MCV-M3: Image Classification (week 3)

Jonatan Poveda, Martí Cobos

Image Classification Pipeline

Previous works (BoVW and SVM)

Input Images and Labels

Feature Extraction

SIFT descriptor

Global image representation

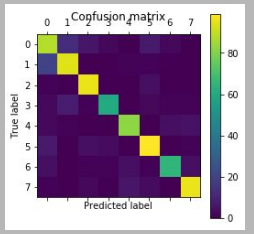
BoVW framework

Classify Image

SVM classifier

Evaluate Code Performance

Compute metrics using ground truth information



Multi-Layer Perceptron

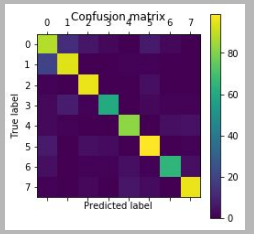
Input Images and Labels

CNN

Neural networks

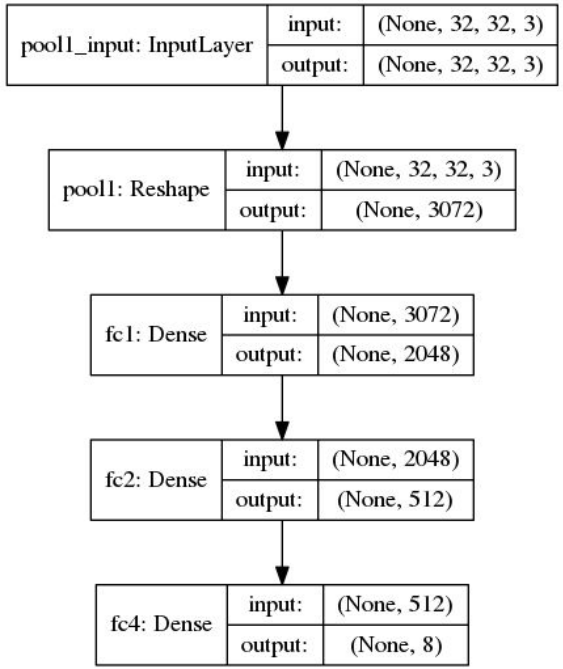
Evaluate Code Performance

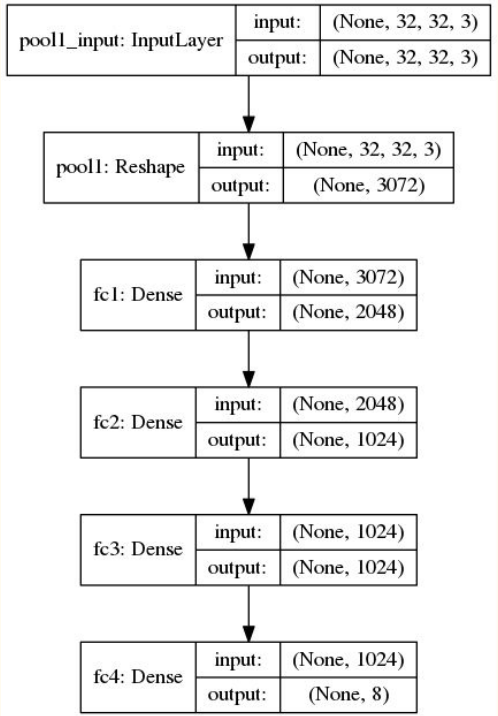
Compute metrics using ground truth information

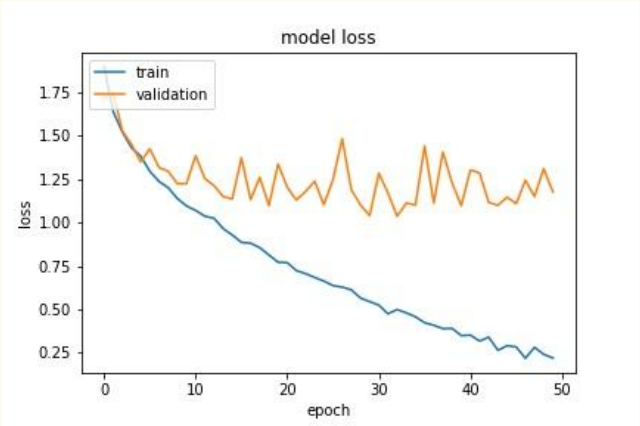


Task 1: Add/change layers in the network topology

- Several topologies have been tested by adding and changing layers on the base network architecture. The best topologies are shown below. Full evaluation results shown on slide 8







