Coursework 3 – Scene Recognition

Evangelos Raftopoulos
University of Southampton
er3n21@soton.ac.uk

Abhilash Pulickal Scaria
University of Southampton
aps1n21@soton.ac.uk

Ajay Arokiasamy
University of Southampton
aea1n21@soton.ac.uk

Aditya Menti Gowrish
University of Southampton
agm2g21@soton.ac.uk

Abstract

There are a lot of classical computer vision techniques and machine learning methods that can be used for the purpose of scene recognition. Lately, deep learning has brought a revolution in image classification producing high accuracy. This paper describes some of the prominent methods used for the classification of images along with their performance and results.

1. Introduction

In this coursework, we have to try some different approaches to image classification. The dataset that was used contains ten classes with 100 images each for training. We used some computer vision methods to solve this task, such as the K-NN algorithm, bag-of-words with k-Means, SIFT and dense-SIFT using a spatial pyramid. Next, we thought it would be interesting to try some Deep learning methods like Convolutional Neural Networks and Transfer learning with some pre-trained networks to compare the results. Finally, we used some of these methods to predict the classes of almost 3000 unseen images. Because we do not have the labels of testing images, we split the training images into two categories. One for training, containing 90 images for each class and one for testing to validate our models containing 10 images per class.

2. Run 1 - K-NN

The first approach used the K-Nearest Neighbours (k-NN) classifier. K-Nearest Neighbours is a supervised machine learning algorithm because it learns from a labelled training dataset by taking in the training data and its labels and mapping the input X to its desired output y.
This algorithm is one of the simplest of the ML algorithms because it simply learns the entire training dataset and, for prediction, gives the output as the class with the majority in the 'k' nearest neighbours calculated according to Euclidean distance. Thus, it classifies unseen images by finding the most common class among the k-closest examples. More specifically, it computes the Euclidean distance to every image belonging to the training dataset and obtains the k-training images closest to the test image. Then it outputs the class after a majority voting procedure from the labels of these 'k' neighbours. The only parameter that we must choose is the value of K, which is the number of nearest neighbours that are taken into consideration when classifying a given image.
We used' tiny image' features to run this algorithm in our dataset. To create these, we simply crop each image to a square about the centre, and then we resize them into 16x16 pixels each. After that, the pixel values were packed into a vector by concatenating each image row. Finally, some of these images were used for model training and the remaining for validation. We tried this algorithm using values of 'k' from 1 to 25, and we found that k=3 was the optimal value with an accuracy of 17.87 %. Notably, this classification result was expected because K-NN is a very simple algorithm and by using tiny images, we lose information too.

3. Run 2

Then we tried another computer vision method using bag of visual words. Bag of Visual Words(BoVW), as shown in Figure 1 [3], is similar to the Bag of Words(BoW) method used in NLP. In BoVW we use image features as the words. We extract features (custom patches, SIFT descriptors etc) from training images and cluster them using K-Means to create codewords. The collection of these cluster centers form the codebook. Using this codebook we can represent images as a histogram in which each bin represent the count of features that fall into that cluster.Once we have this representation we can feed it into a classifier like SVM for

Table 1. Results of Bag of Features

Value of k	Patch size	Step size	Accuracy
80	16x16	8	45.33
80	32x32	16	41.33
80	32x32	32	38.66
200	16x16	8	36
500	32x32	16	28,66

