Feature Extraction for Scene Recognition

Nathan Francis and Towner Hale

1 Introduction

This report will be discussing the effect of colour histograms and tiny images as features for a KNN classifier for scene recognition. The performance of the colour histogram and tiny images features will be used as baseline for future experiments that will be conducted at a later date. Part of the SUN database was used for the training and test data; 15 scenes from the dataset were used as the scenes to be recognised by the classifier. The colour histogram method creates a histogram of the colours in an image and colour hsitograms can be compared for classification tasks. The 'tiny image' technique downsamples an image and places the value of each pixel into a vector, which can be be compared to other tiny image feature vectors for classification. While evaluating the feature extraction techniques with the K Nearest Neighbour classifier, various parameters were changed in order to find the best performing set of parameters. A literature review into other image recognition methods, and in particular scene recognition methods was conducted, with a view on covering techniques that improve upon or offer alternative approaches to colour histograms and tiny images. Section 2 gives a brief description of the colour histogram and tiny image approaches, section 3 gives a brief description of how the KNN classifier works, section 4 gives an evaluation of the performance of the colour histogram and tiny images feature extraction techniques and section 5 is the literature review into other image recognition techniques.

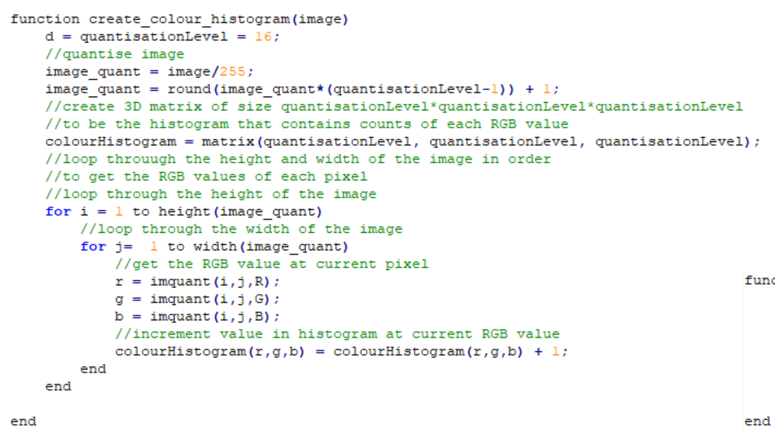
2 Feature Extraction Techniques

2.1 Colour Histograms

Colour histograms can be efficient as an identifying feature for recognizing objects and scenes in images. To create a colour histogram the image is first quantised, which is a type of compression and can avoid overfitting, and then a colour histogram is created which records the number of times each colour occurs in the image. Histograms work because objects tend to have surfaces composed of regions of colour, and while specific matches on the object's surface are not always consistent, there is a high match value if the specific regions match well. In order to identify an object through the colour histogram, there must be a comparison between the colour histogram of an image and the histogram in the database. This comparison can be computed in many different ways such as using the euclidean distance. Swain and Ballard (1991) mentions a way of comparing histograms called the "histogram intersection". The result of the histogram intersection is the number of pixels of the same colour between both the two histograms. This is then stored as the match value. Pixels in the image background can negatively affect the accuracy of the histogram, however the match value will choose to ignore these pixels unless the pixel is the same colour as one of the colours stored in the model, and the amount of pixels of a specific colour are less than the ones in the model. When the object can be segmented from the background the histogram intersection is equivalent to the city block distance metric. Parameters for the colour histograms such as the colour space used, normalisation and quantisation level can be tweaked to get the best performance. Figure 1a shows pseudo code of the algorithm for colour histograms.

2.2 Tiny Images

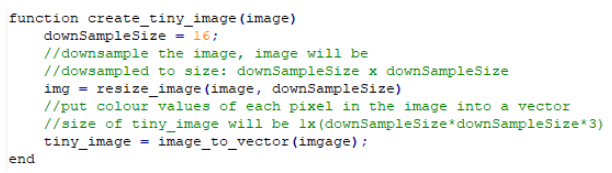
Tiny images are another technique for feature extraction for image classification and it works by downsampling the original image, for example to a 16x16 image, and then stretching out each of the colours into a vector. The feature vector can then be compared to feature vectors of other images for classification. The image is downsampled to save space; if the original size of the image is used, the processing time for classification can be too long which makes images less sensitive to exact alignment. If an RGB image is downsampled to a 16x16 image then the tiny image will have 768 (16x16x3) features, as it has 16x16 pixels and 3 values for the colour. The colour space used for tiny images does not have to be limited to RGB and others can be experimented with, such as HSV and grayscale. An alternative approach to tiny images is to crop the image instead of downsampling it; if the main area of interest in the image is known, then the image can be cropped to that region as this will reduce the size of the image and also mean that the only features used are those of the region of interest. Normalisation can also be used in order to normalise the feature values in the feature vector. Figure 1b shows pseudo code of the algorithm for tiny images.



```
function create_colour_histogram(image)
    d = quantisationLevel = 16;
    //quantise image
    image_quant = image/255;
    image_quant = round(image_quant*(quantisationLevel-1)) + 1;
    //create 3D matrix of size quantisationLevel*quantisationLevel*quantisationLevel
    //to be the histogram that contains counts of each RGB value
    colourHistogram = matrix(quantisationLevel, quantisationLevel, quantisationLevel);
    //loop throuugh the height and width of the image in order
    //to get the RGB values of each pixel
    //loop through the height of the image
    for i = 1 to height(image_quant)
        //loop through the width of the image
        for j=  1 to width(image_quant)
            //get the RGB value at current pixel
            r = imquant(i,j,R);
            g = imquant(i,j,G);
            b = imquant(i,j,B);
            //increment value in histogram at current RGB value
            colourHistogram(r,g,b) = colourHistogram(r,g,b) + 1;
        end
    end

end
```

