University of Edinburgh
School of Informatics

Automated Classification of Butterfly Images

4th Year Project Report
Computer Science and Mathematics

Mariyana Koleva

April 4, 2013

Abstract: In this thesis the problem of identifying the species of a butterfly from a segmented image is tackled using a bag-of-visual-words approach. We construct and evaluate vocabularies to represent the colour and the shape/texture features of a butterfly image. For each feature the size of the vocabulary and the representation are fine-tuned through cross-validation. SVM and KNN classifiers are evaluated on a 10-species and 13-species datasets. Despite the large intra-class variance, uncontrolled conditions and subtle difference between classes, we obtain a good classification performance of $92.5\% \pm 0.5\%$ using SVM on 13-species dataset consisting of 1012 images. The performance is further improved by combining the colour and texture features using concatenated histograms and it achieve $91.2\% \pm 0.6\%$ correct rate and 2.8% error rate. We draw conclusions about automated species classification and comment on possible integration of the identification procedure in a mobile phone app.

Contents

1. Overview

1.1 Introduction

The objective of this project is to classify butterflies from images. Identification of butterflies is an intriguing problem for many because of the worldwide spread fascination with these insects. They have been the object of exhaustive research not only in the area of Biology, but also Engineering and Ecology & Conservation. As a popular cultural motif in the visual and literary arts they have always been part of the life of people everywhere, hence the interest in their classification. However, due to the large number of species and subtle differences between them, identifying the species of a butterfly is rather difficult even for specialist without some kind of a reference guide. These guides usually take the form of handbooks or charts and can be difficult and frustrating to use by non-specialists. Some on-line resources for butterfly identification exist but they require the user to manually identify and fill in information about various specifics of the butterfly.

The features typically used for classification are the region of spreading of the exemplar, the shape of the wing and overall colour of the butterfly. However, as these do not represent a set of feature discriminative enough to identify a butterfly species, manual comparison between the given butterfly and a set of images might still be required. The main visual features, such as texture of the wing, shape and position of patterns (dots, eyespots, stripes etc.) are subtle and cannot be categorised very easily and included in a general guide. An automatic identifier on the other hand would be able to quickly and accurately determine the name of the species and provide the missing link between a photo of a butterfly and the vast on-line resources available to provide information about a particular species.

Such a tool would be valuable even for experts, as it would not require manual input of various characteristics or browsing through images of tens of butterfly classes. The ability to identify classes or at least to quickly round down the number of possible ones would be quite beneficial and could utilize the advances in digital photography and mobile phone technology, which provide the means to take high-quality images of specimens on the go.

The butterfly identification problem is also an interesting task from the object recognition point of view. A large part of the research in this area is focused on recognition of different categories of objects: people, cars, horses (e.g. PASCAL Visual Challenge [10]). However, the classification of butterflies presents the problem of identification of different species in the same category which requires

its own approach. The differences between members of different classes are a lot more subtle and intra-class variance might be as big as the inter-class in some cases.

There have been several attempts to build an insect identifiers, like ABIS – Automatic Bee Identification System [28], DAISY (mainly tested on moths) [24] and SPIDA (used to classify spiders) [6]. Butterfly specific work was done by [35] – Identification of butterfly families using content based retrieval, and a web platform called Butterfly Ecology [21]. Parallels can also be drawn between the extensive project to build an "Automatic visual Flora", a flower classifier [26], and the project presented here, as they share similar characteristics.

Although satisfactory results were achieved by the above-mentioned systems, they have limitations that make them unsuitable for the butterfly identification task. ABIS requires specialist knowledge to identify the region of interest on the bee wings, as it recognises the type of bee by investigating the wing veins of the specimen. Also, the process is tailored to bees and their characteristics and cannot be transferred to other species, as those specific veins cannot be used for characterization of another category of species. The original version of DAISY achieved over 90% accuracy over 49 moth species, however, it again required skilled operators to take suitable images and segment the photo for correct results. Automatic pre-processing was developed later [37] and the updated system achieved comparable results (85% over 35 classes) when combined with Support Vector Machine classifier. The SPIDA system is used to classify Australian spiders and is tailored to that topic. It employs artificial neural networks and does a hierarchical classification first on genus- then on species-level. The Butterfly Ecology web platform requires the user to fill in information on butterfly characteristics such as colour, shape and pattern to retrieve a set of relevant images and does not provide enough identification benefits for the amount of manual work that is required.

What is more, most of these systems were tested on controlled datasets, consisting of images of dead insects taken under the same conditions, with perfect positioning. This means that the results are not really comparable to the problem addressed here, as the aim is to build a classifier robust to movement and positions typical for a butterfly in its natural surroundings. Also, the project goal is to assist non-specialists as well as professionals, so any system which requires more than basic operation on images or knowledge of taxonomy, or significant supervision and manual work, is not suitable.

One successful project which stands out as rather similar to the butterfly identification problem is the "Automatic visual Flora", the flower identification system mentioned above. Both problems require the correct identification of a species among a large number of classes (e.g. there are around 200 distinct species of butterflies in the UK and the flower identification work was done on 117 classes

of flowers [26]); also the variance between different specimens of the same type might be very large due to the season, gender or weather conditions. Furthermore, neither butterflies, nor flowers can be recognised solely based on their colour or shape. Combinations of shape, colour and texture are necessary for accurate description of an exemplar. These, combined with an SVM classifier were used successfully (average precision 80%) on a number of flower species. The feature extraction and selection methods used in that project offer an attractive approach to the problem.

1.2 Project Description

The goal of the project is to develop a butterfly classifier which can be incorporated in a butterfly identification phone app or used on its own. Such an app can be of use to many, especially people working in the area of conservation. It could prove to be very useful for field work and research as it requires minimum technology to work – a mobile phone with a camera and internet connection, which is a standard nowadays. It has been noted that the main challenges for Ecology and Conservation is documenting species [30]. Considering that all images which are submitted for classification would be uploaded to a server to be identified, the server would then possess a large amount of annotated data of the worldwide spread and number of species of butterflies. This could be extremely useful for tracing and documentation of butterflies and would make the process simpler and faster. Non-specialists might also be motivated to take more photos and upload more data to help the conservation of butterflies and this in turn would further improve the application.

On the other hand, butterfly observation and categorization is of interest to non-specialists as a hobby or for educational purposes. For example, a leaf identification app (LeafSnap [39]) has been used by various schools worldwide to increase interest in biology through practical work and games outside. A butterfly identification app might have a similar effect.

The core of such an app is the butterfly classifier. This paper describes the research and development done on this Computer Vision/Machine Learning problem of successfully identifying the species of a butterfly from a photo. Future work may involve the integration of the classifier in a phone or web app, or the application of the methods used here to other species. The main objective which guided the work was to:

> Develop an automatic classifier which identifies correctly the name of a butterfly species based on a segmented image of the butterfly taken in an uncontrolled environment.

In order to accomplish this several steps were implemented and the following topics have been investigated:

- which of the object recognition methods previously employed can be applied to the butterfly identification problem and if any modifications need to be made,
- the best features or a combination of features which could be used to distinguish between species, the different classifiers and the variation in performance, running time and resources.
- whether the results are comparable to previous work (either on butterflies or other similar problems),
- whether the results vary depending on the datasets, and if these variations could be predicted and explained.

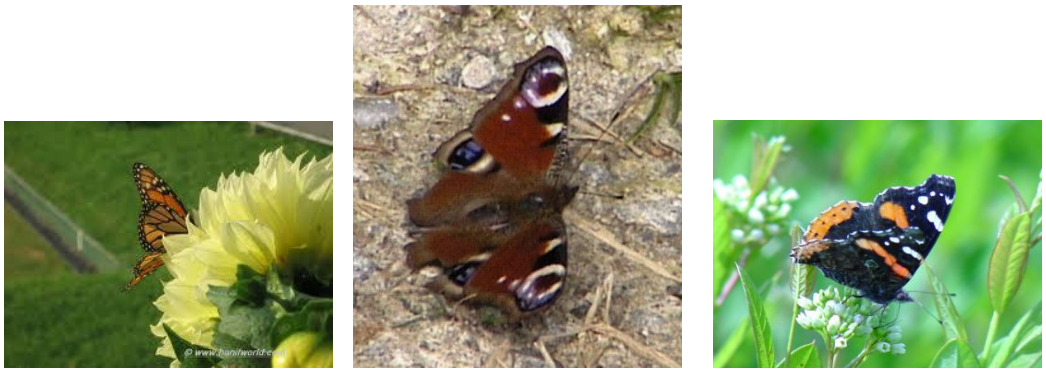
1.3 Challenges

The main challenges which this problem poses can be split into several categories. The typical Computer Vision and Machine Learning issues were present, as well as several problems specific to the task of butterfly identification.

The main objective of a classification system is to not be affected by irrelevant changes in the visual appearance of the samples. These changes might be due to change in the illumination or position of the object. Deformation or occlusion could also be potentially problematic, especially for butterflies, which are often photographed from different points of view and between flowers or other greenery. Low resolution or small size can lead to inaccurate or insufficient feature extraction. This is why it is very important to have a dataset which contains a variety of images. Examples of some of the problems can be observed in Firgure 1.1.

Several challenges arise from the fact that identification is specifically targeted at butterflies. Some of them are outline below.

- Different patterns can be observed depending on which side of the wing is visible. This means that changes in viewpoint may cause large intra-class variance and even significant differences between the same exemplar in different images.
- The overall shape of the butterfly can look very different depending on whether the image was taken when it had its wings open or closed. This makes segmentation as well as classification more difficult, as it is hard to fit a single model to a class or employ unsupervised segmentation techniques based on shape.



(a) Occlusion (b) Deformation (c) Side view

Figure 1.1: Typical problems in butterfly images.

- Typically butterflies are hard to segment from their background as their colours may mimic those of the greenery around. Also they are photographed in variety of settings so the background may include clutter irrelevant to the classification, as opposed to common habitat which might be helpful in some cases.
- Intra-class variance is large due to several characteristics of the butterfly species.
  - Polymorphism – change in the appearance due to different geographical location or season.
  - Gender – sometimes large differences exist between male and female specimens of the same class.
  - Natural events – discolouring of the wings might be problematic for classification based on colour features.

  Usually, these types of problems are solved by the addition of subclasses and the constructions of separate models for each subclass -e.g. open wings and closed wings model, female and male models, etc. [5]. This could potentially solve problems such as viewpoint variance, polymorphism and gender variations. However, even more data is required to build these models and also the number of classes needs to be controlled to ensure scalability and good performance.
- Data collection – it is difficult to collect high-quality photos with varying positions and viewpoints, because the specimens are often moving their wings or flying. Also, the amount of data freely available online is not large.
- Feature selection – the main problem of the butterfly identification is that

a species cannot be conclusively identified based on a single feature, such as colour or shape of the wings. There are hundreds of categories which share similar colours and colour in general is not a reliable feature as there might be large variations due to illumination changes, discolouring or some other camera-specific issue. The outer shape of the butterfly is not discriminative enough on its own as well. The butterflies can be visually divided in 6 major families based on the overall shape of their wings. So a classification on a family-level would be a simpler problem. However, there is a large amount of subfamilies, tribes and common types of butterflies which are from the same family and family classification would bring almost no information about the particular specimen. Hence, some more subtle characteristics are required for the accurate categorization.

- Visual vocabulary – clearly a larger number of more sophisticated features is required for this problem. However, given the large number of classes, this makes the problem computationally expensive. A scalable solution has to be determined to ensure user-friendly experience in practice.

All these issues combined with the general difficulty of object identification make the task quite challenging. However, we have managed to achieve promising results on a 13-class dataset through careful investigation of the problems and the possible solutions.

2. Methodology

In this project we develop a butterfly identification system. The most significant parts of the system are the data and its pre-processing; feature selection and classification.

2.1 Datasets

A significant problem in visual object recognition is obtaining training data. Although images and photos of butterflies are widely available on-line, there isn't a single, annotated dataset which captures all butterflies in UK or North America, let alone larger regions (e.g. Europe) that could be used for extensive training and testing. The images used in the best case would be segmented, as otherwise this would require a significant amount of work on top of the development of an identifier. Furthermore, images would have to present the object in different lighting conditions, viewpoints (single wing, both wings, closed, open, and sideways), with or without occlusions. Although there isn't a single dataset which has all these characteristics, there are two datasets which provide enough image data on a limited number of classes of butterflies.

- Leeds butterfly dataset [36]: 823 images in total, of 10 species of butterflies, distribution of photos ranging from 55 to 100 per class, images are provided with segmentation masks.
- Ponce butterfly dataset [17]: 535 images ranging from 42 to 150 photos per classes, representing 6 butterfly species, no segmentation provided.

Table 2.1 shows the species used for training and testing of the algorithms with some example of each class. Note that initially only the first 10 species presented in the table are used. Later, the extended dataset with all 13 classes is used for evaluation.

Species	Image #	Examples
Admiral	109	

Machaon	46	
Peacock	80	
Giant Swallowtail	20	
Zebra	112	
Cabbage White	55	
0Buckeye	89	
Mourning Cloak	99	
Painted Lady	83	

Longwing	60	
Black Swallowtail	89	
American Copper	88	
Monarch	82	

Table 2.1: The first 10 species of butterflies make up the first dataset and all 13 make up the extended dataset. Experiments were run on both of the datasets. Here we give the number of samples per class and several examples of the segmented images.

Despite the limited number of species captured (13 in total as there is some overlap between the two datasets), these collections are useful because they provide a large number of annotated images in varying conditions, viewpoints and with varying backgrounds. By analysing the outcome of the experiments we were able to conclude if the results would be affected when the problem is scaled to include a lot more classes (for example 200). The computational complexity is discussed to ensure that successful algorithms are scalable and the use of large amount of testing/training data is feasible.

2.2 Features

In this section, the methods used for feature extraction and representation are discussed. A brief overview of the bag-of-words model is provided in the context of the butterfly identification problem as well as some discussion on the significance of local and global features. Finally the feature representation chosen for each

individual feature is explained in detail. Information on implementation and libraries used is provided in Section 2.4.

2.2.1 Bag-of-words approach

The bag-of-words approach is used for text analysis and refers to the fact that documents can be represented as a collection of words with no information on the sequential order of the words in the documents. This is used to determine the topic of a text. The visual analogy is to represent an image using a fixed set of visual words (vocabulary). Csurka et al. [4] describes the 4 steps required to classify images using the bag-of-words approach:

1. feature extraction and representation
2. codebook construction
3. bag-of-words representation of an image
4. training of classifier

The feature representation and extraction is discussed in the next three sections, which give details about colour and shape features as well as features combining colour and shape. The codebook or vocabulary construction involves choosing the right codewords. The codewords are usually obtained by clustering the visual words extracted from a set of training images [4, 8, 15]. Thus the vocabulary is a set of cluster centres. A newly extracted feature is vector quantized to a codeword [29, 15]. A particular training sample is then represented as a visual histogram that stores information about the normalized frequency of codewords in an image.

This approach performs well when combined with discriminative classifiers [40] and has several major advantages over spatial information-preserving approaches. It is spatially invariant because it relies on the presence or absence of features, not their position. Furthermore, it allows for weak supervision, e.g. labelling of individual features or regions of the image is not necessary, which is a common practice in some object identification systems like ABIS or SPIDA. Finally, the bag-of-words method is found to be relatively robust to occlusion [35]. This is particularly important for butterfly identification, because images are usually taken outdoors in an uncontrolled environment and butterflies are often photographed among flowers or other greenery which might cover part of their body.

However, although position and rotation invariance is a desirable characteristic of a feature representation, at least some spatial information might be necessary to build a distinctive model of a butterfly species. For that reason instead of simple histograms, spatial histograms are used for the Shape/Texture descriptors. These

are built by splitting the image into a 2x2 grid, computing the histogram of each grid tile and concatenating those. The histograms used for the colour feature do not preserve any spatial information.

2.2.2 Feature extraction

Local and global features

There are two types of features to be extracted from an image – local and global. Each of them have advantages and disadvantages. Global features, such as colour histograms, are more sensitive to clutter and occlusion, whereas local features give more specific information. However global features are usually easier to extract and no information for the image is required. There is no need for labelling or segmentation of the image as the features are extracted from the whole photo. On the other hand, local features are extracted in a two-step process. It involves finding 'interesting' keypoints and then extracting descriptors for them. There are several methods, usually involving edge detection [14, 3] or detection of large differences in illumination and gradients in nearby regions [25, 11, 22]. Another approach is to densely sample the whole image [18, 38]. This results in a large number of features and is susceptible to clutter and noise. However, the former approach does not guarantee that the extracted keypoints will help discriminate between classes and does not provide as much information as the dense sampling [27]. In this project, global and local features are combined. Local features describing shape and texture are densely extracted over a grid of keypoints and combined in a histogram to produce a global representation. Furthermore, the colour feature is represented as a global histogram.

