DEEP LEARNING BASED CAR MODEL RECOGNITION

Praphul Singh[1], Harshit Saini[2], Sunkari Preetham Paul[3], Aadrish Sharma[4], Vikas Singh[5], and Nishchal Verma[6]

IIT Kanpur, India
spraphul@iitk.ac.in, harshits@iitk.ac.in, preeth@iitk.ac.in,
aadrish@iitk.ac.in, vikkyk@iitk.ac.in, nishchal@iitk.ac.in

Abstract. Deep Learning is a bleeding edge field, which took Artificial Intelligence to the next level. With deep networks expanding their utility for different tasks, deep networks have reserved their place in almost every field. With growing need for technology that aids the human-kind, deep Learning has proved to be loyal in promoting AI in every field. This paper presents a CNN based model retrained on Inception-v3 with HOG, sliding window, Softmax and SVM for the detection of car models.

Keywords: Inception-v3, SVM, HOG, CNN, Sliding Windows, Softmax

1 Introduction

The term deep learning, emerged in the late 2000s, following the influential 2006 paper by Hinton, Osindero, and Teh [1][2]. Multi-label image classification is however a more general and practical problem, since the majority of real-world images are with more than one objects of different categories. Many methods[3][4][5] have been proposed to address this more challenging problem.

Our code demonstrates how deep networks can be used to contribute to one of the most anticipated tasks that every India teenager wants to do - Know the details of a stunning car when it appears in front of him, on road. Our projects goal is to enable him to simply take a picture of it in his smartphone, and get an handful of information, about the car and its model. Well, this may appear to be quite challenging for an ordinary smartphone, but with about 200 lines of code, trained upon over 800 lines of code running in the app's server on the internet, this is not challenging anymore. Deep Learning has made this possible. Our code is written in tensorflow and python.

2 Data Collection

Data collection is one of the challenging things that any machine learning engineer faces. For reaching goals that no one has ever reached (that is, no one has ever collected data for different models of cars), we have spent days collecting data from the internet. As data from internet is uncoordinated, we had to spend to spend lot of time removing irrelevant images from the data. This made the data, in simple terms, quite pure.

We have collected 1,50,000+ (the number of images before the data turned pure is more than 2,00,000) images of about 140 models, belonging to 14 brands of cars which can be found on Indian roads. With all the data collected, organised and sorted, we have set to train the data with Googles Inception-v3 model.

3 Inception-v3

Proposed in 2014 by Christian Szegedy and other Google researchers and used in the GoogLeNet architecture that won both the ILSVRC 2014 classification and detection challenges.[6]

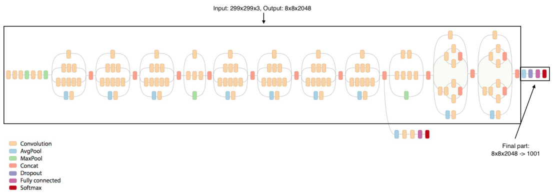


Fig. 1: Inception-v3

4 Transfer Learning

We are using the Inception-v3 model to train each brand of car for its different models which gives a good accuracy for small number of classes approximately less than 30. Inception-v3 model was trained on imagenet data in the 2012 Imagenet challenge, upon 6 GPUs for about 8 weeks. As we lack such huge amount of computation power, we resorted to something better instead of trained the whole model from the scratch - Transfer Learning.

[7]Transfer learning is a technique that shortcuts a lot of this work by taking a fully-trained model for a set of categories like Inception-v3, and retraining from the existing weights pre-trained over a larger data, for new classes. So, we took the saved graph from the imagenet 2012 challenge, and used it for feature extraction. In this project we'll be retraining the final layer from scratch, while leaving all the others untouched.

The problem we faced was that we were having about 14 brands and there were about 10 models for each class. So, finally the number of classes became 140 training which, getting a good accuracy was not possible by implementing just Transfer Learning.

We needed something special. So we decided to train for the brands separately.

5 Retraining Inception-v3

We have added an extra layer at the end of the inception-v3 layer as mentioned before. Weve trained the last layer first for all the models at once. But the models accuracy dropped below 55was high. So, weve thought of classifying images based on brand first, and then trained them for the models in the respective brands.

