Improving Image Classification by Co-training with Multi-modal Features

Kyle Weston

Master of Engineering

Department of Electrical Computer and Software Engineering

McGill University
Montreal, Quebec
April 2011

A Thesis submitted to the Faculty of Graduate Studies and Research in partial fulfilment of the requirements for the degree of Master of Engineering

© Kyle Weston, 2011
ACKNOWLEDGEMENTS

I would like to express my gratitude to all those who made it possible for me to complete this thesis. I want to especially thank my supervisor Professor Martin Levine for his valuable insights and direction while refining the focus of this thesis, and for providing continual feedback throughout the research and writing stages.

I would also like to thank the CIM administration and ECE staff for providing the computer resources and technical support needed to enable me to carry out my experiments.

Finally, I would like to express my great appreciation to my wife Mandy for her proof reading, patience and many words of encouragement. Without her help this thesis would not have been possible.
ABSTRACT

We explore the use of co-training to improve the performance of image classification in the setting where multiple classifiers are used and several types of features are available. Features are assigned to classifiers in an optimal manner using hierarchical clustering with a distance metric based on conditional mutual information. The effect of increasing the number of classifiers is then evaluated by co-training using the assigned feature sets. Experimental results indicate that the feature assignments chosen by the clustering approach afford superior co-training performance in comparison to other logical assignment choices. The results also indicate that increasing the number of classifiers beyond two leads to improved performance provided that the classifiers are sufficiently independent, and are reasonable well balanced in terms of labeling ability.

Additionally, we explore the effect that the initial training set selection has on co-training performance. We find that the quality of training images has a profound effect on performance and provide recommendations for how best to select these images.
Nous explorons l'utilisation de la co-formation pour améliorer la performance de classification d’image dans un milieu où multiples classificateurs s’emploient et plusieurs types de caractéristiques sont disponibles. Les caractéristiques sont associés aux classificateurs d’une manière optimal en employant le groupage hiérarchique avec une mesure de distance basée sur l’information mutuelle conditionnelle. L’effet d’augmenter le nombre de classificateurs est alors évalué par la co-formation, en employant les ensembles de caractéristiques attribués. Les résultats de nos expériences indique que si on augmente le nombre de classificateurs au-delà de deux, la performance s’améliore pourvu que les caractéristiques soient suffisamment indépendantes et assez bien équilibrées en termes de compétence d’étiquetage.

# TABLE OF CONTENTS

ACKNOWLEDGEMENTS ................................................. ii  
ABSTRACT .............................................................. iii  
ABRÉGÉ ................................................................. iv  
LIST OF TABLES ....................................................... vii  
LIST OF FIGURES ..................................................... viii  
1 Introduction ......................................................... 1  
  1.1 The Problem .................................................... 3  
  1.2 Approach ....................................................... 4  
  1.3 Contributions .................................................. 5  
  1.4 Outline .......................................................... 6  
2 Background .......................................................... 7  
  2.1 Related Work .................................................... 7  
    2.1.1 Co-training ............................................... 7  
    2.1.2 Combining Textual and Visual Information .......... 9  
    2.1.3 Using Separate Visual Modalities ................. 11  
    2.1.4 Extending to Multiple Views ................... 12  
  2.2 Support Vector Machines ...................................... 13  
    2.2.1 Dual Form ............................................... 14  
    2.2.2 Soft-Margin SVM ..................................... 15  
  2.3 Information Theory ........................................... 16  
    2.3.1 Shannon Entropy ...................................... 17  
    2.3.2 Mutual Information .................................. 17  
    2.3.3 Conditional Mutual Information ................ 18  
3 Methods .............................................................. 19  
  3.1 Datasets ....................................................... 19  
  3.2 Features ....................................................... 21  
  3.3 Classifier ..................................................... 23  
  3.4 Co-training with N Classifiers .......................... 25  
  3.5 Kernel Reduction ............................................ 27  
  3.6 Feature Clustering Algorithm ............................ 32  
    3.6.1 CMIM-based distance metric .................. 34
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Feature definitions to be used in the experiments. Individual feature types are in columns and feature group definitions are in rows. For simplicity, the 3x1 horizontal decompositions are not shown since they are always grouped with the 1x1 image decompositions of the same feature type.</td>
<td>22</td>
</tr>
<tr>
<td>3-2</td>
<td>Results for the Pascal VOC’07 dataset using text and visual classifiers. Each classifier is trained using the full training set and results are reported for each class using average precision.</td>
<td>24</td>
</tr>
<tr>
<td>3-3</td>
<td>Results for the MIR Flickr dataset using text and visual classifiers. Each classifier is trained using the full training set and results are reported for each class using average precision. The classes marked with * are the relevant class labels and those without are the potential class labels. Relevant labels correspond to a specific interpretation of a concept from a single annotator, while potential labels can apply to the image in a more general sense.</td>
<td>25</td>
</tr>
<tr>
<td>6-1</td>
<td>An example of the cumulative feature groups chosen for $N = 3$ for the Pascal VOC’07 dataset. The top row shows all cluster candidates chosen, where each feature type is represented by a number from 1 to 9. The bottom row shows the number of times each cluster candidate was chosen among all classes. In this case columns 3, 7 and 8 would be selected as most common. These correspond to the colour, grayscale and text feature groups.</td>
<td>61</td>
</tr>
<tr>
<td>6-2</td>
<td>The results of hierarchical clustering of feature types on the Pascal and MIR Flickr datasets with and without image tags. Each assigned cluster is given a letter from A to E. Letters are assigned in the order the clusters were formed.</td>
<td>62</td>
</tr>
<tr>
<td>6-3</td>
<td>Feature splits used in the co-training experiments. The CMIM split is the feature split chosen by the CMIM clustering algorithm. The logical splits were manually chosen by considering the feature strength and the conceptual independence of the features.</td>
<td>64</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Overview of the semi-supervised setting. Each classifier is trained on a small set of labeled images and is then bootstrapped using an additional set of unlabeled images. The goal is to predict the class labels for the test images. Two settings are shown here, one were image tags are included in the training phase and one where they are not. In either case it is assumed that image tags are not available at test time.</td>
<td>5</td>
</tr>
<tr>
<td>2-1</td>
<td>Some example training images shown with their associated Flickr image tags and class labels.</td>
<td>9</td>
</tr>
<tr>
<td>3-1</td>
<td>Sample images from each of the datasets used in the experiments.</td>
<td>20</td>
</tr>
<tr>
<td>3-2</td>
<td>Overview of co-training with N classifiers. Solid lines represent the training phase and dashed lines represent the labeling phase. Classifier $h_t$ is trained on image tags while classifiers $h_v$ are trained on visual features. After the classifiers are trained using the labeled image features, each classifier is used to label additional images from the unlabeled set. These images are then added to the training set and the process repeats.</td>
<td>27</td>
</tr>
<tr>
<td>3-3</td>
<td>The kernel reduction technique assuming a 8x8 full kernel matrix. The grid squares shown in grey are the kernel elements that are copied over to the new reduced kernel.</td>
<td>31</td>
</tr>
<tr>
<td>4-1</td>
<td>Performance scores for a varying number of training images for a subset of classes from the MIR Flickr and Pascal datasets. Performance is shown for the visual combined classifier only. Training sizes of 1, 3, 5, 10, 25, 50, 100, 200, 500, 1000, 2000 and 5000 images were used. The same number of images were chosen for the positive and negative images in each case. Note that some classes only have a few hundred images, which is why the plots end at varying points along the x axis.</td>
<td>39</td>
</tr>
<tr>
<td>4-2</td>
<td>A plot showing the effect of varying the active pool size for a few sample classes from the Pascal VOC’07 dataset.</td>
<td>42</td>
</tr>
<tr>
<td>4-3</td>
<td>Distribution of training images for each dataset.</td>
<td>43</td>
</tr>
</tbody>
</table>
4-4 AP scores shown for 100 rounds of co-training for the text/visual feature split with 50 initial training images. Only a subset of classes are shown for each dataset. 44

4-5 A comparison of the different ways of measuring performance. Co-training was run for 100 iterations using 50 initial training images. The top row shows the mean average precision scores. The bottom row shows the average maximum precision scores for the same data. The columns show the results for each dataset. 45

5-1 Some examples of rated training set images for the Pascal VOC’07 airplane class in increasing order of difficulty. 50

5-2 Some examples of rated training set images for the Pascal VOC’07 person class in increasing order of difficulty. 51

5-3 Some examples of rated training set images for the Pascal VOC’07 bird class in increasing order of difficulty. 52

5-4 The effect of training image selection on performance for an increasing number of training images. The “easy”, “medium” and “hard” plots are the rated training sets, while the “all” plot is for the full unrated training set. 53

5-5 Co-training with the initial training images drawn from one of 3 sets of images: easy, medium or hard. 30 initial training images were used in each case for Pascal VOC’07 while 50 were used for MIR Flickr. Results shown are the average maximum precision for the visual classifier only. The “all” image set corresponds to the full unrated set of images. 56

5-6 Co-training with the initial training images drawn from one of 3 sets of images: easy, medium or hard. 30 initial training images were used in each case for Pascal VOC’07 while 50 were used for MIR Flickr. Results shown are the average precision for each class. 57

6-1 Sample dendrograms produced by the CMIM clustering algorithm on the Pascal VOC’07 dataset using nine feature types as input. The clusters merged first are those on the left of the diagram and the clusters merged last are on the right. The x-axis shows the CMIM distance between each pair of merged clusters. 60

6-2 MAP test scores for an SVM classifier trained using each feature type individually. The full training set was used in each case. 63
6–3 Co-training with full disjoint feature sets with $N = 2$ clusters. Self-training with the combined visual features is included for comparison purposes. .......................... 65

6–4 Co-training with full disjoint feature sets with $N = 3$ clusters. Self-training with the combined visual features is included for comparison purposes. .......................... 66

7–1 Co-training with the feature splits chosen by CMIM clustering for an increasing number of classifiers. The results for including text features are shown on the left side, while those which consider only visual features are shown on the right side. 68

7–2 Co-training with the feature splits chosen by CMIM clustering for an increasing number of classifiers. Performance scores are the average maximum precision for 100 iterations for Pascal VOC'07 and 50 iterations for MIR Flickr. ........... 69

7–3 Self-training performance with each of the feature types used in the experiments. ................................. 72

7–4 Co-training performance with the bottom-up approach for an increasing number of classifiers. The results for including text features are shown on the left side, while those which consider only visual features are shown on the right side. .. 74

7–5 Co-training with the bottom-up approach for an increasing number of classifiers. Performance scores are the average maximum precision for 100 co-training rounds for Pascal VOC'07 and 50 co-training rounds for MIR Flickr. ........... 75
Chapter 1
Introduction

It is well known in the computer vision community that a key to achieving good classifier performance is to collect a large set of labeled images that densely sample the appearance manifold of the target object. However, in practice this is seldom possible since labeling a sufficiently large set of images can be prohibitively expensive in terms of time and effort. While certain collaborative efforts have taken place to provide a large labeled set of freely available images, e.g. [16, 49, 57], these sets often tend to be quite imbalanced, containing a large number of images for certain objects, such as people and cars, and few images for others. Furthermore, there are many cases where the problem is so specific that a generic object recognition dataset is not helpful, such as detecting cars with a fixed-viewpoint camera or managing personal photo collections. To overcome this, researchers often turn to semi-supervised learning approaches to reduce the amount of labeling required. Unlike supervised learning which requires training on a complete labeled dataset, semi-supervised learning only requires a small initial set of labeled samples for training, and then uses a set of unlabeled samples in an intelligent manner to boost classification performance.

One popular way of accomplishing this is to use the Expectation Maximization (EM) algorithm, which was introduced by Dempster et al. in [15] and made popular by Nigam et al. in the area of information retrieval in [39]. Given a generative model of the data and a set of unlabeled observations, EM seeks a maximum a posteriori (MAP) estimate of the distribution of the model parameters. Once these parameters are learned, the model is used to predict
the unlabeled observations. This presents two main problems when applied in
the context of object recognition. First, it requires a generative classification
model which usually tends to trail behind discriminative models in terms of
classification performance. Second, it requires that the unlabeled data respect
the class boundaries of the labeled data, an assumption which often does not
hold for object recognition tasks where the target object class is lost amid the
myriad of background objects.

Co-training is an alternative approach which does not require a generative
model and has been shown to outperform EM in the cases where two classifiers
are used and there is a natural independent feature split [40]. Essentially it
involves training a pair of classifiers with differing views of the target data and
allowing each one to train the other with the images that it can confidently
label. This approach is naturally suited to the situation where several feature
sets are readily available, as will be the case here. In this case a separate
classifier can be trained for each feature set to form differing views of the
training data. While it is tempting to simply combine the views from the
beginning and do self-training on the unlabeled examples, there is significant
advantage to resist doing so since having several specialized classifiers has the
potential to add much more information to the system.

For co-training to be effective, Blum and Mitchell provide a set of under-
lying assumptions that must first be met [6]. These are:

1. Each view must be sufficient for correct classification.
2. The views must be conditionally independent given the class label.
3. The target class must learnable with random classification noise.

Of these three requirements it is the second one that is the most difficult to
achieve in practice and thus, is most often overlooked. Choosing the two views
is often done in an ad-hoc manner without explicitly considering their conditional independence [34, 48, 9]. Choosing the right feature split to maximize the independence is a topic that is also largely unexplored. In addition, almost all of the literature on co-training focuses on the case where just two views are used.\(^1\) The focus of this thesis is to investigate the case where several features sets are available and several views are used.