2.2.3 Feature description

2.2.3.1 Colour descriptors

Originally images are represented as 3D matrices, *height* x *width* x 3, where the last dimension is the RGB-value of each pixel. This representation gives information about the colour features of an image. These, however, are not the most robust characteristics of an image because they are not invariant to illumination or position changes in their original form. Furthermore, working with the images in their original state would mean operating in a very high dimensional space which would make the calculations difficult. A more efficient colour representation is the bag-of-words approach - a colour histogram. Considering that there are 256 values for each of the red, green and blue channels, it is unlikely that more than a couple of pixels would be exactly the same colour. This is because of

illumination and natural colour changes, and mostly due to the fact that natural colours are not that uniform. So, unique combinations of *r-g-b* are not good codewords. Instead we have adopted the 4-step bag of words approach outlined in Section 2.2.1 and constructed a colour codebook by clustering the pixel values. The codewords of the vocabulary are the cluster centres. Given a pixel with RGB values, the closest cluster centre to it is determined and is used as a codeword for that pixel.

This approach resembles image smoothing and makes the representation less sensitive to slight changes in illumination or colour. The colour feature of an image is now an m-dimensional feature vector, where the i^{th} element is the number of occurrences of the i^{th} codeword from the vocabulary, normalized over the number of pixels in an image, where m is the size of the vocabulary. This representation is scale and position invariant.

Although, the RGB colour-map is generally used, it does not mean that it is the best choice for colour representation. In some case, Hue Saturation Value (HSV) or normalized-RG representation might be preferred, as it is generally easier to deal with illumination changes and shadows when working with these. The approach to extract the colour vector is the same, except the original image is first transformed to the desired colour space.

2.2.3.2 Shape and pattern descriptors

Although the colour of a butterfly is important for the classification, it is rarely enough to determine its species (Table 2.2). More discriminative features are the shape of the wing or its pattern (Table 2.3).

Table 2.2: The Longwing has outer shape similar to the Zebra, but different colours on its wings.

By observation of images from different classes, it has become obvious that these local features are quite distinctive and could be used to clearly distinguish between most of the classes present in the dataset. Hence a sophisticated method

Table 2.3: The Zebra and the Swallowtail are coloured similarly but have a different shape.

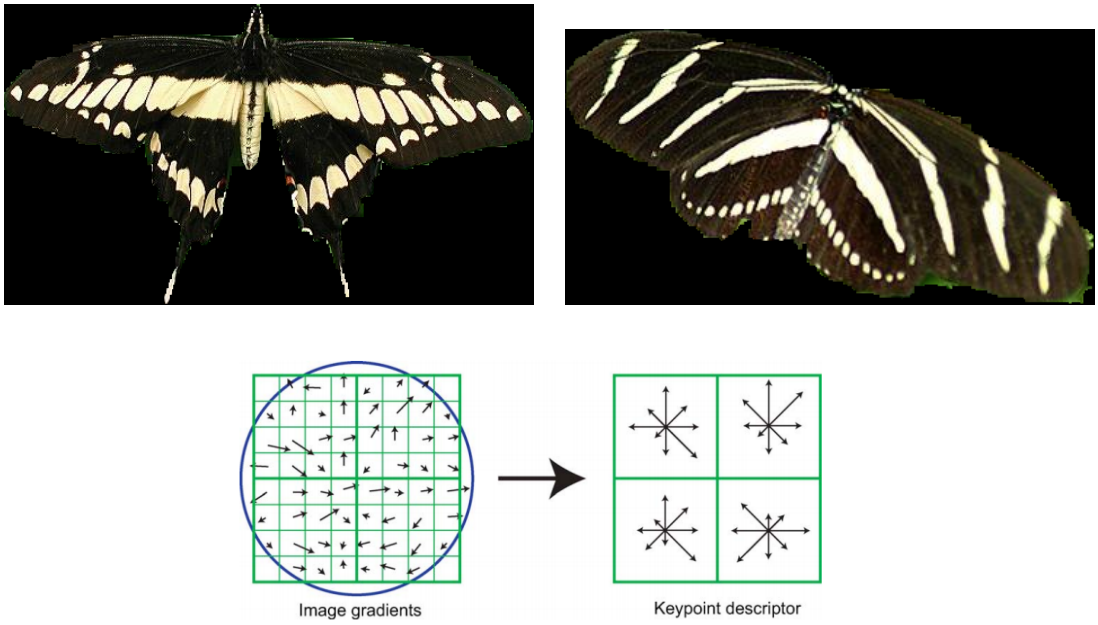


Figure 2.1: The descriptor for a keypoint is created by first computing the gradient magnitude and orientation in a region around the keypoint location, as shown on the left. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right. This figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas the implementation of SIFT uses 4x4 descriptors computed from a 16x16 sample array. [23]

was chosen for their representation to ensure that enough variance is captured but also to make the representation scale and position invariant. The state-of-the-art approach to shape description is the Scale Invariant Feature Transform [23]. SIFT describes a region of an images as a 128-bit histogram of local gradient orientation (Figure 2.1). The gradient of an image is shift-invariant, and under small intensity changes, the gradient direction remains the same. As the descriptors are normalized, the gradient magnitude is irrelevant. If the SIFT descriptors are extracted from a circular patches, they become rotationally invariant.

However as discussed in Section 2.2.1, when extracting local features we need to first determine the 'interesting' points. This is not a trivial task and potentially requires some knowledge of butterfly classification. But even if we knew typical characteristics of a certain butterfly species, it would be hard to determine which of those were the most discriminative given our dataset. Relying on the presence of a particular feature would result in an unreliable system, which performs badly

when part of the butterfly wing is missing, or if we have an atypical image, such as a photograph of several butterflies from the same species. Considering that we would not want to impose any limitations on the algorithms on which features and points we believe are the most interesting, we might try to extract as much information and then reduce it to represent the large variance using clustering.

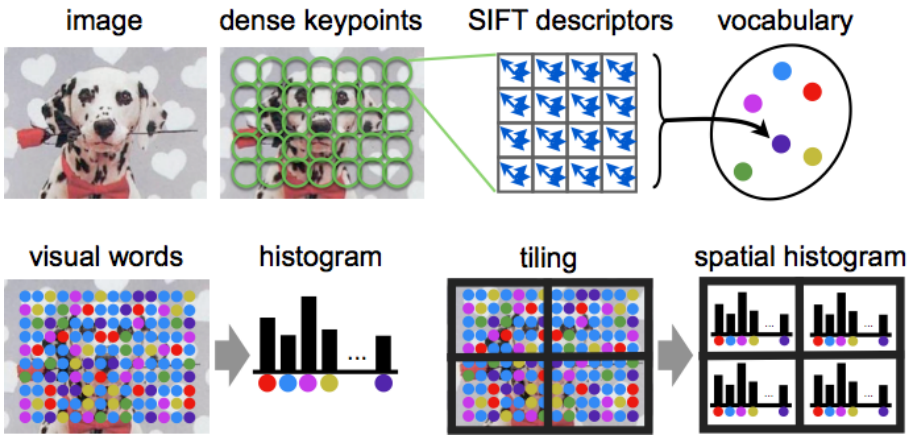


Figure 2.2: The figure [33] shows the steps in bag-of-words method. The first row demonstrates the process of visual vocabulary construction through dense sampling of images and on the second- image representation with the new vocabulary.

Thus, we have decided to use the dense feature extraction method, to ensure that we have enough data to successfully classify a large number of classes. The descriptors are extracted over a dense grid of keypoints. The process is shown in the first three tiles in Figure 2.2. Each gridpoint is the centre of a circular patch of radius R which is represented by a 128-bit SIFT descriptor. These circular patches, represented with SIFT are the visual words that describe the local shape and patterns. This is a dense-SIFT algorithm and is shown to provide more information than bigger sparse key points or areas in object categorization problems [27]. The approach is then typically combined with clustering to reduce the number of visual words and build a visual vocabulary [22, 11], as we did with the colour feature. This vocabulary is global and is later used to describe each image in the dataset regardless of its class. Given a new images, the dense-SIFT are extracted and for each keypoint the closest codeword from the vocabulary is determined through vector quantization [29]. The image is then represented as a normalized frequency histogram of the number of occurrences of each word. This procedure is visualized in the second row of Figure 2.2.

In some cases spatial information might be useful for object identification, however this makes the representation method not rotation- or position- invariant. Using 2X2 spatial histograms can both make the method invariant to small po-

sition changes and at the same time preserve some basic information on spatial distribution of patterns in the image. In this case the shape feature only captures local textures and pattern, so the information about the overall shape and the relations between the different patterns is lost. Thus, a 2x2 spatial histogram is used to preserve some of this information. This is built by simply dividing the image in 4 tiles with the same area size and computing local histograms, which are then concatenated.

2.2.3.3 Combined features

Two or more set of features can be combined to ensure better performance. Also experiments are conducted to determine the best combination of features and to ensure that they are successfully combined. For that to happen, the accuracy of the combined method should be consistently higher than that of any of the individual methods. An obvious way to do that is to concatenate histograms i.e. combine the colour feature vector with the SIFT feature vector. This approach is similar to the histogram concatenation technique discussed in this paper [16].

Another possibility is to use colour SIFT. Usually SIFT descriptors are extracted from a grey-scale images, i.e. computed over the intensity channel. Another approach is to extract the feature over all three HSV or RGB channels. This gives 384 dimensions per feature, 128 for each channel. This increases the computational cost of the classification and is not recommended unless there is an obvious improvement of the accuracy. Other implementations are *rgbSIFT*, *opponentSIFT* etc. and experiments with all of them were conducted. They are similar in implementation and more specific information is provided in Section 2.4. These extensions of the SIFT algorithm provide some additional information for the keypoints and some of them outperform the greyscale SIFT on categorical classification problems [31]. This result is investigated for the butterfly identification problem.

Finally, the most sophisticated method involves a multi-kernel Support Vector Machine [32], where each kernel represents a separate class of features. In this case each kernel is trained with different set of parameters, which in theory means that the best performance of all features would be combined together. However, due to lack of time, this method was not implemented and could be a topic of further research.

2.3 Classification

The Support Vector Machine classifier [12] has been used in a number of similar classification projects and is often regarded as a state-of-the-art classifier [4]. A simpler algorithm that could be used as a baseline is the k-Nearest Neighbour algorithm. It works quite well if there is enough training data, as it bases the classification outcome on a majority vote from the k training set points which are closest to the testing sample based on some distance measurement. The fact that the data is mapped to lower dimensions using one of the proposed feature representation means that distance based methods, such as this one, are viable options.

The advantages of KNN were that it was very easy to implement and experiment with. This is because of the lack of a training stage. Furthermore, from a theoretical point of view, the error rate of a KNN classier tends to the Bayes optimal as the sample size tends to infinity [42]. On the other hand, acquiring a large amount of dense data is extremely difficult. Finding a good distance measure is not a trivial task either and the computationally expensive work in the KNN classifier is done during the testing stage which would not scale well. If the dataset is large the program might be slow to respond to classification queries. Hence, a balance must be found between accuracy and performance. However, combined with the colour features, it was a good option for a simpler classifier to be initially implemented and used as a benchmark for more sophisticated setups.

A very important part of the development of a Nearest Neighbour classifier is the choice of a distance measure. Euclidean distance does not always make sense and other measure must be used. A popular choice for high-dimensional data is the Chi-squared metric (χ^2) [41]. It is used to compute distance between different feature vectors (observation). It is a weighted Euclidean distance in which each terms has weight inversely proportional to its average frequency. The Chi-squared distance between two observations given as $x = \{x_1 x_2 ... x_j\}$ and $y = \{y_1 y_2 ... y_j\}$ is defined by:

$$\chi_{x,y} = \sqrt{\sum_{j=1}^{J} \frac{1}{c_j}(x_j - y_j)^2} \tag{2.1}$$

where c_j is the average proportion of the j^{th} feature in the whole dataset.

It has been experimentally proven to work very well in similar problems [34]. We have investigated this hypothesis and concluded that the Chi-squared metric is indeed better suited for the butterfly identification problem than the Euclidean distance. Further discussion of those experiments is presented in the Evaluation Chapter 3 of this paper.

After the analysis of the result from this experiment a more sophisticated system,

using a multi-class Support Vector Machine was developed. The original SVM classifier is a binary classifier which tries to find the optimal boundary between two classes, based on boundary points, called support vectors. The algorithm tries to maximize the margin between the boundary hyperplane and the support vectors, which leads to a similar boundary as in Figure 2.3 if the datasets are linearly separable. If the data is not linearly separable, kernels can be employed

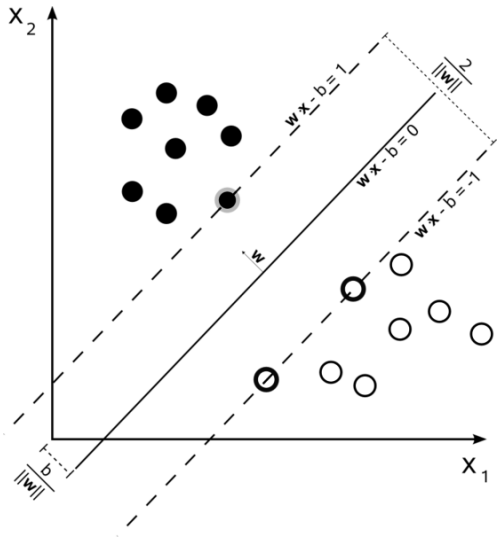


Figure 2.3: SVM decision boundary

to map the data to a different feature space, where the data points are separable.

The Support Vector Machine is a binary classifier and so the butterfly classification problem had to be reduced to a set of binary problems. There are several ways to implement a multi-class SVM classifier and in practise they achieve comparable results [9]. Two common techniques are a 1-to-1 pairwise comparison and a 1-against-all approach, which involves the construction of as many models as classes and then iteratively checks over all of them. The latter method has the 'winner takes it all' strategy, and the positive class is the final output, whereas the first method employs majority vote. The latter strategy was chosen for the work here and implementation details are discussed in Section 2.4. The only method which outperforms the one-against-all SVM in practice, is the SVM with coupling using Platt's probabilities [19]. Due to time constraints, it was not integrated in the system, but is discussed in Chapter 6 as a topic for future work.

2.4 Implementation

In this section the topics discussed are the implementation of the segmentation, feature extraction and representation, classification and evaluation. All of the work was done in MATLAB. The implementation of the different algorithms is a combination of my own work, open source libraries, built-in MATLAB functions and code from open-source on-line tutorials. Details follow below.

2.4.1 Segmentation and data preparation

The first dataset (Leeds Butterfly Dataset) consists of annotated images without any segmentation information. So the Interactive MATLAB segmentation tool developed by the Visual Group, University of Oxford [13] was used to segment the butterflies from the background clutter. The foreground masks, i.e. binary images where the foreground pixels have a value 1 and background are 0, were saved. The work on this dataset involved manually going through each image and marking some area of the background and the foreground with a brush stroke. The program is then supposed to estimate a segmentation boundary and the segmentation mask can easily be saved after that. When the images were focused and with good resolution, this only took a couple of strokes, but low-quality images, or images with background similar to the butterflies required more detailed marking and thus took longer time. Overall, 300 images were manually segmented by me using this approach.

The second dataset (Ponce butterfly dataset) had segmentation masks available. These segmentation masks as well as the ones acquired using the process described above had to be combined with the original image, to deliver a coloured image with the butterfly visible and the background zeroed (as in Table 2.1). The images had to be cropped to the bounding box and resized to have a maximum dimensions 500x500 pixels. I implemented this process and applied it to all 1012 images in the dataset. The preprocessed images make up the final dataset used for all evaluation and training procedures. For each species a text file with the names of all images which belong to it, was compiled. These files are later used to load the image data.

2.4.2 Features

The features employed in this project are the colour and shape features and a feature combining both. Feature extraction implementations:

- **colour** – This involved reshaping an HxWx3 image into *(H*W)*x3 vector,

so that we have a vector of pixels without any information on where this pixels comes from with respect to the image (i.e. bag-of-words). Each pixels is still in the 3D RGB space. For the experiments in the HSV colour space, the image is first transformed to HSV using the build-in MATLAB function *rgb2hsv*, then transformed to the appropriate vector.