Model loss: there is no loss train convergence with 50 epochs. More epochs should be taken into account, or the model should be modified

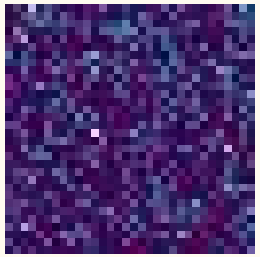
Task 2: Given an image, get the output of a given layer

Input image

MLP Topology

pool1_input: InputLayer	input:	(None, 32, 32, 3)
	output:	(None, 32, 32, 3)
pool1: Reshape	input:	(None, 32, 32, 3)
	output:	(None, 3072)
fc1: Dense	input:	(None, 3072)
	output:	(None, 2048)
fc2: Dense	input:	(None, 2048)
	output:	(None, 1024)
fc3: Dense	input:	(None, 1024)
	output:	(None, 1024)
fc4: Dense	input:	(None, 1024)
	output:	(None, 8)

MLP output is difficult to interpret using only fully connected layers.



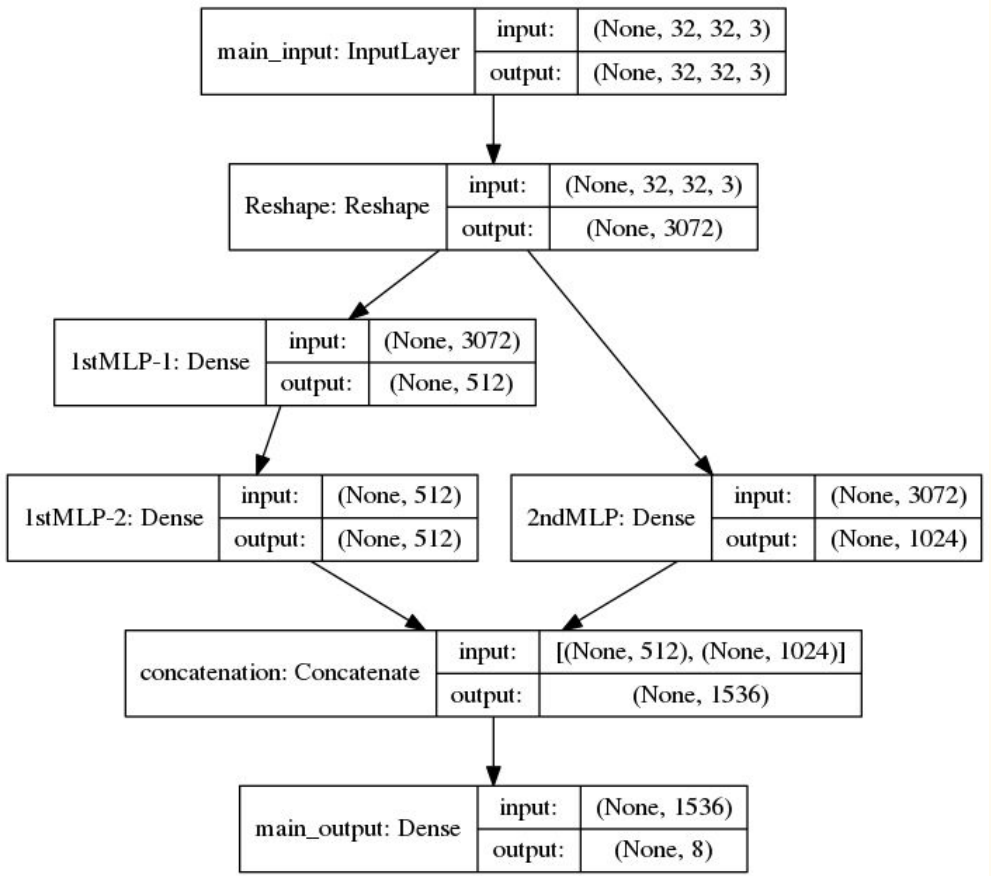
Output image

Reshape array to 32x32 image

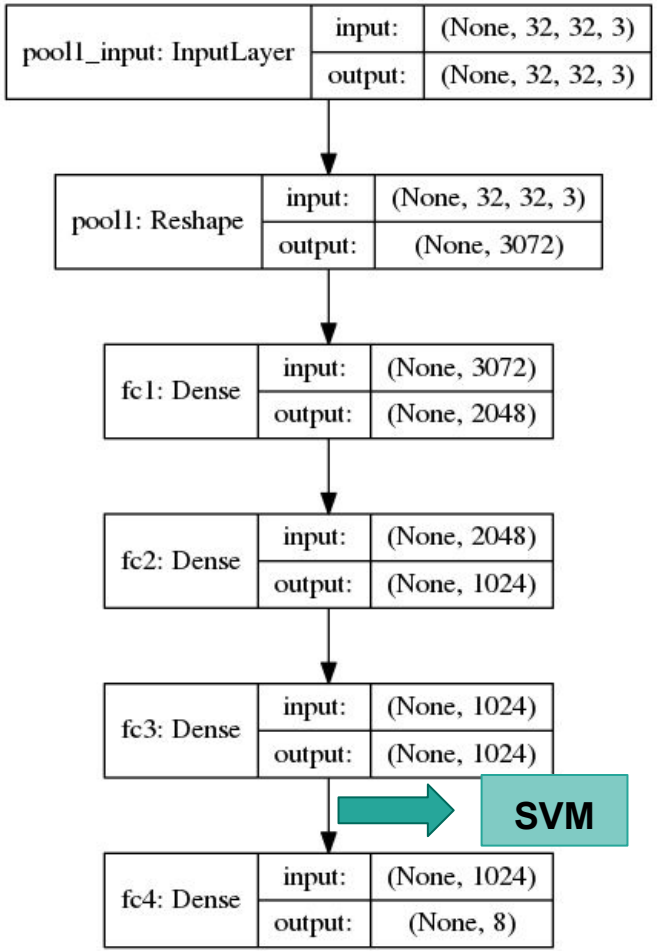
Task 3: Merge multiple outputs from a single image in an end to end network

- Merge two multilayer perceptron by concatenating their outputs
- Architecture implemented shown on right figure
- Accuracy on test images: 0.649
- Architecture implemented using Keras functional API:

```
#Input layers
main_input = Input(shape=(self.IMG_SIZE, self.IMG_SIZE, 3), dtype='float3
inp = Reshape((self.IMG_SIZE * self.IMG_SIZE * 3,),input_shape=(self.IMG_

# First branch layers
first = Dense(512, activation='relu', name='1stMLP-1')(inp)
first = Dense(512, activation='relu', name='1stMLP-2')(first)

# Second branch layers
second = Dense(1024, activation='relu',name='2ndMLP')(inp)

# Concatenate the previous layers
x = concatenate([first, second],name='concatenation')
main_output = Dense(units=8, activation='softmax', name='main_output')(x)

# Compile the model
self.model = Model(inputs=main_input, outputs=main_output)
```



Task 4: Extract a single feature from an input and apply to SVM



Procedure to apply SVM to the MLP output

1. Train multi layer perceptron with train images
2. Get the image output from the last hidden layer
3. Cross-validate SVM with output values to find best SVM kernel and parameters (best kernel: linear)
4. Train SVM with train images
5. Predict test images and evaluate classifiers

Evaluation results on test images:

Image based	Accuracy	Precision	Recall	F1-score	time[s]
MLP	0.640	0.640	0.657	0.646	1122.0
SVM	0.660	0.674	0.667	0.670	1141.8