(a) Pseudocode of the colour histogram algorithm



```
function create_tiny_image(image)
    downSampleSize = 16;
    //downsample the image, image will be
    //dowsampled to size: downSampleSize x downSampleSize
    img = resize_image(image, downSampleSize)
    //put colour values of each pixel in the image into a vector
    //size of tiny_image will be 1x(downSampleSize*downSampleSize*3)
    tiny_image = image_to_vector(imgage);
end
```

(b) Pseudocode of the tiny image algorithm

Figure 1: Pseudocode of the tiny image and colour histogram algorithms

3 Classification Algorithm

A k-nearest neighbour classifier (KNN) can be used for image and scene recognition. The KNN works by using the features of a test image, which are in this case from either the colour histograms or tiny images, and then computing the distance between the test image and every train image. The class of the k training examples closest to the test point determine the test images class through a most vote method. For small k values, training samples similar to the test sample will command its classification, while large k values will have training samples less like the test sample influence its classification. Therefore, choosing a good value for k involves an odd value that isn't too small (over-fitting) and isn't too large (under-fitting).

4 Evaluation of Feature Extraction Techniques

The performance of tiny image feature vectors and colour histogram feature vectors for image classification, and in particular scene recognition, was evaluated by using them with a KNN classifier. The data was partitioned into two sets, a training set and test set, with both sets containing 1500 images. Experiments with various parameter changes to the colour histograms, tiny images and KNN classifier were conducted to see how the classifiers performance was affected by them, and also in attempt to find the parameters that gave the classifier optimal performance. The SUN database [1] was used for the train and test data and it contained images of the following scenes: Kitchen, Store, Bedroom, Living Room, House, Industrial, Stadium, Underwater, Tall Building, Street, Highway, Field, Coast, Mountain, and Forest. The goal of the classifier was to correctly classify a test image as one of the scenes using the features provided by either the tiny images or colour histograms. Both feature extraction techniques had a set of benchmark parameters for any parameter changes to be compared to.

4.1 Colour Histograms

The parameters experimented with for the colour histograms were the quantisation level, removal and inclusion of the colour black in the histogram, normalisation, colour space, number of neighbours in the KNN classifier and the distance method used in the classifier. The benchmark parameters were a quantisation level of 16, a one nearest neighbour euclidean distance classifier, the inclusion of the colour black and an RGB colour space. A quantisation level of 16 was chosen because there was room to both increase and decrease the quantisation level, while a one nearest neighbour should have room for improvement as one nearest neighbour can often lead to overfitting. The assumption that colour is important for scene recognition was held by the RGB colour space, and this could be tested by comparing it to other colour spaces such as grayscale. Black was included in the image as this maintains the natural state of the image, while a euclidean distance was used as this is the most popular distance method used for a KNN classifier. The benchmark achieved an accuracy of 30.3% and in each experiment that is discussed the benchmark model is used changed parameters.

4.1.1 Quantisation level

The quantisation level was changed to determine its effect upon the classifier. Changing the quantisation level changes the compression of the image; the smaller the quantisation level, the more compressed the image. This gives the image less colours. For example, if an RGB image is quantised to 16 then the image becomes a 16x16 image. A heavily quantised image may receive more accurate results and give a more accurate result because it will leave the general colours in the scenes and not the colours specific to that image.

The largest quantisation level, 32, gave the worst accuracy at 24.9%, while the quantisation level 16 gave a much improved accuracy at 30.3%. A quantisation level of eight and four gave an improved accuracy of 31.6% and 32.3%. However, a quantisation level of two achieved an accuracy of 23.4% which shows that an image too heavily quantised loses too much information and has a negative affect on classifier performance. Overall, these results met our assumption that the smaller the quantisation level, up to a certain point, the better the classification performance. Figure 2 shows the results of the quantisation levels.

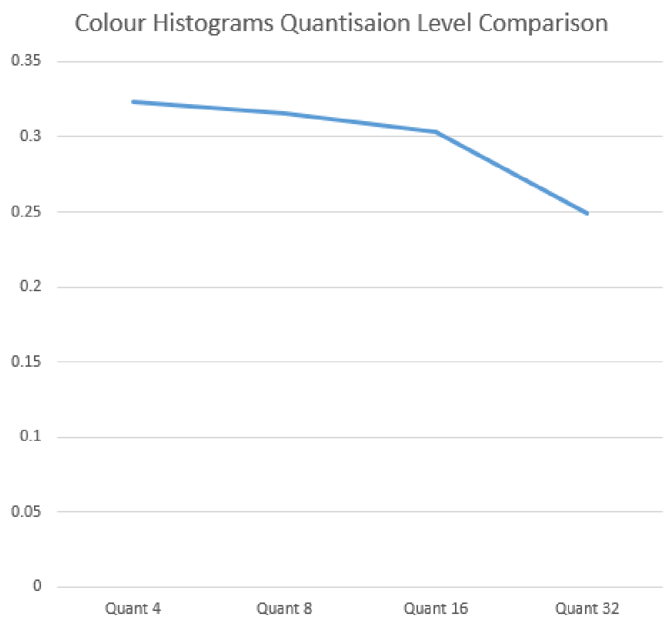


Figure 2: Accuracy of colour histograms with different levels of quantisation.