- **shape** – Here we required an implementation of the dense-SIFT method described in Section 2.2.3.2 . I have used the Image Classification Practical [33], where this method is employed. The Practical exercise used the software package *vlFeat*[1]. This is an open-source library, written in Matlab and C . In particular, the PHOW features [1] were extracted. They are a variant of the SIFT descriptors extracted on a dense grid and are computed using the *vl_phow* function, which is a wrapper around an image smoothing function (*vl_imsmooth*) and a dense-SIFT extractor (*vl_dsift*). The function accepts as arguments the step size and whether the gray-SIFT, hsv-, rgb- or opponent- SIFT should be used. The evaluation of different combined features discussed in Chapter 3 involved varying the arguments of this function.
- combined - the histogram concatenation is done in the training step, as it doesn't involve a different features extraction procedure. It is only combines the features described above.

2.4.3 Vocabulary and image representation

Vocabulary

The colour and shape vocabularies are developed using the typical bag-of-words approach discussed in Sections 2.2.1 and 2.2.3.2. I used the implementation of the algorithm available in Image Classification Practical [33], which is also similar to the tutorials provided for the *vlFeat* library. As that was originally only for the shape feature (utilizing the SIFT descriptors), I had to developed a similar procedure for the colour vocabulary,which involved the use of a different feature extraction and histogram representation methods. My implementation uses the *vlFeat* library again and in particular their *k-means* clustering implementation, the *kd-tree* building and vector quantization algorithms.

Image representation

The shape feature is represented with a 2x2 spatial histograms. To build this, I used the *vlFeat* tutorial and the Image Classification Practical mentioned above.

[1] http://www.vlfeat.org/

For the colour feature, I used simple histograms which did not preserve any spatial information. For that I employed the built-in MATLAB function to build histograms - *hist()*. The concatenated histograms representation was developed by me, by concatenating the histograms delivered from the above approaches in a single histogram.

The tasks of vocabulary building and the transformation of the images to the new feature space (defined by the new vocabulary) are relatively time consuming. Considering that sometimes the image representation would stay fixed, while different experiments with the classifiers are performed, it made no sense to learn the vocabulary and compute the image representation for each test. So the vocabulary building and the evaluation of classifiers was split in two separate scripts, as in the Image Classification Practical. The first script describes the *Preprocessing* phase an the second - the *Evaluation* of the algorithms. These are described in details below.

Preprocessing phase

In the first script the names of all files are read and a uniformly distributed among classes sample of images is taken. This sample is used to compute the new vocabularies – colour and shape. These new vocabularies are saved and can be used later if required, so they have to be recomputed only if there is a need for different feature representation or the vocabulary size needs to be modified. For my experiments, vocabularies with different setup – such as different vocabulary size for example - were saved with different names, not overwritten, so that experiments were reproducible later. After we compute or load the vocabulary, we need to transform each image using the codewords in the vocabulary – i.e. represent it as a frequency histogram of codewords. So for each class, the names of all images from that class are read from the predefined *.txt* files. Each of these images is loaded and its colour histogram and shape histogram are computed using the new vocabularies. This information is stored in a cache folder, so unless new images are added or the vocabulary has been changed, the data does not have to be recomputed. For this step I have used the code from [33] which uses vector quantization techniques from the *vlFeat* library. I have made some modification of the script, like the fact that both the shape and the colour histograms are extracted at the same time. The code can be easily extended to include even more features. Then at testing time, the user can choose which of the precomputed features to use in their evaluation. In our case for each class a matrix with the names, colour and shape histograms of all images of that class is stored and later loaded in the classification step. This concludes the preprocessing script.

2.4.4 Classifiers

SVM classifier

The MATLAB build-in function (*svmtrain*) combined with my implementation of one-against-all multi-class SVM was used in this project. For the multi-class SVM, I have looked into various implementations available on-line and I have used the typical approach with some changes which I thought were reasonable. Generally, the one-against-all N-class SVM is built by first computing different models M_c for all $c \in N$ classes. For class C the model is built by training an SVM on data, split into C and *not* C groups (which is a binary classification). In the testing stage, a testing sample is classified using each of the models. The best case is when a sample is classified as positive by only one model M_i, so the sample clearly belongs to class i. If the sample is classified as belonging to more than one class, the prediction output is determined randomly among the positive classes. In some of the implementations the first positive class was returned instead. However, this makes the the classification biased to the order of the classes so we wouldn't be able to comment on the misclassification rate on a per-class basis. Hence, the random approach was chosen. The last possible case is when a sample cannot be classified as any of the classes. Some of the implementations would simply return the last class label, and in this way artificially increase the misclassification rate of the last class. This was easy to notice in the Confusion matrices. This is not desirable because often images which could not be classified as any of the classes could be of interest to the developer or user, so they have to be marked in some way. So, I decided to assign *N+1* class label to such images, where N is the number of classes in the dataset. In this way, they could be easily extracted from the classification results and analysed. Moreover, when using MATLAB tools for classifier evaluation, these labels count as an inconclusive results, so it was easy to keep track of the amount of such images in the dataset.

KNN classifier

The base of our KNN classifier is the build-in MATLAB classifier - *knnclassify*. However, in the MATLAB distribution that was used in this thesis (R2011a), *knnclassify* function didn't accept a custom distance function and the choices provided did not include χ^2 function. Thus, the *knnclassify* function was modified by me to accept as an argument a custom function. Fortunately, this functionality was supported by the more low-level MATLAB functions, so this was the only function that required modification. After that, I used a chi-squared distance implementation from Piotrś Image and Video Matlab Toolbox [7], which is available freely online and complements the MATLAB image processing toolbox. The modified *knnclassify* function accepts as an argument whether the χ^2 or the

default Euclidean distance metric should be used. This was one of the parameters varied in the evaluation step.

2.4.5 Evaluation

To run the experiments with the two classifiers, two similar scripts were developed. In them, the data prepared in the preprocessing step is loaded and read. Then based on several parameters, the right setup for the experiment is chosen – e.g. whether the shape, colour or both histograms have to be loaded; which kNN distances measure would be used etc.. To ensure valid results, all experiments are performed using the built-in MATLAB tools for cross-validation and classifier performance evaluation – *crossvalind* and *classperf*. With these, 10-fold cross-validation is performed and the average accuracy and inconclusive rate is printed on the screen. Further information used for some of the experiments is the *Confusion Matrix* or *Error Distribution Per Class*. This information is provided as properties of the *classperf* class. The cross-validation process is repeated 5 times, with the accuracy and the inconclusive rate (in case of SVM) returned after each iteration. From these results, the final scores and deviations are computed and reported in Chapter 3.

The code was implemented in such a way to ensure that no out-of-memory exception would happen, despite the large amount of data. So at each step only the necessary data is kept, all other memory is freed. Furthermore, because the preprocessing (vocabulary and data representation) step is split from the classification step, testing different parts of the system was relatively easy and consumed less time, than if that was not the case. Also the fact that the preprocessing data was cached, meant that experiments were reproducible later, without recomputing of vocabularies or histograms. All this allowed a large amount of experiments to be run in a limited time and ensured optimal performance of the methods tested.

3. Evaluation

3.1 Features

All features are represented using visual words. The set of visual words, or *codewords*, is the visual vocabulary of a particular feature. It is constructed by extracting the descriptors of a particular feature from all images (or a smaller sample if there is too much data) and clustering those descriptors using k-means clustering. The resulting cluster centres are the codewords which make up the vocabulary. An image is transformed to the newly defined feature space by replacing each descriptor with its closest cluster centre. Given a set of cluster centres $\{w_1 w_2 ... w_n\}$, each image is represented as a n-dimensional normalized frequency histogram of the cluster centres in the image. The number of visual words for each feature is optimized through cross-validation.

3.1.1 Colour

Colour is an important feature of the butterfly as it can be used to narrow down the number of possible classes. However, it cannot be used as a sole feature for classification. There are several reasons for that, the most crucial one being the fact that there are multiple pairs of species which share the same colour patterns (see Figure 3.1).

Furthermore, object classification based on colour is often problematic, due to the fact that there might be a significant variation in the captured colour of an object in different images caused by illumination changes. Since most of the butterfly images are taken outside, natural lighting changes should not cause the performance of the classifier to deteriorate, or at least we need to aim to minimize such cases. A way to partially overcome a variation in the performance is to use the Hue-Saturation-Value colour space instead of RGB, as the HSV is less sensitive to illumination changes. Later in this section the performance of the classifiers using RGB and HSV is compared. Interestingly, the usage of HSV does not result in a significant improvement on the datasets we worked with.

The implementation here employs the bag-of-words approach to feature representation. This method is invariant to translations and rotations, as discussed in section 2.2.1, which is necessary as butterflies might be photographed in many different positions. However, in this way all spatial information regarding the relative position of the colours to one another is lost.

(a) Admiral (b) Peacock (c) Painted Lady

Figure 3.1: Admiral, Painted Lady and Peacock are some of the classes which have similarly coloured wings.

The number of visual words is optimized with 10-fold cross-validation. Figure 3.2 shows how the classification performance varies with the number of words used. The results plotted were achieved with SVM and K-Nearest Neighbour classifiers. The experiment with the K-Nearest-Neighbour was run using Euclidean distance and χ^2 distance in the RGB colour space and the results from both are plotted separately. It can be observed that the number of words has little significance on the performance of the classifiers. The optimal result for the SVM classifier was achieved using between 1200 and 2000 words. There is a slight improvement of the accuracy when increasing the number of words from 400 to 1200 and after that the results achieved are very close to each other. An interesting observation is the fact that the inconclusive rate also increases with the increase in the number of words. The highest recognition rate, 63.5%±1%, with the lowest inconclusive rate –18.4%±0.6% is obtained when the colour vocabulary consists of 1200 codewords. This is determined to be the optimal setup, as it delivers the largest percentage of correctly identified images. The KNN classifier achieved optimal performance with 200 words using Euclidean distance ($52\% \pm 0.8\%$) and with 200-800 words ($65\% \pm 1.5\%$) when χ^2 distance was preferred. The size of the vocabulary has a smaller effect on the performance of the KNN classifier compared to that of the SVM. The difference between the results with the different setups is 3% and 2% for Euclidean and χ^2 distance respectively. Despite the small variation, it is interesting that the accuracy rate decreases with the increase of the vocabulary size, in contrast to the SVM method, where larger vocabulary ensures better performance.

Although HSV is often preferred in object recognition projects because of its illumination invariance, it did not improve the overall performance in our model. The results are not plotter because they were almost identical to the results obtained in the RGB colour space. All tests were performed with a Nearest

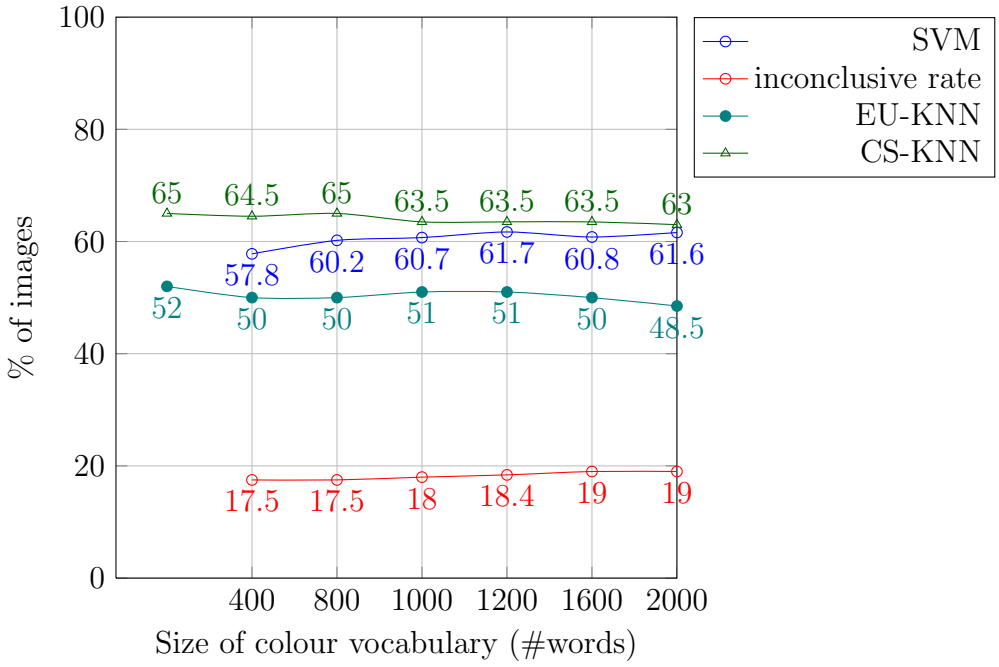


Figure 3.2: Correct rate of the SVM, KNN with Euclidean distance(EU-KNN) and KNN with χ^2(CS-KNN) with varying number of codewords used to represent the colour feature in RGB space. The inconclusive rate corresponds to the percentage of inconclusive predictions from the SVM and should be taken into account when calculating the error rate. The results are averaged over 5 runs of 10-fold cross-validation. The variance in the results is negligible so is not plotted.

Neighbour classifier using χ^2 distance as it was obviously better suited for the purpose of classification using the colour feature compared to Euclidean. With 1000 visual words extracted from HSV images, the kNN classifier achieved 63% accuracy compared to $63.5\% \pm 0.5\%$ for RGB. For Support Vector Machine the accuracy was slightly less as well ($59\% \pm 1\%$), but the HSV colour space led to a 2% increase in the inconclusive rate (20%). Although the average correct rate was practically the same with RGB and HSV, some difference might be observed in the recognition and confusion rates per class.

Table 3.1 shows the confusion matrix after 10-fold cross validation, using Nearest Neighbour classifier with 1000 RGB words and table 3.2 shows the confusion matrix achieved by the same setup except the fact that the visual words were extracted from images in the HSV colour space. The columns of the confusion matrix give the true classes and the rows the predicted ones. Hence, the true positive, or the correctly classified samples, are given in the diagonal of the matrix. The confusion matrices are averaged over 5 runs to ensure stability of the results.

	Admiral	Machaon	Peacock	Swallowtail	Zebra	Cabbage White	Buckeye	Mourning Cloak	Painted Lady	Longwing
Admiral	70	2	14	10	9	0	1	19	2	17
Machaon	0	26	1	0	4	0	0	2	3	0
Peacock	10	0	35	2	1	0	5	1	4	1
Swallowtail	8	4	1	56	36	0	1	5	0	0
Zebra	6	3	0	33	54	0	0	5	0	6
Cabbage White	0	0	0	0	0	55	0	0	0	0
Buckeye	4	0	10	1	0	0	73	2	21	0
Mourning Cloak	3	3	1	5	4	0	1	63	1	1
Painted Lady	1	8	16	0	0	0	8	2	52	1
Longwing	7	0	2	2	4	0	0	0	0	34

Table 3.1: Confusion matrix of NN classifier with 1000-word vocabulary in **RGB** colouspace.

	Admiral	Machaon	Peacock	Swallowtail	Zebra	Cabbage White	Buckeye	Mourning Cloak	Painted Lady	Longwing
Admiral	79	4	7	5	9	4	4	13	7	11
Machaon	2	24	2	8	2	0	0	0	0	2
Peacock	5	0	56	2	0	1	1	1	2	1
Swallowtail	1	10	0	65	35	0	4	1	0	6
Zebra	0	3	0	19	45	0	1	2	0	5
Cabbage White	1	0	0	0	0	50	0	0	0	0
Buckeye	5	1	2	4	2	0	61	1	22	3
Mourning Cloak	6	0	5	4	8	0	0	74	2	7
Painted Lady	6	2	7	0	0	0	18	3	50	1
Longwing	4	2	1	2	11	0	0	4	0	24

Table 3.2: Confusion matrix of NN classifier with 1000-word vocabulary in **HSV** colouspace.

From the matrices it can be observed that the HSV method performs consistently better that the RGB method on the Peacock class (increase from 44% to 70% on average). This is a significant improvement in the performance. Furthermore, the Peacock is almost never confused with the Buckeye (figure 3.3), and less so with Admiral and Painted Lady. The three of them made up for most of the misclassifications when using RGB images. In the HSV the Peacock images are still most often misclassified as Painted Lady (10% misclassifications), which is quite reasonable given the similar colour shade and eyespots on the wings. However, Peacock images are no longer misclassified as Buckeye, but as Mourning Cloak, even more often than as Admiral. This is an interesting difference between the two models.

Also an interesting variation can be observed in the incorrect predictions for the Admiral class when we compare the models in the HSV and RGB colour space. In the RGB space Admiral samples are most often mislabelled as Longwing, Peacock and Swallowtail (Figure 3.5). Peacock seems like a reasonable misclassification based on colour, whereas Swallowtail and Longwing might initially look very different from the Admiral. However, as this classification is based only on colour, one might understand how on different lighting the body of an admiral butterfly might look quite dark. When this is combined that with the white spots (which are sometimes yellowish), the colours observed become quite similar to the colour pallet of the Swallowtail and Longwing species.