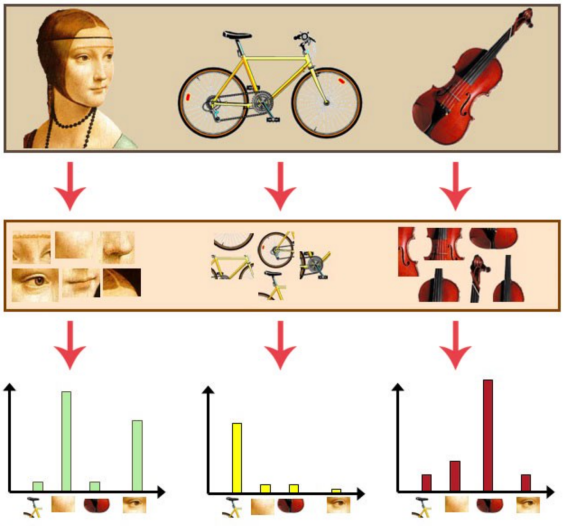

Figure 1. Bag of visual words

training. We did the classification using a linear SVC and 1 vs 15 SVM classifiers. T The results with different values of k, different size of patches and step size which were tried are shown in Table 1. Extracting the features using small patches and small step sizes consumed a lot of time and memory. Moreover, clustering the features(flattened patches) with large k values also took a lot of time. We were able to obtain the best result in our hardware, i.e. accuracy of 45.33% by using 16x16 patches moved across the image in x and y directions with a step size of 8 and by using 80 clusters in the K-Means algorithm.

4. Run 3

4.1. Computer Vision methods

4.1.1 dense SIFT with SVM

As the first approach for run 3, we developed a classifier which uses Dense Sift for feature extraction and Support vector for classification [3]. Sift(Scale Invariant Feature Transform) is an algorithm used to locate local features in an image. These features are scale and rotation invariant.We used dense sift for extracting the features. The features are calculated from densely sampled keypoints.Then we applied K-means to create the codewords and codebook as we did in run 2. Using this codebook we can represent

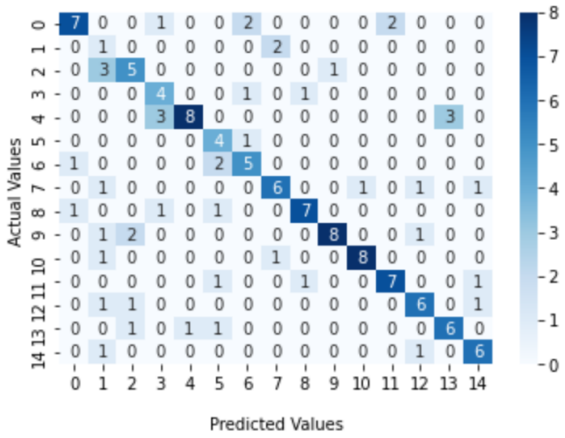

Figure 2. Confusion matrix for k=80

images as a histogram in which each bin represent the count of features that fall into that cluster just like we did in run 2. Then the bag of visual words created can be used for training. Training was done using a Support vector classifier. After training and testing with different K values, we obtained the best accuracy of 66.66% at K = 80. We also tuned the regularisation parameter (c) of SVC. The accuracies for different values of k and c are shown in the table 2.

Table 2. Results using D-Sift

Value of k	Best value of C	Accuracy
40	0.095	65.18
50	0.095	62.96
60	0.093	64.44
80	0.0082	66.66
100	0.027	64.44

To have a complete understanding of our model, we created the confusion matrix for k=80. As we can see, most of the unseen images are classified into the correct class, but there are some images that are misclassified.

4.1.2 Spatial Pyramid Matching

Since the Bag of visual words representation represents an image as an orderless collection of local features. The features are encoded into a single code vector and as a result we end up losing all the spatial information. So we implemented Spatial Pyramid Matching to improve the performance. Spatial Pyramid Matching is an idea proposed in Lazebnik et al 2006 [4] which helps incorporate spatial information into the final vector. The image is subdivided at different levels of resolution. The count of features falling into each bin at each level of resolution and each channel are calculated. The spatial histograms are then combined