### 1.1 The Problem

There are two main problems that will be addressed in this thesis. The first one regards the optimal division of features sets and can be formally defined as follows. Given a choice of \(M\) feature sets and \(N\) classifiers, how can we best divide the feature sets among classifiers so that the performance of co-training is maximized? There are two considerations to keep in mind when making this choice. One is the strength of the individual features themselves and the other is the conditional independence of the features given the class label. These stem from the key assumptions suggested by Blum and Mitchell and listed earlier. Ideally we would like both the strength of each classifier and their conditional independence to maximized but practically there may have to be some trade-off between the two.

The second problem is related to the first and involves finding the optimal choice of \(N\), given that we know the optimal division of features for any choice of \(N\). For the same reason that two classifiers are better than one when it comes to co-training we might reason that having more than two classifiers would improve the results even further.

A third problem that will be addressed is that of selecting the training images so that co-training performance is maximized. Although this problem

\(^1\) The only exception to this was found in [18] and will be discussed later.
is not directly related to the first two, it is of practical interest for our system and may have implications that are relevant for other semi-supervised learning techniques.

1.2 Approach

To solve the problem of feature assignment we use two separate approaches. The first tries to maximize the conditional independence between the classifiers and the second tries to maximize the strength of each classifier.

The first approach is the most elegant of the two and employs a hierarchical clustering algorithm to assign features to one of $N$ groups based on the amount of conditional independence that exists between them. This results in $N$ clusters or sets which are maximally independent given the class label, but may not be balanced in terms of strength. A separate classifier is then trained using each feature set and co-training proceeds. This approach assumes that all feature sets are disjoint and no feature set is discarded.

The second approach involves ranking all feature sets by their performance on a test set and greedily picking the one which gives the highest test accuracy. Thus for $N$ classifiers, the top ranking $N$ feature sets will be selected, a classifier will be trained for each and co-training will proceed. Since this approach discriminates against feature sets which do not perform well, it has the potential to improve performance over the case where all feature types are considered.

Once the feature sets have been defined for each choice of $N$, the task remains to observe and analyze the effect that increasing the number of classifiers has on performance.

Performance will be evaluated using the Pascal VOC 2007 [19] and MIR Flickr [30] datasets. These datasets are both obtained from Flickr and include annotation in the form of image tags supplied by users. We will explore two
Figure 1–1: Overview of the semi-supervised setting. Each classifier is trained on a small set of labeled images and is then bootstrapped using an additional set of unlabeled images. The goal is to predict the class labels for the test images. Two settings are shown here, one where image tags are included in the training phase and one where they are not. In either case it is assumed that image tags are not available at test time.

settings, one where the image tags are not used, and one where they are used (Figure 1–1). In both settings it will be assumed that image tags are not available at test time and can only be used for training.

We find that when image tags are excluded, sub-dividing the pool of features into smaller disjoint sets and creating separate views for each tends to improve classification accuracy over the case where all features are combined, and that accuracy generally increases with the number of views. Conversely, when image tags are included, we find that accuracy generally decreases with the number of views for reasons which will be later discussed.

1.3 Contributions

The main contributions of this thesis are summarized as follows.

- A known clustering algorithm [38], originally intended for feature selection with a single feature type, is employed and extended to allow the grouping of several feature types into sets that are conditionally independent given the class label.
• The common paradigm of co-training with two views of the data is extended to the situation where multiple views are used and the effect on performance is observed.

• A novel kernel reduction technique is introduced that reduces the computational complexity of co-training with support vector machines by eliminating redundant kernel computations.

• Experiments are carried out that provide insight on how to select the initial set of labeled training images so that co-training performance is maximized.

1.4 Outline

This thesis will be organized as follows. In Chapter 2 we will discuss related work and introduce the background material necessary for understanding the main matter of the thesis. In Chapter 3 we will explain the methods used in the experiments. Chapter 4 will deal with selecting the parameters for the co-training algorithm. Chapter 6 will detail the results of feature assignment using the hierarchical clustering algorithm. Chapter 7 will present the results of co-training as the number of classifiers increases. Finally, in Chapter 8 we will discuss the results and point to areas of future work.
Chapter 2
Background

In this chapter we discuss related work and provide the background information necessary for understanding the methods used in this thesis. We start by reviewing related work on co-training using multi-modal features as it pertains to image classification. Then we provide some brief background information on support vector machines and information theory as they are utilized by our approach.

2.1 Related Work

2.1.1 Co-training

The features used to represent an object have a major impact on how well a particular object recognition algorithm will perform. Some objects are more easily recognized by their texture, while others are more easily recognized by their shape, or colour. In many cases, two different types of features can lead to similar performance at predicting the target label but have very different outputs for a given set of images, thus having a high conditional independence given the target label. Co-training is naturally suited to this type of situation since there is a natural independent feature split available in the data. It exploits this feature division in the learning stage to boost performance using a set of unlabeled data.

The algorithm consists of first training a pair of classifiers with differing views using a small selection of labeled training data. In most cases this is
done by using different features for the same set of data\textsuperscript{1}. For example, an image could be represented using either texture or colour information. After training, each classifier is used to label a small number of unlabeled samples for which it has the most confidence, and add them to the training set. The classifiers are then re-trained using the updated training set and the process repeats. The samples which are most informative are those which would have been classified correctly by one classifier but misclassified by the other.

The earliest work on co-training was done by Blum and Mitchell in [6] with the objective of web-page classification. The task was to differentiate between Computer Science course home pages and other web sites within the same department. A Naive Bayes classifier was trained on keywords from the home pages themselves and another was trained on pages with hyper-links to the home pages. They demonstrated that co-training is effective at reducing the error rate given these two classifiers and provided a set of underlying assumptions that must be met for co-training to be effective in general (listed in chapter 1). Their algorithm forms a fundamental building block for this thesis and will be discussed more in Section 3.4.

Subsequently, the co-training algorithm has been applied successfully in several different domains, such as detecting cars in traffic-cam video [34], people detection [48], speech and gesture recognition [10], action recognition [28], image annotation [24, 23] and image classification [9, 27]. Image classification will be the focus of this thesis and in the next few sections we will review, in more detail, some of the co-training methods which pertain to this topic. We will consider the methods which employ textual information separately from

\textsuperscript{1} It is also possible to create different views by using different classifiers for the same set of features as in [60].
those that do not. We will start with some brief background information on
textual features and review their use in the literature. Following this, we will
discuss approaches to co-training which use visual information alone. Finally
we will briefly discuss the possibility of extending to more than two views and
review related work in this area.

2.1.2 Combining Textual and Visual Information

Almost all images found on the internet are accompanied by some form
of weak annotation. Examples of such annotation include image meta-data
such as the filename and directory, the words surrounding the image link and
Flickr image tags. This type of information is usually noisy for a couple of
different reasons. First, the text describing an image can refer to its contents
in an abstract way rather than explicitly referring to some element in the
scene, i.e. a person can be labeled as a monkey if they are exhibiting behavior
that is “monkey-like.” Second, the labels can be ambiguous, i.e. a mouse can
refer to either a small animal or a computer input device (see Figure 2–1 for
some examples). Despite these issues, many have found that using textual
information in conjunction with visual information is beneficial because these
two information sources are naturally independent [31, 3, 35, 27, 58, 36].

![Figure 2–1: Some example training images shown with their associated
Flickr image tags and class labels.](image-url)
Many of the methods which employ these two modalities have automatic data collection as a primary objective, and use a web search engine such as Google as a source of images. Since the images returned from a typical web search are often inaccurately labeled, machine learning techniques are used to improve on the results. In [5], a labeled face dataset was automatically created from captioned images gathered from Yahoo news. A face detector was used to extract faces from news images and the potential names were extracted from the associated captions. A clustering procedure was then used to resolve ambiguously named faces, thereby removing some of the label noise. A more general approach was taken in [51], with the objective of automatically building a dataset for any given object category using a Google web search. A Naive Bayes classifier was first trained on textual information and used to rank the search results for a given object. An SVM classifier was then trained on visual information using the highest ranked images from the text classifier and used to re-rank the entire set of downloaded images.

The same two modalities have also been used for co-training, with the objective being to reduce the amount of labeling required for a given performance level, rather than to collect a large dataset of labeled images. The earliest known example of this was applied to image annotation and is found in [23]. Here, two SVM classifiers were trained, one with text features and another with visual features. The text features were similar to those used in a typical image search (i.e., the file name, page title, and text surrounding the image), while the visual features consisted of colour histograms in conjunction with adaptive texture features [53]. The dataset consisted of a set of 5418 images downloaded from Google images for 15 different concepts, with an average of 216 training images per concept class. A variation to the original co-training algorithm based on “agreement” of the two classifiers was used.
following [11], and confidence rules were included that had to be satisfied for each view in order for labeling to occur. This reduces the chance of mislabeling but also adds complexity since these rules must be estimated using a validation set, and possibly updated throughout the learning process. It was shown that co-training with 50 initial seed images lead to comparable results to full supervision but with much fewer training examples.

The approach in [27] is most similar to ours and forms a starting point for this thesis. It employs several types of visual features along with text features in the form of Flickr image tags to improve image classification. A single visual SVM classifier was trained on a combination of 15 different types of image features, and a separate text classifier was trained using Flickr image tags. Performance was evaluated on the Pascal VOC 2007 and MIR Flickr datasets using the visual classifier alone since it was assumed that image tags are not available at test time. Co-training was used here only as a benchmark to compare with a regression method which appeared to have better performance, but the results are useful nonetheless. We extend the work in this paper by dividing the visual features into smaller subsets and forming a separate visual classifier for each.

2.1.3 Using Separate Visual Modalities

While it is true that textual information is easily obtained for images found on the internet, this is not the case for other types of images such as personal photos. Thus, it is of practical interest to also investigate the case of co-training with only visual information.

One example of this is found in the co-SVM algorithm of [9], wherein separate visual classifiers are used to improve image classification in a semi-supervised manner. One classifier was trained on colour histogram features and another was trained on texture features extracted using a multi-level discrete
wavelet transform. Each of these classifiers was used to label a set of unlabeled images and those which the classifiers disagreed on were presented to the user for labeling. All labeled examples were then added to the training set and the process was repeated.

Another example is found in [24] and applies to the task of semi-supervised image annotation. This time co-training is applied at the region level instead of at the image level. Images were first segmented into regions using two separate algorithms, Blobworld [7] and JSEG [17]. Colour, texture and shape features were extracted for each region and fed to a pair of independent SVM classifiers, with the first trained on colour features and the second on texture and shape features. The confidence scores for these two classifiers were used to jointly determine the most likely label for each unlabeled region and as before, the users were asked to label regions where the classifiers were in disagreement. The resulting set of labels for each region in an image were combined using a decision tree to produce the final image annotation.

Both of the above methods achieved significant performance gains with co-training but the drawback is that they can require a great deal of supervision from the user if the classifiers do not perform well. Ideally, we would like to limit the amount of supervision so that user intervention is not required after the initial training phase.

2.1.4 Extending to Multiple Views

While the prospect of co-training with more than two views was mentioned by Blum and Mitchell in [6], there is only a scant amount of information on this topic in the literature. The only example found by the author is in [18]. Rather than use a different feature set for each classifier, this method uses the same feature set for each and uses different types of classifiers to create the differing views as in [60]. Three classifiers were trained and tested on several
datasets from the UCI machine library repository. The chosen classifiers varied depending upon the actual dataset used since an effort was made to keep the compatibility or the amount of agreement between classifiers above 90 percent. No firm conclusions were drawn from this other than to say that co-training with multiple views allows exploiting the unlabeled data in a useful manner.

Although not a co-training example, we are shown evidence in [4] that combining multiple visual cues along with textual information can be beneficial in a supervised learning scenario. The task in this case was to gather a collection of animal images from the web using the Google search engine as a primary source of information. Text features consisted of the words surrounding the image link and visual features were chosen based on shape, colour and texture. A separate classifier was trained for each feature type and used to score the set of collected images. The overall score for each image was then obtained by averaging the individual feature scores. It was shown that combining all feature types in this manner lead to significant performance gains over the case where each feature type was considered independently.

2.2 Support Vector Machines

Since we plan to use a support vector machine (SVM) as the underlying classifier in this thesis, some understanding of the theory behind this technique is required. A support vector machine is a binary classifier that seeks a separating hyperplane \( f(x) \) between a set of labeled data \( (x_i, y_i) | y_i \in \{-1, +1\} \) by maximizing the margin, or the distance between the decision hyperplane and the nearest correctly-classified sample on either side. Maximizing the margin in this manner leads to improved generalization ability over a linear classifier since the training samples are kept further from the decision boundary. One other major advantage that an SVM has over other types of classifiers is
its unique ability to generate non-linear decision boundaries using a method known as the kernel trick [2].

