Active Learning for Visual Concept Model Creation

Hoda Alemi
Active Learning for Visual Concept Model Creation

Master of Science Thesis

For the degree of Master of Science in Artificial Intelligence at University of Amsterdam

Hoda Alemi

Supervisor: dr. Gwenn Englebienne
Mentor: dr. Harro Stokman

August 22, 2013

Faculty of Science · University of Amsterdam
The work in this thesis was supported by EUVISION. Their cooperation is hereby gratefully acknowledged.
Abstract

Active learning is an important machine learning framework to deal with the cost of labeling. In this master thesis, we employ an active learning framework to train an SVM visual concept detector. We apply uncertainty sampling query approach and combine it with the representativeness measure of the images. We propose a new, more accurate measure of annotation cost which allows us to propose a novel, lower cost, method of active querying, based on the k-means algorithm.

In our study, the experimental results show that employing active learning for a visual concept detector can reduce the amount of training data up to 90% while maintaining the performance of the model. We demonstrate that by employing active learning, only by using 10% of the data for training the model, we can also achieve slightly higher performance than when is trained on the whole data.
# Table of Contents

Abstract i
Acknowledgements v

1 Introduction: What is Active Learning? 1
   1-1 Contribution .................................................. 3
   1-2 Overview .................................................... 4

2 Related Work 5
   2-1 Stream-based Active Learning .............................. 5
   2-2 Pool-based Active Learning ................................ 6
   2-3 Query Strategies ............................................. 6

3 Active Learning Methods 9
   3-1 Learning with Support Vector Machines .................. 9
   3-2 Margin Based Query Approaches ............................ 10
       3-2-1 MaxMin Margin Query Approach ....................... 10
       3-2-2 Uncertainty Sampling Query Approach ................ 11
   3-3 Representativeness Measure for Queries .................. 12
       3-3-1 Similarity Measure .................................... 13
   3-4 Clustering and Active Learning ............................ 14
       3-4-1 K-Means Clustering .................................... 14
       3-4-2 Proposed Method ....................................... 15

4 Data 19
   4-1 Dataset ..................................................... 19
   4-2 Data Foundation ............................................. 20

Master of Science Thesis Hoda Alemi
## 5 Experiments and Results

5-1 Evaluation .................................................. 23  
   5-1-1 Building the Evaluation Sets .......................... 23  
   5-1-2 Evaluation Metric .................................. 24  
5-2 Experimental Setup ..................................... 26  
5-3 Initial Seeds .............................................. 26  
5-4 Results and Analysis ..................................... 26  
   5-4-1 Experiment 1: Query Strategies ....................... 27  
   5-4-2 Experiment 2: Training Time ......................... 32  
   5-4-3 Experiment 3: K-means .............................. 35  

## 6 Conclusions and future work ............................. 41  
   6-0-4 Future Work ......................................... 42  

**Bibliography** ................................................. 43
Acknowledgements

I thank my supervisor dr. Gwenn Englebienne, for giving me the freedom to explore such an interesting topic, and for his kind guidance during the work of this thesis.

I’m grateful to Euvision Technologies team, for sparking the idea of this work, and for their continuing support throughout my internship.

I would also like to thank my family and friends for their support during my project.

University of Amsterdam

August 22, 2013
To my Mom and Dad.
Chapter 1

Introduction: What is Active Learning?

Three general classes of problems in machine learning are supervised learning, unsupervised learning, and semi-supervised learning. In supervised learning, we learn a mapping from observation to predicted output from a set of (input, output) pairs: each training input has an associated output. In unsupervised learning, there are no known outputs, and the goal is to discover structure in the set of (input) points. Finally, in semi-supervised learning, we have a set of (input, output) pairs, and a separate set of unlabeled (input) points. The goal is then to use the structure of the unlabeled data points to learn a better mapping than what we would get from labeled data alone.

In a supervised machine learning task we need to obtain labeled data to train a model. In most cases, obtaining the data is very cheap; however, obtaining the label is quite expensive in terms of time or labor. We can choose to label a subset of the available data, but the choice of the subset will affect the quality and performance of the final model. The question, then, is how to actually select the subset of data that will achieve the best model performance?