Interestingly, the top misclassifications of the Admiral class are almost completely

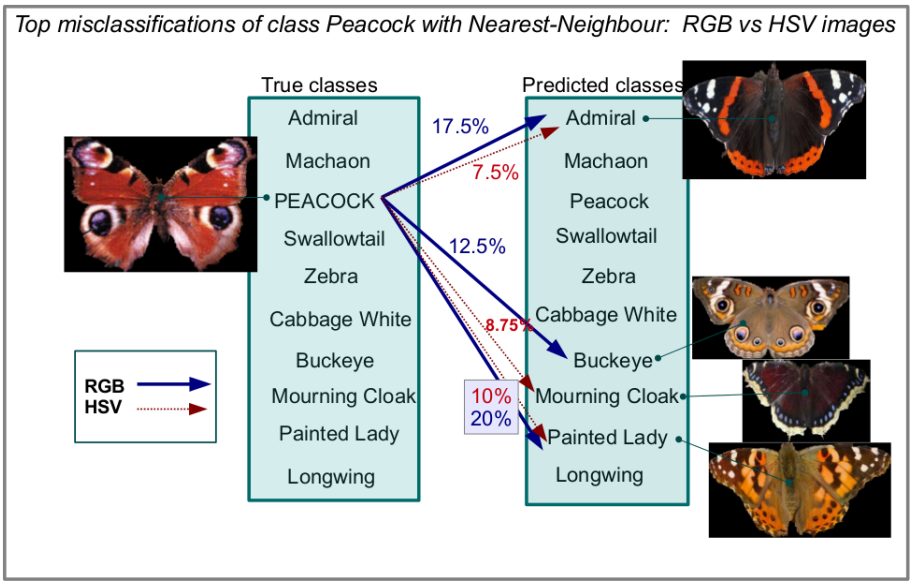


Figure 3.3: The solid line shows which are the top classes as which images from the Peacock class are mislabelled if colour feature is in RGB space and the red dotted line demonstrates the results for the HSV colour space.

different when the colour feature is extracted from HSV images (Figure 3.4). The Peacock is the only label present in the top incorrect classes of both models. In the HSV model, the Admiral is more often mislabelled as Painted Lady and Buckeye, which share some of the characteristic brown patches with the Admiral species. However, because HSV is not so sensitive to illumination, the fact that the body of an Admiral is darker shade of brown compared to that of the Buckeye and Painted Lady was probably ignored by the classifier. These observations lead to the conclusion, that although HSV is generally better performing when the images are taken in outdoor settings, the difference between some of the classes are too subtle and even small changes in the intensity of the colour should not be ignored. Furthermore, although χ^2 distance tries to estimate the weight of each feature based on its frequency, it still does not guarantee that the most discriminative features would have the most weight.Hence, common features(colours, shapes, patterns) which occur often but do not contribute to the class separability might have a negative effect on the predictions. This problem will be discussed further in the Discussion Section.

In fact, most of these variations are no longer present if instead of Nearest-Neighbour, we choose Support Vector Machine for classification. Below are shown the confusion matrices of the SVM classifier used with the HSV colour feature (table 3.3) and the RGB feature (table 3.4). Most of the correct and incorrect classifications are the same. The only bigger variance is in the identification of the Longwing class, which is mislabelled as Zebra 5 times when using HSV

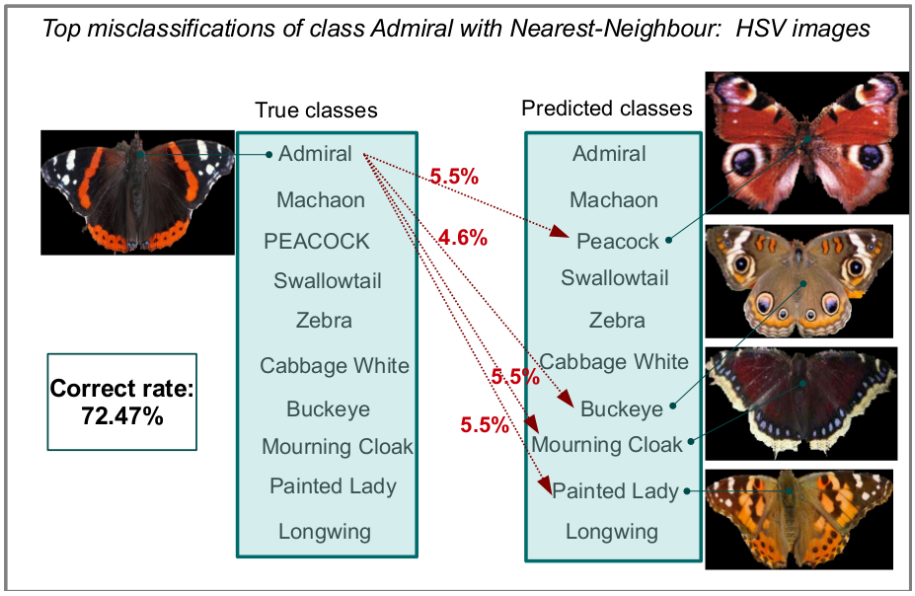


Figure 3.4: The arrows demonstrate which are the top classes as which images from the Admiral class are mislabelled if colour feature is in HSV colour space.

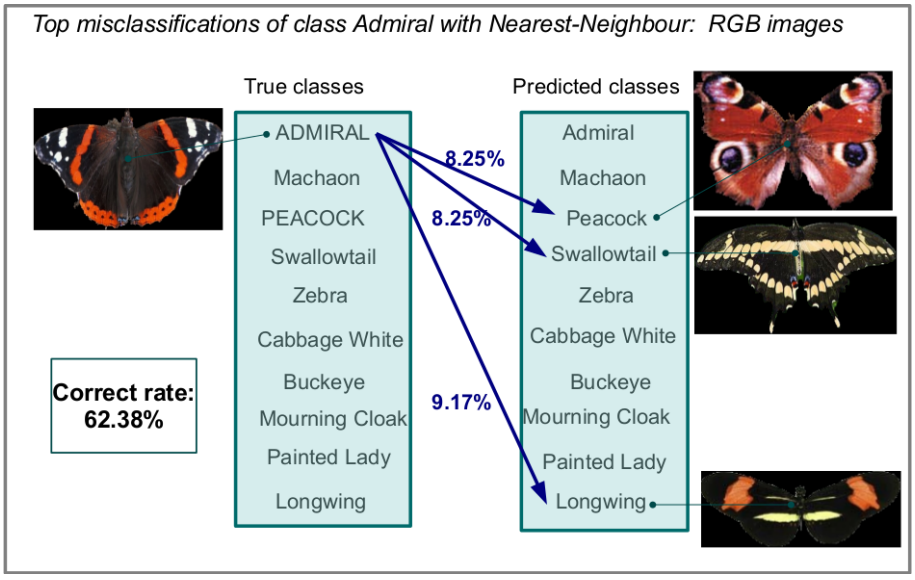


Figure 3.5: The arrows demonstrate which are the top classes as which images from the Admiral class are mislabelled if colour feature is in **RGB** colour space.

	Admiral	Machaon	Peacock	Swallowtail	Zebra	Cabbage White	Buckeye	Mourning Cloak	Painted Lady	Longwing
Admiral	66	1	1	3	7	3	3	2	6	5
Machaon	1	20	1	3	2	0	0	1	0	0
Peacock	4	0	55	1	0	1	3	3	0	0
Swallowtail	2	8	0	52	18	0	2	1	0	3
Zebra	0	2	1	17	58	0	2	0	1	5
Cabbage White	0	0	0	0	0	46	0	0	0	0
Buckeye	1	1	1	4	1	0	60	0	11	1
Mourning Cloak	2	2	2	0	0	0	0	78	1	0
Painted Lady	2	1	2	0	0	0	6	1	51	3
Longwing	1	0	0	2	6	2	0	1	0	24
inconclusive rate	30	11	17	27	20	3	13	12	13	19

Table 3.3: Confusion matrix of SVM classifier with 1000-word vocabulary in **HSV** colour space. Average accuracy is $59.2\% \pm 1\%$ and 20% inconclusive rate.

feature, compared to 1 in the case of RGB. However, the overall performance of the SVM classifier with HSV feature on the Longwing class is a lot worse (40%), compared to its performance when used with RGB colour features (66.7%), so we could presume that the HSV model is not very well suited for identification of that particular species. From these results we can conclude that SVM classifier is not as sensitive to small changes in the feature space as kNN.

On average the RGB model performs just as well as the HSV model despite the fact that most of the images are taken in uncontrolled environment during different times of day and/or lighting conditions. Also the misclassifications above prove that a classification based on only colour would not be successful regardless of the method of colour extraction, as this might cause butterflies which are visually very different to be classified as the same species, only because they have several colours in common.

	Admiral	Machaon	Peacock	Swallowtail	Zebra	Cabbage White	Buckeye	Mourning Cloak	Painted Lady	Longwing
Admiral	67	1	6	4	3	1	3	8	4	6
Machaon	0	21	0	4	1	0	2	0	1	0
Peacock	2	1	43	0	2	1	5	0	2	0
Swallowtail	2	10	0	51	22	3	1	4	0	1
Zebra	0	3	0	11	62	0	0	1	0	1
Cabbage White	0	0	0	0	0	47	0	0	0	0
Buckeye	0	0	7	0	0	0	56	0	10	0
Mourning Cloak	4	0	0	1	2	1	1	75	4	0
Painted Lady	2	1	6	2	2	1	9	2	43	0
Longwing	3	0	0	2	3	1	0	0	0	40
inconclusive rate	29	9	18	34	15	0	12	9	19	12

Table 3.4: Confusion matrix of SVM classifier with 1000-word vocabulary in **RGB** colour space. Average accuracy is $60.7\% \pm 2\%$ and 18% inconclusive rate.

3.1.2 Shape and texture

From the experiments performed initially it became obvious that the colour feature was insufficient for good performance and that it was not discriminative enough. The shapes and the patterns of the butterfly wing were an obvious feature to use. Although there are patterns which are common among more than

one species, the combination of shapes is often enough to classify a sample. Furthermore, the shapes of the wings or any particular spikes like in the Swallowtail class are very distinctive and are characteristic to particular species (see Table 2.3). Both texture of the wing and shape can be represented using scale- and translation-invariant SIFT features. The process of feature extraction is similar to the colour feature. Here the visual words are circular patches with a fixed size, which are described using SIFT descriptors. The vocabulary is estimated using k-means clustering and its size is optimized using cross-validation. The size of the patches and the vertical and horizontal distance between patches on the images (the step of the grid) are also optimized using 10-fold cross validation. Figure 3.6 shows the results from vocabulary and Firgure 3.7 for step size optimization.

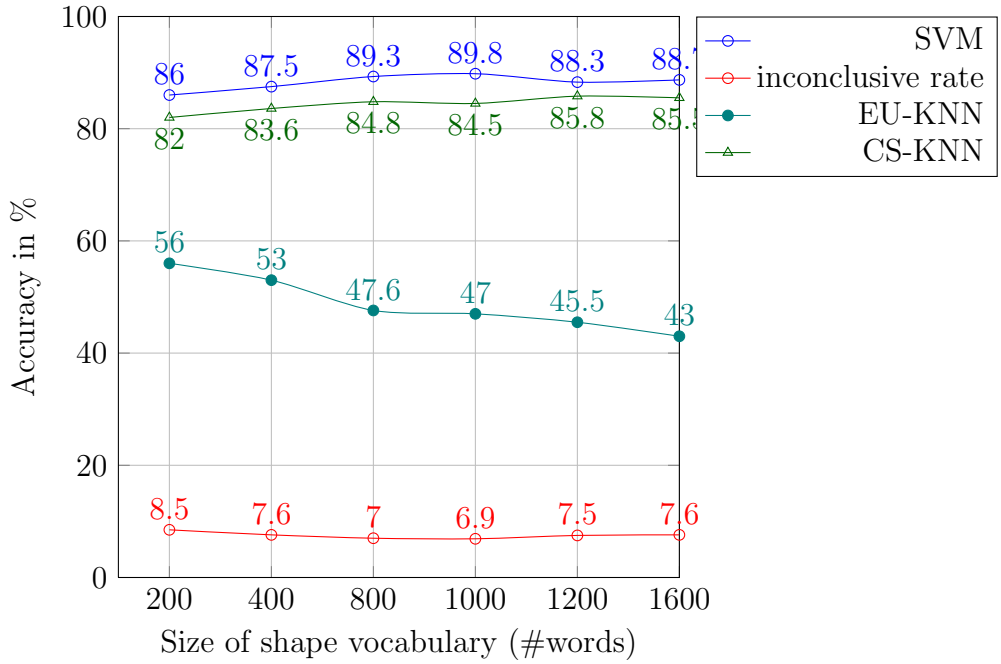


Figure 3.6: The accuracy of the SVM, KNN with Euclidean distance(EU-KNN) and KNN with χ^2(CS-KNN) with varying number of codewords used to represent the shape/texture feature. The inconclusive rate corresponds to the percentage of inconclusive predictions from the SVM. The results are averaged over 5 runs of 10-fold cross-validation.

Again the size of the vocabulary has little significance on the performance of the SVM classifier as with the colour feature. The inconclusive rate is also relatively stable over multiple tests with 200-1600 words. The optimal results were achieved with 1000 words. Although the difference in the accuracy achieved by the top- and worst performing setup is small (3.8%), it should be noted that the increase in the performance is observable when the vocabulary is increased to at least 800

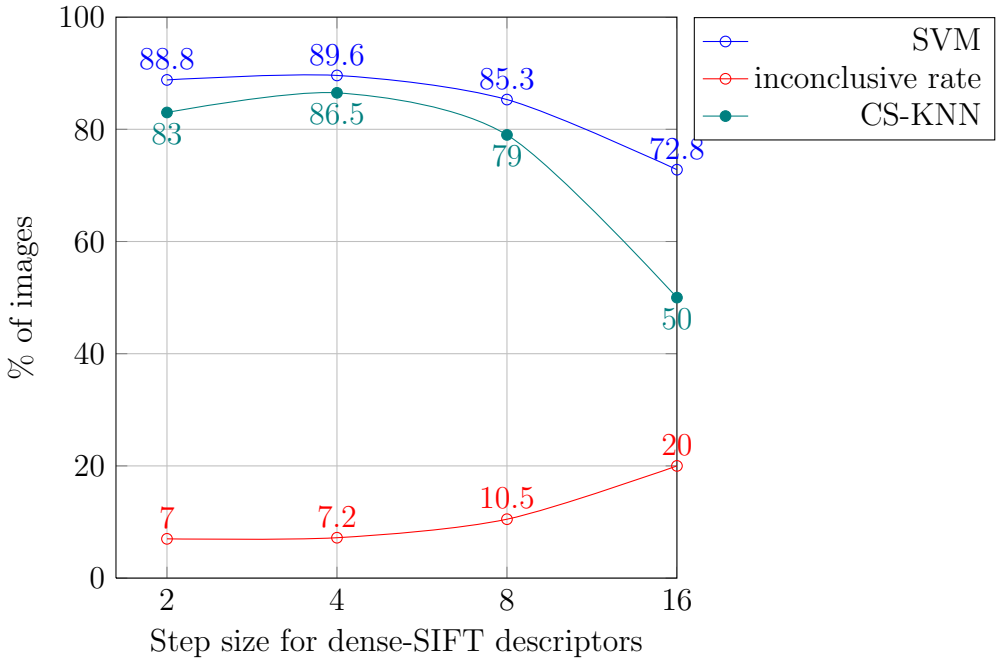


Figure 3.7: Accuracy of the SVM and KNN with χ^2(CS-KNN) with varying size of step used in the dense grid extraction of visual codewords for the shape/texture feature. The inconclusive rate corresponds to the percentage of inconclusive predictions from the SVM. The results are averaged over 5 runs of 10-fold cross-validation.

words. With the optimal setup (1000 words, step size of 4 pixels) the Support Vector Machine classifier achieves $89.8\% \pm 0.5\%$ accuracy and 6.9% inconclusive rate on average over 5 runs of 10-fold cross-validation. From the accuracy rates per class it is clear that the Shape/Texture feature outperforms the colour feature in every case (compare Figure 3.8 and Figure 3.9).

As far as the Nearest-Neighbour classification goes, the performance of the classifier with Euclidean distance metric is quite low. In fact, it performs worse that the same classifier when using the colour feature and the accuracy deteriorates with the increase of the vocabulary and decreases by more than 10% from 200 to 1600 words. The fact that Euclidean distance is not suitable for the kind of high-dimensional problem where not all dimensions (codewords) are equally relevant is more than obvious. With larger vocabulary there are more irrelevant visual words for each class, which makes it even harder for the classifier to compute similarity between samples. On the other hand, the Nearest-Neighbour classifier performs extremely well when combined with χ^2 distance measure as was the case with the colour feature. Moreover, whereas in the case of the colour feature, the difference between the performances of the NN classifier with the two distance metrics was $\approx 10\%$, here χ^2 proves to be far superior to the Euclidean metric.

