COMP3204 Coursework 2: Scene Recognition

Dan Roberts
University of Southampton
dr1n22@soton.ac.uk

Eric Ding
University of Southampton
sd8g22@soton.ac.uk

Oliver Aldred-Regan
University of Southampton
oar1g21@soton.ac.uk

Samuel Hyder
University of Southampton
sh5g22@soton.ac.uk

1. Introduction

This report explores scene recognition across 15 different categories. Three distinct classifiers were implemented, the first using a k-nearest neighbour classifier with the 'tiny image' feature. The second classifier recognises specific scenes using a bag-of-visual-words (BoVW) representation with 15 linear 1-vs-all classifiers. Finally, the third classifier uses a convolutional neural network to hopefully achieve the best possible accuracy.

The progression of the tasks sees both model complexity and model performance increase; from traditional, basic techniques to state-of-the-art. An overview of the performances of the various models can be seen in Figure 1.

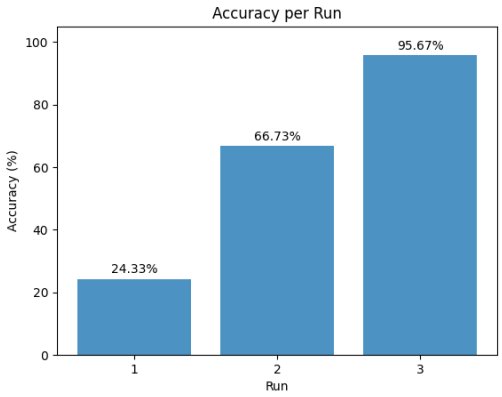


Figure 1: Accuracy of the various models.

2. Run 1

In the first task, a k-nearest-neighbour (kNN) classifier was developed using the provided training images. All images were cropped about the central 16×16 pixels to create a very basic 256×1 feature for the kNN model. An input matrix 1500×256 was constructed with each column representing the feature of a training image. The feature vectors were normalised to zero-mean and unit length 1, to improve the performance of the classifier.

The performance of the model was measured using k-fold cross-validation, with 5 folds. The accuracy of the model was calculated for various values of k (from 1 to 20, inclusive) by computing the ratio of correctly predicted instances to the total number of instances. The results are shown in Figure 2 to identify the optimal k. As shown, $k = 10$ achieves the highest accuracy (around 24%) on the validation sets. The overall performance of the kNN model is relatively low, likely due to the small size of the training set and the simplicity of the feature used to identify images.

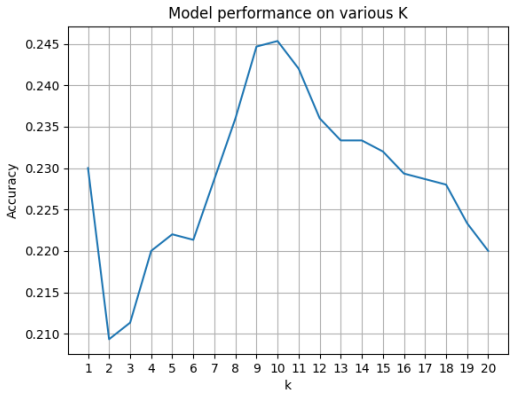


Figure 2: Run 1 model performance

3. Run 2

In this task, 15 one-vs-all linear classifiers were built to determine whether an image instance belongs to a given class. These classifiers make predictions based on features derived from visual words, which are extracted from the im-

ages.

The visual words were obtained by training a K-Means clustering model on densely sampled pixel patches. To extract patches, training images were divided into 8×8 windows with a stride of 4 (as suggested in the coursework description). Examples of individual patches and their representative vector are shown in Figure 3. Each patch was mean-centred and normalized to unit length. The model identifies 750 distinct visual words, meaning a K-Means clustering model with k = 750 was trained on the extracted patches. The resulting visual words, corresponding to the centroids of the clusters, are illustrated in the left column of Figure 3. $k = 750$ provided marginally better accuracy results while not significantly increasing complexity and was therefore chosen over the recommended $k = 500$.

Training images were represented using the visual words detected within them. Each image was divided into patches using the same parameters (8×8 sliding windows and stride of 4) as the BoVW model. These patches were classified using the BoVW model, representing each image as a 750-dimensional vector with each channel indicating the frequency of a visual word's occurrence (essentially a histogram, as shown in Figure 4). Images that share many visual words (i.e. images with similar histograms) likely contain similar features and thus will appear visually similar (as shown in Figures 4 and 5).

The *LogisticRegression* model from the sklearn library was used for linear classifiers for each scene. The model takes a matrix of vectorised images and an annotation vector as inputs. The input matrix was created by converting all training images into 750-dimensional vectors and stacking them to form a matrix of size 1500×750. The annotation vector (1500×1), indicates whether each corresponding image in the input matrix belongs to the target class (0 or 1). A total of 15 classifiers were trained, one for each of the 15 scenes.