The decision hyperplane can be expressed as the set of points that satisfy \( w \cdot x_i - b = 0 \) and the hard constraint that we would like to impose on the data is as follows (see [12] for further explanation).

\[
y_i(w \cdot x_i - b) \geq 1, \forall i \in \{1, \ldots, n\} \tag{2.1}
\]

with the weights \( w \) and \( b \) to be determined. This ensures that all positive samples \( x_i \) will lie above the hyperplane \( w \cdot x_i - b = 1 \) and all negative samples will lie below the hyperplane \( w \cdot x_i - b = -1 \) assuming separable data. It can be shown using geometry that the distance between these two planes is \( \frac{2}{\|w\|} \). Thus maximizing the margin amounts to the following optimization problem for linearly separable data:

\[
\min_{w,b} \|w\| \tag{2.2}
\]

subject to the constraint

\[
y_i(w \cdot x_i - b) \geq 1
\]

This optimization problem is difficult to solve since \( \|w\| \) involves a square root term. Fortunately, it is possible to alter the original equation by replacing the \( \|w\| \) by \( \frac{1}{2} \|w\|^2 \) without changing the solution (with the \( \frac{1}{2} \) being used only for mathematical convenience). With this substitution the problem can be readily solved using quadratic programming techniques.

2.2.1 Dual Form

Writing the optimization problem in its dual form reveals that the maximum margin hyperplane is only a function of its support vectors, the training
samples that lie on the margin. Using the fact that $w = \sum_{i=1}^{n} \alpha_i y_i x_i$, it can be shown that the dual form of the problem in equation 2.2 can be written as:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j k(x_i, x_j)$$

subject to

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

and

$$\alpha_i \geq 0, \forall i \in \{1, \ldots, n\}$$

The kernel function $k(x_i, x_j)$ defines an inner product between feature vectors $x_i$ and $x_j$ in an induced feature space and can be interpreted as a measure of similarity between two vectors. We have used the linear kernel in equation 2.3 as defined by $k(x, x_i) = x_i \cdot x_j$, but many other non-linear kernels can also be defined. We mention only the radial basis function (RBF) kernel defined by $k(x, x_i) = \exp(-\gamma \|x - x_i\|^2)$, as one example of a non-linear kernel as it will be used later on.

The solution to the dual optimization problem in equation 2.3 is a classification function of the form:

$$f(x) = \sum_{i} \alpha_i y_i \cdot k(x, x_i) - b$$

(2.4)

With this defined, the class label $y \in \{-1, +1\}$ for any given test sample can now be predicted as $\text{sign}(f(x))$.

### 2.2.2 Soft-Margin SVM

The SVM formulation has the drawback that it can only be used on data that is separable. This is unrealistic for many problems where adversarial data exists and no clean decision boundary can be found. For this reason
an alternative version was presented in [12] which accounts for misclassified examples by introducing the slack variables $\xi_i$ which measure the degree of misclassification for the data $x_i$. With these variables the primal form of the optimization problem becomes

$$\min_{w, \xi} \left\{ \frac{1}{2} \|w\|^2 + C \cdot \sum_{i=1}^{n} \xi_i \right\}, \quad (2.5)$$

subject to the constraints

$$y_i(w \cdot x_i - b) \geq 1 - \xi_i \text{ and } \xi_i > 0, \forall i \in \{1, \ldots, n\}$$

The solution is a decision hyperplane which maximizes the margin for correctly classified samples while minimizing the error penalty of the misclassified samples.

The variable $C$ is a cost parameter that determines how heavily the misclassified samples are weighted. This can be used in conjunction with a weighting factor $\rho$ to form a prior that assigns the cost of misclassification separately for each class.

$$\rho = \begin{cases} w_{pos} & \text{if } y_i = 1 \\ w_{neg} & \text{if } y_i = -1 \end{cases} \quad (2.6)$$

This is especially useful in situations where there is a highly unbalanced class distribution and one does not want to give undue favor to the class with the higher population.

Since the soft-margin SVM is used exclusively throughout this thesis we will simply use the term SVM to refer to the soft-margin SVM.

2.3 Information Theory

We describe some key concepts relating to information theory as these will be used in the material that follows. Information theory deals with the
quantification of information and has broad application in many areas including data analysis, as it will be applied here. Fundamentally, it uses the notion of entropy to describe the amount of information or uncertainty contained in a signal.

2.3.1 Shannon Entropy

Given a discrete random variable $X$ with possible values $\{x_1, x_2, \ldots, x_n\}$, the Shannon Entropy $H(X)$ describes the amount of information contained in the signal $X$ and is given by the following equation.

$$H(X) = -\sum_{x \in X} p(x) \log p(x) \quad (2.7)$$

This equation can also be interpreted to represent the average amount of uncertainty in $X$, with the uncertainty being expressed as $\log p(x)$. When all possible discrete values occur with equal probability then the average uncertainty is maximized, whereas if we can predict with certainty that the $X$ will take on a specific value then the average uncertainty is minimized.

2.3.2 Mutual Information

The mutual information $I(X;Y)$ is known as the Kullback-Liebler divergence $^{13}$ between the joint distribution $p(x,y)$ and the product distribution $p(x)p(y)$ and is given by the following formula.

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left( \frac{p(x,y)}{p(x)p(y)} \right) \quad (2.8)$$

This describes the information that is shared between random variables $X$ and $Y$. It can also represent the amount of information that one variable can predict about the other. The mutual information becomes zero when $X$ and $Y$ are independent and reaches a maximum when the two variables are identical.
2.3.3 Conditional Mutual Information

In general terms, conditional mutual information (CMIM) describes the expected value of the mutual information of two random variables given a third. Assuming the random variables $X_1$, $X_2$, and $Y$, this can be written as follows.

$$I(Y; X_2 | X_1) = \sum_{x_1 \in X_1} \sum_{x_2 \in X_2} \sum_{y \in Y} p(x_1, x_2, y) \log \frac{p(x_1)p(x_2, y|x_1)}{p(x_1, y)p(x_2|x_1)}$$ (2.9)

If $Y$ is the target variable that we are seeking information about, then we can interpret the above equation as describing the amount of additional information that $X_2$ can provide about $Y$ that cannot be provided by $X_1$. We will refer to this in a later section as we describe our method for grouping feature types.
Chapter 3
Methods

Our investigation into co-training with multi-modal features requires having a set of $N$ classifiers and $M$ subsets of features, where the numbers $N$ and $M$ can be arbitrarily chosen, and the feature subsets can be arbitrarily assigned. The goal is to assign feature sets to the classifiers in such a way that co-training performance is maximized. This chapter will outline the datasets, features, classifiers and methods that will be used to accomplish this goal. First, we describe the datasets and multi-modal features that will be used and outline the feature definitions used in our approach. The details of our chosen classifier and co-training algorithm will then be provided. A kernel reduction technique will then be described which reduces the running time of our algorithm. Next, we describe an approach to automatically assign feature types to classifiers using clustering. Finally, we describe our method for benchmarking the co-training performance with different feature combinations.

3.1 Datasets

For our experiments we will use the Pascal VOC 2007 [19] and MIR Flickr [30] datasets, both of which contain images downloaded from the Flickr website. Sample images for each are shown in Figure 3–1. The details of each of these datasets are described below.

- **Pascal VOC 2007** - A set of 10000 images which were downloaded from Flickr by querying for images from 20 different object categories. All the images were then manually annotated for these object categories. Approximately 5000 images are used for testing and the other 5000 for training. The user tags for the 9587 images that were still available on
Flickr were downloaded. The tags that appear less than 9 times were discarded, leaving 804 tags in total.

- **MIR Flickr** - A set of 25000 images collected by downloading images from Flickr over a period of 15 months. Contains images that scored highest according to Flickr’s "interestingness" score. The images were annotated in two ways: first, a set of 24 "potential" labels were chosen which mostly represent object categories, but also general scene elements such as sky, water and sunset. Each image received a potential label if the label was deemed to apply in any way possible. Second, a set of 14 "relevant" labels were chosen, where a single annotator applied a label to an image only if the image matched his specific interpretation of the label. The tags that appeared at least 50 times were kept, resulting in a vocabulary of 457 tags. Train/test splits vary depending on the
In this thesis we use a random split of 12500 training images and 12500 test images.

3.2 Features

The features used in these experiments are the same as those used in [27, 26] and were downloaded from the LEAR website. These include the Flickr image tags already described, visual word histograms, colour histograms in RGB, HSV and LAB colour spaces, and the Gist descriptor described by Torralba in [42].

The visual word histograms were constructed using SIFT [37] and hue SIFT [56] descriptors. This was done both on regions found by a Harris-Laplace local interest-point detector and on a dense multi-scale grid. The local descriptors were then quantized using k-means and a histogram was formed for each image by collecting quantized descriptors, creating a bag-of-features image representation.

In addition to using the whole image for feature extraction, the histogram-based representations were also computed using a 3x1 horizontal decomposition as in [33] and concatenated to form a new representation that encodes some of the spatial layout of the scene.

In total, 15 image representations were used which can be grouped into categories according to whether they employ colour or grayscale information and whether they use local or global descriptions of the image. In order to reduce the total number of features, and thus the complexity of the experiments we use the feature grouping shown in Table 3-1.

---

1 We use the same train/test split as in [27], as this was made available on the LEAR website.

2 http://lear.inrialpes.fr/people/guillaumin/data.php
<table>
<thead>
<tr>
<th>Feature Groups</th>
<th>Feature Types</th>
<th>Tags</th>
<th>Harris SIFT</th>
<th>Dense SIFT</th>
<th>Harris Hue SIFT</th>
<th>Dense Hue SIFT</th>
<th>RGB</th>
<th>LAB</th>
<th>HSV</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Text</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Visual Combined</td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>SIFT</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Harris SIFT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Dense SIFT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hue SIFT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Colour Histograms</td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gist</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>•</td>
</tr>
<tr>
<td></td>
<td>Colour</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Grayscale</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td>•</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Local</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Global</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random 2 Set 1</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random 2 Set 2</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random 3 Set 1</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random 3 Set 2</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Random 3 Set 3</td>
<td></td>
<td></td>
<td></td>
<td>•</td>
<td></td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3-1: Feature definitions to be used in the experiments. Individual feature types are in columns and feature group definitions are in rows. For simplicity, the 3x1 horizontal decompositions are not shown since they are always grouped with the 1x1 image decompositions of the same feature type.
3.3 Classifier

An Support Vector Machine is used as the classifier for all experiments, since it gives excellent performance and is widely used [20]. Following [27], we define two different types of classifiers: one based on text features and the other based on visual features. Each classifier learns a function of the form shown in Equation 2.4, where the kernel varies depending on the type of features used.

The text classifier \( h_t \) is trained on Flickr image tags and uses a linear SVM kernel \( k_t \) to compare samples \( x_i \) and \( x_j \). Image tags are represented using a binary vector \( t_i = \{0, 1\}^W \) which encodes the presence or absence of each of the \( W \) words in the dictionary. The inner product defined by the linear kernel is the standard dot product and counts the number of common tags between two binary vectors.

\[
k_t(x_i, x_j) = t_i \cdot t_j
\]

The visual classifier \( h_v \) is trained on a combination of visual feature types, with the feature groupings defined in Table 3.1 (second row). In this case, we use a radial basis function (RBF) kernel \( k_v \) as defined below.

\[
k_v(x_i, x_j) = \exp\left(-\frac{d(x_i, x_j)}{\lambda}\right)
\]

\[
\lambda = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} d(x_i, x_j)
\]

Each feature pair is normalized by \( \lambda \), the average pairwise distance for the set of \( N \) samples, as shown in Equation 3.2. The distance \( d \) is calculated by summing the normalized distances for each of the \( M \) feature types using \( d(x_i, x_j) = \sum_{m=1}^{M} \frac{d_m(x_i, x_j)}{\lambda_m} \), where \( \lambda_m = \max_{i,j} d_m(x_i, x_j) \). The individual feature distances
$d_{mn}$ are calculated differently depending on the type of feature being used. We use the $L_1$ distance for colour histograms, $L_2$ distance for Gist and $\chi^2$ distance for the visual word histograms as in [27].

To benchmark the performance of the classifiers, we train one text classifier and one visual classifier as described above, and test on both the Pascal VOC'07 and MIR Flickr datasets. For the visual classifier, we combine all visual feature types as indicated in Table 3–1, while the text classifier uses only the image tags. We train on the full training set and report the average precision (AP) scores for each class as well as the mean average precision (mAP) score for the entire set of classes. The SVM is implemented using LibSVM with the cost parameter $C$ set to 10. The class weights for the positive and negative classes are set to $w_{pos} = \frac{n_{pos}+n_{neg}}{n_{pos}}$ and $w_{neg} = \frac{n_{pos}+n_{neg}}{n_{neg}}$ respectively, where $n_{pos}$ is the number of positive training images and $n_{neg}$ is the number of negative training images (see Equation 2.6). The results are shown in tables 3–2 and 3–3 and are consistent with those in [27].