The importance of active learning emerges in applications that deal with large amounts of data; since labeling such data can be very costly and exhausting. Active learning is an iterative machine learning algorithm in which the main issue is to evaluate the informativeness of an unlabeled instance. The active learner is a classifier that is initially trained on a few labeled instances. Then, iteratively, by its knowledge derived from the labeled data it request a label for one of the instances of the unlabeled pool of data. Successful active learning should lead to a significant reduction of the amount of training data needed for a supervised learning task with no significant reduction of the machine’s performance over a machine trained with a fully-labeled dataset.

We indicate most of supervised machine learning algorithms as passive learning in terms of fully-labeled training data. The efficiency of active learning has been qualified by two approaches. The more common approach is the reduction of training data required to reach a certain level of performance. The other approach is to increase the performance of the model.
2 Introduction: What is Active Learning?

Figure 1-1: Active learning cycle for image classification task. The cycle starts by labeling few examples as initial seeds then active learner model is created. By applying query strategy on the unlabeled pool of data, the active learner asks the human annotator for the label of the most informative example. The new labeled example will then be added to the training set.

for a certain amount of training data (Vlachos, 2004). The applicability of either of these approaches depends on the use case.

Active learning has been studied in many different contexts Settles (2009) provides a comprehensive literature review on the subject. In the application of speech recognition D. Yu and Acero (2010) employed active learning to select a limited subset of utterances to transcribe from a large amount of un-transcribed utterances. They reduced the amount of utterances needed for transcribing by 60% to achieve the same recognition accuracy obtained using random sampling.

Another application of active learning is in message filtering. For instance, in spam filtering applications, trustworthy labels for messages can be costly to acquire. Sculley (2007) reports that online active learning can significantly reduce labeling and training costs while maintaining high levels of filtering performance.

Active Learning has been also employed for the task of classification for documents or media like images, audio and videos. For the classification of media not only the annotation can be very time consuming and tiring but also, in some domains like sexual child abuse it can be mentally challenging. This thesis tackles the problem of annotation in the domain of image classification. Figure 1-1 illustrates the active learning cycle in this problem domain.
1-1 Contribution

For this thesis, we explored the problem of annotating images for ImageMiner application at EUVISION company. ImageMiner is a software developed by EUVISION company on top of Impala software which is a visual concept detector (Snoek et al., 2012). Impala has the ability to recognize a wide range of concepts in digital media content. The term concept refers to any scene or object in the image; some of the visual concepts are “cats”, “dogs”, “sunsets”, “handguns”, “text messages”, “fireworks”, etc. In order to recognize such concepts, Impala needs to be trained with thousands of labeled images for each of individual concepts.

ImageMiner provides a user interface for annotating concept models. Moreover, it provides a commercial tool for analyzing image sets. The ImageMiner’s current approach to build a concept is to first train a model with a few labeled images. Then based on the confidence of the classifier, the ImageMiner ranks the unlabeled images. An annotator is then presented with a ranked list of images, with most confident first, as illustrated in the Figure 1-2. Then, the annotator labels some positive and some negative images and retrain the model in a series of iterations. Figure 1-2 illustrates the ImageMiner’s first iteration ranking results for the concept of “car” in which the classifier has been trained with five positive and five negative images. The issue with the current approach of ImageMiner is that there is no clue for the annotator, other than experience to select the best positive and negative images for the next retraining iteration.

In this work we consider the following research questions:

Research Question 1: By employing active learning in order to obtain labeled images, can we reduce the annotation cost? This requires carefully defining the “cost” of annotation, as different ways to qualify the cost will yield different results. In this work, we explore 3

Figure 1-2: ImageMiner application. The ranking results for the concept “car” after first training iteration with five positives and five negatives examples.

1http://www.euvt.eu/
Introduction: What is Active Learning?

Definition of cost: 1) amount of training data, 2) time needed per each iteration of active learning, and 3) time needed for whole process of active learning.

Research Question 2: What is the best query strategy, that leads to the best trade-off between annotation cost and model performance? This question can also restated as, which is the best example to be labeled at each iteration of active learning? As illustrated in Figure 1-3.