Again, the performance of the model was measured using k-fold cross-validation, with 5 folds. The accuracy of the model was averaged over the different folds and correctly identified 66.73% of the validation fold(s).

4. Run 3

Deep learning has emerged as a popular and effective approach to image classification. For this project, we initially attempted to create and train a Convolutional Neural Network (CNN) from scratch on the greyscale scene recognition dataset. However, we observed that training the CNN was slow to converge and prone to overfitting. The overfitting seemed to be primarily due to the limited size of the training dataset, which contained only 1500 images across 15 categories; merely 100 per class. Deep learning models rely on large datasets and an insufficient number of samples can often lead to overfitting, where the model memorises

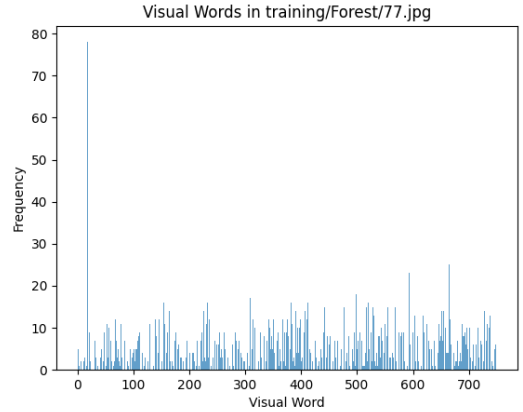
Figure 3: Sample visual words (left column) and patches matching them (right columns). The cosine similarity to the visual word is shown above each patch.

the training data instead of learning general features[3]. Additionally, the slow convergence was likely attributed to the lack of diversity and complexity in the small dataset, which limited the amount of meaningful information available for the model to learn.

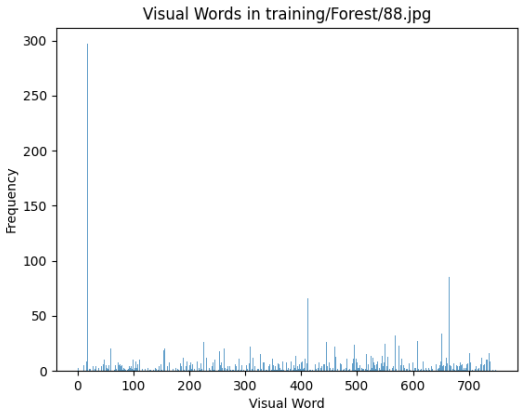
With these considerations in mind, we opted to use transfer learning as an effective solution to mitigate overfitting and improve training time. Transfer learning allowed the model to leverage features previously learned from larger datasets, such as ImageNet, and adapt them specifically to our task[4]. This implementation was carried out using PyTorch; providing a flexible framework for adapting, training and evaluating the model.

For model evaluation, we employed cross-validation, where 20% of the images from each class were extracted to form the validation set. We experimented with several well-known models, primarily variation of ResNet, including ResNet18, ResNet50, ResNet101 and ResNet152 [2]. The final layer of each model is modified to output the required number of 15 classes. We tested the performance of each model on the first 10 epochs, as show in Figure 6. ResNet50 and ResNet152 both converge quickly and achieve high accuracy after 10 epochs. We choose ResNet50 as our backbone model for the balance between accuracy and training cost. It is also less prone to overfitting due to lower model complexity [1].

These pretrained models were originally trained on RGB images from the ImageNet dataset, while our task involved greyscale images. To adapt the models to greyscale input, we explored two approaches: modifying the first convolu-



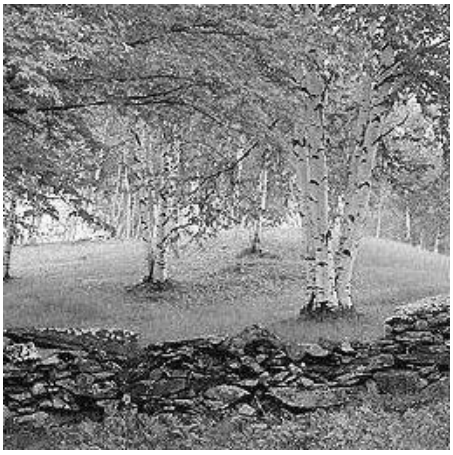
(a) Histogram for image 'training/Forest/77.jpg'



(b) Histogram for image 'training/Forest/88.jpg'

Figure 4: Visual word histograms for two images. Both images contain a large number of visual word #18.

