The Simpsons' Character Recognition

Rishav Mittal
Department of Computer Science
International Institute of Information Technology.
Bhubaneswar, India
B116044@iiit-bh.ac.in

Bipul Suman
Department of computer Science
International Institute of Information Technology.
Bhubaneswar, India
B116015@iiit-bh.ac.in

Tanmaya Sahu
Department of Computer Science
International Institute of Information Technology.
Bhubaneswar, India
B416059@iiit-bh.ac.in

.

1.0. Introduction

Python is an Interpreted language which in lay man's terms means that it does not need to be compiled into machine language instruction before execution and can be used by the developer directly to run the program. This makes it comprehensive enough for the language to be interpreted by an emulator or a virtual machine on top of the native machine language which is what the hardware understands.

It is a High-Level Programming language and can be used for complicated scenarios. High-level languages deal with variables, arrays, objects, complex arithmetic or Boolean expressions, and other abstract computer science concepts to make it more comprehensive thereby exponentially increasing its usability.

Python is also a General-purpose programming language which means it can be used across domains and technologies. Python also features dynamic type system and automatic memory management supporting a wide variety of programming paradigms including object-oriented, imperative, functional and procedural to name a few.

Artificial intelligence is a conglomeration of concepts and technologies that mean different things to different people – self-driving cars, robots that impersonate humans, machine learning, and more – and its applications are everywhere you look.

An AI is a computer system that is able to perform tasks that ordinarily require human intelligence. These artificial intelligence systems are powered by machine learning. Many of them are powered by machine learning, some of them are powered by specifically deep learning, some of them are powered by very boring things like just rules.

1.1. Objectives of Research

Nowadays internet is filled with anabundance of images and videos, which is encouraging thedevelopment of search applications and algorithms that can examine the semantic analysis of image and videos for presenting the user with better search content and their summarization. There have been major breakthroughs in image labeling, object detection, scene classification areas reported by different researchers across the world. This leads to making it possible to formulate approaches concerning object detection and scene classification problems. Since artificial neural networks have shown a performance breakthrough in the area of object detection and scene classification, specially convolutional neural networks (CNN), this work focuses on identifying the best network for this purpose. Feature extraction is a key step of such algorithms. Feature extraction from images involves extracting a minimal set of features containing ahigh amount of object or scene information from low-level image pixel values, therefore, capturing the difference among the object categories involved.

CNN has been presenting anoperative class of models for better understanding of contents present in an image, therefore resulting in better image recognition, segmentation, detection,and retrieval. CNN's are efficiently and effectively used in many pattern and image recognition applications, for example, gesture

recognition face recognition object classification and generating scene descriptions.CNN has well known trained networks that uses these datasets available in open source networks and increases its efficacy of classification after getting trained over millions of images contained in the datasets .The datasets used are composed of millions of tiny images. Therefore, they can simplify well and accurate and hence successfully categorize the classes' out-of-sample examples. It is important to note that neural network classification and prediction accuracy and error rates are all most comparable to that of humans when such comparisons are made on a large data set

.

1.3 Problem Statement

As a big Simpsons fan, I have watched a lot (and still watching) of The Simpson episodes -multiple times each- over the years. I wanted to build a neural network which can recognize characters. My approach to solve this problem will be based on convolutional neural networks (CNNs) : multi-layered feed-forward neural networking able to learn many features.

3. Data Collection

The selected dataset here is the Simpsons dataset, downloaded from Kaggle. Next, the following is data distribution.

Character Name	Training Image Number	Testing Image Number
abraham_grampa_smpson	913	48
apu_nahasapeemapetilon	623	50
bart_simpson	1342	50
charles_montgomery_burns	1194	48
chief_wiggum	986	50
comic_book_guy	469	49
edna_krabappel	459	50
homer_simpson	2246	50
kent_brockman	498	50
krusty_the_clown	1210	50
lisa_simpson	1354	50
marge_simpson	1291	50
milhouse_van_houten	1080	49
moe_szyslak	1452	50
ned_flanders	1454	49
nelson_muntz	358	50
principal_skinner	1194	50
sideshow_bob	878	47

The number of each character's image in training set is different, but that in testing set is equal.
row_num = [48.0, 50.0, 50.0, 48.0, 50.0, 49.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 50.0, 49.0, 50.0, 49.0, 50.0, 50.0, 47.0]

4. Methodology

The first step for the following is to perform transfer learning on the networks with image datasets. This is followed by checking the prediction rate. The different accuracy rates are observed. Third important criteria for evaluating the performance were to check whether prediction accuracy varies across all CNNs chosen for the study. Hence we are looking for best image classifier where the object is the main attribute for classification of scene category.