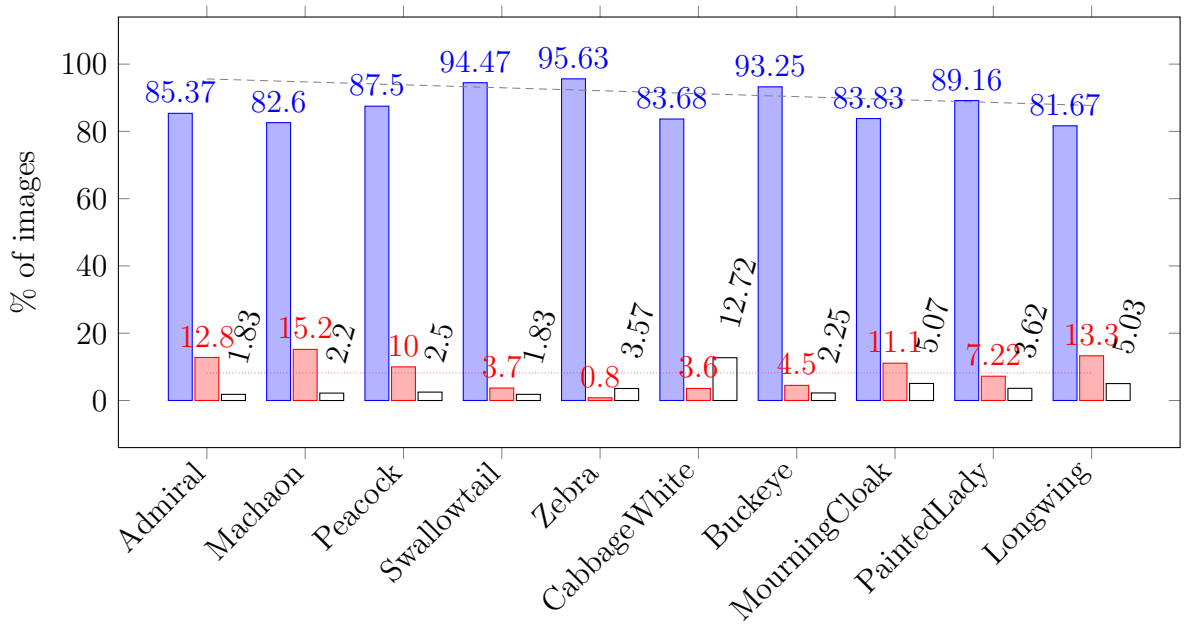


Figure 3.8: Per-class performance of SVM with the shape feature. Plotted are the correct rate, the inconclusive rate and the error rate for each class. The red dotted line represents average inconclusive rate (8.2%) and the blue dashed line - the average correct rate (87.1%).

The accuracy achieved with it is almost twice that of the Euclidean-based NN ($85.6\% \pm 0.5\%$ and $43\% \pm 0.5\%$ for χ^2 and Euclidean respectively). Even with the optimal setup for the Euclidean model (200 visual words) it still makes $\approx 30\%$ less correct predictions.

The size of the grid step has a more significant role in the correct rate of both KNN and SVM classifiers (Figure 3.7) compared to the vocabulary size. The optimal result is achieved with step size of 4 pixels. Obviously bigger size leads to a drop in the correct rate, more drastically for the KNN classifier. Although the decrease in the error rate of the SVM is not that large (3% drop when switching from 4-pixel step to 16-pixel), the correct rate decreases with 16% and the inconclusive rate has dramatically increased as well, reaching 20% for step size = 16. This is a significant amount of inconclusive decisions and leads to the conclusion that not enough information is provided to the classifier to make confident decision. This makes sense, given the fact that butterflies are relatively small in size and subtle patterns located close to each other are usually what allows differentiation between species.

All the above data is from experiments performed on grey-SIFT, that is when the SIFT descriptors are extracted from the grey-scale version of the image. Although this setup performs extremely well, there is still room for improvement,

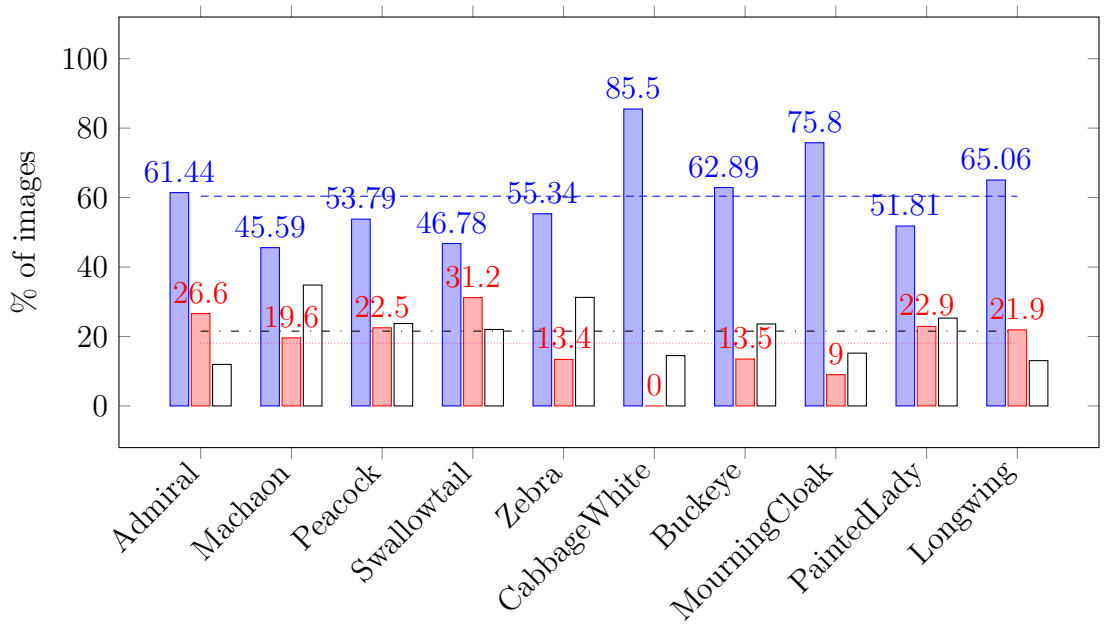


Figure 3.9: Per-class performance of SVM with the colour feature. Plotted are the correct rate, the inconclusive rate and the error rate. The average correct, inconclusive and error rates are represented by the blue dashed, the red dotted and the black dashdotted lines respectively. The values of the average rates are 60.4%,18.1% and 20.5%.

particularly in the rate of inconclusive decisions of the SVM classifier and of the overall accuracy of the KNN classifier. Hence, it was interesting to test whether any of the colour-SIFT methods discussed in Section 2.2.3.3 can improve the performance. The results are given in the section which follows.

3.1.3 Combined features

The main goal of feature combination is to achieve superior performance compared to that of the individual features. Furthermore, it is interesting to observe whether an improvement occurs on classes for which the confusion matrices of different methods don't overlap. As explained in Section 2.2.3.3, there are several way in which features can be combined. One way to utilized both the colour and the pattern characteristics of a butterfly, is to use the colour-SIFT descriptors to represent circular image patches (the shape codewords). The SIFT descriptors are usually extracted from gray-scale images [23]. However this algorithm was extended to colour-SIFT [2], where the descriptors are extracted from the three different colour channels of an image independently (red, green and blue; hue, saturation and value etc.) and stacked in one vector. Hence, a circular image patch

would be described with a 384-dimensional vector (128 for each channel). In this project, experiments with the *vlFeat* implementations of colour-SIFT were performed, and the descriptors tested were rgbSIFT, hsvSIFT and opponentSIFT. The opponentSIFT is similar to the HSV and RGB one, except for the fact that the images are transformed from the RGB space to the Opponent space using the transformation defined by equation 3.1.

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} \frac{R-G}{\sqrt{2}} \\ \frac{R+G-2B}{\sqrt{6}} \\ \frac{R+G+B}{\sqrt{3}} \end{pmatrix} \tag{3.1}$$

Experiments with the three SIFT variants were performed using SVM and KNN and compared against the results from the gray-SIFT descriptors. All experiments were run with shape vocabulary with 1000 words, as this was determined to be the optimal setup from the evaluation of the shape vocabulary.

	SIFT	rgb-SIFT	opponent-SIFT	hsv-SIFT
SVM	$89.8\% \pm 0.5\%$ (6.9%)	$90.2\% \pm 0.7\%$ (7%)	$88\% \pm 1\%$ (7.8%)	$88.5\% \pm 0.5\%$ (8.3%)
KNN	$86.5\% \pm 0.5\%$	$87.1 \pm 0.5\%$	$84\% \pm 0.5\%$	$81.5 \pm 0.6\%$

Table 3.5: Correct rate of SVM and χ^2 KNN averaged over 5 runs of 10-fold cross-validation with 1000-word shape vocabulary and varying SIFT descriptors. The inconclusive rate of the SVM is given in brackets and should be taken into account when the error rate is computed. The optimal performance for each classifier is highlighted.

From Table 3.5 it can be seen that the choice of descriptors affects the performance of the classifiers, however not very significantly. When the SVM is used, the performance varies very little, and only *rgbSIFT* improves the performance of the gray-SIFT. The optimal performance is achieved with *rgbSIFT* and is $90.2\% \pm 0.7\%$ with $7\% \pm 0.5\%$ inconclusive rate. This is $\approx 0.5\%$ improvement compared to the original SIFT. The performance of both the SVM and the KNN classifier decreases with *opponentSIFT* and *hsvSIFT* descriptors respectively. The *rgbSIFT* improves slightly the performance of the KNN as was the case with SVM, but it is with less than 1%. As a whole an improvement can be observed if *rgbSIFT* descriptors are preferred, but it is not significant.

Another way of combining features, which was discussed in Section 2.2.3.3 is to concatenate the histograms for each feature into one combined histogram. This approach makes sense because the colour and the shape words are independent of each other, so in theory this should simply be a more descriptive representation. The results are shown in Table 3.10.

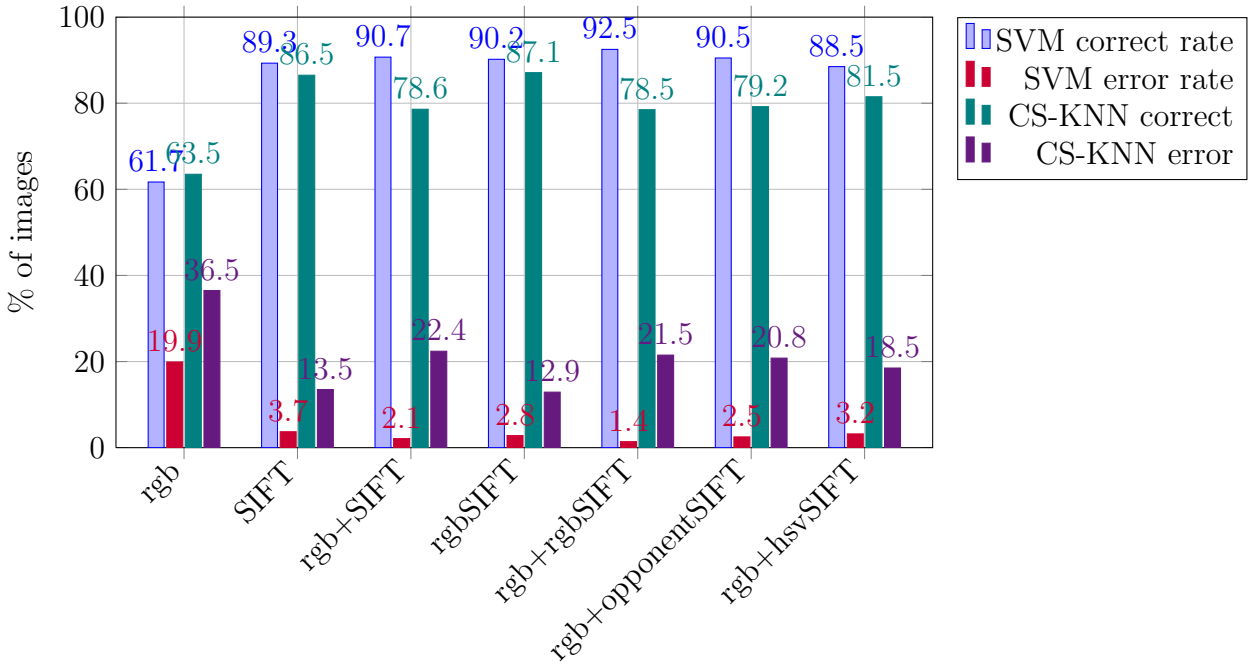


Figure 3.10: Performance of the SVM and KNN with χ^2(CS-KNN) with different descriptors used to represent the colour, the shape and the combined feature. The correct and the incorrect rates of both classifiers are shown. The inconclusive rate of the SVM is not plotted. The results are averaged over 5 runs of 10-fold cross-validation, the variance was neglibile so is not plotted. The colour vocabulary consists of 1200 codewords and the shape - of 1000 words.

The combined feature has a completely different effect on the two classifiers' accuracy rates. When the SVM classifier is used, the method of combining features through concatenated histograms is clearly successful. The histogram of concatenated features method outperforms the individual descriptors in terms of accuracy in every case except rgb+*hsvSIFT*. Also, the inconclusive rate of the SVM with the combined approach is lower. For example the performance of the rgb+SIFT feature histogram leads to a 1.4% better accuracy than the SIFT shape feature and to a 30% improvement when compared to the accuracy of SVM with the *rgb* colour feature. This is a clear proof of successful feature combination. The optimal result is achieved when the *rgb* colour feature is combined with the shape feature described with rgbSIFT method and is $92.5 \pm 0.5\%$ with $1.4\% \pm 0.2\%$ error rate. This is not surprising considering that the top performing descriptors for the shape feature are precisely the *rgbSIFT* descriptors ($90.2\% \pm 0.7\%$ with SVM) for shape and for colour–*rgb* . An interesting point to investigate and discuss in the next chapter would be whether the concatenated histogram approach improves the accuracy in classes for which the shape feature is inconclusive. Results on the per-class performance of the SVM with different

features is presented in Section 3.3 when the classifier is evaluated on an extended dataset (specifically Figure 3.13).

As far as the KNN classifier is concerned, this feature combination method causes significant decrease in the accuracy regardless of which features are combined. A possible explanation for that is the fact that the KNN cannot discriminate between different dimensions so well as the SVM. This would be discussed in more details in Chapter 4.

3.2 Classifiers

The two classifiers used in this thesis are a one-against-all Multi-class Support Vector Machine and a K-Nearest-Neighbour classifier. They are evaluated in this section.

3.2.1 K-Nearest-Neighbour

The simpler classifier, which was implemented initially, was a KNN classifier. It does not have a training phase and there are two main parameters which have to be optimized on a validation set. These are the distance metric and k – the number of nearest-neighbours used to determine the prediction outcome. The results from the experiments with varying k are in table 3.6. The data provided is obtained by averaging over 10 experiments with the shape features, using a fixed number of 1000 words and χ^2 distance measure. In this implementation the majority vote of the k neighbours is used to determine the classification outcome. Hence, each sample has the same weight in the decision regardless of its distance from the testing example.

# of neighbours	1	3	5	7	9
CS-KNN	86.4%	86.3%	84.6%	83.5%	83.5

Table 3.6: Average accuracy of χ^2 KNN averaged over 10 runs of 10-fold cross-validation with 1000-word shape vocabulary and varying number of k.

With the increase of k, the performance slowly starts to decline. Hence, the optimal performance is achieved using only the nearest point. Although the difference between the accuracy with k=1 and 3 is negligible, the decline is obvious with $k \geq 5$. A possible explanation for that is the fact that the differences between samples from different classes might occasionally be very subtle so they might lie relatively close to each other in the feature space. However, it must be noted that small number of neighbours results in classifiers sensitive to outliers and

noise. Because there is not much difference in the accuracy achieved with 1 and 3 nearest-neighbours, we can conclude that a Nearest-Neighbour classifier can be very suitable for such classification problems when combined with appropriate features.

The most important characteristic of the KNN is the distance measure used to compute similarity between points. It depends on the feature space and can have a significant effect on the performance of the classifier as confirmed from our experiments. Generally, if we have no knowledge of the feature space, the Euclidean distance measure is appropriate [34]. χ^2 distance measure discussed in Section 2.3 is found to be more successful with data with a lot of features, which is the case here. The results from the experiments show that χ^2 is indeed superior to the Euclidean distance, so far that it achieves double the accuracy of the default Euclidean metric in some case.