4.1.2 Removing black

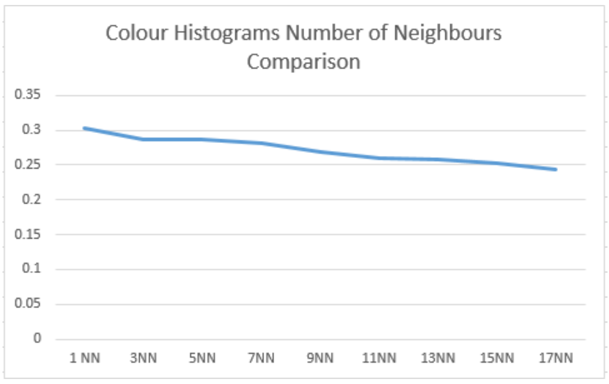
When an image has a substantial number of black pixels or a black background, it can create a spike in the histogram and reduce performance for image classification. To see if this was a problem with the SUN database and for scene recognition, the colour black was removed from the histogram. After running image classification with the changed parameter, the result

[1]See, http://vision.princeton.edu/projects/2010/SUN/,

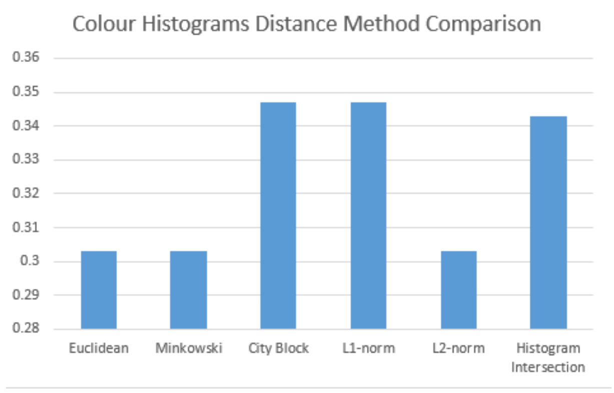
gave an accuracy of 30.6%, a 0.3% increase from the benchmark accuracy of 30.3%. This showed that removing black from the colour histogram improved accuracy in image classification.

4.1.3 Distance methods

Various distance methods were experimented with in the KNN classifier to find its effect upon classification performance. The distance methods that were tested were euclidean distance, Minkowski, city block, L1-norm, L2-norm and the histogram intersection. The City Block distance method was expected to give a better degree of accuracy over the Euclidean method because it performs better for higher dimensional data (Aggarwal et al., 2001). This was proven by the experiment results as both the Euclidean and Minkowski distance methods resulted in an accuracy of 29.8%, while City Block and L1-norm achieved an accuracy of 31.9%. L2-norm's accuracy was lower than L1-norm by 2.1%, at 29.8%. The histogram intersection also had a high level of performance as it achieved an accuracy of 34.3%. Figure 3b displays the accuracy of the classifier for each distance metric that was tested.



(a) Accuracy of colour histograms with different numbers of neighbours



(b) Accuracy of colour histograms using distance methods

Figure 3: Example of a test and train label file

4.1.4 KNN Number of Neighbours

A large set of k values tested for the KNN Classifier to find the highest degree of accuracy. The benchmark, where k equalled one, was not expected to have the highest accuracy because the training samples would dominate its classification and it could overfit. However, the results contradicted this assumption; the benchmark performed with the highest degree of accuracy at 30.3%. As the k values increased, the accuracy decreased, demonstrating that there was a direct relationship between the k value and its image classification accuracy. One reason why the one nearest neighbour could be the most accurate is because the training data and test data are similar, which would mean that overfitting wouldn't be a problem. When changing the benchmark parameters, however, the one nearest neighbour was no longer the most accurate and other k values performed better, which can be seen in section 4.2.7. Figure 3a displays the accuracy of the classifier as the number of neighbours increased.

4.1.5 Colour Space

Images were converted into colour spaces such as RGB, RG (red green), HSV (Hue, Saturation, Value) and grayscale to determine if colour plays a role in affecting the accuracy of the classifier. RG was hypothesized to be less accurate compared to RGB because removing the colour blue could negatively affect the classification as there will be less colour information for the classifier to use.

The RGB colour space gave the highest accuracy of 30.3%, while RG gave the lowest accuracy at 7.1%. Grayscale and HSV were at 20.1% and 20.6% respectively. Therefore, the hypothesis that colour information is important for classification was true as RG and grayscale gave low accuracy levels, although it would have been interesting to see if a data set whose natural colours were dominated by RG or grayscale would give a higher performance than RGB.

4.1.6 Normalisation

Zero mean and unit variance normalisation was tested on the colour histogram in an attempt to make the classifier less sensitive to the scale of features. Zero mean normalisation negatively affected classification performance as it received an accuracy of 30.1%, which is lower than the benchmark. However, unit variance normalisation positively affected classification performance as it received an accuracy higher than the benchmark at 31.3%. This shows that the classifier can be affected by the scale of features. Normalisation might have a different effect on altered parameters, which will be discussed in section 4.1.6.

4.1.7 Optimal Set of Parameters

A histogram feature vector along with the KNN classifier was tested with the optimal set of parameters to see the highest degree of accuracy that could be achieved. The methodology used was to pick the highest accuracy in each category as the parameters. The parameters chosen were: a quantisation level of four, removed black, RGB colour space, city block distance method, and the first nearest neighbour. These parameters gave an accuracy of 32% percent, which was lower than using the benchmark with a quantisation level of 4 at 32.3%. This showed that placing the best performing parameters into one model wouldn't necessarily lead to the best performing model. The parameters were tweaked with and the best performing model received an accuracy of 40.1% with a quantisation level of 7, the inclusion of black and using a seventh nearest neighbour city block classifier. There was also a 0 mean normalisation used on the colour histogram. This shows that the effect of black on classifier performance is very sensitive to the other parameters that have been used, and that normalisation had to be used when there was a lower level of quantisation was as it may have lead to big differences in the colour histogram's scale. Figure 4 is the confusion matrix for the best performing colour histogram model. From looking at the confusion matrix it can be seen that the scenes in the top left were misclassified often, this is most likely because they are all indoor scenes. A forest and field were also misclassified often and this is most likely due to both scenes containing a lot of green colours.

