGNet, GoogleNet, ResNet, ZFNet [10]. In this paper we used a simple deep network which is explained and demonstrated below.

Model Architecture:

The proposed solution has four convolutional and pooling layers, one flattened, two full connected layers and an output layer. The first convolution layer has a filter of 32, kernel size of 3, three channel image size of 64x64, and a relu activation function. The second layer is Max pooling with pool size and strides set to 2. One flattened layer, two fully connected layers with relu activation function, a final output layer with 10 neurons for each one of the 10 classes and a SoftMax activation function for multiclass, which is suitable and better than sigmoid.



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 12, 12, 32)        9248

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 32)         0
 2D)

 conv2d_3 (Conv2D)           (None, 4, 4, 32)          9248

 max_pooling2d_3 (MaxPooling  (None, 2, 2, 32)         0
 2D)

 flatten (Flatten)           (None, 128)               0

 dense (Dense)               (None, 128)               16512

 dense_1 (Dense)             (None, 128)               16512

 dense_2 (Dense)             (None, 10)                1290

=================================================================
Total params: 62,954
Trainable params: 62,954
Non-trainable params: 0
_________________________________________________________________
```

Figure 8: Model architecture source: project notebook

4. Results

The SVM model had a validation accuracy of 57.01% on the training set and 57.2% on the test set, unlike CNN which had 78.12% on the training set and 75.48% on the test set, clearly showing why svm are not the best for 2-dimensional data. The CNN performance was incredibly good with image recognition. Computational time for machine learning (SVM) was 4.7hrs unlike deep learning (CNN) which was 1.26hrs given the same hardware resources (Google Colab TPU with high RAM of 35.2GB), breaking the myth that it takes more time to train and develop CNN models. A summary of our analysis is shown below in Figure 9.

Element	Machine Learning (SVM)	Deep Learning (CNNs)
Data	CIFAR-10	CIFAR-10
Accuracy	57.2%	75.48%
Training time	282.04 mins (4.7hrs)	75.38 mins (1.26hrs)
Required hardware	Python 3 Google Compute Engine backend (TPU)	Python 3 Google Compute Engine backend (TPU)
Features	Manually (HOG)	Automatically (learned)
Interpretability	Good	Low

Figure 9: SVM and CNN comparison

5. Conclusions

It is true that computer vision is in all aspects of our daily lives from the smart phones we use to our fun social media sites, health, military, law enforcement authorities and business enterprises. This field is making a huge improvement. In our paper we have proven that CNNs are better than SVM in image classification, recognition, and object detection tasks.

Also, another groundbreaking discovery is that with more computational resources like GPUs (Graphics Processing Unit) and TPUs convolutional neural networks perform better and take less time to train and deploy than SVM.

However, we should mention that SVM is still widely used in machine learning and one of the most powerful and most effective algorithms in high dimensional spaces [11].

6. Acknowledgement Thanks to Professor Guido Borghi for his guidance and provision of initial code templates on which this paper was built on.

Group organization

GitHub repository: L. Ghiani , E. Yilmaz, M. A. Kamara
Markdown & Web template: L. Ghiani , E. Yilmaz
Jupyter Notebook : Mohamed Alie Kamara

References

[1] IBM, “What is computer vision”.

[2] Paperswithcode, “Cifar-10”, https://paperswithcode.com/dataset/cifar-10

[3] A. Krizhevsky https://www.ibm.com/topics/convolutional-neural-networks sky and V. Nair and G. Hinton, “The CIFAR-10 dataset”, s, Department of Computer Science, University of Toronto, https://www.cs.toronto.edu/~kriz/cifar.html

[4] Kaggle, “Cifar-10 raw image data”, https://www.kaggle.com/datasets/oxcdcd/cifar10

[5] Yunfei Lai, “A comparison of Traditional Machine Learning and Deep learning in Image Recognition”, https://iopscience.iop.org/article/10.1088/1742-6596/1314/1/012148

[6] Wikipedia, “Histogram of Oriented Gradients”

[7] Scikit-learn, “Support vector machines”, https://scikit-learn.org/stable/modules/svm.html

[8] Wikipedia, “Neocognitron”, https://en.wikipedia.org/wiki/Neocognitron

[9] IBM, “What are convolutional neural networks?”, https://www.ibm.com/topics/convolutional-neural-networks

[10] IBM, “Types of convolutional neural networks”, https://www.ibm.com/topics/convolutional-neural-networks

[11] Scikit-learn, “Support Vector Machine”