When the data is represented by the colour feature, the difference between CS-NN(χ^2-Nearest-Neighbour) and EU-NN (KNN with Euclidean distance) is not that large, however it is still significant (see Figure 3.2). χ^2 distance outperforms the Euclidean regardless of the number of words in the vocabulary. Both of them achieve lower average accuracy with larger vocabulary size. The optimal performance is with 200 words – $52\% \pm 0.8\%$ and $65\% \pm 1.5\%$ for EU-NN and CS-NN, although the performance of the CS-NN with 200-1000 words is relatively consistent. The difference between the accuracy rates is significant but still small compared to when the shape feature is used (c.f. Figure 3.6). CS-NN outperforms the EU-NN again with vocabulary of any size. The performance of EU-NN with the shape feature is quite similar to its performance with the colour feature. The optimal accuracy is only $56\% \pm 1\%$ and it declines with over 13% compared to 5% in the case of χ^2 with the increase of vocabulary size. The shape features are a lot more sophisticated and the fact that there is no significant improvement in the recognition rate confirms the unsuitability of the Euclidean metric for this feature space and task in general.

On the other hand, the performance of the CS-NN has significantly improved with the introduction of more discriminative features. Its optimal performance is achieved with 1200 shape words and is $85.8\% \pm 0.5\%$. This is a major improvement and leads to a performance slightly lower than that of the SVM classifier. Although the KNN outperforms SVM with colour features in terms of correct rate, its lowest error rate is 35%, compared to 20% for the SVM. With the introduction of the shape feature, the error rate of the χ^2-KNN decreases to 18% which is a significant improvement. Further comparison of CS-NN and SVM is made later in this chapter.

The fact that the performance of the NN classifier deteriorates with the increase of the vocabulary size regardless of the features used when the Euclidean distance is utilized is expected. This Euclidean classifier is not designed to work very well

in higher dimension as each dimension (visual word) is equally weighted and there is a lot of noise which does not improve the class separability. χ^2 on the other hand is a lot more stable. Its performance decreases with 2% and increases with 3% when using the colour and shape feature respectively.

This might be explained with the fact that it is a weighted Euclidean distance, so noise and irrelevant words should not affect the performance that much as long as the weights of each dimension (codeword) are appropriate. In the case of χ^2, the weight of a codeword is proportional to the inverse of the relative frequency of that codeword. Thus, we can conclude that there are a large number of visual words which occur often and do not contribute to the class separability. This in turn affects the performance of the Euclidean-based NN.

To sum up, the optimal setup of this classifier is to use the closest nearest neighbour and the chi-squared distance measure. With the appropriate vocabulary size, the NN classifier achieves $65\% \pm 1\%$ and over $85\% \pm 1\%$ accuracy with colour and shape feature respectively.

3.2.2 SVM

The SVM classifier provides one of the best overall performances in object classification, because of the fact that it determines points which lie near class boundaries and uses only them for the estimation of the decision boundary. With a soft boundary, it is robust to noise and the size of the datasets does not play such a major role as with the KNN. Furthermore, no assumptions about the feature space are required, as long as points are linearly separable. Even if they are not, they could be mapped to another feature space with a kernel transformation. It is also very successfully combined with bag-of-words representations, as the one that is employed here [42, 26]. All these features of SVM make it one of the top performing classifiers in object classification and an attractive choice for the butterfly identification task.

The only problem with the SVM is that it is a binary classifier and needs o be extended to a multi-class classifier. This was discussed in Chapter 2. The implementation results in the possibility of a sample not being classified as any of the classes. This is generally a good property, as it is an important feature of a classifier to be able to predict a class with some confidence and also to distinguish samples which are of unknown classes. It also provides us with another statistic which can be observed with the variation of vocabulary size and feature type inconclusive rate. This is the percentage of samples from the training set which were determined to be from an unknown class. The accuracy rate is then computed only on those samples for which a conclusive prediction is made. The results in Figure 3.11.

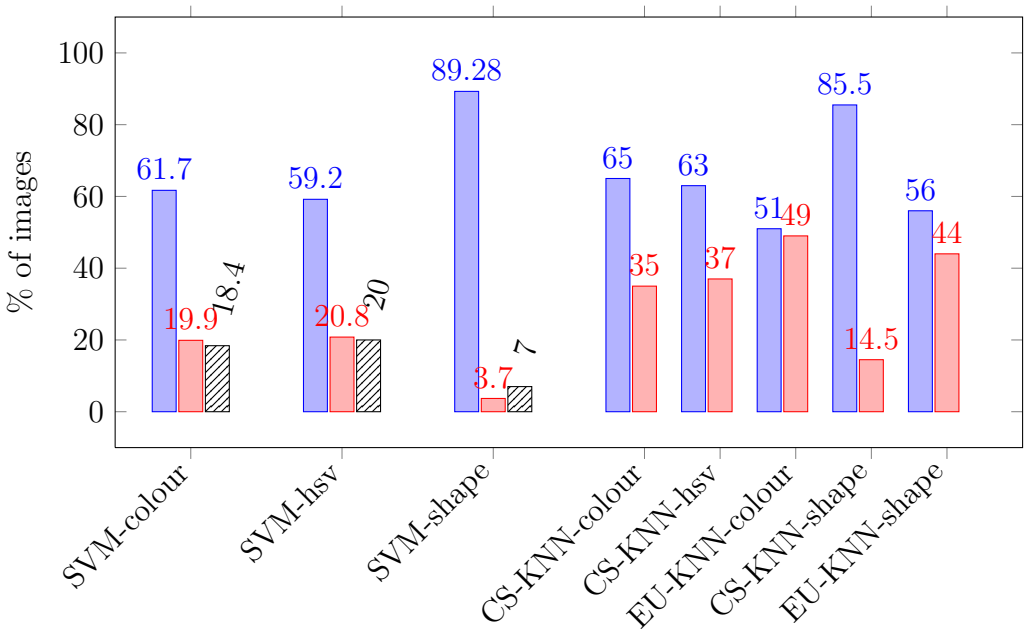


Figure 3.11: Average correct ( ), incorrect ( ) and inconclusive rate (where available) for different classifiers with different features. Results are given with the top performing setups for each of the classifiers and averaged over multiple runs and 10-fold cross-validation. Variance is negligible.

The SVM classifier outperforms the NN classifier in every experiment in terms of error rate. However, due to the high inconclusive rate the KNN has higher correct rate on the colour feature. The optimal performance with the colour feature is $61.7\% \pm 1\%$ and is achieved with 1200 words. Using the shape features the SVM achieves $90.7\% \pm 0.5\%$ accuracy which is a very good result. In both cases SVM achieves $\approx 10\%$ better error rate compared to CS-NN. However, in the case of the the colour features the inconclusive rate of the SVM is unacceptably high – it reaches $19\% \pm 1\%$ using RGB images and 20% for HSV. This means that $\frac{1}{5}$ of the images could not be classified. This leads to the CS-NN outperforming the SVM in terms of correct rate as obviously the colour feature does not provide enough information for conclusive decisions of the SVM classifier. Possible explanations for this are discussed in Chapter 4.

Despite this, the SVM classifier achieves a very high performance when utilizing the shape features $90.7\% \pm 0.5\%$ with an inconclusive rate of only 6.9% (out of 842 images over multiple runs of 10-fold cross validation).

An interesting observation regarding the inconclusive rates is that in the case of the colour features, the rate increases with the increase of the vocabulary size, whereas it decreases by a small bit, but is relatively stable, with the shape feature.

Classifier	Shape feature	Colour feature
SVM	$86.3\% \pm 0.5\%$ ($8\% \pm 0.2\%$)	$60.4\% \pm 0.5\%$ ($20\% \pm 0.5\%$)
CS-KNN	$86\% \pm 0.5\%$	$59\% \pm 0.5\%$

Table 3.7: Performance of SVM and χ^2-NN on the 13-class dataset with the rgb colour feature and SIFT shape feature with the optimal parameters. The values in brackets show the inconclusive rate of the SVM. All results are averaged over 5 runs of 10-fold cross-validation.

3.3 Extended dataset

The evaluation of the features and classifiers discussed above was all done using a 10-species dataset. It was interesting to see how the performance of the classifiers and features changed when this dataset was extended to include more classes of butterflies. For even better evaluation of whether the features are discriminative enough, the newly added classes have been selected so that they share similar patterns and colours with some of the species in the 10-class dataset.

From Table 3.7 it can be seen that there is a reduction in the correct rate of both classifiers when the dataset was extended. The optimal performance of the SVM with the shape feature is 90.2% on 10 classes, which is around 4% more than on 13 species. Also the inconclusive rate is 7% compared to 8% with the larger dataset. If we use the colour feature, the performance of the SVM on 13 and 10 classes is the same, with a slight increase (1.5%) in the inconclusive rate. In contrast, with the KNN the performance deteriorates when the colour feature is used for classification and remains almost the same when the shape feature is used. Considering that the shape feature provides superior performance, it makes sense to aim to stabilize its results, so that it can perform as well on larger datasets as on the 10-species dataset, or at least in some reasonable margin. Hence, the fact that the KNN's correct rate does not decrease when tested on more species with of the shape feature is promising. However, the variation in the performance of the SVM is a bit worrying, as even a slight decease in the performance would not scale well if a lot more classes are added and this was the top-performing classifier so far. This decline might be expected considering the similarity of the 3 new species to some of the already existing species in the smaller dataset.

Although the combined feature resulted in a rather small improvement, we have decided to evaluate its performance on the larger dataset to try and find a more stable feature set for the SVM classifier. Figure 3.12 shows a comparison between the performance of the SVM on the two datasets using the concatenated histograms.

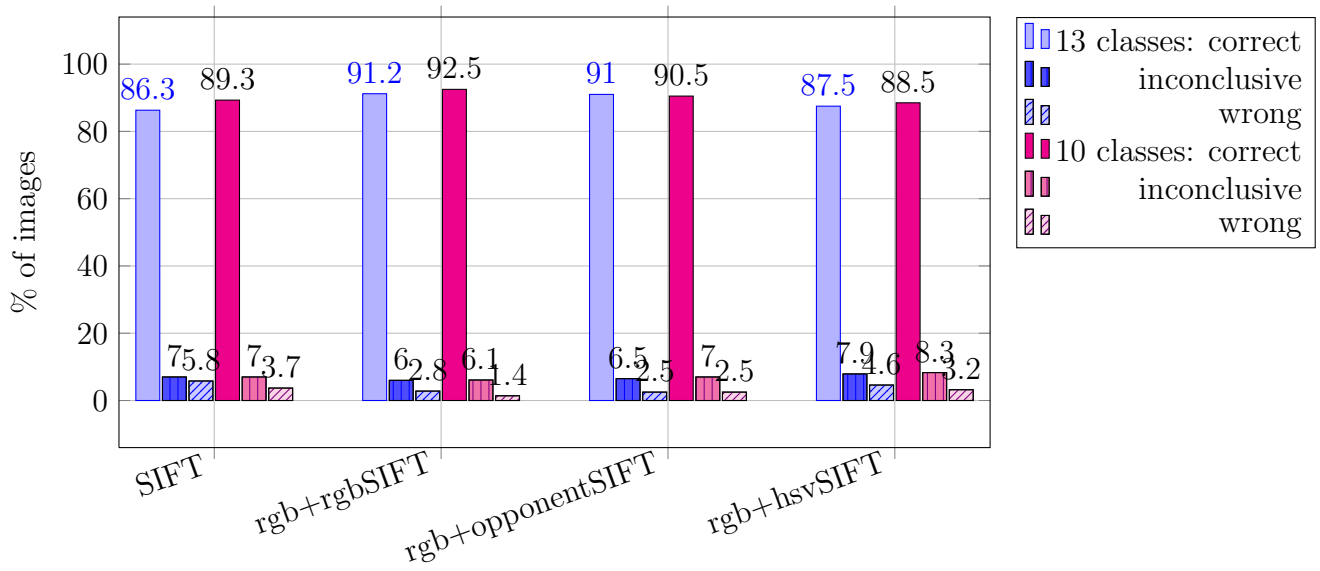


Figure 3.12: Comparison between the performance of SVM on the two dataset with various features. The correct rate, error rate and inconclusive rate are plotted from each experiment. The variance was negligible so it is not plotted.

Interestingly, the performance of the SVM varies less when we combine the colour and the shape feature. The optimal accuracy is achieved with the combination of RGB colour feature and shape feature described using the either *opponentSIFT* or *rgbSIFT* descriptors. The correct rate is the same on both datasets - 91.2% $\pm$ 0.5%, and the inconclusive rate is with 0.5% less on the larger dataset with the rgb+*opponentSIFT*. The rgb+*opponentSIFT* feature manages to achieve 0.5% higher accuracy on the extended dataset. The other feature combinations result in an approximately 1% decline in the performance on the larger dataset and smaller inconclusive rate, which is better compared to the 3% decline when only the shape feature is utilized.

The per-class performance of SVM with rgb, rgbSIFT and rgb+rgbSIFT is plotted in Figure 3.13 and discussed in detail in the following chapter.

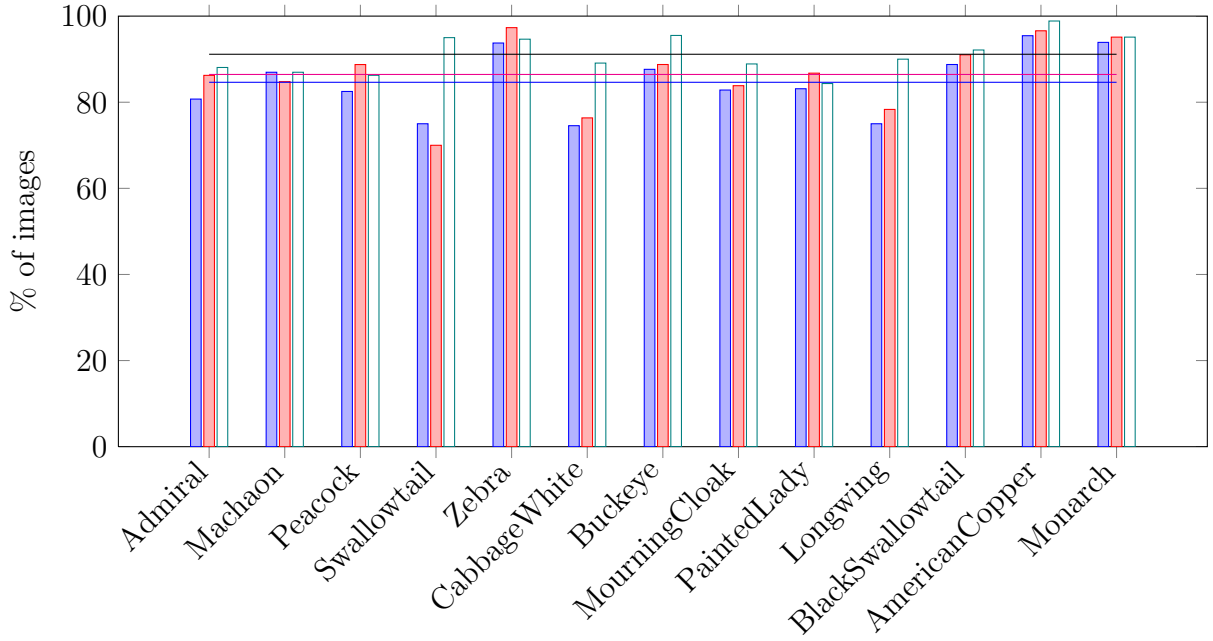


Figure 3.13: Correct rates for SVM classifier using the SIFT shape feature ( ), the rgbSIFT ( ) and the combined feature of rgb+rgbSIFT ( ). The average correct rates are 84.63%, 86.45% and 91.14% respectively and are plotted as straight lines.

4. Discussion

In this chapter, the important results from Chapter 3 are discussed and summarized and we try to draw conclusions about the butterfly identification problem.

There are several major discussion points that could be outlined on the basis of the evaluation of the experimental design and so this chapter is split into 4 sections:

- features
- classifiers
- data
- comparison to other work

We would discuss each of those in detail in there and summarize the most important contributions of this thesis in Chapter 5.

4.1 Features

A significant part of the work on this project was concentrated on feature engineering. We tried to find features which would allow us to differentiate between classes. Then we had to find an appropriate representation for each feature. The goal was to have representations which were invariant to various factors and in the same time captured as much of the variance of the data in as low dimensions as possible. Given the high accuracy and low error rate of the shape and the combined features we can conclude, that we have managed to find suitable features and feature representations. Moreover, the fact that both KNN and SVM achieve comparable high results points to the conclusion that the good classification results are indeed due to the features. Below we have outlined some of key facts inferred from the conducted experiments.