Here, we are providing the accuracies for different brands, trained only for models within. We have trained for models individually in each brand : Fiat, Ford, Honda, Hyundai, Mahindra, Maruti Suzuki, Volkswagen, Land Rover, Chevrolet, Tata, Datsun etc. There has been problem of over-fitting in the beginning, which has been resolved using a dropout layer .

Fig. 2: Retraining Inception-v3

6 Histogram Oriented Gradients

[**?**]This is a feature descriptor discussed 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. This method is similar to that of edge orientation histograms, scale-invariant transform descriptors, and shape contexts, but differs in that it is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy. We have seen the convolutional layers in which filters are used to map different parts of images to feature vectors. We needed a feature which could be taken as the unique thing to distinguish the brands. HOG can be taken similar to one of those filters which we have used to map logos of the cars to extract features. We trained the logo data using HOG filters in a standard model named SVM which gave us an accuracy of 88% .

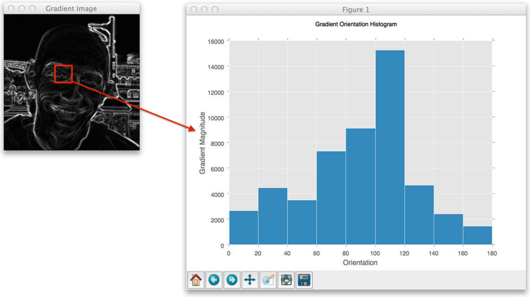


Fig. 3

7 SVM

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new examples.

For a linearly separable set of 2D-points which belong to one of two classes, find a separating straight line. A line is bad if it passes too close to the points because it will be noise sensitive and it will not generalize correctly. Therefore, our goal should be to find the line passing as far as possible from all points. Then, the operation of the SVM algorithm is based on finding the hyperplane that gives the largest minimum distance to the training examples. Twice, this distance receives the important name of margin within SVMs theory. Therefore, the optimal separating hyperplane maximizes the margin of the training data.

How does SVM work?

For a dataset consisting of features set and labels set, an SVM classifier builds a model to predict classes for new examples. It assigns new example/data points to one of the classes. If there are only 2 classes then it can be called as a Binary

SVM Classifier.

There are 2 kinds of SVM classifiers:
1.Linear SVM Classifier
2.Non-Linear SVM Classifier

7.1 Svm Linear Classifier:

In the linear classifier model, we assumed that training examples plotted in space. These data points are expected to be separated by an apparent gap. It predicts a straight hyperplane dividing 2 classes. The primary focus while drawing the hyperplane is on maximizing the distance from hyperplane to the nearest data point of either class. The drawn hyperplane called as a maximum-margin hyperplane.

7.2 SVM Non-Linear Classifier:

In the real world, our dataset is generally dispersed up to some extent. To solve this problem separation of data into different classes on the basis of a straight linear hyperplane cant be considered a good choice. For this Vapnik suggested creating Non-Linear Classifiers by applying the kernel trick to maximum-margin hyperplanes. In Non-Linear SVM Classification, data points plotted in a higher dimensional space.

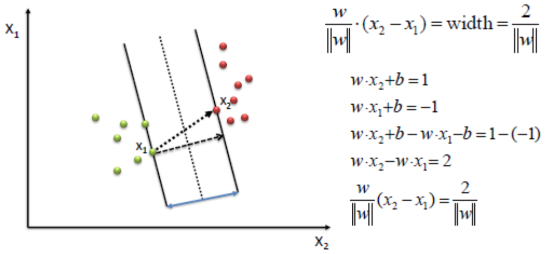

x_1
x_2
x_1
x_2
$\frac{w}{\|w\|} \cdot (x_2 - x_1) = \text{width} = \frac{2}{\|w\|}$
$w \cdot x_2 + b = 1$
$w \cdot x_1 + b = -1$
$w \cdot x_2 + b - w \cdot x_1 - b = 1 - (-1)$
$w \cdot x_2 - w \cdot x_1 = 2$
$\frac{w}{\|w\|}(x_2 - x_1) = \frac{2}{\|w\|}$

Fig. 4: Hyperplane Calculation

8 CNN

A Convolutional Neural Network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units.