<table>
<thead>
<tr>
<th>Pascal VOC'07</th>
<th>aeroplane</th>
<th>bicycle</th>
<th>bird</th>
<th>boat</th>
<th>bottle</th>
<th>bus</th>
<th>car</th>
<th>cat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>0.665</td>
<td>0.411</td>
<td>0.598</td>
<td>0.388</td>
<td>0.203</td>
<td>0.281</td>
<td>0.505</td>
<td>0.666</td>
</tr>
<tr>
<td>Visual</td>
<td>0.722</td>
<td>0.534</td>
<td>0.493</td>
<td>0.665</td>
<td>0.253</td>
<td>0.531</td>
<td>0.608</td>
<td>0.498</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>chair</th>
<th>cow</th>
<th>diningtable</th>
<th>dog</th>
<th>horse</th>
<th>motorbike</th>
<th>person</th>
<th>pottedplant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>0.152</td>
<td>0.435</td>
<td>0.077</td>
<td>0.569</td>
<td>0.669</td>
<td>0.542</td>
<td>0.632</td>
</tr>
<tr>
<td>Visual</td>
<td>0.464</td>
<td>0.149</td>
<td>0.447</td>
<td>0.419</td>
<td>0.740</td>
<td>0.588</td>
<td>0.333</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sheep</th>
<th>sofa</th>
<th>train</th>
<th>tvmonitor</th>
<th>Average</th>
<th>[27]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td>0.460</td>
<td>0.186</td>
<td>0.710</td>
<td>0.275</td>
<td>0.434</td>
</tr>
<tr>
<td>Visual</td>
<td>0.390</td>
<td>0.395</td>
<td>0.713</td>
<td>0.432</td>
<td>0.529</td>
</tr>
</tbody>
</table>

Table 3–2: Results for the Pascal VOC'07 dataset using text and visual classifiers. Each classifier is trained using the full training set and results are reported for each class using average precision.
Table 3–3: Results for the MIR Flickr dataset using text and visual classifiers. Each classifier is trained using the full training set and results are reported for each class using average precision. The classes marked with * are the relevant class labels and those without are the potential class labels. Relevant labels correspond to a specific interpretation of a concept from a single annotator, while potential labels can apply to the image in a more general sense.

3.4 Co-training with N Classifiers

For our experiments, we will use an extension of the original co-training algorithm described in [6] that allows it to work with more than just two classifiers. We assume that a set of $M$ feature sets exists for the set of input images and is denoted by $T = \{T_1, T_2, ..., T_M\}$. Moreover, we assume that the $M$ feature sets are divided into $N$ disjoint subsets $S = \{S_1, S_2, ..., S_N\}$, where $S_i \subseteq T$. These feature subsets are chosen so that textual features are treated separately from visual features and are never combined. Before we proceed to the description of the algorithm, a set of labeled training examples $\mathcal{L}$ and a set of unlabeled examples $\mathcal{U}$ must first be chosen. In this case $\mathcal{L}$ is chosen by randomly selecting a small subset of the training images and the remaining
Algorithm 3.1 Co-training with N Classifiers

Given:
A set \( L \) of labeled training examples
A set \( U \) of unlabeled examples
A set \( T \) of feature sets \( \{T_1, T_2, ..., T_M\} \)
A set \( S \) of disjoint feature subsets \( \{S_1, S_2, ..., S_N\} \), where \( S_i \subseteq T \).

Create a pool \( U' \) of examples by choosing \( u \) examples at random from \( U \)

Loop for \( k \) iterations:
   For \( i \) in \( 0, 1, ..., N \)
      - Use \( L \) to train a classifier \( h_i \) that considers only the \( S_i \) portion of \( T \)
      - Allow \( h_i \) to label \( p \) positive and \( n \) negative examples from \( U' \)
      - Add these self-labeled examples to \( L_1 \)
      - Add \( L_1 \) to \( L \)
      - Randomly choose \( N(p+n) \) examples from \( U \) to replenish \( U' \)

images are assigned to \( U \). In addition, a smaller pool of unlabeled images \( U' \) is randomly selected from \( U \), to be used as the active set to be labeled\(^3\).

The algorithm proceeds by training a set \( \{h_1, h_2, ..., h_N\} \) of SVM classifiers from \( L \) using the feature subsets already defined. For example, classifier \( h_i \) would be trained using feature subset \( S_i \). We use the linear kernel for textual features and the RBF kernel for visual features as defined in Equation 3.1. After training, each classifier scores the set of images in \( U' \) and selects the ones corresponding to the \( p \) highest scores and adds them to \( L \) with a positive label. Likewise, the images corresponding to the \( n \) lowest scores are labeled negative and added to \( L \). The active set \( U' \) is then replenished with \( N(p+n) \) randomly chosen images from \( U \) and the process repeats. This is formally outlined in Algorithm 3.1 and illustrated in Figure 3-2.

Besides the main novelty of enabling co-training with more than two views, this algorithm is the same as the the one used in [6]. The only other

\(^3\) Blum and Mitchell indicate that restricting the labeling to the active set leads to better performance over the case when all unlabeled images are considered at each iteration. The reason for this is not entirely clear.
**Figure 3–2:** Overview of co-training with N classifiers. Solid lines represent the training phase and dashed lines represent the labeling phase. Classifier $h_t$ is trained on image tags while classifiers $h_v$ are trained on visual features. After the classifiers are trained using the labeled image features, each classifier is used to label additional images from the unlabeled set. These images are then added to the training set and the process repeats.

A minor difference is that we allow $N(p + n)$ images to be added to the training set per iteration rather than $2(p + n)$ as was done before.

### 3.5 Kernel Reduction

Co-training can be a time-consuming process since it involves re-training a set of classifiers over several iterations of the algorithm, each time with a different selection of training images. The vast majority of this time is spent computing the SVM kernel for the set of training images. For example, to compute the training kernel as in Equation 3.1 for all pairs of training images can take well over an hour for the MIR Flickr dataset on a 3 GHz Intel I7 processor while finding the learning parameters typically takes under a minute. If this is to be done for several classes and over several iterations, running experiments can quickly become unfeasible.
Fortunately, many of these computations are redundant, which means that it should be possible to boost efficiency by eliminating them. The redundancy occurs because at each round of co-training, only a few images are added to the training set, while the SVM classifier must be re-trained with the entire set of training images. Ideally, we would like to compute the kernel between any given pair of images only once so that the running time is minimized. An algorithm which can learn incrementally would allow us to accomplish this since the classifier could then be updated one sample at a time, without having to re-learn previous information. Unfortunately, the SVM classifier is a batch learning algorithm by nature and does not allow this.

An incremental version of the SVM learning algorithm was presented in [8] and would have been suitable for our purposes. However, due to the complexity involved in implementing this, we opt for a simpler approach which does not involve changing the fundamental learning algorithm. Instead, it uses a store and load technique we call kernel reduction to eliminate redundant kernel computations. This technique computes the kernel matrix only once for all elements and forms a new reduced kernel matrix corresponding to the labeled portion of the data, by copying specific elements. Each time the labeling changes, a new reduced kernel is created by copying a different set of elements from the full kernel matrix.

In the co-training algorithm, there are three specific times when an SVM kernel matrix must be computed. Each of which happens at every iteration:

1. Training an SVM using a set of labeled images.
2. Labeling a set of unlabeled images using the trained SVM.

\footnote{A more exhaustive review of other methods for reducing the time complexity of SVMs can be found in [8].}
3. Testing performance of the SVM using a set of test images.

We will address how kernel reduction can be used in each of these situations in what follows. It should also be pointed out that the methods in [8] and [55] address only the training stage of an SVM classifier and thus, our method is more appropriate for the co-training situation. To the best of the author’s knowledge, techniques similar to this are not found in the literature.

As a precursor, it is assumed that the input kernel matrix $K_{\text{input}}$ has been computed for every possible pair of $m$ input images as follows. Note that no labeling is required at this stage. Let

$$K_{\text{input}} = \begin{pmatrix}
  k_{1,1} & k_{1,2} & \cdots & k_{1,m} \\
  k_{2,1} & k_{2,2} & \cdots & k_{2,m} \\
  \vdots & \vdots & \ddots & \vdots \\
  k_{m,1} & k_{m,2} & \cdots & k_{m,m}
\end{pmatrix} \quad (3.3)$$

likewise, assume that test kernel matrix $K_{\text{test}}$ has been computed for every possible pair of $m$ input images and $n$ test images as follows:

$$K_{\text{test}} = \begin{pmatrix}
  k_{1,1} & k_{1,2} & \cdots & k_{1,n} \\
  k_{2,1} & k_{2,2} & \cdots & k_{2,n} \\
  \vdots & \vdots & \ddots & \vdots \\
  k_{m,1} & k_{m,2} & \cdots & k_{m,n}
\end{pmatrix} \quad (3.4)$$

In the above equations, the kernel function $k_{i,j}$ can be either the text kernel $k_t$ or the visual kernel $k_v$ as described in the Section 3.3. We define $\mathcal{I}_{\text{input}} = \{1, 2, \ldots, m\}$ to be the indices of the input images, and $\mathcal{I}_{\text{test}} = \{1, 2, \ldots, n\}$ to be the indices of the test images.

We now consider the case where the input images are split into labeled subset $\mathcal{L}$ and unlabeled subset $\mathcal{U}$, where $\mathcal{L}$ and $\mathcal{U}$ consist of input images with index sets $\mathcal{I}_l$ and $\mathcal{I}_u$ respectively, with $\mathcal{I}_l \cup \mathcal{I}_u = \mathcal{I}_{\text{input}}$ and $\mathcal{I}_l \cap \mathcal{I}_u = \emptyset$. 

29
In the training situation, we require a kernel matrix of the form in Equation 3.3 as input to the SVM, but with the row and column indices drawn from $\mathcal{I}_l$ rather than from $\mathcal{I}_{\text{input}}$. We start with $K_{\text{input}}$ and copy the elements $k_{i,j}$ for which $i \in \mathcal{I}_l$ and $j \in \mathcal{I}_l$, producing a reduced training kernel matrix which is symmetric as shown in Figure 3–3a.

In the labeling situation, we rank a set of images from $\mathcal{U}$ according to their score as determined by the output of the SVM classifier. This requires a kernel matrix of the form shown in Equation 3.3, but with the row indices drawn from $\mathcal{I}_l$ and the column indices drawn from $\mathcal{I}_u$. We start with $K_{\text{input}}$ and copy only the elements for which $i \in \mathcal{I}_l$ and $j \in \mathcal{I}_u$, producing a reduced kernel matrix which is asymmetric as shown in Figure 3–3b.

In the testing situation, we require a kernel matrix of the form in Equation 3.4, but with the row indices drawn from $\mathcal{I}_l$ and the column indices drawn from $\mathcal{I}_{\text{test}}$. Note that the test images are never changed. We start with $K_{\text{test}}$ and create a reduced test kernel matrix by copying only the rows corresponding to $\mathcal{I}_l$ as shown in Figure 3–3c.

Once the initial kernel matrices $K_{\text{input}}$ and $K_{\text{test}}$ have been computed, the kernel reduction techniques described above take only a few seconds in comparison with an hour or more per kernel matrix when calculating each one from scratch. This means that the speed by which we can carry out experiments is greatly increased. However, this also means that the normalization factors $\lambda$ and $\lambda_m$ described in Section 3.3 are not the same as they would have been had the reduced kernels been constructed from scratch. This is because the copied elements were normalized using the full set of input images rather than the reduced subset $\mathcal{L}$. However, in theory this should not pose a problem, since one could argue that normalizing using a larger set of images allows the normalization factors to be more reliable.
Figure 3.3: The kernel reduction technique assuming a 8x8 full kernel matrix. The grid squares shown in grey are the kernel elements that are copied over to the new reduced kernel.
3.6 Feature Clustering Algorithm

The problem of optimally assigning features to classifiers for use in a co-training situation is one which has not received much attention in the literature. The only reference found by the author is in [40], wherein a strategy based on graph cutting is proposed to split a set of features into two sets which are conditionally independent, although no experiments were carried out to test the effectiveness of this strategy. In our case, the goal is to cluster a set of $M$ feature types $T = \{T_1, T_2, ..., T_M\}$ into $N$ distinct sets which are maximally independent from one another given the class label $Y$, where each feature type $T_i = \{\bar{f}_1, \bar{f}_2, ..., \bar{f}_{K_i}\}$ is itself a set of feature vectors. Clustering feature types based on conditional mutual information (CMIM) would seem like an appropriate way of accomplishing this since CMIM can be used to measure the amount of class-conditional independence between feature sets.

An approach which uses this strategy is found in [38] and applies to the problem of feature selection to improve classification for a single classifier. Here, the goal was to select from a set of $K$ features of a single type, the subset that retains the most amount of information about the class label. Hierarchical clustering is used to cluster the entire set of features according to a CMIM-based distance metric. Then, the feature cluster with the lowest amount of mutual information with respect to the class label is discarded, leaving the remaining features to be used for classification. We will extend this approach and apply it to the situation where several feature types are available and several feature clusters are to be obtained. In particular, we add a pre-processing stage which allows CMIM to be measured between feature types, where each feature type consists of a set of feature vectors.

We now provide the details of the pre-processing stage just mentioned. Since we want to cluster feature types and not the constituent features for
Algorithm 3.2 Feature Type Abstraction

Given:
A set of feature types $T = \{T_1, T_2, ..., T_M\}$

For each feature type $T_i$
- Train an SVM classifier $h_i$ on the full training set.
- Collect SVM output scores $\theta$ for the set of test images using $h_i$.
- Find the threshold $\lambda$ that balances the error rate on the test set using $\theta$
  (i.e., find $\lambda$ s.t. false positive rate = false negative rate).
- Binarize the test outputs to obtain $X_i = \{x_1, x_2, ..., x_n\}$
  where $x_k = \begin{cases} 
1 & \text{if } \theta_k > \lambda \\
0 & \text{otherwise}
\end{cases}$ for $k \in \{1, 2, ..., n\}$.

a single feature type, we first convert the feature types $T_i$ into meta-features $X_i$ that can be used as input to the clustering algorithm. This is done using the feature type abstraction method listed in algorithm 3.2. First an SVM classifier $h_i$ is trained for each feature type $T_i$ using the full training set. The classifier $h_i$ is then used to generate output scores $\theta$ for the set of test images. These scores are then binarized by finding the threshold $\lambda$ that balances the error rate between the positive and negative classes. This results in a single binary column vector for each feature type. Having the results binarized in this manner allows us to more easily measure the amount of shared information between a particular type of feature and the class labels, since the class labels are also binary.