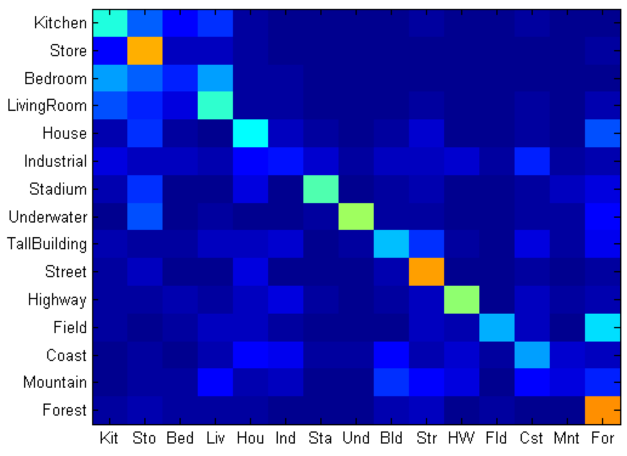


Figure 4: Confusion matrix for best performing parameters for colour histogram.

4.2 Tiny Images

The parameters that were experimented with for the tiny images were the downsample size, cropping of the image opposed to dowsampling it, the colour space of the images, the number of neighbours in the KNN classifier and the distance method used in the classifier. The benchmark parameters for the tiny images were a 16x16 image downsample, an RGB colour space and a 1 nearest neighbour euclidean distance classifier. The RGB colour space was chosen as this was the colour space of the images in the SUN database and an assumption was held that colour is important for scene recognition. A one nearest neighbour classifier should have room for improvement as using one neighbour is prone to overfitting, while the euclidean distance is the most common distance method used so this was included in the benchmark parameters. A 16x16 image downsample size was chosen as there is still room to increase and decrease the downsample size, and there is also the option of cropping the image as an alternative to downsampling it. The benchmark achieved an accuracy of 29.8%, and in each experiment discussed the benchmark model has changed stated parameters.

4.2.1 Colour Space

Images were converted to the RGB, RG (red green), HSV and grayscale colour spaces to see their effect on the classifier. The RG and grayscale colour spaces were not expected to perform better than RGB, however if they had a similar accuracy, then it would show that colour is not that important in scenery classification when compared to pixel intensity. HSV (Hue, Saturation, Value) is a colour space that is similar to the way humans perceive colour and describes colours in terms of their brightness, value, and shade. It would be interesting to see how HSV affects tiny images.

The RGB colour space gave the highest accuracy at 29.8%, however the HSV colour space had only a slightly lower accuracy of 29.5%, so no definitive conclusion can be made as other datasets may give slightly different results. The RG and grayscale colour spaces had a far lower accuracy of 20.3% and 19.1% respectively, which shows that colour is important for scenery classification and the more information there is in the colour space the better the classifier can perform. Figure 5b displays the accuracy of the classifier for each colour space.

4.2.2 Downsample size

Downsampling experiments were conducted by downsampling each image to a set size. Experiments were ran with downsample sizes of 4x4, 8x8, 16x16 and 32x32. The size of the image should have a significant effect on the classification performance and it was expected that the lower the downsample, up to a certain point, the better the classifier performance because it makes images less sensitive to exact alignment. However, a downsample size that is too low can mean there is not enough information in the image to make a good classification decision.

The experiment results met the assumptions as the smaller the downsample size, the higher the accuracy of the classifier, however any downsample sizes of 2x2 and lower received a much lower accuracy score than that of 4x4. The optimal downsample size was 4x4 with an accuracy of 31.7%. Figure 5a displays the accuracy of the classifier as the downsample size increased.

4.2.3 Normalisation and Image Cropping

Zero mean normalisation and unit variance normalisation were both tested. Cropping the image so that the centre of the image was used instead of downsampling the image was also tested. Normalising the features meant that no features were more important than others for the classifier as this would give certain pixels more influence on the classifier. Cropping the image significantly decreased the accuracy at 15.6%, and this is most likely because not enough of the image was retained to make a good classification choice. Zero mean normalisation improved performance compared to the benchmark as it received an accuracy of 30.6%, whereas unit variance normalisation decreased performance with an accuracy of 27.2%.

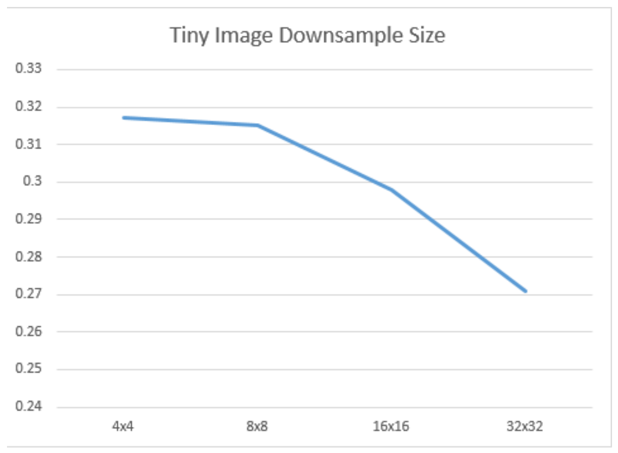
4.2.4 Distance method for KNN Classifier

Various distance methods were used for the KNN classifier, and their performance for image classification with the tiny image feature vectors was evaluated. Euclidean, city block, L1-norm (least absolute deviations) and L2-norm (least squares) and n were the distance methods used.