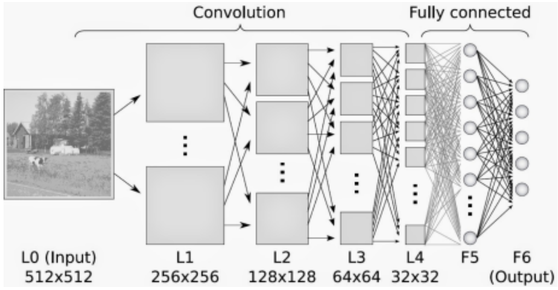


Fig. 5: CNN

9 Softmax Classifier

Softmax regression is a generalized form of logistic regression which can be used in multi-class classification problems where the classes are mutually exclusive. The hand-written digit dataset used in this tutorial is a perfect example. A softmax regression classifier trained on the hand written digits will output a separate probability for each of the ten digits, and the probabilities will all add up to 1.
Softmax regression consists of ten linear classifiers of the form:

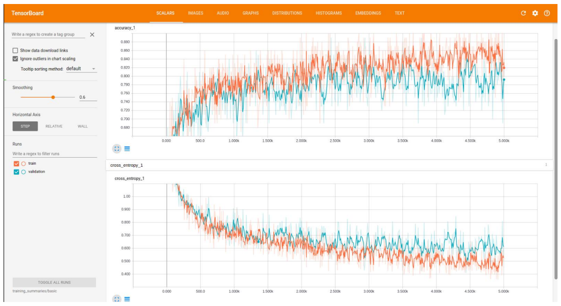
$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)};\theta) \\ p(y^{(i)} = 2|x^{(i)};\theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)};\theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

Fig. 6

The output of this equation is a vector, with one value for each hand-written digit. The first component is the probability that an input image of a digit, x, is the number 1 (belongs to class y = 1).

10 Test Results

10.1 Fiat: Accuracy- 75%



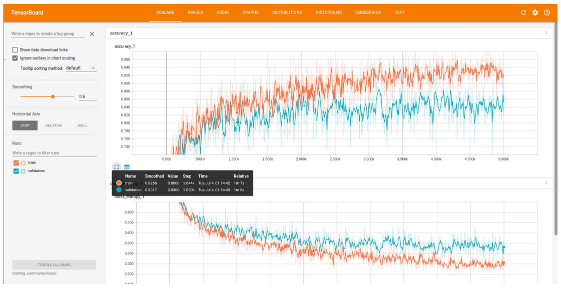
☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default

Smoothing 0.6

☑ train
☑ validation

Fig. 7

10.2 Ford: Accuracy- 76%



☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default

Smoothing 0.6

☑ train
☑ validation

Name	Smoothed	Value	Step	Time	Relative
train	0.8258	0.8000	1.044k	Tue Jul 4, 01:14:42	1m 7s
validation	0.8017	0.8300	1.050k	Tue Jul 4, 01:14:43	1m 8s

Fig. 8

Similarly we got the following accuracies for the other brands:
Honda : 78 %
Hyundai : 73%
Tata : 80%
Mahindra : 80%
Chevrolet : 77%
Maruti Suzuki : 75%
Volkswagen : 78%
Land Rover : 82%

Datsun : 90%
Renault : 92%

11 Conclusion and Future Work

Finally, as weve planned, first a sliding window uses hog descriptor and searches for logo and classifies the image based on brands. Then, the relevant node from the brands hidden layer will be activated at the end of the inception model, and further classification is done based on models, in the last layer, which has been trained individually for models in each brand. To make the project more precise and descriptive we are trying to implement the entire model by an Android app. We will give a description of the image using image captioning which covers the topic LSTM and link the sentence to the google search API to recommend cars of similar features like colour, brand and models.

References

1. Hinton, G. E.; Osindero, S.; Teh, Y. W. (2006). "A Fast Learning Algorithm for Deep Belief Nets" . Neural Computation.
2. Bengio, Yoshua (2012). "Practical recommendations for gradient-based training of deep architectures".
3. F. Perronnin, J. Sanchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In European Conference on Computer Vision, pages 143156, 2010.
4. Q. Chen, Z. Song, Y. Hua, Z. Huang, and S. Yan. Hierarchical matching with side information for image classification. In Computer Vision and Pattern Recognition, pages 34263433, 2012
5. J. Dong, W. Xia, Q. Chen, J. Feng, Z. Huang, and S. Yan. Subcategory-aware object classification. In Computer Vision and Pattern Recognition, pages 827834, 2013.
6. Rethinking the Inception Architecture for Computer Vision Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna
7. Wikipedia Transfer Learning
8. Histograms of Oriented Gradients for Human Detection Navneet Dalal and Bill Triggs