Once this pre-processing stage has completed, we proceed to cluster the feature types using hierarchical clustering with a distance metric based on CMIM as in [38]. The only major difference is that we retain all clusters rather than discard the one with the lowest mutual information. We provide further details on this distance metric and on the clustering algorithm itself in the next sections.
3.6.1 CMIM-based distance metric

Referring to Equation 2.9, the conditional mutual information, \( I(Y; X_2|X_1) \), is the amount of information that \( X_2 \) provides about the class label \( Y \) given the information about \( Y \) that \( X_1 \) already provides. In order to perform clustering using this measure we must first convert it to a distance metric by ensuring that it fulfills all of the required properties, i.e., positive-definiteness, symmetry and sub-additivity. As it stands, it is not symmetric since \( I(Y; X_2|X_1) \neq I(Y; X_1|X_2) \), and thus it is not a distance metric. The symmetric form of equation 2.9 can be defined by reversing the inputs to \( I \) and summing as follows:

\[
D_{CM}(X_1, X_2) = I(Y; X_1|X_2) + I(Y; X_2|X_1)
\]

We refer the reader to [38] for proof that this equation also fulfills the other properties of a distance metric.

The normalized form of this becomes

\[
D_{CM1}(X_1, X_2) = \frac{D_{CM}(X_1, X_2)}{2H(Y)}
\]

where the \( H(Y) \) in the denominator is the entropy of the class label \( Y \) and constrains the function to have a range between 0 and 1. Again, refer to [38] for proof.

The distance metric shown in equation 3.5 can be interpreted to describe the amount of information that feature \( X_1 \) provides about the label \( Y \) that \( X_2 \) does not and vice-versa. This is described in [38] as relevant independence, the amount of information that two variables \( X_1 \) and \( X_2 \) can predict about \( Y \) that they do not share.

An ideal feature split is one with conditional independence for the two sets of features given the class label, i.e., \( P(X_1 \cap X_2|Y) = P(X_1|Y)P(X_2|Y) \).
While a proof is not provided here, it stands to reason that a feature split which provides maximum relevant independence as defined by the distance metric in Equation 3.6 is also one which has class-conditional independence. Thus, clustering using this distance metric should provide feature sets which are maximally independent from each other given the class label.

### 3.6.2 Hierarchical Clustering

Now that the distance has been defined we can describe how the features are clustered. We use agglomerative hierarchical clustering since it provides the most amount of information about the relative distance between each cluster and allows late selection of the number of clusters. Initially each of the feature types is assigned its own cluster. The process then continues by repeatedly selecting the pair of clusters with the smallest distance $D_{CM1}$ and merging them together to form a new cluster. Thus, the number of clusters is reduced by one at every iteration. The algorithm terminates when $N$ clusters are obtained.

We use Ward’s linkage [59] to compare one feature cluster to another as in [38] since it has the property of producing clusters with minimal variance. This defines the distance function

$$D(r, s) = \sqrt{\frac{2n_r n_s}{n_r + n_s} D_{CM1}(\bar{x}_r, \bar{x}_s)}$$

where $n_r$ and $n_s$ are the number of elements and $\bar{x}_r$ and $\bar{x}_s$ are the centroids for clusters for $r$ and $s$.

Once the clustering has completed, the feature set defined by each cluster will be used to train a separate classifier and co-training will proceed.

### 3.7 Measuring Performance

A common method of measuring performance for image classification is to use the mean average precision (MAP), or the mean of the average precision
scores of all classes evaluated; However, as we will see, placing too much emphasis on the MAP as a performance metric may be unfair when it comes to co-training, as the peak performance can occur at very different times from one class to the next and the MAP curves themselves can be somewhat erratic, being characterized by many sudden changes in performance. For some object classes the performance deteriorates while for others it is improved. This makes comparisons between different approaches difficult.

In a real-world system one would want to maximize the performance that can be achieved for each class by choosing the optimal training iteration for each, rather than choosing an arbitrary fixed iteration for all classes. This leads us to introduce a new metric called the average maximum precision (AMP), or the average of the maximum average precision (AP) score for each class across a given number of iterations. This is formally defined as follows.

$$AMP(k) = \frac{1}{C} \sum_{i=1}^{C} \max_{1 \leq j \leq k} AP(i)$$ \hspace{1cm} (3.7)

where $C$ is the number of classes and $k$ is the number of co-training iterations.

Using this metric tends to provide smoother looking performance graphs, but it does bias the results to the positive side somewhat as it implies that the performance for each class is lower bounded by the performance at the first iteration.

Another problem that must be addressed is that of comparing the results of co-training using different feature splits. If the feature splits were the same in each experiment we could simply plot performance for one or all feature sets. However, when the feature splits are different, the choice is not as clear. Whatever the choice may be, it must be consistent for all experiments.

We want to be sure that what we are measuring is the performance of co-training and not the strength of the feature sets themselves. For example,
consider a feature set which has a narrow view of the image content. It is possible for this feature set to perform well for co-training since it provides unique information, but poorly when considered on its own. Thus we must choose a feature set which allows a broad view of the image content and allows for accurate classification.

For this reason we use the visual combined classifier as a benchmark to measure performance regardless of the actual feature sets used for each classifier. This means that the visual combined classifier is not involved in the labeling process but is updated with new images at each iteration. Thus co-training is seen as a data collection process with the combined visual classifier being used to measure the strength of the data collected at any given stage.

3.8 Summary

This chapter has described the features, datasets, and methods that will be used in our experiments. We plan to use co-training with multi-modal features and multiple classifiers to improve classification on two popular image datasets: Pascal VOC’07 and MIR Flickr. Hierarchical clustering will be used to assign feature sets to classifiers in an optimal way using a CMIM-based distance metric. The performance of each co-training instantiation will be measured using average maximum precision.
Chapter 4
Parameter Selection

The co-training algorithm brings with it several parameters which must be tuned. These are:

- The size of the initial training set $L$.
- The number of positive images $p$ and negative images $n$ to be added at each iteration.
- The size of the active pool of unlabeled images $U'$.
- The number of co-training iterations $k$.

In this section we will carry out experiments that will enable us to make informed decisions on how best to choose these parameters. We will deal with parameter selection for co-training in the order listed above.

4.1 Training Set Size

The size of the initial training set is important as the set must be large enough to allow the labeling process to have a reasonable degree of accuracy but not so large that performance gains become difficult to measure. To explore this we choose a subset of object classes from the Pascal and MIR Flickr datasets and plot the performance as the training size increases (see Figure 4–1).
Figure 4–1: Performance scores for a varying number of training images for a subset of classes from the MIR Flickr and Pascal datasets. Performance is shown for the visual combined classifier only. Training sizes of 1, 3, 5, 10, 25, 50, 100, 200, 500, 1000, 2000 and 5000 images were used. The same number of images were chosen for the positive and negative images in each case. Note that some classes only have a few hundred images, which is why the plots end at varying points along the x axis.
It appears that for less than 10 training images the performance is somewhat erratic with many classes seeing almost no performance gain. Beyond 10 images the performance seems to increase logarithmically with the number of training images, requiring an ever increasing number of training images for a given performance gain. Extrapolating the logarithmic trend, one could estimate that it would require around 100000 training images to achieve a perfect AP score for the Pascal person class, although the actual number is likely to be higher since perfect performance is difficult to achieve. For other classes, such as bird and dog, it is likely that many more training images would be required to achieve a reasonable level of performance.

For the relatively small number of training images used in these experiments it appears that there is no saturation in performance. This bodes well for our approach since it suggests that there is a benefit to semi-supervised learning even after all available training images have been exhausted. However, we did not focus on maximizing performance by collecting a large dataset, but to explore the use of different feature sets in the co-training process. In order to accomplish this goal, it is necessary that the computations terminate within a reasonable number of co-training iterations, $k$. Thus, we choose 50 training images for each class to initialize the co-training process as the performance trend seems to have stabilized by this point, and this leaves a reasonable number of remaining positive images to be placed in the unlabeled set. Incidentally, this is the same number that was chosen in [27] for many of their reported results.

4.2 Learning Rate

The choice of the number of positive images $p$ and negative images $n$ to be added at each iteration reflects the learning rate of co-training. Selecting
low values may prolong the learning process while high values may make adjustments too coarse, affecting the stability of the algorithm. To be prudent, these numbers should be low in proportion to the size of the initial training set \( L \), and should reflect the underlying class distribution (the proportion of positive and negative images). We choose \( p = 1 \) and \( n = 3 \) for all experiments as in [6, 27] for consistency and for a fair evaluation of the algorithm.

### 4.3 Active Pool Size

To see what effect the size of the active pool \( U' \) has on performance, we choose a few classes from the Pascal dataset and run the co-training algorithm with the active pool size varying from from 100 images to the size of the full set of unlabeled images \( U \). We use 50 training images per class and 100 co-training iterations. It is obvious from these results in Figure 4-2 that a large active pool is best, perhaps due to the relatively small number of positive images per class in \( U \). This is contrary to the claim by Blum and Mitchell that limiting the active set of unlabeled images improves performance; However, their experiments were based on web page classification and ours are based on image classification, so the two are not directly comparable. We will use the full set \( U \) in the remainder of the experiments without limiting the size.
4.4 Number of Co-training Iterations

In order to measure performance of co-training, we need to choose a suitable number of iterations, $k$, for which we will allow the algorithm to run. When multiple classes are involved, the performance can be quite different from one class to the next so choosing $k$ to fairly evaluate each class can be challenging. Other authors have chosen to select $k$ based on the performance of a small group of classes and assume that the same performance trend applies to all classes [27, 24, 9].

Here we evaluate the effect that $k$ has on performance for a select group of classes using 50 randomly selected training images. For the Pascal VOC’07 dataset we choose classes which represent a broad range of concepts (i.e., animals, transportation, people, plants) and have a sufficient number of input images (labeled+unlabeled) for co-training to be effective. In this case the cow class had the lowest number of input images at 141, or close to three times the size of the chosen training set. For the MIR Flickr dataset we choose the 14 relevant (manually annotated) classes since that is a choice that has
been recommended by Hare et al. in [29] and gives a manageable number to work with. Note however that the baby_r1, sea_r1 and tree_r1 classes have only 59, 74 and 116 images respectively so these classes are not expected to perform well (see Figure 4–3 for the full distribution of training images for each dataset).

Figure 4–3: Distribution of training images for each dataset.

We use two classifiers for this experiment, one based on image tags alone and the other based on all visual features combined as in [27]. It is immediately apparent from the results in figures 4–4a and 4–4b that co-training is not effective for all classes, the reasons for which will be discussed in a later section. For the classes where it is effective there seems to be no general pattern for where the performance reaches a maximum. For some classes, such as clouds_r1 and car_r1, the performance reaches a maximum early on then decreases following an erratic pattern. For others such as people_r1, flower_r1, dog_r1, aeroplane and bus, the performance continues to increase beyond 100 iterations. One thing that does seem apparent is that performance gains are usually modest beyond 50 iterations but this is not true for all classes.
Figure 4–4: AP scores shown for 100 rounds of co-training for the text/visual feature split with 50 initial training images. Only a subset of classes are shown for each dataset.

To shed further light, we try co-training with the text/visual feature split, this time using the full 20 classes for Pascal VOC’07 and the full 38 classes for MIR Flickr. We plot both the mean average precision (MAP) and average maximum precision (AMP) scores as described in Section 3.7, (see figures 4–5a
and 4–5b). For the Pascal dataset, the MAP for the visual classifier reaches a peak at 20 iterations and shows an obvious performance gain, while for the MIR Flickr dataset the peak happens much sooner at 8 iterations and the performance gain is modest at best.

Figure 4–5: A comparison of the different ways of measuring performance. Co-training was run for 100 iterations using 50 initial training images. The top row shows the mean average precision scores. The bottom row shows the average maximum precision scores for the same data. The columns show the results for each dataset.

In contrast, the AMP results are much more positive, the curves are monotonic and show a clear benefit to using co-training for both datasets (Figures 4–5c and 4–5d). It is still the case that the Pascal dataset shows more improvement than the MIR Flickr dataset but this is likely due to the fact that
the text classifier is much more effective for Pascal than it is for MIR Flickr. Since performance continues to improve with more iterations using this new metric, we will plot the AMP performances curve using the maximum number of iterations that time will allow and compare the performance curves, rather than list performance numbers for a fixed number of iterations $k$ as others have done.

4.5 Summary

This chapter has dealt with selecting appropriate parameters for the co-training algorithm. Based on our experiments, we will use 50 training images per class to initialize the co-training process with the learning rate parameters: $p = 1$ and $n = 3$. The active pool size will be set to the full set of unlabeled images. The number of co-training iterations will be set to 100 and performance will be tested at each iteration using average maximum precision.

Now that the parameters have been established, we should be able to reasonably estimate what the upper bound on co-training performance would be assuming perfect labeling accuracy. If 50 initial positive training images are used then after 100 co-training iterations we should end up with 150 correctly labeled positive images. By matching the corresponding points on the graph shown in Figure 4.1c, we can estimate that the upper bound on the performance gains should be somewhere around 5 to 10 percentage points, for both the Pascal and MIR Flickr datasets. Having this in mind should help us to better evaluate the significance of the experiments that follow.
Chapter 5
How to Choose the Training Set

In this chapter, we will show experimental results that provide insight on how the training images can be selected so that co-training performance is maximized. First, we investigate supervised learning using different selections of training images. Then, we investigate co-training in a similar manner. While the results of this chapter are not used in subsequent experiments, they would be of practical use for many supervised learning approaches.