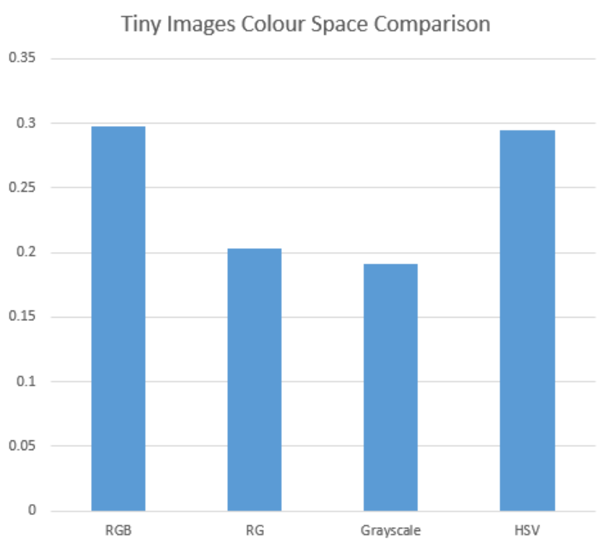
The city block and L1-norm distance methods gave the best accuracies and at 31.9%, which was a 1.9% improvement on the euclidean distance method. The euclidean distance method used in the benchmark achieved the same accuracy as the Minkowski and L2-norm distance methods. The L1-norm method out performed the L2-norm method because it is much more robust and isn't as prone to outliers.

4.2.5 Number of Neighbours in KNN Classifier

Various number of neighbours for the KNN classifier were tested in order to find the best performing number of neighbours. It was expected that using one nearest neighbour wouldn't give the best performance, and an increase in performance would



(a) Accuracy of tiny image feature vectors with different downsample sizes.



(b) Accuracy of tiny image feature vectors with different colour spaces

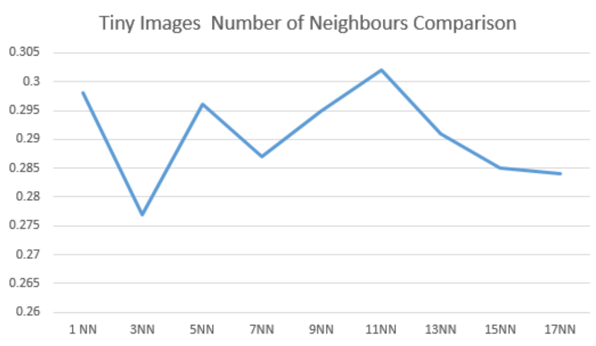
Figure 5: Example of a test and train label file

be seen as the number of neighbours increased up until a certain number of neighbours.

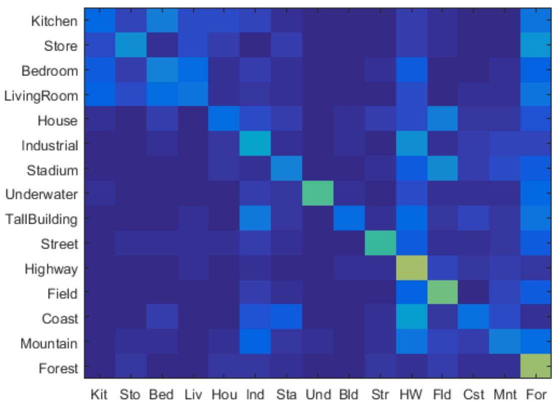
The classifier's accuracy level fluctuated up and down as the number of neighbours increased and no pattern was seen. Eleven nearest neighbours gave the best performance level, however this only meant that for the parameters used in the benchmark, 11 neighbours were optimal and results may vary depending on different parameters. Figure 6a displays the accuracy of the classifier as the number of neighbours increased. This experiment was conducted again once the optimal parameters for the tiny images feature vector were found in section 4.2.6.

4.2.6 Optimal Set of Parameters

A tiny images feature vector and KNN classifier with the optimal set of parameters were used to see how well it improves performance. Zero mean normalisation of the tiny images was applied, with each image downsampled to a 4x4 image in the RGB colour space. The city block distance method was also used for the classifier. While evaluating this model, when normalisation and the cityblock distance method were used, the optimal downsample size was no longer 4x4 as a 7x7 downsample had better performance. Removing normalisation from the model also improved the performance. This was interesting as it showed there is not a fixed downsample size with the best performance and removing normalisation may improve performance as a 7x7 downsample may have meant the feature values lack a large range that could skew the classifier. As the best value of k for the KNN classifier varied for each set of parameters, the optimal k was tested from k equals one to thirty and the best value of k was 9, which had an accuracy of 34.9%. Figure 6b is the confusion matrix for the best performing tiny images model; as has been described in section 4.1.7, the indoor scenes and the field and forest scenes misclassified each other which is most likely due to there being similar colours in those scenes.



(a) Accuracy of tiny image feature vectors with number of neighbours used.



(b) Confusion matrix for best performing parameters for tiny images.

Figure 6: Example of a test and train label file

5 Literature Review

As the accuracy results of the tiny image feature vectors performed poorly and didn't receive an accuracy above 40.1%, a literature review was conducted into other image classification and scene recognition techniques that build on the the tiny images and colour histograms or present alternative solutions. The basic ideas of each technique is described in this section.