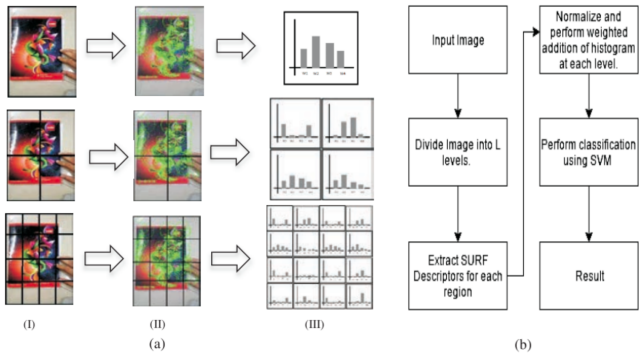

Figure 3. Algorithm for spatial Pyramid matching architecture

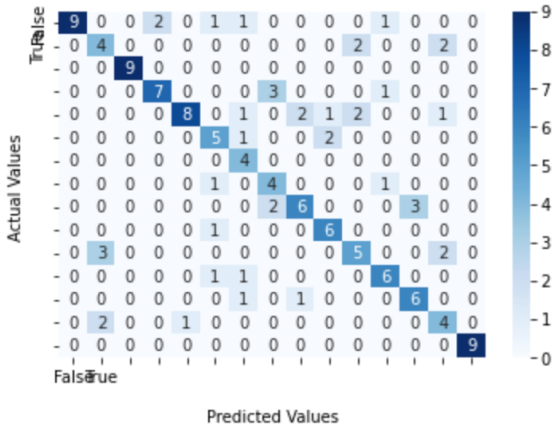

Figure 4. confusion matrix for k=60

after weighting. In our case the image is broken into different sub regions and descriptors(Dense SIFT) at each region are calculated. Using this histogram of visual words at each region is calculated and combined to a single 1D vector. In our case we take L = 2 and break the image into 3 regions. At each layer it is further broken into 2^{2l} regions. The size of codebook M is 80(number of clusters). The histogram of visual words for each region is then calculated and combined into a 1D vector. In our case the length of vector obtained is 1700 ($Length = M * (4^l + 1 - 1) \div 3$). In the figure 2 it is shown the describer algorithm [5]
The results for different values of k are depicted in the following table

Table 3. Results using Spatial Pyramid

Value of k	Best value of C	Accuracy
40	0.0003	63.7
60	0.00049	70.37
80	0.00035	68.88
100	0.0003	65.92

The confusion matrix for k=60 is shown in the figure 4

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_6 (Conv2D)            (None, 222, 222, 64)      1792

dropout_4 (Dropout)          (None, 222, 222, 64)      0

conv2d_7 (Conv2D)            (None, 220, 220, 448)     258496

max_pooling2d_2 (MaxPooling  (None, 55, 55, 448)       0
2D)