5.1 Training Set Quality

It is well known that the number of training images has a huge impact on the performance of any classifier, but the quality of the training images (i.e., the amount of variation in appearance, scale, pose and position of the relevant target objects) is a factor that is often overlooked. This is important to semi-supervised learning approaches since it may be possible to get by with fewer training images if those images are of sufficient quality.

Some aspects of image quality, like occlusion and lighting, can be determined by observing each image individually while others apply to the set of images collectively, such as the amount of correlation between background and foreground objects. If we knew what characteristics to look for when selecting the training images, we could choose a set which more profoundly influences classifier performance than if we were to make a random selection. However, there seems to be very little information in the literature on how to make this choice.

A related issue is how best to measure performance, or select the test set, for which there has been an abundance of research. It was found recently
that many of the datasets that have been used in the past (i.e., Caltech 4 [25], Caltech 101 [22], UIUC [1]) for benchmarking performance are severely limited in the appearance aspects mentioned in the first paragraph, and are therefore too easy. It was found in [47] that many of the algorithms that were able to achieve excellent performance on these early datasets failed miserably when tested with more challenging datasets such as Pascal VOC’05 [21]. The reason for this is that the models trained with these easier datasets were too rigid, since they learned to exploit the limited appearance variation of the training images (see [46] and [45] for a thorough evaluation of this phenomenon).

From this we can conclude that to build a robust model, the training images cannot be overly constrained. However, there is a difference between building a robust model and one which performs well for a particular dataset. A poor model can perform well on a particular dataset if the dataset is too easy. For fair evaluation, the dataset chosen must contain enough appearance variation to represent the amount that would exist in the real world. We would expect the Pascal VOC’07 dataset to allow for fair evaluation since it is a commonly used performance benchmark and contains a high level of appearance variation.

In order to answer the question of which training images to use, we divide the training images into three rated sets: easy, medium and hard. This is done using a similar approach to that in [43]. First the training set is divided into 5 random subsets or folds of roughly equal size. An SVM classifier is then trained on 4 folds using the visual features and the kernel described in Equation 3.1. An SVM output score is then assigned to each image in the remaining fold using the function in Equation 2.4. This is repeated 5 times, each time leaving out a different fold for scoring.
Once all images are scored they are assigned into sets based on the following criteria: The easy positive images are those with a score greater than one standard deviation above the mean score for all positive images. The hard positive images are those with a score less than one standard deviation below the mean score for positive images. Positive images with scores between these two thresholds were assigned to the medium set. The negative images were chosen analogously to the positive images.

An SVM classifier was then trained on each of the easy, medium, and hard images and the mean average precision was reported for a subset of classes. Only those classes with at least 30 positive images in each set were chosen for Pascal VOC’07 while those with at least 50 positive images in each set were chosen for MIR Flickr. Examples of these rated training sets are shown in figures 5–1, 5–2 and 5–3. The easy sets feature images where the target object is prominently shown, with little or no occlusion and backgrounds are typical (e.g. airplanes in a blue sky, birds in trees, people in the city), whereas the more difficult sets feature images where the target object is less prominent, the amount of occlusion is increased, and the backgrounds become less typical.

The results in figures 5–4a and 5–4b show that training on the easy images alone leads to much higher performance than training on either the medium or hard image sets. This is not too surprising because these images were selected by the classifier as being highly confident, meaning that there was enough statistical support among the other training images to enable a high degree of confidence in these images being positive. Since the test images are similarly distributed this also means that more test images will be classified correctly. Because many of the feature types used are based on bag-of-features approaches, the fact that scenes are typical for these easy sets can also be a
Figure 5.1: Some examples of rated training set images for the Pascal VOC'07 airplane class in increasing order of difficulty.
Figure 5–2: Some examples of rated training set images for the Pascal VOC’07 person class in increasing order of difficulty.
Figure 5–3: Some examples of rated training set images for the Pascal VOC’07 bird class in increasing order of difficulty.
Figure 5-4: The effect of training image selection on performance for an increasing number of training images. The “easy”, “medium” and “hard” plots are the rated training sets, while the “all” plot is for the full unrated training set.

boon to performance since the relevant target object can be inferred to some degree by the background information [54].

Overall, training on hard images provides the worst performance. From the hard image sets in the above figures, it appears that the foreground objects are often very far away or obscure and out of context. Thus, it is understandable that for images with a high degree of clutter it would be hard for a classifier to pick out the foreground information relevant to the target class since we are classifying unsegmented images. In the absence of prominent foreground objects the system must rely on scene context to make its decision. However, when scene context is also inconsistent confusion ensues.

It is interesting to note that for hard images the performance actually worsens with more training, indicating that what is being learned in these cases is not relevant to the information in the test images. This might stem from the fact that these datasets contain a relatively small number of target classes, many of which co-occur in the same image. If another object class appears often alongside the target object then it may be that object that is
being learned rather than the target object. This seems plausible from an observation of the hard images in figures 5-1, 5-2 and 5-3, since it seems that buildings often appear with airplanes, cars or trains often appear with people and people often appear with birds. Note that the background set of both of these datasets consists of images where other target objects are present, which means there is greater chance for confusion than if a general and broad background set was used.

Also of interest is the fact that the classifiers trained with easy images perform significantly better than those trained with the full unconstrained set of images for a given training set size, indicating that the inclusion of hard images in the training set actually worsens performance. This is a bit surprising given that for support-vector machines and other margin-based classifiers the images on or near the classification boundary are the most useful for performance[12]. A possible explanation for this may be that the hard images chosen were not actually close to the classification boundary but were broadly dispersed throughout the hypothesis space on either side of the boundary. These results do not show conclusively that including hard images are always detrimental but for the small size of training sets used here this seems to be the case. Observing what happens as the hard images are gradually removed from the full training set would be another area of research and is not explored here.

The fact that training on easy images performs better than training on hard images begs the question “How easy should our training images be?” The answer to this question depends upon the definition of “easy,” but based on the easy images shown above we can interpret this to mean “typical”. At one extreme all training images would be focused on one tight cluster of appearance, with the most typical image for that class near the centre of that
cluster. At the other extreme, all training images would be far from typical appearance of the target class and would feature no consistent background context. Having too tight a cluster would ignore many aspects of the object’s appearance, while having too many images far from the cluster centre would make learning more difficult. Clearly there must be some balance but from what is seen here, a general guideline might be to choose images where the target object appears prominently with little or no occlusion, the background is typical, and the pose of the object allows for easy recognition.

5.2 Co-training with Rated Training Sets

We have seen in the previous section that the choice of training set has a profound effect on classifier performance, with the result being that for small training sets, training on easy images leads to higher performance than does training with any other choice of images. Here we investigate how the choice of training set affects the co-training results by using the same easy, medium and hard training sets defined previously. We will call these rated training sets. On the MIR Flickr dataset we also compare with the results of co-training using the unrated dataset (listed as “all” in Figure 5-5b) using the same number of training images. Two classifiers were trained as before, one using the image tags and the other using the combined visual features. To limit the computational burden the number of co-training rounds was limited to 100 and only a subset of classes was used. The classes were chosen to have a large number of training images and to collectively represent a broad range of concepts, i.e., vehicles, animals, people, scenery, etc. It is important to note that the three image sets were rated using only the visual classifier, so an image that is “easy” for the visual classifier may not be so for the text classifier.
Figure 5–5: Co-training with the initial training images drawn from one of 3 sets of images: easy, medium or hard. 30 initial training images were used in each case for Pascal VOC’07 while 50 were used for MIR Flickr. Results shown are the average maximum precision for the visual classifier only. The “all” image set corresponds to the full unrated set of images.

It appears from the results in Figures 5–5 and 5–6 that training on easy images initially means that the absolute performance will be higher but the co-training gains will be more modest as compared to the medium and hard sets. The fact that co-training is less effective for easy images may come as a surprise. However, this makes sense based on the fact that the number of easy images in the full training set for each class is typically quite small (On the order of 30-50 images). If an easy image is put into the training set then it is no longer available as an image to be labeled and the probability of selecting an easy image in a co-training round is lessened. This means that the potential gain from co-training is also lessened since the easy images are most likely to increase performance. It is expected that the results would improve if more easy images were included in the unlabeled set. Likewise, for a classifier trained on hard images, the probably of choosing a hard image during co-training is lessened so it stands to gain more from co-training, given that hard images tend not to be as beneficial. It is interesting to note that
Figure 5–6: Co-training with the initial training images drawn from one of 3 sets of images: easy, medium or hard. 30 initial training images were used in each case for Pascal VOC’07 while 50 were used for MIR Flickr. Results shown are the average precision for each class.
even after 100 rounds of co-training with 50 unconstrained images on the MIR Flickr dataset, the average performance still does not exceed that of the visual classifier trained on the easy set without co-training. This emphasizes that the importance of proper training set selection far outweighs that of the selection of particular co-training parameters, a fact which may also apply broadly to other semi-supervised learning techniques.

5.3 Summary

In this chapter, we have carried out experiments which were intended to provide guidance concerning the type of training images which should be chosen to maximize classification accuracy. The first section dealt with the supervised learning scenario and the second dealt with co-training. In both scenarios, we find that optimal performance is achieved by selecting training images where the target object appears prominently with little or no occlusion, the background is typical, and the pose allows for easy recognition. It was found that, in general, including images which deviated from this ideal led to worse performance.
Chapter 6
Choosing the Right Features

Up until now the co-training experiments have dealt only with the text/visual feature split. This was a logical choice for two classifiers since these features are naturally independent and image tags have been shown to be complementary to visual features in other literature [36, 58]. However, it is not clear whether fusing all visual features is the best choice. In this chapter we explore other feature splits chosen automatically using the CMIM clustering algorithm described in Section 3.6. First, we outline the experimental procedure and present the results of the algorithm. Then, we compare the feature splits chosen by the clustering algorithm to some other logical choices using co-training to validate the approach.

6.1 CMIM Clustering

The CMIM-based hierarchical clustering algorithm described in Section 3.6 was implemented in MATLAB using the MutualInfo library provided in [44]. This was run separately for each of the target classes using the nine main feature types listed in Table 3–1 as input. These are: Flickr tags, Harris SIFT, dense SIFT, Harris hue SIFT, dense hue SIFT, RGB, LAB, HSV and gist. The hierarchical clustering process can be visualized using a dendrogram, that shows the order in which the clusters were merged and the distance between each merged pair. Examples of this are shown in Figure 6–1. In this case the clustering algorithm was allowed to continue running until all clusters were merged.
Figure 6–1: Sample dendrograms produced by the CMIM clustering algorithm on the Pascal VOC’07 dataset using nine feature types as input. The clusters merged first are those on the left of the diagram and the clusters merged last are on the right. The x-axis shows the CMIM distance between each pair of merged clusters.

At any point along the x-axis we can draw a vertical line which defines a cut-point and marks the clustering progress at a specific point in time. This can give a sense of which feature types would be grouped together for a given choice of \( N \). For example, for \( N = 3 \) we could choose a cut-point of 0.15 for the aeroplane class to define the clusters: \{text\}, \{gist, Harris SIFT, dense SIFT\} and \{Harris hue, dense hue, LAB, HSV, RGB\}. These same clusters are defined in Table 3–1 as text, grayscale and colour.

For each target class we can observe the clusters formed for \( N = 2, 3, 4 \) and 5 clusters to get a sense of what the feature groupings would be as the number of clusters varies. In order to fairly investigate the situation where textual information is excluded, we test two scenarios: one with Flickr tags included as input to the algorithm, and one without.

The feature groups chosen by the clustering algorithm are somewhat class dependent, as can be seen in Figure 6–1. Therefore, we estimate the most common groups by accumulating the feature groups chosen for each class. Then, for a given choice of \( N \), we select the \( N \) most common groups among all classes. An example of this is shown in Table 6–1.
Table 6-1: An example of the cumulative feature groups chosen for \( N = 3 \) for the Pascal VOC’07 dataset. The top row shows all cluster candidates chosen, where each feature type is represented by a number from 1 to 9. The bottom row shows the number of times each cluster candidate was chosen among all classes. In this case columns 3, 7 and 8 would be selected as most common. These correspond to the colour, grayscale and text feature groups.

The results in Table 6-2 show that the clusters assigned in this way form logical groups that are fairly intuitive when one considers the manner in which the features are extracted. For example, the Pascal VOC’07 results confirm that for \( N = 2 \) with text included, the text and visual feature groups (Table 3-1) are most commonly chosen to have the greatest distance between them. This means that these feature groups are deemed to have the most relevant independence [38] between them, as one might expect given the distinctly different nature of these features.