Research Question 3: Atypical image in the training set can negatively affect the final model's performance. Can active learning remove those images from the training data to improve performance over a model trained on a fully-labeled dataset?

1-2 Overview

The remainder of the thesis is organized as follows. In Chapter 2, we present relevant related work. In Chapter 3, we describe active learning approaches and the proposed method. In Chapter 4, we describe the dataset and the data foundation. To test and evaluate the explained methods, we apply them on a standard dataset in Chapter 5 and discuss the results. Finally, in Chapter 6, we present our conclusions and discuss possible future work.
Active learning can be categorized by its way of synthesizing queries, into either stream-based or pool-based active learning (Settles, 2009). In this work, the focus is on pool-based active learning with SVMs. Work on stream-based active learning is only mentioned to provide more relevant context.

2-1 Stream-based Active Learning

Atlas et al. (1990) introduced stream-based or sequential active learning, in which the key idea is to select one instance at a time from the sequence of instances which are sampled from the actual distribution. Then, the active learner, based on the measure of informativeness of the instance, must decide whether to query this instance or ignore it. Stream-based active learning is usually used when data cannot be easily stored so the only advantage of stream-based approach over pool-based is quick query decision making.

Dagan and Engelson (1995) also propose a stream-based active learning in which they have applied a committee-based sampling which the committees are drawn randomly from a probability distribution. This has been done to learn the Hidden Markov Models known as HMMs (Baum and Petrie, 1966) for part of speech tagging of English sentences. They evaluate the informativeness of an example by measuring the degree of disagreement between several model variants. The use of generative probabilistic models has its advantage, as it makes the scoring of unlabeled instances easy. On the other hand, they do not necessarily perform as well as discriminative models, such as SVM.

More recently, Yu (2005) has employed a support vector machine (SVM) classifier with selective sampling technique for learning ranking functions and applied it to the data retrieval applications. The main idea of their work is to determine the most ambiguous example for ranking at each iteration, and the user’s feedback on the ranking will maximize the degree of learning. In this work, we build on this and apply SVM to our classification task.
2-2 Pool-based Active Learning

Pool-based active learning, which is the more common approach in today’s machine learning and data mining applications, was introduced by Lewis and Gale. (1994). In this approach instead of sampling one instance at a time (stream-based), a large pool of instances are sampled then the model selects the best query to be labeled. Hence, there exist a very small set of labeled data and a large pool of unlabeled data.

Pool-based active learning has been used in many real world applications. In the application of facial age classification, i.e. a technique for classifying face images into different predefined age groups, Indu et al. (2012) propose to find the most informative facial image from the unlabeled pool to use the furthest nearest neighbor criterion, i.e. the unlabeled instance which is the furthest from its nearest labeled neighbor, and add it to the training set. They propose an active learning approach to be applied to the $k$ nearest neighbor classifier and the SVM classifier and then compare the performance of these two classifiers. Their method gives good results but it is not very well formally investigated.

Settles and Craven (2008) presents a detailed analysis on an active learning pool based approach for sequence labeling tasks, i.e. information extraction and document segmentation. Their methods include information density, sequence vote entropy, and Fisher information where their evaluation gives us more hints at which criteria to use for querying.

Also, for the application of text classification, Mccallum and Nigam (1998) present a combination of active learning and Expectation Maximization (EM) on a pool of unlabeled data. They use query-by-committee strategy to select the text documents for labeling and then EM with a naive Bayes model to improve the accuracy of the text classifier. The proposed method is interesting, however, they make a strong assumption that data is produced by a mixture model, and there is a one-to-one correspondence between mixture components which makes the work limited to the assumptions.

In this work, from the two mentioned categories of active learning for synthesizing queries, we use the pool-based active learning.

2-3 Query Strategies

The main issue with active learning is to find the most informative instance from the unlabeled pool of data. These informative instances are iteratively used to retrain the model. Finding the most informative example is based on the query-making strategies. Different query strategies have been proposed such as: uncertainty sampling, query by committee, expected model change, expected error reduction, variance reduction, and density weighting (Settles, 2009).