dropout_5 (Dropout)          (None, 55, 55, 448)       0

conv2d_8 (Conv2D)            (None, 53, 53, 64)        258112

flatten_13 (Flatten)         (None, 179776)            0

dense_48 (Dense)             (None, 256)               46022912

dense_49 (Dense)             (None, 15)                3855

=================================================================
Total params: 46,545,167
Trainable params: 46,545,167
Non-trainable params: 0
```

Figure 5. CNN architecture

4.2. Deep learning methods

After we tried some computer vision methods for the image classification in this sector, we moved onto some deep learning techniques. Deep learning has broken the mold and ascended the throne to become the state-of-the-art computer vision technique. CNN are the most known neural network for image data spaces, and they are used exceptionally well in computer vision problems such as image classification and face and object detection. For that reason, we thought it worthwhile to try some deep learning methods for image classification.

4.2.1 Convolution Neural Network

First of all, a Convolutional Neural Network was created that contained three convolutional layers with relu as an activation function. Two dropout layers for regularization and two dense layers, one with relu and one with softmax as activation functions. We can see the model in figure 5.
For training, we used adam as an optimizer and 100 epochs. The results were not good. It was about 10%. That was because the training dataset was small. More specifically, we used only 90 images for each class and ten images for testing. To be appropriately trained, CNN has to be trained in a large dataset. For instance, when CNNs used for image classification in Cifar10 dataset with 35000 images for training, the results were about 70%.

4.2.2 Transfer Learning

Because we did not have a big enough dataset to train our model, we tried another method named transfer learning. Some modern convolutional networks like ResNet50,

VGG19 [2], Inception and AlexNets were trained on the ImageNet dataset, which has over 14 million images sorted in 1000 categories. Compared to that, our dataset, which has 1500 images is extremely small. Thus, we used some of these pre-trained networks using their weights, and only the final convolutional layers were trained with our dataset and changed the weights. First of all, we tried VGG19 network. In the next figure, its architecture is shown. [2]

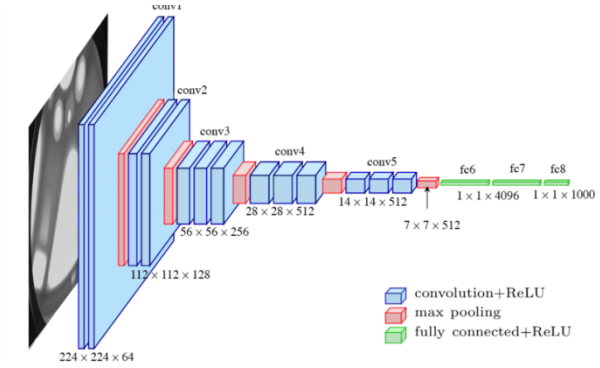

Figure 6. VGG19 network architecture

It takes an input image with 224x224x3 dimensions. In the first part of the network, there are five convolutional blocks with relu as an activation function and between them, there are max-pooling layers. In the final part, there are three fully connected layers [2]. These are the layers that will be trained with our image dataset. This classifier will have only 15 output neurons in the last layer, one for every class, with softmax as an activation function. Finally, all the layers of the convolutional part were frozen, so we only had to train the parameters of our classifier on our small dataset. We have 90 images from every class for training and ten for testing.

After 10 epoch training with SGD as an optimizer and categorical cross-entropy as loss the result in unseen test images was 77

Next, we tried ResNets50 [1]. The architecture of this neural network is pictured in the following image.

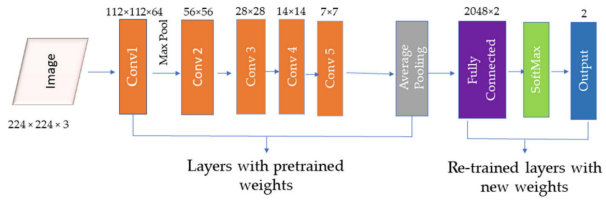

Figure 7. ResNet50 network architecture

Using the same options with VGG19, the accuracy was 87%. More specifically 118 out of 135 unseen images were classified into the correct class.

These promising results were expected because we used state-of-art pre-trained models for our classification task. For that reason, for run 3 results, we used the ResNet50 [1] method to classify the testing images.

5. Conclusion

In this work, we discussed and evaluated the different approaches for scene recognition. We used different computer vision and deep learning approaches on a dataset with 100 images for different scenes(classes). The first approach used a tiny image approach with K-NN, but the results were not good enough, but the accuracy was increased when we used bag of visual words approach with 16x16 patches. Then we used a dense sift-based approach, resulting in a slight increase in accuracy. To improve the performance, we used spatial pyramid matching. To improve the accuracies further, we used different deep learning approaches. We used transfer learning with VGG19 and Resnet50, out of which Resnet50 gave the best performance out of all the methods.

References

[1] Md. Mahbubul Islam, Nusrat Tasnim, and Joong-Hwan Baek. Human gender classification using transfer learning via pareto frontier cnn networks. *Inventions*, 5, 2020.

[2] Yu Ting Li and Jiun-In Guo. A vgg-16 based faster rcnn model for pcb error inspection in industrial aoi applications. *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pages 1–2, 2018.

[3] D.G Lowe. Distinctive image features from scale-invariant keypoints. international journal of computer vision, 2004.

[4] C. Schmid S. Lazebnik and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, 2006. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

[5] Kushal Vyas, Yash Vora, and Raj Vastani. Using bag of visual words and spatial pyramid matching for object classification along with applications for ris. *Procedia Computer Science*, 89:457–464, 2016.