For \( N = 3 \), the visual features are split to contain grayscale feature types in one group (Harris SIFT, dense SIFT, and gist) and colour feature types in another (Harris hue SIFT, dense hue SIFT, RGB, LAB and HSV). It is quite conceivable that given these features types there are images which will be classified correctly using one feature type but not the other, and vice versa. Using \( N = 4 \), the grayscale features get split further into SIFT and gist feature types, highlighting the fact that SIFT features are extracted at local interest points while gist is a global scene descriptor. For \( N = 5 \), the SIFT features are split into Harris SIFT and Dense SIFT. The fact that these two feature types are split later on indicates that they contain a relatively large amount of redundant information about the class label. A fact which is not surprising

<table>
<thead>
<tr>
<th>Cluster Candidates</th>
<th>1,2,3</th>
<th>4,5</th>
<th>1,2,3,4,5</th>
<th>6,7</th>
<th>8</th>
<th>1,2,3,4,5,8</th>
<th>6,7,8</th>
<th>9</th>
<th>4,5,9</th>
<th>1,2,3,4,5,9</th>
<th>6,7,8,9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>2</td>
<td>1</td>
<td>13</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>14</td>
<td>16</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
### Table 6-2: The results of hierarchical clustering of feature types on the Pascal and MIR Flickr datasets with and without image tags. Each assigned cluster is given a letter from A to E. Letters are assigned in the order the clusters were formed.

<table>
<thead>
<tr>
<th># of clusters</th>
<th>Tags</th>
<th>Harris SIFT</th>
<th>Dense SIFT</th>
<th>Harris Hue SIFT</th>
<th>Dense Hue SIFT</th>
<th>RGB</th>
<th>LAB</th>
<th>HSV</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>A</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td>B</td>
<td>B</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
<td>B</td>
<td>E</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>D</td>
</tr>
</tbody>
</table>

(a) Pascal VOC '07

<table>
<thead>
<tr>
<th># of clusters</th>
<th>Tags</th>
<th>Harris SIFT</th>
<th>Dense SIFT</th>
<th>Harris Hue SIFT</th>
<th>Dense Hue SIFT</th>
<th>RGB</th>
<th>LAB</th>
<th>HSV</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>A</td>
<td>A</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
<td>D</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
<td>D</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

(b) MIR Flickr

given the similar nature of these features. Similar results were obtained for the MIR Flickr dataset.

### 6.2 Co-training with CMIM Clusters

Now that a theoretical best split between feature types has been obtained, the task to evaluate these theoretical splits in a co-training framework remains. Again, for these experiments only disjoint sets of feature types were considered and no feature type was excluded, with the text/grayscale split being the only
exception to this. The goal here was not to exhaustively compare every possible feature combination as this would not be feasible given the large number of options. Instead, for each choice of \( N \), a small set of feature splits candidates were chosen in a logical manner and compared with the best theoretical split according to the CMIM clustering algorithm. This was done with and without text included as one of the features, as outlined previously. For the logical feature splits an attempt was made to balance the strength of the individual features as well as to choose feature sets that were conceptually independent from each other. In order to assess the strength of each type of feature, we train an SVM classifier for each one individually using the full training set, and report the mean average precision on the test set. The results are shown in Figures 6–2a and 6–2b.

![Figure 6–2: MAP test scores for an SVM classifier trained using each feature type individually. The full training set was used in each case.](image)

When text is included, the only available choice for \( N = 2 \) is the text/visual feature split since we do not consider feature sets comprised of both text and visual feature types. Given that in general the grayscale features tend
to perform better than the colour features (see Figure 6–2a) we also try a
text/grayscale feature split even though it does not include all feature types.
For $N = 3$, we choose the text/local/global split for comparison with the
text/colour/grayscale split chosen by the clustering algorithm. These choices
are indicated in Table 6–3a.

<table>
<thead>
<tr>
<th>$N$</th>
<th>CMIM Split</th>
<th>Logical Split</th>
<th>$N$</th>
<th>CMIM Split</th>
<th>Logical Split 1</th>
<th>Logical Split 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>text/visual</td>
<td>text/grayscale</td>
<td>2</td>
<td>colour/grayscale</td>
<td>2 random</td>
<td>local/global</td>
</tr>
<tr>
<td>3</td>
<td>text/colour/grayscale</td>
<td>text/local/global</td>
<td>3</td>
<td>SIFT/colour/gist</td>
<td>3 random</td>
<td>none</td>
</tr>
</tbody>
</table>

(a) with text  
(b) without text

Table 6–3: Feature splits used in the co-training experiments. The CMIM split
is the feature split chosen by the CMIM clustering algorithm. The logical splits
were manually chosen by considering the feature strength and the conceptual
independence of the features.

When text is excluded, we choose the local/global split for comparison
with the CMIM clustering choice for two clusters. We also try random feature
splits created by randomly assigning feature types to sets, keeping the same
number of feature types in each set. This was done separately for $N = 2$ and
$N = 3$ clusters. These logical splits are indicated in Table 6–3b along with
the splits chosen by the CMIM clustering algorithm.

Each of the aforementioned feature splits was evaluated by training a
separate SVM classifier for each of their constituent feature sets and allowing
these classifiers to run in a co-training framework for 100 iterations. For fair
evaluation, results are reported for the visual combined classifier in each case
as was explained in Section 3.7. The results of self-training with the combined
visual classifier are also included for comparison purposes.
When text is included, we find that the text/grayscale split outperforms the text/visual split chosen by the clustering algorithm for $N = 2$ (Figure 6–3). This indicates that either the feature combination rule in equation 3.1 is less than ideal or that including colour features is detrimental to performance (more on this in the next chapter). This also suggests that it might be beneficial to allow the clustering algorithm to discard feature sets which do not perform well on their own as in [38]. For $N = 3$, we find that the text/colour/grayscale split chosen by clustering outperforms the text/local/global split on the Pascal dataset, while the two end up roughly equal on the MIR Flickr dataset (Figure 6–4).

![Graphs](image1.png)  
(a) Pascal VOC’07  
(b) MIR Flickr

**Figure 6–3:** Co-training with full disjoint feature sets with $N = 2$ clusters. Self-training with the combined visual features is included for comparison purposes.

When text is excluded, the results in Figure 6–3 show that the colour/grayscale split performs best of the three choices listed in Table 6–3b for $N = 2$, validating the effectiveness of the clustering approach, with the random feature split coming in second and the local/global split coming in last. Interestingly, co-training with the local/global split performs worse than self-training does
Figure 6–4: Co-training with full disjoint feature sets with $N = 3$ clusters. Self-training with the combined visual features is included for comparison purposes.

with the combined visual features alone. For $N = 3$, we find that the clustering split, SIFT/colour/gist, significantly outperforms the random feature split on both datasets (Figure 6–4).

It is clear from these results that the feature sets chosen by CMIM-based hierarchical clustering allow for effective co-training and lead to superior performance in comparison with the other manually chosen feature splits evaluated here. This is convenient because it allows us to generate an effective feature split for any given choice of $N$, a fact which will become useful in later experiments.

It is also clear that co-training with text included as a feature yields results that are far better than those of co-training without text included, a finding which was also verified in [27]. Results are consistent for the both the Pascal VOC’07 and MIR Flickr datasets.
Chapter 7
Increasing the Number of Classifiers

The previous chapter has dealt with the problem of assigning feature types to groups with the number of groups or classifiers, \( N \), given a priori. We now explore the impact of increasing the number of classifiers on co-training performance. Again, to exhaustively test every possible combination of feature types for every choice of \( N \) would be unfeasible, so we simplify the problem using two different approaches. The first is to consider only the feature splits chosen by the CMIM clustering algorithm presented in Chapter 6.1. These have been shown to perform well and could represent a theoretical best that can be achieved for a given choice of \( N \). Using this approach means that no feature type is left out and reflects a realistic scenario of dividing a pool of already obtained feature sets into optimal subsets. We will call this the top-down approach. The second approach is to consider the feature types already divided into independent sets, construct a classifier for each, and test the performance of co-training as the number of classifiers increases. In this case, the order in which the classifiers are added is fixed according to the performance of each on a test set. Since not all feature types are considered, this will allow us to see which feature types are beneficial for co-training and which ones are not. We will call this the bottom-up approach. First, the experimental procedure and results will be presented for the top-down approach. Then, we will deal with the bottom-up approach in a similar manner.

7.1 Top-Down Approach

For the top-down approach we run the co-training algorithm using the feature splits chosen by CMIM hierarchical clustering and listed in Table 6–2.
For comparison purposes, we also try self-training using the combined visual classifier alone. This shows the effect of increasing the number of classifiers $N$ from 1 to 5. We use 100 iterations for Pascal VOC’07 and 50 iterations for MIR Flickr since the latter took much longer to run. We use the full set of 20 classes for Pascal and only the 14 relevant classes for MIR Flickr. As before the combined visual classifier is used as a benchmark in each case and leads to the performance curves shown in Figure 7–1.

![Graphs showing performance curves for co-training with different numbers of classifiers.](image)

**Figure 7–1:** Co-training with the feature splits chosen by CMIM clustering for an increasing number of classifiers. The results for including text features are shown on the left side, while those which consider only visual features are shown on the right side.

It should be pointed out that co-training with $N$ classifiers means that $N$ times as many images will be added at each iteration in comparison with self-training. This means that for a given co-training iteration the results are
not directly comparable since they differ in the number of images employed. Nonetheless, a fair comparison can be made once a plateau has been reached for each test since by that point the results have reached a maximum and any further gains are unlikely. For easier analysis, we show the trend as the number of classifiers is increased, reporting only the final average maximum precision scores for each dataset (Figure 7–2).

(a) With text. Feature splits are chosen in the following order:
1=text, 2=text/visual,
3=text/colour/grayscale,
4=text/SIFT/colour/gist,
5=text/dense SIFT/Harris SIFT/colour/gist.

(b) Visual only. Feature splits are chosen in the following order:
1=visual, 2=colour/grayscale,
3=SIFT/colour/gist,
4=dense SIFT/Harris SIFT/colour/gist, 5=dense SIFT/Harris SIFT/colour histogram/hue SIFT/gist.

Figure 7–2: Co-training with the feature splits chosen by CMIM clustering for an increasing number of classifiers. Performance scores are the average maximum precision for 100 iterations for Pascal VOC’07 and 50 iterations for MIR Flickr.

In the results of Figure 7–2, we observe strikingly different trends for the case where text features are included from those where they are not. Without text there is a general increase in performance as the number of classifiers increases, with the most significant gains seen for co-training with four classifiers and the lowest gains seen for self-training with the combined visual classifier. With five classifiers, the performance declines for the Pascal dataset and appears to level off for the MIR Flickr dataset. When text is included the trend
appears to be reversed, this time decreasing as the number of classifiers is increased.

For the visual-only scenario, the increasing performance trend confirms the claim that repeatedly splitting the feature sets allows co-training to exploit the independence that exists among the features, yielding better performance than if the feature sets were combined. However, there seems to be an upper limit to how many times the feature sets should be split. In our case, splitting the colour features into Hue SIFT and colour histograms lead to worse performance (Figure 7–1b). The reason for this could have to do with the fact that these are weaker features to begin with (see Figures 6–2a and 6–2b). As was pointed out in [38], the clustering process tends to group weaker features together since they tend to have high correlation with each other, although they share little information about the target class. If this weaker feature set is split into subsets, the number of images labeled using this weak feature set at each iteration is effectively doubled, which can increase the labeling noise.

With text features, the fact that the co-training performance decreases with more classifiers seems baffling at first. However, this starts to make sense in light of the assumption that Flickr tags are very informative features by themselves, especially when they contain the target class label. They are also not sensitive to overspecification, i.e. the same label can produce a vast range of different images of the target class, leading to better generalization. These attributes can make other visual feature types seem pale by comparison. If the number of visual classifiers is increased it means that these less effective classifiers are allowed to label more images at each iteration which are less likely to be labeled correctly. This causes confusion on the part of the text classifier and the performance starts to degrade.
Interestingly, the performance of self-training using text features alone is significantly better than co-training with any feature grouping tested so far and shows no sign of leveling off after 100 iterations as is the case with the other tests (Figure 7-1). This is unexpected given that text features do not show the same level of superiority when measured on the test set using supervised learning (see Figure 6-2a). However, this can be explained by reasoning that if the text classifier has learned the label of the target class then it becomes highly effective at labeling other images with tags which also contain the class label. Labeling errors can still occur due to polysemy (i.e., mouse=animal mouse or computer mouse), but this did not seem to be an issue for these experiments.

While using text features may lead to superior labeling performance, some of the test images have tags that are sparse and not all of them are informative, so clearly there is an upper bound to the test performance that can be achieved using these features. By comparison, the visual features are sampled densely for each image so that the upper performance bound is higher, which could explain why they lead to better test performance (Figure 6-2).

7.2 Bottom Up Approach

Approaching the problem from the bottom requires that a set of conditionally independent feature sets has already been obtained. We use the CMIM clustering algorithm with $N = 5$ to determine the feature sets. These are sorted according to their performance on the test set using the results shown in Figures 6-2a and 6-2b. The consequent feature sets are listed as follows, in descending order of test performance: tags, Harris SIFT, dense SIFT, gist, colour. As before, two cases are considered, one with text included and one without. In each case, we start by training an initial classifier for the highest performing feature set and self-training on the unlabeled images.
We then train a classifier for the second highest performing feature set and co-train using this and the classifier obtained previously. We then add a third classifier using the next highest performing feature, and so on. Using a sorted list of feature sets in this manner means that the results are likely to have a greater improvement initially and less of an improvement later on, and will make the results easier to interpret. However, this also means that certain feature splits are ignored such as the Harris SIFT/colour split.

In the previous section we found that self-training with text features led to higher performance than expected based on the feature strength results in Figure 6–2. Thus, as a preliminary step for these experiments we also try self-training using each of our main five feature types, as well as a few others. We also include the results of self-training using the combined visual features for comparison purposes. The results are shown in Figure 7–3.