Uncertainty sampling which is one of the most common querying approaches, was first introduced by (Lewis and Gale., 1994) for the application of text classification. In contrast to random sampling, which selects instances randomly and has been used in traditional supervised learning approach, uncertainty sampling selects the instances that the current model is most uncertain about. Uncertainty sampling is more common for probabilistic learning models (Settles, 2009). However, it can be employed for non-probabilistic models as well. Lewis
and Catlett (1994) employ uncertainty sampling for a decision tree classifier in which the decision rules of the tree has been produced from uncertainty samples. Similarly, Lindenbaum et al. (2004) addresses the problem of active learning by proposing a lookahead algorithm for nearest neighbor classifiers.

Mirroshandel et al. (2011) propose a novel active learning strategy based on SVMs for temporal relation classification task. In their approach, they consider three measures of uncertainty, representativeness, and diversity for making the query from unlabeled pool of data. They define the measure of representativeness as it was previously defined by Settles and Craven (2008) with the name of information density. In their experiment they show that the state-of-the-art results can be reproduced by their method with a significantly less amount of training data.

Similarly, Xu et al. (2009) present an uncertainty sampling-based active learning approach for SVMs to annotate the most uncertain unlabeled instances. They firstly employ the decision margin of SVMs output as the uncertainty measure then to further reduce the amount of unlabeled instances for training, they employ the ratio of center-distance of each example.

The other query strategy is query by committee which employs a committee of models where all the models are trained on the currently available labeled data. In order to find the most informative query example, all the models with different hypothesis vote their predictions on the unlabeled pool of data. Then, the example with maximum disagreement is chosen as the most informative example (Seung et al., 1992). In this work, we focus on a single “strong” classifier, rather than on combining multiple “weak” classifiers. We do not, therefore, investigate QBC further.

In expected model change approach, the example whose inclusion brings about the maximum change in the current model will be the most informative example. The query can be the gradient of the loss function with respect to the parameters of the model (Settles and Craven, 2008). Expected error reduction query method, selects the example which reduces the expected generalization error as much as possible. Therefore, the main idea of this approach is to estimate the expected future error of the current model on the remaining unlabeled instances in unlabeled pool (Roy and Mccallum, 2001). But this approach is an expensive approach and we can still reduce generalization error indirectly by minimizing output variance. This has been called the variance reduction approach (Settles, 2009). Another query strategy method is based on using weights to choose the informative example which is called as density weighting approach. This approach, weigh the informativeness of an example by its average similarity to the entire unlabeled pool of examples. Therefore, the examples will be selected based on both uncertainty and representativeness measure and as an advantage, an outlier will not get a substantial weight (Settles and Craven, 2008).

In this thesis, we use a pool-based active learning to construct the training data, for a visual concept model creation. The visual concept model is an SVM classifier. In this work, we investigate different query strategies, and propose a novel idea in a form of semi-supervised learning to discard a subset of unlabeled data, without negative impact on the performance of the system.
Chapter 3

Active Learning Methods

In this work of active learning, in order to be consistent with the current approach of ImageMiner, we have used a linear Support Vector Machine (SVM) classifier. SVMs are well motivated supervised machine learning algorithms that have been developed from statistical learning theories (Cortes and Vapnik, 1995). In both theoretical and empirical domains, SVMs have proven a success and that makes them an attractive choice as a learning method to use with active learning. SVMs has been widely used in variety of application such as bioinformatics, text and image classification where here our focus is on image classification.

Therefore, in this chapter first we describe in theory how we used SVM in the context of active learning. Then, we explain different active learning query strategies using SVMs.

3-1 Learning with Support Vector Machines

SVMs, mostly has been considered as a non-probabilistic binary linear classifier. However, they can also efficiently perform non-linear classification by applying the kernel trick, mapping the data into high dimensional feature space. In active learning we need to train the classifier in each iteration. In this work, our focus is on the linear SVMs which are computationally less expensive than non-linear SVMs.