5.1 Generative Probability Models

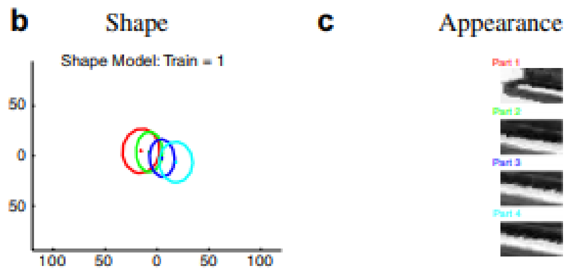
Current approaches towards learning object categories involve a large data set of training images. Previously, learning was not in an incremental manner, and consequently required a large amount of time. Generative Probability Models (GPM) aim to solve this problem by learning object categories based on a small set of training images. The generative probability model itself represents an object's features such as its shape and appearance. Object categories are based on the constellation model, which consists of a number of parts that encode information on the shape and appearance. The appearances and shapes are based on probability density functions. All non-object images are assumed to be modelled by a background with a fixed set of parameters, while the location and appearances of interesting features are noted in the training data. Each feature's appearance is represented as a point, while its shape is a set of coordinates represented by a joint Gaussian density of locations of features. An example of feature detection with GPM in figure 7a

The problem of batch learning, where the the training set is often 5-10 times the number of object parameters, can be solved by the incremental Bayesian algorithm with the GPM. The Bayesian algorithm is used as an alternative to batch learning to train the GPM. The incremental Bayesian algorithm works by updating the model based on its previous images, where a set of statistics is given for each part of the image. The new images then compute its own statistics and compares and then combines it with the previous image statistics. When the model is evaluated on test the data, the incremental Bayesian algorithm is significantly faster at classification than the batch Bayesian algorithm. It is therefore possible to train models of object categories with a small set of images due to a fast learning speed. However, one drawback is that the accuracy of image recognition is lower when working with large training sets compared to the batch algorithm. One possible reason is that the batch algorithm contains all the training images at the same time, while the incremental algorithm passes up less information from one period to the next (Fei-Fei et al., 2007).

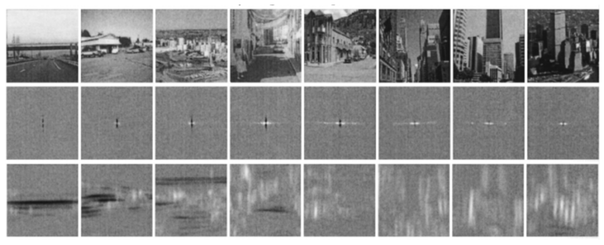
5.2 Spatial Envelopes, getting the 'gist'

This approach was made for scene recognition and is first presented in Oliva and Torralba (2001). It seeks to bypass the segmentation and processing of individual objects or regions in a scene and is based on a very low dimensional representation of the scene that is called the Spatial Envelope. A set of perceived properties which are meaningful to human observers when recognising a scene were chosen; these properties are naturalness, openness, roughness, expansion and ruggedness. These properties represent the dominant spatial structure of a scene and they are used as dimensions and are estimated using spectral and coarsely localised information. A multidimensional space is made where each scene using its dimensions is projected and in this space, scenes sharing semantic categories such as streets or coasts are close together due to the values in each dimension.

The spatial envelope is able to model the shape of a scene and because images of the same scenes have similar shapes, spatial envelopes are able to function. For example, most freeways look like a large surface stretching to a horizon line, and this can be modelled by using the perceived dimensions this can be modelled. The properties for a scene are estimated using either the Discriminant Spectral Template (DST) or the Windowed Discriminant Spectral Template (WDST). An example of a spatial envelope using DST can be seen in figure 7b. This approach has good performance and showed that information about object shapes or identities in a scene is not required for scene recognition.



(a) Example of feature detection with the x and y axis representing the position when measured by pixels



(b) Images of scenes at the top, the degree of the images openness according to DST in the middle and the energy images at the bottom

Figure 7: Generative part models and a spatial envelope

5.3 SIFT

Scale Invariant Feature Transform (SIFT) was developed as a way of getting image features to be used for object recognition, and they have also been used for scene recognition (Xiao et al., 2010). Due to research leading to the belief that visual systems for primates were invariant to location, scale, and illumination, but are very sensitive to combinations of local shape, colour, and texture, a way of doing this in a computer visual system was sought after and SIFT was one of these approaches. In this approach images are transformed into a large collection of local image feature vectors. These feature vectors are invariant to many image parameters such as image scaling, translation, illumination, rotation and 3D projection. This approach was different to previous local feature techniques as previous approaches were negatively affected by changes to scale and illumination. The performance time of this approach is very fast as the SIFT feature vectors can be computed in less than one second. The local features are found by a stage filtering approach, in this approach the maxima or minima of a difference-of-Gaussian function is used to find key feature points and feature vectors are made which describe the regions around those points in a way that is similar to the responses of complex cells in the primary visual cortex in primates. This approach recognises objects by bounding the features to object interpretations, in a process called indexing, and then going through a best-fit solution, which is similar to part of the process of human object recognition. This further demonstrates how this approach is heavily modelled on a primates visual system. Due to this process objects are able to be recognised in cluttered backgrounds from any viewpoint and with varying levels of illumination (Lowe, 1999)

5.4 Self-similarity Descriptors

This method disagreed with the assumption that most object and scene recognition systems have, which is that objects or scenes in an image will share a common underlying visual property across other images that contain that same object or scene. Local self-similarity descriptors instead are modelled on the assumption that the local intensity patterns in objects and scenes are repeated in nearby image locations across all images that contain the same object or scene. In this case local means about 5% of the image, as opposed to global descriptors which would cover the whole image. An image contains many self-similarity descriptors to model the local intensity patterns and these descriptors can be called features. To get the features, patches of pixels are used for each descriptor and enough descriptors are made to cover the whole image. The self-similarity descriptors capture the local geometric layouts of in images such as the colour, edges and patterns in an image. This means that a coloured region in one image can be matched with a textured region in another image if they have similar spatial layouts.