(a) Training image 'training/Forest/77.jpg'



(b) Training image 'training/Forest/88.jpg'

Figure 5: Two training images of the 'Forest' scene. Both images contain a large number of visual word #18, and visually appear quite similar.

tional layer to accept a single input colour channel, or duplicating the single greyscale channel across the three RGB channels. Based on our experiments, the latter approach resulted in better training performance, likely due to the first CNN layer retaining its strong feature-learning capabilities when working with three input channels.

Images were resized to 224×224 to meet the model's input requirement and augmented using random horizontal flip during the training process. This artificially expanded the dataset to twice it's original size, allowing the model to learn more effectively. Pixel values were normalized using channel-wise means of 0.485, 0.456, and 0.406, and standard deviations of 0.229, 0.224, and 0.225. This normalization reduces noise and ensures compatibility with ResNet50's default preprocessing requirements. Similar to Run 1 and Run 2, classes are labelled with numerical indices and cross-entropy loss is used as the loss function. The Adam optimizer was used, to follow the original train-

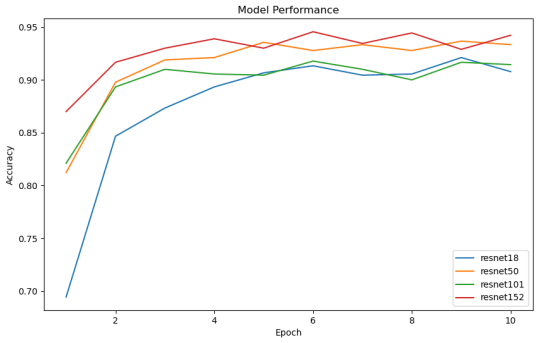


Figure 6: Comparison of various base models

ing approach of ResNet[2]. Initially, the learning rate was set at 0.001, but later adjusted to 0.00001 to improve train-

ing results. The smaller learning rate proved more effective due to the use of transfer learning, which benefits from fine-tuning with smaller step sizes.

The model was trained for 100 epochs, and its performance was evaluated on the validation set after each epoch. The highest validation accuracy achieved was 95.67%. To analyse the model's predictions, we generated a confusion matrix (shown in Figure 7) to demonstrate that the model categorised the majority of instances correctly. However, it struggled with ambiguous samples in classes such as street/highway and mountain/forest/open country. These misclassifications occur because these categories share very similar visual features, such as textures, shapes, noise and edges, making them harder to distinguish. Additionally, the lack of colour information in greyscale images adds to the challenge, as colour can greatly help to separate visually similar scenes. For example, a large amount of blue would easily help separate a 'coast' from a 'forest'. This kind of overlap in features is common in scene recognition tasks and is difficult to avoid completely. Further improvements could involve using additional data or fine-tuning the model, but the misclassifications that are present are expected given the nature of the dataset.

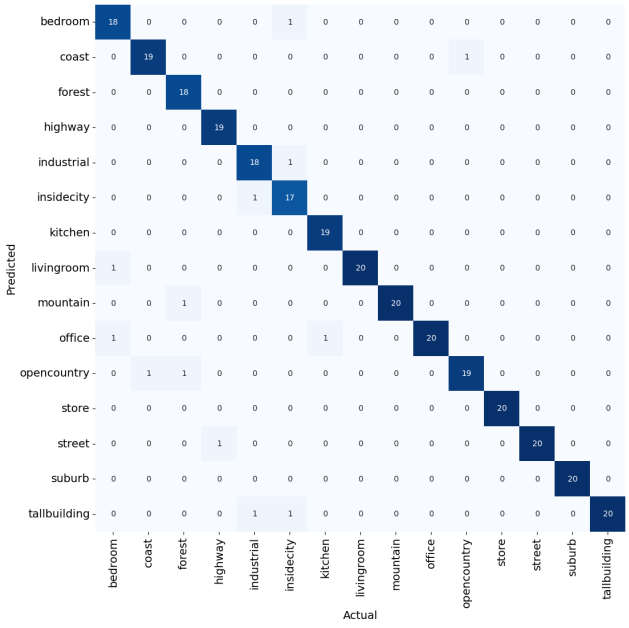


Figure 7: Confusion matrix for Run 3

5. Contributions

Daniel implemented run 1 and run 2. They provided all figures for section 3.

Eric implemented run 3. They provided all figures for section 4.

Oliver and Sam assisted in implementing run 3. They contributed to the report by adding figures and polishing the structure and content.

References

[1] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[3] Chao Luo, Xiaojie Li, Lutao Wang, Jia He, Denggao Li, and Jiliu Zhou. How does the data set affect cnn-based image classification performance? In *2018 5th International Conference on Systems and Informatics (ICSAI)*, pages 361–366, 2018.

[4] Karl Weiss, Taghi M. Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3:9, 2016.