1. *Bag-of-words works well on this problem.*

   Given the variability of images, viewpoints, positions and lighting conditions and the high accuracy achieved by some of the methods, we can conclude that the bag of features approach is appropriate for this type of problem. Its flexibility allows a large number of images to be classified correctly and very similar classes to be distinguished, which is indeed remarkable given the limited number of features that we are using.

2. *The size of the vocabulary does not affect the performance significantly. Still it is important to find balance between enough variance representation and overfitting. For our dataset this is between 800-1200 words depending on the feature used.*

   The size of the vocabulary has little significance to the classification accuracy and vocabulary of size between 800 and 1200 seems to be enough to successfully represent the colour and the shape feature. Interestingly, although the increase in the vocabulary size generally leads to better accuracy of the SVM, it does not affect positively the inconclusive rate, i.e. more visual words do not make classes more separable. The assumption we made was that more words would mean that greater variance of the dataset was captured so a better model of each class could be created. However, this result might mean that the intra-class variance is too large, and that there is some noise, which is captured when more words are used. Smaller vocabulary size means that we are clustering the extracted descriptors in less clusters, or in other words that we are smoothing the image in the new feature space. More smoothing and thus smaller vocabulary means better generalization. Large vocabulary on the other hand might lead to overfitting of the model and this is a possible explanation for the increased inconclusive rate when larger vocabulary is used. This might also explain the decline in the performance of KNN as the size increases. It should be noted that if the dataset is significantly increased with a lot of new species, the optimal size of the vocabulary might require re-evaluation as a smaller size might not be able to capture some subtler difference.

3. *Smaller step size is better. However, it makes feature extraction slower. It also depends on the image size, so image size should be standardized.*

   The evaluation of the step size resulted in the expected outcome, that generally smaller step size is better. The optimal performance was achieved with step of 4 pixels. Smaller distances means that more features would be extracted from each image and that there would be large overlap between consecutive patches. This can results in increased time complexity in extraction and clustering. Also overlap between patches might lead to redundancy in the visual words or overfitting. This might explain why the performance of both SVM and KNN is worse with step size 2 compared to step size 4. Hence, 4 pixel step size was determined to be optimal. It is important to keep in mind that the size of the images in the datasets plays a significant role in the choise of step size. Obviously smaller images would need smaller step size. In this project, the images are standartized to be at most 500x500 pixels. Thus, it is reasonable to assume than a fixed step size would work well. However, as discussed later in this chapter, very small images, 100x100 pixels for example, are often misclassified, which might be

due to the step size. The number of such images in the dataset is small and so smaller step size doesn't improver the average accuracy. The fact the image size affects the choice of step size is one more reason to try and standardise the dataset by resizing the images.

4. *The pattern and the shape of a butterfly are more representative of its species than its colour. However, the colour is still important, especially when we want to classify more species.*

   The fact that the use of the shape/pattern feature leads to better performance is not surprising, especially given the naive method of colour description that was implemented. In every experiment conducted with the shape feature, the correct rate of the chi-squared kNN and SVM classifiers was with around 20% higher compared to when the colour feature was used. Also, the inconclusive rate of the SVM classifier is 18-20% with the colour feature compared to 6-7% with the shape/texture. All this leads to the conclusion that the colour feature is not descriptive enough and does not enable good class separability. There was only one class for which the classification results based on colour were as good as those based on the pattern feature and that was the Cabbage White class. The SVM classifier with colour feature resulted in 85.5% accuracy with 0% inconclusive rate compared to 83.7% with 3.6% inconclusive rate when the shape feature was preferred. Looking at the sample of the Cabbage White butterfly (Table 2.1), we can deduce that the reason for the better performance of the colour feature is the fact that a Cabbage White butterfly has a rather plain wings. There is a single black spot on each wing and the shape of the wing is not that outstanding. So, naturally, the shape and texture feature would provide just as much information as the colour if not even less, unlike with other species. Also this is the only almost purely white butterfly, so understandably it would be the easiest to classify based on colour. It should be pointed out, that the classification based on shape and pattern still managed to deliver comparable accuracy, probably due the lack of certain visual words present in other butterflies.

5. *Colour SIFT improves the performance of SVM and KNN. Concatenated histograms improves SVM even further.*

   Despite the fact that the assumption that the colour feature could not be used to conclusively distinguish butterfly species proved correct, through the numerous experiments we have also demonstrated its significance when combined with the shape feature. If we look into the error rates of SVM and compare results of colour against shape, it is clear that there isn't an absolute overlap. On the contrary, the worst performing classes with the shape feature are the top performing when the colour feature is used. Particularly, the classification with the shape feature has accuracy below

average on Cabbage White, Mourning Cloak and Longwing. These in turn are the most recognised classes by the SVM with colour feature. Hence, we expected to achieve an improvement in the recognition rates for those particular classes if we managed to successfully combine the colour and the shape feature. Therefore, we would look into how the performance of the classifier changes on these particular classes.

Obviously, when the confusion matrices of two features do not overlap, there might be some benefit in their combination. It is important to keep in mind, that the error rate obtained with SVM is quite low, so even slight improvement would be noteworthy. An improvement in the inconclusive rate would be beneficial. We have looked into two different methods to incorporate the colour feature in the classification process using colour SIFT to describe the visual patches and concatenating the shape and colour feature histograms in one histogram. Both of this methods lead to improvement and confirm the hypothesis that the colour feature is beneficial to the classification. Furthermore, the results demonstrate that the colour feature is very important to keep the results stable if one would aim to scale the problem of butterfly classification to include more classes.

From the Figure 3.13 it becomes obvious that the integration of colour using the colour SIFT (*rgbSIFT* in particular) helps to increase the recognition rate of some of the problematic classes for gray-SIFT. The concatenation of the colour feature and the SIFT leads to even further improvement, so much that some of the low performing classes, achieve accuracy rates close to 90%. For the Cabbage White for example, we obtain accuracy of 89%, which is an improvement of 15% on the gray SIFT.

6. *Combining features leads to a more stable performance with larger datasets (with SVM).*

   We have conducted an experiment with the top performing setups (SVM with shape feature and combined feature) on a larger, 13-class dataset. The particular characteristics of the extended dataset are discussed later in this section, but there was an obvious trend when we compared the performance of the SVM with different features. That was the fact that using a single feature, the shape feature in this case, and extending the dataset to include 3 more classes caused the average accuracy of the programme to decrease with $2.1\% \pm 0.5\%$. This decline was due to more errors, as the inconclusive rate was 7% for both datasets. When we use a combination of shape and colour features, the difference between the correct rates of the classifier on the two datasets is smaller. For the combination of *rgb*-colour feature and *opponentSIFT*, the error rate is 2.5% on both datasets, but the combined feature leads to a better accuracy on the larger class. Thus, we have a reason to believe that the combined features would further improve the

performance of the shape feature classifier on larger datasets. It would also make the algorithm more stable, so that it produces consistently high results even with a lot more species.

7. *Histogram concatenation causes a decline for KNN, so this feature space is not suitable for the Chi-squared distance metric.*

   It is interesting that the behaviour of the KNN with the concatenated histogram is very different from that of the SVM classifier. The colour SIFT features still lead to an improvement of the accuracy, but the concatenation of histograms causes a decline in the performance in every case regardless of which features are concatenated. The χ^2 distance measure, despite being a weighted distance is still based on the Euclidean distance. So more dimensions would generally make it more difficult for the classifier, especially when not all are equally important. The weights of the visual words representing the colour should be a lot smaller than those representing shape, because the shape feature is far superior to the colour. However, as we have not implemented any prior weights for the different features, the weighting algorithms has to do all the work. Obviously this doesn't always work as expected. As described in Section 2.3 χ^2 deems codewords which are rare to be more important. This might lead to it giving the colour codewords higher weight than necessary, but this should almost never be the case considering that they are associated with low accuracy.

To sum up, the shape/texture feature is the most important for successful discrimination, as it captures crucial information about the pattern of the butterfly wing. The colour feature is necessary to achieve high recognition rates for certain classes. The good combination of both becomes more important with the increase of the number of species.

4.2 Classifiers

Although a significant amount of time was dedicated to selecting and fine-tuning the feature set and its representation, the importance of the classifier is not underestimated. So two very different classifiers – KNN and SVM, were carefully evaluated and the conclusions derived from those experiments are summarized below. Firstly, we comment on the KNN classifier and then on the better performing SVM.

4.2.1 KNN

The main parameters of the KNN classifier are the number of nearest neighbours used in the classification and the distance metric. These were optimized through cross-validation. Also we observe how the classification performance varies with different features.

1. *Choose small k.*

   The optimal results were achieved with 1 nearest neighbour. With 3 the correct rate was almost the same, however with 5 or more a visible decline could be observed. There are several possible explanations, the first one being that often intra-class variance might be larger than inter-class. In other words, sometimes the difference between samples from different classes are very subtle, so they might lie close to each other in the feature space. This is a reasonable assumption considering the type of classification that is being performed – species identification and not categorization of completely different object. So we can deduce that lower number of neighbours is better suited to fine-grained classification as is the case here. Another possible explanation might be that we have a sparse set of sample, and the 5^{th} or 6^{th} nearest neighbour might be in a completely different part of the feature space. This is also a valid presumption, as there is not that much training data of some of the species. In both case, it would be better to use either the nearest-neighbour or kNN with k=3.

2. χ^2 *distance metric is far better suited for the problem than the Euclidean distance.*

   χ^2 is clearly better suited for this problem than the Euclidean distance and this is not surprising considering the large number of dimensions and the fact that not all are equally important. Although, through clustering and optimization of the vocabulary size we try to estimate a set of visual words which would represent the variance in the dataset without redundancies, there is no guarantee that the particular set of codewords would improve class separability. This combined with the fact the not all dimensions are equally important in every case can lead to bad performance of Euclidean distance KNN. On the other hand, the chi-squared distance metric is a weighted Euclidean distance and with it the accuracy of the KNN neighbour reaches 87% (30% more than the highest score achieved by Euclidean-KNN). The success of the KNN depends mostly on the amount of training data, because the classifier bases its prediction on similarities between the samples. The fact that we managed to obtain such high accuracy rate with a relatively limited dataset, proves again the discriminative power of the features we have chosen – in particular the shape/texture feature. The cor-

rect rate is even slightly higher when three new classes are introduced in the dataset, which can be explained by the fact that KNN usually benefits from more training samples. It is still outperformed by the SVM and has issues with the combined feature which we discuss below.

3. *KNN does not work well with concatenated histograms. However, this is necessary when there are more species to classify, so it would be better to implement another feature combination technique or use SVM instead.*

   Despite good performance of the KNN with the shape feature, one obvious problem of this classifier is that the method of concatenating histograms leads to a decline in the accuracy. This might be because the weights calculated by the chi-squared metric are not appropriate. Because the shape features delivers better performance, the shape/pattern codewords should almost always weight more than the colour codewords. Most likely, this is not very well represented by the method of weight estimation of the chi-squared distance, which is to give more weight to rare features. If we want to use combined features with the KNN, the combination technique has to be changed. We have reasons to believe that this would be important, because the colour SIFT led to an improvement with the KNN as with the SVM, so obviously colour is beneficial for the classification as discussed in the previous section 4.1. What is more, combined features provide a feature space which is more scalable and allows us to obtain similar accuracy rates with the extended dataset as with the original dataset. As we would be aiming to include more species in the classification, we would need to develop another feature combination method if we want to use the KNN classifier. Given the superior performance of the SVM and its flexibility to different feature spaces, we can conclude that the SVM is the better choice.

4.2.2 Support Vector Machine

1. *Outperforms KNN with shape and combined feature on correct rate and works almost as good as KNN on colour. Moreover, SVM obtains lower error rates that KNN in every case – ranging from 19% to 1.4%, compared to 50%-14.6% for the KNN.*

   The optimal performance on both 10-class dataset and 13-class dataset was achieved using the SVM classifier 92.5% and 91% respectively, with the combined feature. The corresponding error rates were 1.4% and 2.7%, which is very low considering that 800 and 1000 images were classified. The SVM manages to work very well with the features which were chosen to represent the images and combining the features worked as expected the combined feature method outperformed the single-feature classification. Consider-

ing the low error rate of the SVM, little has to be desired in that area. A more obvious problem is the inconclusive rate, which despite not being very high with the combined feature (6-7% depending on the specific configuration), still accounts for 60-70 images on the larger dataset. As discussed in Section 2.4.4, the inconclusive decisions result from the implementation of multi-class svm which we have decided to use. The alternative was to randomly assign the sample to a class. However, tagging the images as inconclusive was more useful, because in classification we might occasionally be presented with objects which do not fall in any category and we need a good way to separate those from the incorrect guess. Furthermore, the inconclusive rate can provide information about the feature and descriptors used.

2. *The inconclusive rate can be used to evaluate how good a vocabulary is. Higher inconclusive rate means unsuitable vocabulary. Hence, the optimal vocabulary is the union of the colour an shape/texture. The second best is the shape/texture and the colour vocabulary is the worst performing when used on its own.*

   High inconclusive rate might be interpreted as an inability of the vocabulary or feature to represent a species in such a way so that it is distinguishable from other species. Large class variance might also result in high inconclusive rates. Given the fact that images are taken in different lighting conditions, and that colour variations of the same species sometimes occur, it is not surprising that the inconclusive rate of the SVM classifier was so high when the classification was based on the colour feature.

   Interestingly, the only method which resulted in a decline of the inconclusive rate was when the combined features were used. Every other approach – colour SIFT, increase in vocabulary size, increase in step size – resulted in either increase in the number inconclusive decisions or did not affect them. There are two possible reasons for this. One way to look at it, is that combining two features provides more information, so edge cases are easier to classify. On the other hand, the colour SIFT itself does not result in much difference in the inconclusive rate, so there must be something more. If we observe the inconclusive rate per class for the colour and shape features, we can see that the inconclusive rate for two of the classes – Cabbage White and Mourning Cloak, are lower with the colour feature than with the shape feature. This is quite interesting considering that the average inconclusive rate of the colour feature is 18.5% compared to 7% for the shape feature. The fact that the inconclusive rates don't always overlap means that an improvement is expected when the two features are combined and exactly this happens, the Cabbage White inconclusive rate drops to 0% and Mourning Cloak to 9% from 11.1%. Although the overall

decrease in the inconclusive rate is small, around 1%, it is reasonable to assume that the combination of features is the right approach to minimize the number of inconclusive samples and consequently increase the correct rate.

Despite the good results, the one-against-all SVM still has room for improvement. It would be useful if we could rank the top 3 most probable classes, or reduce the number of possible classes even in the case of inconclusive predictions or in other words – provide a best guess. This can be done using Platt's probabilities method for converting SVM scores to probabilities [19]. This has not been implemented at this point, but would provide a lot more functionality and flexibility to the system, so it is a noteworthy topic of future work.

4.3 Data

Initially we evaluated the algorithms on a 10-class dataset with 843 images. In order to test the scalability of the top performing methods, this dataset was extended with 3 more species. These were chosen so that they are similar to some of the classes in the initial dataset. The American Copper and the Monarch are brownish in colour and share some of the common patterns observed in other classes like Admiral, Peacock, Painted Lady. What's more, the Black Swallowtail class is of the same family like the Giant Swallowtail and they look visually very similar; they differ only by a couple of blue spots and a white strip across the body of the butterfly (c.f. Table 2.1). This new dataset consisted of 1012 images and was generally more complex because of the similarities between the species and the increased number of classes and images to classify. Below we have summarized some of the conclusions reached from experiments on the two datasets.

1. *In general more species cause decline in the performance and more precisely an increase in the error rate.*

   There was only one combination of features with which the same accuracy was achieved on both the smaller and the larger datasets. This was not the optimal performance, but differed from it only by 0.2% which is in the range of result variation of the classifier. Although the average accuracy decreased it was only with around 1% when the combined feature is used and 2% with the shape feature. As discussed above the decline in the performance could be avoided by employing combinations of more features, as they provide more information and capture greater variance.

2. *Small images are often misclassified – grid size might affect this. Also closed wings are hard to classify.*

Table 4.1: Misclassified samples.

Table 4.2: Inconclusive samples.

In Table 4.1 are some of the images which were misclassified with the top performing methods in multiple runs. By observing them, we can deduce that often the misclassified images are small in size and show butterflies pictured from the side. The obvious challenges when classifying side-view images of butterflies could explain the misclassification. Another problem when dealing with images of different size is that since the dense-Sift method extracts descriptors from a fixed sized grid, the step size might not be optimal for smaller sizes. This is why, the image size is standardized over the dataset. This is to ensure that the maximum dimension is of size 500 pixels and doesn't deal with smaller images. However, the error rate is very small – 1-3%, so we can conclude that the overall performance of the algorithm is satisfactory and some errors are to be expected considering the uncontrolled data.