To match the local descriptors of two images, F which is a template of an object and G which contains that same object, local descriptors for each image are created. The local descriptors of image F are put together to form a global ensemble of descriptors which maintains the relative geometric positions of the image. F is found in G if a similar ensemble of descriptors is found in G. This means that the descriptor values will be similar and the relative geometric positions will be similar, while taking into account any small local shifts to account for small deformations. Self-similarity descriptors applied to two images can be seen in figure 8a. This approach allows a wide range of objects to be matched that would be normally be difficult to match, for example complex objects in cluttered images can be matched with a rough hand sketch of the same object and also a differently textured version of the same object can be detected even if it doesn't have clear boundaries. Also complex actions performed by two people wearing different clothes and with different backgrounds can be detected (Shechtman and Irani, 2007). This shows how powerful self-similarity descriptors can be for object recognition and they had also been used for scene recognition in Xiao et al. (2010) and achieved a high level of accuracy compared to other state of the art methods.

This approach is similar and closely related to other methods such as the notion of statistical co-occurrence of pixel intensities across images and internal joint pixel statistics, however these methods only compared individual pixels instead of patches of pixels which can be used to match patterns. Local self-similarities have been used before for texture edge detection and detecting symmetries in images.

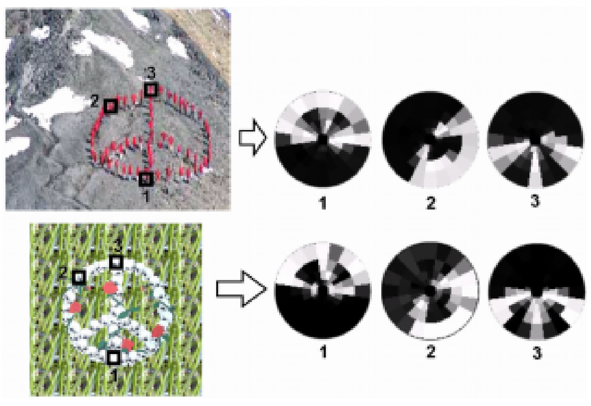
5.5 Bag of Keypoints

Orderless bag of keypoints approaches have been used for scene and object recognition and have gotten impressive results (Lazebnik et al., 2006). They are simple and efficient, however they do not take into account the location of the features of the image nor do they distinguish between the background and foreground of an image (Zhang et al., 2005). Bag of keypoints represent the local features of an image in an orderless fashion. They have been used with many different classifier and with different approaches such as in pyramid matching that is described in section 5.5.1.

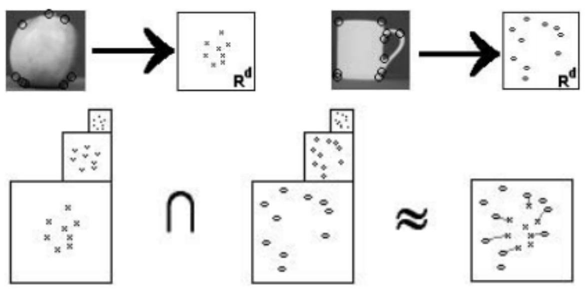
5.5.1 Pyramid Matching

Pyramid matching uses unordered feature sets such as a bag of keypoints, which in the case of image classification is features about the image that are not in any particular order. These feature sets are mapped to multi-resolution histograms, called histogram pyramids, with a new fast kernel and a weighted histogram intersection is then computed. This process can be seen in figure 8b. The kernel is robust to clutter as it doesn't penalize the presence of any extra features. With this approach the pyramid kernel can map features sets of different sizes to a multi-resolution histogram that maintains the distinctness of each feature and the kernel is positive-definite which means it can be used with learning methods such as an support vector machine which is very powerful. This approach to image classification was dramatically faster than other approaches as its kernel can be computed in time that is linear to the feature sets' cardinality and works well with large feature set sizes, and maintained the accuracy of other approaches (Grauman and Darrell, 2005).

This approach set it apart from other approaches that were developed at the time because other approaches suffered from problems such as being infeasible to work with large feature set sizes due to the computational complexities, having kernels that were not positive-definite and therefore being unable to work with learning methods such as support vector machines and being limited to only working with feature sets of equal sizes.



(a) Corresponding self-similarity descriptors for points 1, 2 and 3. Even though the images look different the descriptors for the 'peace' symbol in both images are similar



(b) Pyramid match kernel intersecting histogram pyramids over local features in images

Figure 8: Example of a test and train label file

5.6 Spatial Pyramid Matching

Spatial Pyramid Matching builds upon the idea of pyramid matching and works well specifically with scene recognition. A spatial pyramid works by partitioning an image into increasingly fine sub-regions, locating local features in the sub regions and then creating histograms based off of the local features. In this approach features are no longer an orderless set as they are in pyramid matching as it uses global cues as indirect evidence about the presence of an object. The feature vectors of each image are quantised into a set number of discrete types, which can be seen in figure 9, and make the assumption that only features of the same type can be matched to each other. The weighted histogram intersection is computed for the resulting histograms of the local features just as in pyramid matching. During feature extraction two kinds of features can be extracted. They are weak features which are similar to the features described in the GIST section and are also similar to global SIFT descriptors and the other feature type are strong features which are SIFT descriptors and these provide more discriminative power (Lazebnik et al., 2006). A support vector machine works well with spatial pyramids for classification of images and they were with spatial pyramid matching by Lazebnik et al. (2006).

Spatial pyramid matching performed better than state of the art and more sophisticated techniques on the Caltech-101 database and performed better than other methods based on detailed geometric correspondence and performed much better than orderless bag of features methods (Lazebnik et al., 2006).