We set the image size as 64 * 64 and select 19548 images from training dataset and 990 images from testing dataset.

4.1 CNN

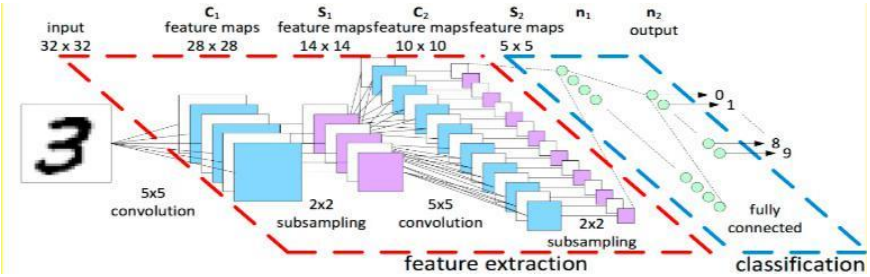
Doing image classification with 3-Layers CNN

Convolutional Neural Networks are very similar to ordinary Neural Networks: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply.

ConvNet architectures make the explicit assumption that the inputs are images, which allows us to encode certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the amounts of parameters in the network.

The CNN is a sequence of layers. Those layers are included Convolutional Layer, Pooling Layer, ReLU Layer, Normalization Layer, Full Connected Layer, and Loss Layer. What's more, Convolutional Layer, Pooling Layer, and Full Connected Layer are three main layers. Now let's stack them up one by one.
Different layers of the convolutional neural network used are:

- **Input Layer:**
  The first layer of each CNN used is 'input layer' which takes images, resize them for passing onto further layers for feature extraction.
- **Convolution Layer:**
  The next few layers are 'Convolution layers' which act as filters for images, hence finding out features from images and also used for calculating the match feature points during testing.
- **Pooling Layer:**
  The extracted feature sets are then passed to 'pooling layer'. This layer takes large images and shrink them down while preserving the most important information in them. It keeps the maximum value from each window, it preserves the best fits of each feature within the window.
- **Rectified Linear Unit Layer:**
  The next 'Rectified Linear Unit' or ReLU layer swaps every negative number of the pooling layer with 0. This helps the CNN stay mathematically stable by keeping learned values from getting stuck near 0 or blowing up toward infinity.
- **Fully Connected Layer:**
  The final layer is the fully connected layers which takes the high-level filtered images and translate them into categories with labels.



The steps of proposed method are as follows:

1. **Creating training and testing dataset:**
   The super classes images used for training is resized [64,64] pixels and the dataset is divided into two categories i.e. training and validation data sets.
2. **Modifying CNNs network:**
   Replace the last three layers of the network with fully connected layer, a softmax layer, and a classification output layer. Set the final fully connected layer to have the same size as the number of classes in the training data set. Increase the learning rate factors of the fully connected layer to train network faster.
3. **Train the network**:
   Set the training options, including learning rate, mini-batch size, and validation data according to GPU specification of the system. Train the network using the training data.
4. **Test the accuracy of the network:**
   Classify the validation images using the fine-tuned network, and calculate the classification accuracy.
   Similarly testing the fine tune network on real time video feeds for accurate results.

4.2 Data Modelling

We create the neural network, first we create the module, so if we make *model = Sequential()* we have declaring the constructor of NN.

After it we will adding the first entry layer, the *Conv2D* mean the convolutional 2D layer, well for them we have *32 neurons*, followed by the *tensor*, after the *input shape* is *64* pixels with RGB system color *(3)* and to the finalize we have the *activation function' ReLu*.
So with this we will adding the *MaxPooling2D* which as the name already says it's the max pooling of CNN with size 2x2. We repeat the steps 3 times with 64 and 128 neurons in sequentials layers, and finalize with *Flatten()* to "flatten" the layer and make the "layer vector". With this we will having created the input layers.