![Figure 7-3: Self-training performance with each of the feature types used in the experiments.](image)

It is interesting that self-training with either of the SIFT or gist features alone performs significantly better than using all the visual features combined. It is tempting to conclude from this that feature combination using the RBF kernel in Equation 3.1 is ineffective, but as the results show in Figure 6–2,
the combined visual classifier yields better test performance for supervised learning than any of its constituent features alone, so this is clearly not the case. A likely explanation could be that feature types that are more focused are more effective at labeling since they become specialized at picking out certain visual cues. For example, a feature based on texture would be confident at labeling close up images of an object, whereas a feature based on shape would be more confident at labeling images where the object is far away. If these features are combined, their confidence scores would get averaged out, resulting in a classifier that is better on average but less confident on images where either shape or texture is a dominant feature. Since only the most confident images are chosen at each co-training iteration, the average performance of the classifier is not nearly as important as its ability as a specialist.

We now proceed with the co-training experiments using these results. For the case where text is excluded, we start with a classifier trained on Harris SIFT features alone since this gave the best test results for supervised learning and self-training as shown in Figure 7-3. A classifier is then trained using the dense SIFT features and included in the co-training process along with the Harris SIFT classifier tested previously. These feature types both have high test performance but are similar in nature, the only difference being that dense SIFT features are extracted on a dense grid at regular intervals and the Harris SIFT features are extracted at regions found using a Harris-Laplace interest point detector. The results in figures 7-4b and 7-4d show that co-training using these two classifiers leads to higher performance initially, due to the fact that more images are labeled at each iteration, but in the end the results show

---

1 Dense SIFT features were used instead of Harris SIFT on the MIR Flickr dataset since they gave higher self-training performance.
little difference from those using a single classifier. Next, a classifier trained on gist features is included. This leads to a marked improvement over the two classifier case where only the two SIFT feature types are used, indicating that the gist features add useful information that was not already available. Finally, a classifier trained on colour features is included, this time causing a sharp decline in performance, for reasons which will be discussed below.

Figure 7-4: Co-training performance with the bottom-up approach for an increasing number of classifiers. The results for including text features are shown on the left side, while those which consider only visual features are shown on the right side.

The performance trend as the number of classifiers increases can more easily be seen in Figure 7-5, where we show the average maximum precision
after the final co-training iteration in each case. This will aid in the analysis that follows.

(a) With text. Feature splits are chosen in the following order:
1=text, 2=text/Harris SIFT,
3=text/Harris SIFT/dense SIFT,
4=text/Harris SIFT/dense SIFT/gist,
5=text/Harris SIFT/dense SIFT/gist/colour.

(b) Visual only. Feature splits are chosen in the following order:
1=Harris SIFT, 2=Harris SIFT/dense SIFT, 3=Harris SIFT/dense SIFT/gist, 4=Harris SIFT/dense SIFT/gist/colour.

Figure 7–5: Co-training with the bottom-up approach for an increasing number of classifiers. Performance scores are the average maximum precision for 100 co-training rounds for Pascal VOC’07 and 50 co-training rounds for MIR Flickr.

The fact that pairing Harris SIFT and dense SIFT classifiers does not improve results indicates that for co-training to outperform self-training, it is not enough to have strong classifiers alone if those classifiers are based on features that are too similar in nature. This highlights the importance of the conditional independence assumption in [6]. By contrast, including gist features in the co-training process does seem to improve results, although the change is quite small.

The performance decrease seen with the inclusion of colour features indicates that the colour classifier adds more labeling noise than can be tolerated by the other classifiers being trained. This is surprising given that colour features were shown in [50, 56] to have superior performance over their grayscale
counterparts, both of which compared colour SIFT to the standard grayscale SIFT. Given this fact, we cannot say that colour features perform poorly in general. We must conclude then that either (i) the datasets used are poor judges of performance, or (ii) the colour features used are simply not strong enough to capture the wide amount of variation that exists in our datasets.

If the datasets contained a large amount of grayscale images then understandably, a colour classifier would be not be able to learn a consistent recognition model. However, this cannot be the case as the vast majority of the dataset images are in colour. This leaves the features themselves as the likely source of error.

In the supervised learning setting, both the hue SIFT and colour histogram features fall behind the others in terms of test performance on the Pascal dataset (Figure 6-2a). This does not come as a surprise for colour histogram features since these are global descriptors that do not encode shape information and typically do not perform well on their own [50]. On the other hand, hue SIFT features combine colour and shape information, so in theory they should perform better than the standard grayscale SIFT since they contain more information. Specifically, they consist of a concatenation of the standard SIFT descriptor and a local hue histogram [56]. These are extracted on a dense multi-scale grid and at regions found using a Harris-Laplace interest point detector and clustered to form a histogram of visual words. They were shown to outperform the pure SIFT-based bag-of-features model on the bird dataset in [32] where the task was to classify 6 distinct bird species, and on a soccer dataset where the task was to distinguish one soccer team from
another.\textsuperscript{2} However, both of these datasets contain classes which have consistent colours and are not representative of most real-world objects, such as the ones in the Pascal and MIR Flickr datasets where colour is more variable. Thus performance could have been overestimated.

One other likely possibility is that there are simply not enough visual words included in the dictionary to capture the higher dimensionality of this descriptor. For reasons unknown to the author, the bag-of-features model for the hue SIFT descriptors downloaded from the LEAR website consists of only 300 visual words while the grayscale SIFT descriptors consists of 3000 visual words. One would expect that more visual words would be required to capture the more abundant information contained in the colour features. By contrast the winning entry to the Pascal VOC’08 challenge [50] used a combination of invariant colour SIFT descriptors and clustered them using K-means to form a 4000 visual word model.

Without further testing, the true cause of the failure of these features cannot be definitively identified. One thing that is clear is that including colour features, as they are found here, degrades the performance of co-training and in general, better results could be obtained by incorporating a feature selection step into the CMIM clustering algorithm as was done in [38].

When text features are included, the results differ somewhat depending on the dataset. For the Pascal dataset, there is a steady decrease in performance as the number of classifiers increases, while for the MIR Flickr dataset, the results increase for the inclusion of SIFT features but decrease for the inclusion of gist and colour features (Figure 7–5a). The reason for this differing behavior is likely due to the fact that text features are far better at labeling for Pascal

\textsuperscript{2} Available from http://lear.inrialpes.fr/people/vandeweijer/data
than they are for MIR Flickr (Figure 7-3), perhaps because the number of
tags in the dictionary differs significantly between the two datasets (the Pascal
dataset uses 804 tags while MIR Flickr uses only 457).

Note that the trends for the MIR Flickr dataset differ between the top-
down results in Figure 7-1c and the bottom-up results in Figure 7-5a. For the
top-down approach, the trend decreases monotonically while for the bottom-up
approach this is not always the case. This is because the top-down approach
includes all the feature types for each test, including those that don’t perform
well, and the bottom-up approach does not.

7.3 Summary

We have investigated the impact of increasing the number of classifiers on
co-training performance using two approaches, both of which rely on feature
splits chosen by CMIM clustering. The first approach starts with a pool of
feature sets and repeatedly subdivides these according to their relevant inde-
pendence in a top-down manner. The second starts with a set of subdivided
feature sets and works from the bottom up, adding feature types according to
their test performance.

In general, we found that including more classifiers in the co-training
process can lead to improved performance over the case where only one or
two classifiers are considered, but only when (i) the classifiers are based on
independent features, and (ii) the classifiers are not too unbalanced in terms of
labeling ability. Specifically, we found that performance tends to increase with
the number of views in the case where only visual features are considered, but
tends to decrease when text features are included due to the fact these features
have superior labeling ability when considered on their own. We also found
the performance to decrease as colour features were included as these had
poor labeling ability to begin with. This situation may have been remedied
by incorporating a labeling rule based on classifier confidence as was done in [34, 23, 9], but was not done here due to the added complexity.

For the visual-only scenario, we found that including more classifiers in the co-training process by repeatedly subdividing the input features generally leads to improved performance. However, further gains were found by prioritizing the features types and systematically adding each one until performance gains began to diminish, thus eliminating certain lower performing feature types. The best results for visual features were obtained by co-training with three classifiers using the Harris SIFT/dense SIFT/gist feature split.

It should be noted that while co-training with visual features alone improves with more classifiers, the performance is still far below what could have been obtained if text features had been incorporated. With text included, the best results for Pascal were obtained by self-training with text features alone, while for MIR Flickr the best results were obtained by co-training with the text/Harris SIFT/dense SIFT feature split.
8.1 Discussion

In this thesis, we have investigated the use of co-training to improve image classification in the situation where multiple classifiers are used and multi-modal features are available. We assumed that text features were provided in the form of Flickr tags, and several different types of visual features also existed. Two main objectives were given. The first was to investigate how to optimally assign features to classifiers such that co-training performance is maximized. The second was to explore the effect of increasing the number of classifiers given that the features are already assigned. An auxiliary objective was to investigate the effect that the initial training set selection has on performance.

To address the issue of feature assignment, we clustered all feature sets using a distance metric based on conditional mutual information, and assigned each feature cluster to a separate classifier. Co-training was then evaluated using the resulting set of classifiers and compared to a set of manually chosen feature splits. In each case, the feature splits determined by the clustering algorithm led to favorable results. It was also found that the classifiers which were most effective at labeling were those which focused on one particular aspect of an image (i.e., colour or shape), rather than combining several different feature modalities.

By using the feature sets assigned by clustering, we found that increasing the number of classifiers was beneficial to co-training performance provided that the classifiers were sufficiently independent, and reasonably well balanced.
in terms of their labeling ability. This trend was made most apparent when features were prioritized and those with the highest test performance were systematically included in the co-training process. In most cases, the best results were obtained by using three classifiers, although it is expected that adding more classifiers would continue to improve results as more features are included.

When text features are used, our results show that co-training performance can approach that of full supervision when one assumes that the number of labeled training images that results from co-training is the same as that used by the full supervision. The same qualitative result was also found in [23], which uses text features of a different form and tests performance on a set of images downloaded from Google.¹ Note that, in our case, the expected gains are modest (roughly 8 percentage points for Pascal and 6 percentage points for MIR Flicker) because of the low upper limit on the number of training images imposed by the co-training parameters. Further gains could likely be achieved by labeling more images at each iteration or increasing the number of iterations.

When only visual features are considered, we found that co-training could provide a significant performance gain over supervised learning if the features were appropriately assigned, but far less than could be achieved if text features were also included. Other approaches which combined separate visual modalities [24, 9] employed an active learning stage which required user input to boost accuracy and showed excellent results. The same thing could also have been done here to further improve the results, but was not due to the additional supervision required.

¹ This dataset was not made publicly available for comparison.
As was mentioned in Chapter 3, the datasets and features used in this thesis were the same as those used in the co-training approach of [27]. A fair quantitative comparison between our approach and theirs is not possible due to the different random selections of training images employed, and the instability involved with using mAP as a performance metric. However, our results suggest that using text features alone for Pascal and the text/Harris SIFT/dense SIFT feature split for MIR Flickr can provide a significant improvement over their approach, where only the text/visual split was considered.

In our experiments on training set selection, we found that, for small training sets, the images which were most beneficial to classification performance were those where the target object appeared prominently in the scene and the background was typical. By contrast, it was found that including cluttered images in the training set was detrimental to performance. Interestingly, our results show that proper training set selection can lead to better test performance in the first iteration of co-training than a randomly chosen training set could after 100 iterations. This suggests that training set selection is something that may currently be under-emphasized by the semi-supervised learning community.

8.2 Conclusion

To conclude, we have shown that image classification can be improved in a semi-supervised manner by co-training with multi-modal features provided that each classifier uses an appropriate selection of features. We have also shown that using more than two classifiers can be beneficial to co-training performance provided that the classifiers are sufficiently independent, and that they are well balanced in terms of labeling ability.
8.3 Future Work

The co-training framework presented in this thesis can been extended in several different ways. First, it should be noted that the datasets used in these experiments are relatively small, which means that the potential gains from co-training are reduced since, for some classes, there are not many images to draw from in the unlabeled set. Using a Google search to collect more relevant images would be one way of raising the upper bound on performance.

Second, although the SIFT and gist features proved effective at labeling, the colour features (hue SIFT and colour histograms) used here did not, for reasons which are not entirely clear. Given the excellent performance of invariant colour SIFT features [50] on the Pascal VOC’08 challenge it would be interesting to include these types of features to see how co-training performance is affected. Other features could also be tried, such as HOG [14], LBP [41] and biologically inspired features [52].

Third, it was apparent from our results that co-training with certain feature set combinations led to worse performance than self-training with either set alone. There are several possible ways that our co-training algorithm could be modified to overcome this. One might be to allow each classifier to label a number of images in proportion to its performance on a validation set. Another might be to use a confidence threshold for each classifier as was done in [34].

Finally, further investigation into the issue of training set selection might prove fruitful. In particular, it is not clear whether the assumption that cluttered images are detrimental to classification still holds as the size of the training set increases. If these images lie on the boundary of the classification
hypothesis then they should, at some point, be beneficial. It would be enlight-
ening to explore gradually removing these images from the full training set to
observe the effect on classification accuracy.
Bibliography

[1] S. Agarwal and D. Roth. Learning a sparse representation for object

of the potential function method in pattern recognition learning. *Automation


world: A system for region-based image indexing and retrieval. In *Visual

[8] G. Cauwenberghs and T. Poggio. Incremental and decremental support