5.7 Histograms of Oriented Gradients for Object Recognition

Histograms of Oriented Gradient (HOG) grids offer a method to significantly improving feature sets for detecting objects and they were presented in the paper Dalal and Triggs (2005). An object's appearance and shape can be defined by its local

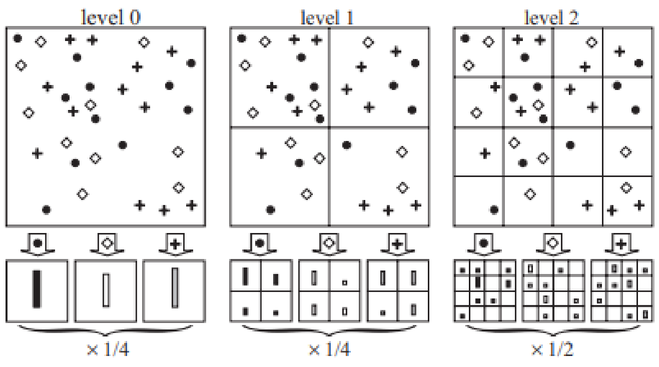


Figure 9: Example of a three-level pyramid that has three feature types, indicated by circles , diamonds and crosses. Each image at the top shows the image being subdivided to a smaller resolution. For each level of resolution and channel the number of features that fall into each spatial bin is counted

gradient intensity and edge directions, which allows HOG's to capture a gradient structure that represents the local shape of the object. HOGs are similar to SIFT features however HOGs are computed on a dense grid of uniformly spaced cells. A basic overview of how HOGs are calculated is by splitting the image into small regions and calculating a histogram of gradient directions or edge orientations for each region and then normalising the histograms. This process can be seen in figure 10. The combined histograms then form the representation of the image. Since gradients derive most of the information from abrupt edges at fine scales, it should be calculated with the finest scale. After testing the HOG on a data set of 2,478 images in total, the HOG outperformed the wavelet and SIFT methods, and gave good results for detecting people while reducing false positive rates. One drawback is that blurring the image drastically reduces HOG's accuracy, so it must run with the finest scale available. However, this also means that the image does not need to be blurred (often in hopes of reducing the sensitivity to its spacial position), because all the information can be found from abrupt edges. HOGs can be used with a support vector machine for classification and during the evaluation by Dalal and Triggs (2005) they were used.

Some possible improvements to HOGs include developing a coarse-to-find detector for the HOG's to identify and using motion information. HOGs have also been used for scene recognition and this was shown in the the experiments in the paper by Xiao et al. (2010) where HOGs achieved very high results, and were better than all other feature types tested such as SIFTs, self-similarity descriptors and spatial envelopes. For scene recognition the histogram intersection of two images is used to define the similarity of two images.

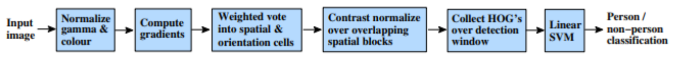

Input image
Normalize gamma & colour
Compute gradients
Weighted vote into spatial & orientation cells
Contrast normalize over overlapping spatial blocks
Collect HOG's over detection window
Linear SVM
Person / non–person classification

Figure 10: An overview of how HOGs are computed

5.8 Convolutional neural networks for image classification

Convolutional neural networks (CNNs), which are deep feed-forward artificial neural networks, have been useful at solving the problem of object classification, however they haven't been widely applied to the problem of scene recognition. This section will be covering the findings made by Zhou et al. (2014) as they applied CNNS to scene recognition and achieved a state of the art results for scene recognition. Convolutional neural networks have improved upon the performance of current object recognition approaches by using a larger database, learning more effective models, and employing different techniques to prevent overfitting. A CNN can make accurate assumptions on the nature of images because they have less parameters and consequently are easier to train. For scene recognition, the CNN proposed by (Zhou et al., 2014) learns deep features in each scene. The CNN then needs an input of a set size of images, so each image will have to be down-sampled to a fixed resolution of 256x256. CNNs can be prone to overfitting, however data augmentation solves this problem by extracting random 224x224 patches from the images. The network then trains the patches, which drastically increases the size of the training set. After, the CNN averages its predictions on five of these patches. This method solves overfitting because the network now grows at a drastic rate. The CNN was trained on the Places database which is a database that was developed and is more dense and diverse than the SUN database for scene recognition. With the methods in place to prevent overfitting and improve efficiency of the neural network, the CNN was able to use supervised learning to achieve an accuracy of 66.2% on the SUN database and 50% on the Places database. The deep features learned from the CNN can be also be used for other visual recognition tasks and when the features learned from the CNN were used with a a support vector machine and evaluated on the SUN database, an accuracy of 54.32% was achieved.

6 Conclusion

In this paper, the performance of the tiny image and colour histogram feature vectors were evaluated to see which technique had the highest accuracy for scene recognition when using the SUN database. The colour histogram had the best performance with an accuracy of 40.1%, in comparison the tiny images highest accuracy was 34.8%. It was seen that zero mean normalisation, a city block distance metric and small quantisation levels between four and 8 worked well for colour histograms.

A literature review was conducted on other methods that build upon or offer an alternative approaches to colour histograms and tiny images for scene recognition and image classification. Some of these methods achieved state of the art results and can be implemented in the future to find their performance for scene recognition with the SUN database.

References

Aggarwal, C. C., Hinneburg, A., and Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*, pages 420–434. Springer.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

Fei-Fei, L., Fergus, R., and Perona, P. (2007). Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70.

Grauman, K. and Darrell, T. (2005). The pyramid match kernel: Discriminative classification with sets of image features. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1458–1465. IEEE.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 2169–2178. IEEE.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

Oliva, A. and Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175.

Shechtman, E. and Irani, M. (2007). Matching local self-similarities across images and videos. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Swain, M. J. and Ballard, D. H. (1991). Color indexing. *International journal of computer vision*, 7(1):11–32.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. (2010). Sun database: Large-scale scene recognition from abbey to zoo. In *Computer vision and pattern recognition (CVPR), 2010 IEEE conference on*, pages 3485–3492. IEEE.

Zhang, J., Marszałek, M., Lazebnik, S., and Schmid, C. (2005). *Local features and kernels for classification of texture and object categories: An in-depth study.* PhD thesis, INRIA.

Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., and Oliva, A. (2014). Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495.