For the output layers we will adding a *Dense* layer with 64 neurons and other with 30 neurons (number of characters). The first will still have the ReLu activation function and the second have had the softmax activation function.
The *compile* configure the models for training, we optimizer function will be adam, the loss function, which is represent of value that will be minimized by the model will then be the sum of all individual losses, so with this we use *categorical_crossentropy* for multiple outputs and to finalize we use the *metrics* accuracy to multiple outputs.
So we will creating the generate batches of tensor image data with real-time data augmentation, this is what's be doing in *ImageDataGenerator()*. The arguments of this processing will not be describe in here, for more informations access the Keras documentations.
Continuing we have select we databases and describe where the images be. The *target_size* is where we "view" the image and the *batch_size* is the batches of select imagens, consluing with the class_model which is categorical.
Realize that we found 20,582 images in training base and 3,667 in testing base, divided in 30 classes (number of characters)

5. Result

The performance analysis of CNN's is done by testing each of the networks on Simpsons datasets.

```
Epoch 35/40
90/90 [==============================] - 28s 313ms/step - loss: 0.7122 - acc: 0.8107 - val_loss: 1.0914 - val_ac
c: 0.7167
Epoch 36/40
90/90 [==============================] - 28s 315ms/step - loss: 0.7010 - acc: 0.7963 - val_loss: 0.9287 - val_ac
c: 0.7500
Epoch 37/40
90/90 [==============================] - 29s 318ms/step - loss: 0.6524 - acc: 0.8215 - val_loss: 1.1002 - val_ac
c: 0.6833
Epoch 38/40
90/90 [==============================] - 28s 314ms/step - loss: 0.6692 - acc: 0.8070 - val_loss: 0.8996 - val_ac
c: 0.7667
Epoch 39/40
90/90 [==============================] - 28s 314ms/step - loss: 0.6199 - acc: 0.8244 - val_loss: 0.9548 - val_ac
c: 0.7333
Epoch 40/40
90/90 [==============================] - 28s 312ms/step - loss: 0.6026 - acc: 0.8230 - val_loss: 0.7476 - val_ac
c: 0.8250

Out[48]: <keras.callbacks.History at 0x7fe5faff6898>
```

For the end we FINALLY are training the NN.
It's important see that the accuracy it's 82% for this case it's good!

6.Conclusion

The work analyzed the prediction accuracy of convolutional neural networks (CNN) on training and test datasets We focused our study on 20 classes of each dataset only. Our main purpose was to find out the accuracy and evaluating the consistency of prediction by each of these CNN. We have presented a thorough prediction analysis for comparing the networks' performance for different classes of objects. It is important to note that complex frames often create confusion for the network to detect and recognize the scene. The results suggested that trained networks with transfer learning performed better than existing ones and showed higher rates of accuracy. From our experiments, more the number of layers, more will be the training and therefore, higher the rate of accuracy in prediction will be achieved. It can further be summed up that neural networks are new and best

emerging techniques for making a machine intelligent for solving many real-life object categorization problems. Many types of research and works are being done on it. It has wide applications and it is easy and flexible to integrate into various platforms. The hardware requirements may not allow the network to be trained on normal desktop work but just with nominal requirements one can train the network and generate the desired model.

References

[1] Kou, F., Du, J., He, Y., & Ye, L. (2016) “Social Network Search Based on Semantic Analysis and Learning.” *CAAI Transactions on Intelligence Technology*.

[2] Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V.,& Garcia-Rodriguez, J. (2017) “A Review on Deep Learning Techniques Applied to Semantic Segmentation.”

[3] Li, L. J., Su, H., Lim, Y.,& Li, F. F. (2010, September) “Objects as Attributes for Scene Classification.”*ECCV Workshops***(57-69).**

[4] Srinivas, S., Sarvadevabhatla, R. K., Mopuri, K. R., Prabhu, N., Kruthiventi, S. S., &Babu, R. V. (2016) “A taxonomy of deep convolutional neural nets for computer vision.”

[5] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., &Torralba, A. (2014) “Object detectors emerge in deep scene cnns.”

[6] Wang, Y., & Wu, Y. “Scene Classification with Deep Convolutional Neural Networks.”

[7] Lowe, D. G. (2004) “Distinctive image features from scale-invariant keypoints.” *International journal of computer vision***60(2).**

[8] Dalal, N., &Triggs, B. (2005, June) ”Histograms of oriented gradients for human detection.”*In Computer Vision and Pattern Recognition, 2005. CVPR 2005.*

[9] Yang, J., Jiang, Y. G., Hauptmann, A. G.,& Ngo, C. W. (2007, September)“Evaluating bag-of-visual-words representations in scene classification.” in Proceedings of *the international workshop on Workshop on multimedia information retrieval*.

[10] Cheung, Y. M., & Deng, J. (2014, October) “Ultra local binary pattern for image texture analysis.” in Security*Pattern Analysis, and Cybernetics (SPAC), 2014 International Conference*.