3. *Bad quality photos, sideways or upside down images are sometimes labelled as unknown.*

   In Table 4.2 some of the inconclusive images are presented. By observing those we can conclude that low quality of images – with scratches, very high or low illumination etc., are hard to classify. Bad images can lead to problems with the feature extraction as with any computer vision problem. For example when the image is unfocused, the descriptors might not be properly extracted. Low resolution might also be problematic. Although these cannot be avoided, the hope is that given the high-quality cameras in most mobile phones, such occurrences would be rare. Furthermore, butterfly images are usually taken outdoors during the day, so although illumination might vary, there should be enough lighting to clearly see the butterfly pat-

Table 4.3: Samples of classes which are below average.

tern, which is what is important. One problem, specific to the butterfly identification task is the fact that when picture in flight butterflies might look very different and patterns on their wings can be hidden/distorted. Although the SIFT descriptors are suppose to deal with such cases, it seems that such images often cause inconclusive predictions. The inconclusive rate is higher than the error rate, so trying to minimize this is crucial to the improvement of the classification. As discussed above, a feature combination approach would most likely improve the situation.

4. *Even very similar classes can be successfully distinguished with good features.*

   A particularly remarkable observation is that our dataset includes two species of the same family – Black Swallowtail and Giant Swallowtail (c.f. Table 2.1). However, surprisingly they are not often mislabelled as one another, in fact almost never. This leads to the conclusion that the proposed classification method has a good chance to work well on classes which have subtle difference between each other, as long as the class variance is not very large. If it is, we can look into building several models per class – e.g. open wings, closed wings or sideways view.

5. *Below average results are obtained on classes which share similar features. Despite the lower accuracy, the error rate is small.*

   There are several classes which are brownish in colour and have rounded patches (eyespots) on their wings (See Table 4.3). Some of the samples are hard to classify even manually and the fact that we achieve such a low error rate demonstrates the suitability of the algorithm to this problem. Thus, it is important to combine as many features as possible in order to minimize error in similar classes.

4.4 Comparison to other work

Unfortunately, no experiments on butterfly identification have been performed on precisely the datasets that have been used for evaluation here, so direct comparison is difficult. Furthermore, although there are some similar projects developed – bee identification, spider identification, classification of different moths, many of these implement a retrieval scheme instead of a classification one. This means that results reported are usually accuracy of prediction in top 3, 5 or 10 guesses. With our current implementation of SVM, this could not be done easily, so direct comparison is not possible. One of the project most similar to the butterfly identification performed here, is the flower identification project [26]. Table 4.4 summarizes the results of other relevant projects.

System	Number of species	Reported accuracy	Details
ABIS [28]	35	90%	requires specialist knowledge to extract features
Concatenated histograms [16]	4	82%	
DAISY [37]	49	90%	supervised
DAISY [24]	49	85%	requires manual highlighting of ROI
Butterflies and Ladybug [20]	unknown	80-90%	only on yellow and white butterflies
Identification on family level [35]	unknown (large)	84%	classifies up to a Family level, works well only for one family
Flowers [26]	17	88.04% ± 1.9%	per-class correct rate ranges between 40-100%
Flowers	103	85%	shortlist of 10 predictions is retrieved
Proposed method	10	92.5±0.5%	inconclusive rate - 6.1% ± 0.2%, error rate is ≈ 1.4%
Proposed method	13	91.2% ± 0.6%	inconclusive rate is 6% ± 0.2% so error rate only ≈ 2.8%

Table 4.4: Summary of reported accuracy of similar projects. Most of those are mentioned and referenced in Section 1.1. The proposed method performance reports the scores from classification using the combined *rgb* colour + *rgbSIFT* shape feature on the two datasets presented in the thesis.

Although many of the methods include more classes than our dataset, the setup of some of these experiments greatly simplifies the problem from Computer Vision point of view. For example the DAISY programme achieves 90% accuracy on images which were taken in a supervised environment – fixed background, spread wings, the insect is positioned in a specific way. This is not only time consuming, but also requires specialist information and is not suitable for the problems discussed here. The method was later extended to work by extracting Regions-Of-Interest on the insect wings where the ROI was manually highlighted. Obviously, our situation allows a lot more variety in pose, lighting and does not require specialist knowledge.

Another system which achieves comparable correct rate on more species is ABIS. The system requires specialists to take the photos of the bees, because the classification is based on specific veins in the bee's wing. Again, this approach requires more supervision that we aim for. It also uses features specific to bees, so the method is not transferable to our problem. The other methods presented also have limitations, like being able to recognise only yellow or white butterflies, or classifying the butterfly on a family level (there are 6 butterfly families in total).

The only project which provides as much freedom in terms of number of species and image variety is the Flower identification project. The results on a dataset of comparable size 17 species, are similar to the ones we achieved, 88.04% vs 91.2%. However, the Flower Identification work reports larger variation in the scores on a per-class basis. The correct rate ranges between 40% and 100% depending on the class. In our case, there is no class with correct rate of less than 80% with the optimal set of features. The result of the flower identification on 103 species dataset is extremely good and the system generalizes very well. It was achieved with a combination of 5 different features and a custom position systematization algorithm. It should be noted, however, that this correct rate is computed on a shortlist of 10 classes. This means that the correct class has to be present in the top ten prediction. Hence, it is not possible to directly compare the scores.

To sum up, although we did not have much basis for comparison, as the problem of classifying species is not so extensively studies as that of general object categorization, from the information available we can conclude that the methods proposed in these thesis achieve good results. The correct rates we obtained are comparable to those of the flower identification project which is the only one to tackle a problem of similar scale. However, there is still room for improvement in the number of species included in the classification.

5. Conclusion

This thesis has addressed the task of identifying butterfly species from a photo. We have collected two datasets of segmented images of 10 and 13 commonly occurring butterfly species and developed a method of identifying them by extracting colour and texture features. The main contributions of this thesis are:

- **Datasets** : a 13-species dataset was created by combining two smaller datasets – PONCE and LBD. The species are commonly occurring and the images provide enough classification challenges through variation of viewpoints, pose of the butterfly and natural conditions at the time the photos have been taken. To build this dataset and make it directly usable for image classification problems, two smaller datasets had to be combined. This included finding overlapping classes, removing mislabelled images and segmenting more than 300 images (from the LBD). This makes it the largest freely available butterfly dataset, which includes masks to segment the butterflies from the image background, and thus directly use it for classification.

- **Feature Engineering** : colour and texture features were thoroughly examined and evaluated in the context of butterfly identification. We have evaluated different feature representations and features. In Chapter 2, we discuss the construction of a visual vocabulary – initially a colour vocabulary and then the more complex shape/texture vocabulary. For the colour feature, descriptors in the RGB and HSV colour space were evaluated. The pattern/texture visual words are circular patches represented by SIFT descriptors. The vocabularies were build through *k-means* clustering. Both of these vocabularies were fine-tuned and evaluated in Chapter 3 and we concluded that the shape feature outperforms the colour feature regardless of the classifier. However, the shape feature did not perform very well on some of the classes on which the colour feature excelled, so we had reason to believe that combining the colour and shape/texture feature would be useful. One way of using the colour was to describe the texture/pattern visual words with descriptors that also captured colour information – colour SIFT, such as *rgbSIFT*, *hsvSIFT* and *opponentSIFT*. Those improved the performance on some of the classes. We then developed a more direct approach of feature combination – histogram concatenation, which improved the performance further and even managed to decrease the rate of inconclusive classifications.

- **Classification** : a one-against-all Support Vector Machine was implemented and evaluated against a k-Nearest-Neighbour classifier on different features and on a 10- and 13-species datasets. The parameters of the KNN

were optimized through 10-fold cross-validation (Chapter 3). χ^2 distance measure was found to be far superior to the Euclidean distance measure and the optimal result of KNN on the 13 class dataset was with the *rgbSIFT* shape feature – $86\% \pm 0.5\%$. The SVM obtained lower correct rate than the KNN on the colour feature, but outperformed the nearest-neighbour classifier in all other experiments. Its performance was evaluated by observing the correct, error and inconclusive rate obtained with different features. The optimal result on the 13 class dataset was $91.2\% \pm 0.6\%$ with 2.8% error rate and was achieved by a combination of RGB colour feature and the *rgbSIFT* shape feature. The RGB colour feature combined with the *opponentSIFT* also led to a high result on the 13 species dataset $91\% \pm 0.5\%$ (2.5% error rate) and provided more stable performance as the correct rate was the same on the smaller, 10 class, dataset as on the extended one. Misclassifications were due to small size of the images and large variation in viewpoint.

6. Future work

Possible directions for future research will be presented now:

Extend to more species : The framework for classification of similar species was presented in this thesis and in order to be useful it needs to be extended to include more classes. For that, a larger dataset needs to be collected, preferably with large variation in the images to ensure that enough variance is captured. This dataset would in the best case include segmentation mask of the images, otherwise they would need to be manually segmented or a segmentation procedure has to be developed. A good number of species would be 200-250, but a smaller dataset could potentially work if the classification is concentrated on a particular geographical area.

Add more features : Although we provided a good set of features which allowed successful classification of 13 classes, if the programme was to be extended to include more species, we would require more features. For example descriptors could be used to represent the outer shape of the butterfly, such as SIFT descriptors on the boundary of the butterfly. Another approach would be to find a better way to combine features. A multi-kernel approach which allowed different feature weights to be specified on a per-class basis produced very promising results in the flower identification project and could possibly work well for the butterflies as well.

Classifiers : A lot of similar problems use retrieval approaches instead of pure classification methods, to achieve higher recognition results. It would be convenient to be able to give several plausible classification outcomes as well as the top prediction. This would be especially useful in the case of inconclusive classification, where we could at least provide our best guess for the possible species. User feedback could then be used to select the class which looks more similar. This could also be used to extend our database of images. A possible way to do this is to implement Platt's probabilities for the Support Vector Machine. This would provide posterior class probabilities and would allow ranked classification.

Integration : Although this thesis focuses on the Computer Vision/Machine Learning part of the problem, the main direction of future work would be to integrate the classifier in a mobile phone app. The app would be used to take a photo of a butterfly and upload it to a server. The server would classify the butterfly and return the species name and a sample photo (or top 3-5 most likely species). In case of ranked classification, the user would either agree with the prediction or mark one of the other likely species as correct based on their similarity to the uploaded photo. The information on that species would then be downloaded on the device. Numerous extensions are available and various directions can be taken

with the app – educational, conservation etc.. An important thing to note is that once the image is uploaded to the server and the classification result is confirmed by the user, the image could be added to the existing database. Thus, the programme would gradually learn and get better. Another point to consider is that the data collected on the server from numerous worldwide butterfly observations could be used in various scientific projects in the field of Butterfly Conservation and more.

Bibliography

[1] Anna Bosch, Andrew Zisserman, and Xavier Muoz. Image classification using random forests and ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[2] Anna Bosch, Andrew Zisserman, and Xavier Muoz. Scene classification using a hybrid generative/discriminative approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(4):712–727, 2008.

[3] J. F. Canny. Finding edges and lines in images. Master's thesis, MIT, 1983.

[4] Chris Dance, Jutta Willamowski, Lixin Fan, Cedric Bray, and Gabriela Csurka. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[5] S Das, B Bhanu, and CC Ho. Generic object recognition using multiple representations. *IMAGE AND VISION COMPUTING*, 14(5):323 – 338, n.d.

[6] M.T. Do, J.M. Harp, and K.C. Norris. A test of a pattern recognition system for identification of spiders. *Bulletin of Entomological Research*, 89:217–224, 2 1999.

[7] Piotr Dollár. Piotr's Image and Video Matlab Toolbox (PMT). "http://vision.ucsd.edu/ pdollar/toolbox/doc/index.html".

[8] Gyuri Dorkó, Cordelia Schmid, et al. Object class recognition using discriminative local features. 2005.

[9] Kai-Bo Duan and S Sathiya Keerthi. Which is the best multiclass svm method? an empirical study. In *Multiple Classifier Systems*, pages 278–285. Springer, 2005.

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

[11] Robert Fergus, Li Fei-Fei, Pietro Perona, and Andrew Zisserman. Learning object categories from google's image search. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1816–1823. IEEE, 2005.

[12] Terrence S Furey, Nello Cristianini, Nigel Duffy, David W Bednarski, Michel Schummer, and David Haussler. Support vector machine classification and

validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

[13] Varun Gulshan, Carsten Rother, Antonio Criminisi, Andrew Blake, and Andrew Zisserman. Geodesic star convexity for interactive image segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3129–3136. IEEE, 2010.

[14] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[15] Frederic Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 604–610. IEEE, 2005.

[16] Natalia Larios, Hongli Deng, Wei Zhang, Matt Sarpola, Jenny Yuen, Robert Paasch, Andrew Moldenke, David A Lytle, S Ruiz Correa, E Mortensen, et al. Automated insect identification through concatenated histograms of local appearance features. In *Applications of Computer Vision, 2007. WACV'07. IEEE Workshop on*, pages 26–26. IEEE, 2007.

[17] S. Lazebnik, C. Schmid, and J. Ponce. Semi-local affine parts for object recognition. In *BMVC -CONFERENCE*, volume 2 of *British Machine Vision Conference*, pages 959 – 968, 2004.

[18] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.

[19] Zeyu Li, Shiwei Tang, and Shuicheng Yan. Multi-class svm classifier based on pairwise coupling. In *Pattern Recognition with Support Vector Machines*, pages 321–333. Springer, 2002.

[20] Jaehong Lim, Jongman Cho, Taewoo Nam, and Soohong Kim. Development of a classification algorithm for butterflies and ladybugs. In *TENCON 2006. 2006 IEEE Region 10 Conference*, pages 1–3, 2006.

[21] B. H. Liu. Semi-automatic feature selection for fine-grained image retrieval. master thesis. Master's thesis, Department of Computer Science and Information Engineering, National Taiwan University, Taibei, 2003.

[22] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[23] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91 – 110, 2004.

[24] Michael Mayo and Anna T Watson. Automatic species identification of live moths. *Knowledge-Based Systems*, 20(2):195–202, 2007.

[25] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In Anders Heyden, Gunnar Sparr, Mads Nielsen, and Peter Johansen, editors, *Computer Vision ECCV 2002*, volume 2350 of *Lecture Notes in Computer Science*, pages 128–142. 2002.

[26] M-E. Nilsback. *An Automatic Visual Flora – Segmentation and Classification of Flowers Images*. PhD thesis, University of Oxford, 2009.

[27] Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In *Computer Vision–ECCV 2006*, pages 490–503. Springer, 2006.

[28] S. Schrder, W. Drescher, V. Steinhage, and B. Kastenholz. *An Automated Method For the Identification of Bee Species (Hymenoptera: Apoidea)*, pages 6–7, 1995.

[29] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477 vol.2, 2003.

[30] J. A. Thomas C. D. Thomas P. C. Hammond T. R. New, R. M. Pyle. Buannual review of entomology. 40:57–83, 1995.

[31] K. E A Van de Sande, T. Gevers, and C. G M Snoek. Evaluating color descriptors for object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1582–1596, 2010.

[32] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[33] Andrea Vedaldi and Andrew Zisserman. Image classification practical. http://www.robots.ox.ac.uk/ vgg/share/practical-image-classification.html, 2011.

[34] Janett Walters-Williams and Yan Li. Comparative study of distance functions for nearest neighbors. In *Advanced Techniques in Computing Sciences and Software Engineering*, pages 79–84. Springer, 2010.

[35] Jiangning Wang, Liqiang Ji, Aiping Liang, and Decheng Yuan. The identification of butterfly families using content-based image retrieval. *Biosystems Engineering*, 111(1):24 – 32, 2012.

[36] Josiah Wang, Katja Markert, and Mark Everingham. Learning models for object recognition from natural language descriptions. In *Proceedings of the British Machine Vision Conference*, 2009.

[37] Anna T. Watson, Mark A. O'Neill, and Ian J. Kitching. Automated identification of live moths (macrolepidoptera) using digital automated identification system (daisy). *Systematics and Biodiversity*, 1(3):287–300, 2004.

[38] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *INTERNATIONAL CONFERENCE ON COMPUTER VISION*, volume 2 of *Computer vision; ICCV 2005*, pages 1800 – 1807, 2005.

[39] I. Yahiaoui, O. Mzoughi, and N. Boujemaa. Leaf shape descriptor for tree species identification. In *Proceedings - IEEE International Conference on Multimedia and Expo*, pages 254–259, 2012.

[40] Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, MIR '07, pages 197–206, 2007.

[41] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, pages 1–51, 2006.

[42] Hao Zhang, Alexander C Berg, Michael Maire, and Jitendra Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2126–2136. IEEE, 2006.