- SVM results are slightly better than MLP

Task 5: Extract multiple features from an image and apply BoW

- Use patches of images instead of the whole image
- Simulates learning using areas of the image ~ convnets
- Using 64 patches of size 32x32 per image results in a dataset 64x bigger
- Implemented codebook learning with patches to avoid the out-of-memory error

Method not finished because a problem training the classifier in the GPU server (seems an issue because of a different versions of Keras: 2.1.2 on laptop, 2.0.4 on server)

```
Traceback (most recent call last):
  File "session3.py", line 49, in <module>
    layer=neural_network.LAYERS.LAST, image_set='train')
  File "/home/master10/scene-classificator/source/MLP.py", line 223, in get_layer_output
    features = model_layer.predict_generator(generator)
  File "/usr/local/lib/python2.7/dist-packages/keras/legacy/interfaces.py", line 88, in wrapper
    return func(*args, **kwargs)
TypeError: predict_generator() takes at least 3 arguments (2 given)
```

Fixed updating to 2.1.2 but still has some unresolved error on running

Even rebuilding the keras doc for v2.0.4 gave no clue

Classification performance evaluation (on test images)

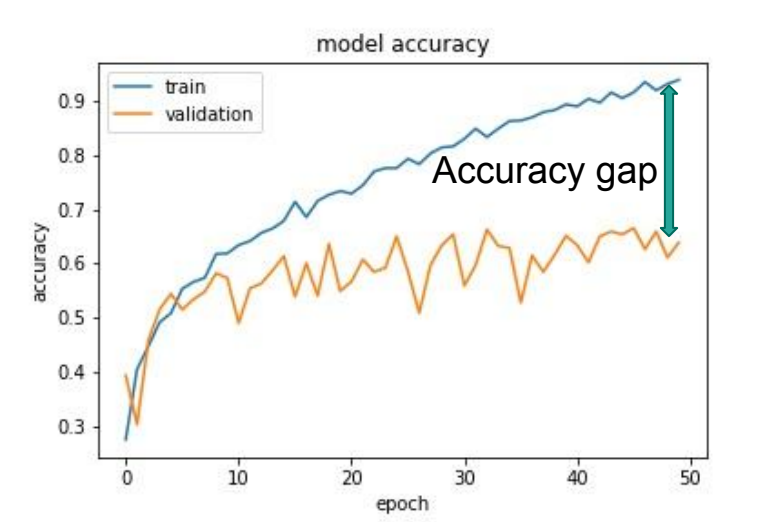
Image based	Accuracy	Precision	Recall	F1-score	time[s]
Method 1	0.647	0.647	0.700	0.647	1631.5
Method 2	0.640	0.640	0.657	0.646	2122.0
Method 3	0.649	0.649	0.674	0.646	1115.7
Method 4	0.660	0.674	0.667	0.670	2141.8

Algorithm performance evaluated on test images using the following architectures:

- Method 1: 2 hidden layer multi layer perceptron
- Method 2: 3 hidden layer multi layer perceptron
- Method 3: Concatenation of two multi layer perceptrons outputs (task 3)
- Method 4: 3 hidden layer multi layer perceptron + SVM (task 4)

Conclusions

- On all the multi layer perceptron architectures tested there was a huge difference between train and test accuracy. See accuracy plot below:



The accuracy gap is an in an indicator that a Multi layer perceptron with fully connected layer do not generalize well and are not the best architectures to classify images. Results are expected to improve when implementing CNNs

- Best results achieved using method

Pixel based	Accuracy	Precision	Recall	F1-score	time[s]
Method	0.660	0.674	0.667	0.670	2141.8

Method 4: 3 hidden layer multi layer perceptron + SVM (task 4)

Conclusions

Best method Confusion Matrix plot

- Normalizing by True Labels, we got a probability of predicting the right class between ~0.53-0.75 among all classes.
- The worst case is predicting the class 3 which is predicted as 4 the 13% of times.
- The best cas is predicting class 7 which is predicted as 3 10% of times.