The main idea of SVM is to find an optimal hyperplane, parameterized by $w$, that separates the training data in a way of maximizing the margin between two classes. Margin is the distance between decision boundary and the closest training data as illustrated in Figure 3-1.
Given the training data \( \{x_1, ..., x_n\} \), which are feature vectors where \( x \in \mathbb{R}^m \), and their labels \( \{y_1, ..., y_n\} \) where \( y_i \in \{-1, +1\} \) (in which all the examples on one side of the decision boundary are labeled as -1 and all the examples on the other side are labeled as 1), we would like to optimize the problem defined in Equation 3-1:

\[
f(x) = \min \frac{1}{2} ||w||^2
\]

s.t. \[
\begin{align*}
y_i(w \cdot x_i + b) & \geq 1 & \forall i: y_i = 1 \\
y_i(w \cdot x_i + b) & \leq -1 & \forall i: y_i = -1
\end{align*}
\]

By solving this problem we achieve \( w \) which is a vector of feature weights and \( b \) which is the bias of the discriminant.

### 3-2 Margin Based Query Approaches

In this section, we describe the candidate query strategies for performing active learning with SVMs. First, we explain MaxMin query approach which has been introduced by Tong and Koller (2001). Then, we describe uncertainty sampling query approach used with the SVM classifier. It is worth to mention that Tong and Koller (2001) refer to uncertainty sampling as simple margin query approach.

#### 3-2-1 MaxMin Margin Query Approach

In MaxMin margin query method first we should explain the notion of version space (Mitchell, 1982). Given a set of labeled training data, \( \{x_1, ..., x_n\} \), which are feature vectors
in some space, \( X \in \mathbb{R}^m \), and their corresponding labels, \( \{y_1, ..., y_n\} \) where \( y_i \in \{-1, +1\} \), the version space is a set of consistent hyperplanes or hypotheses which separate the data in the feature space as defined in Equation 3-2:

\[
V = \{ f \in H | \forall i \in 1..n : y_i f(x_i) > 0 \}\]

(3-2)

where \( H \) is a set of hypotheses that separate the class data in the feature space.

The basic idea is to find the point in the version space that maximizes the minimum distances to any of the hypotheses. By imagining the version space as a hypersphere, SVMs try to find the center of the largest radius hypersphere whose center is in the version space and whose surface does not contain the hyperplanes. The idea of MaxMin margin query is to look for query examples that halve the version space. Therefore, the size of the version space where \( w \) lies, reduces as fast as possible. The algorithm of the MaxMin margin query method is explained in Algorithm 3-1.

**Algorithm 3-1** MaxMin Margin Approach

<table>
<thead>
<tr>
<th>Input:</th>
<th>( U = {u_1, ..., u_n} ),</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:</td>
<td>( Q ) : query instance</td>
</tr>
<tr>
<td>2:</td>
<td>for ( x = 1 \rightarrow \text{size}(U) ) do</td>
</tr>
<tr>
<td>3:</td>
<td>( y_x \leftarrow -1 )</td>
</tr>
<tr>
<td>4:</td>
<td>calculate ( M^- ) {negative margin}</td>
</tr>
<tr>
<td>5:</td>
<td>( y_x \leftarrow +1 )</td>
</tr>
<tr>
<td>6:</td>
<td>calculate ( M^+ ) {positive margin}</td>
</tr>
<tr>
<td>7:</td>
<td>keep ( \min(M^-, M^+) )</td>
</tr>
<tr>
<td>8:</td>
<td>end for</td>
</tr>
<tr>
<td>9:</td>
<td>( Q \leftarrow \max(\min(M^-, M^+)) )</td>
</tr>
</tbody>
</table>

The method literally is described as following: for each unlabeled image in the pool, once label the image as +1 and once as -1 and each time calculate margins. The margin is defined in Equation 3-3. Then, query the instance where the minimum of the two positive and negative margins is the greatest.

\[
M = \frac{2}{||w||}
\]

(3-3)

where \( w \) is achieved by optimizing the SVM problem.

### 3-2-2 Uncertainty Sampling Query Approach

In this query strategy we make query on examples that the SVM classifier is uncertain about. The most uncertain examples are the ones which are closer to the decision boundary. Given a SVM model trained on a few labeled data with the vector of feature’s weight, \( w \), and the bias, \( b \), we define the uncertainty score (or confidence of the classifier), \( \Omega(x) \), as the following:
12

Active Learning Methods

Figure 3-2: The triangles and rectangles are respectively positive and negative data. Black circles are unlabeled data and solid line is the decision boundary. In this scenario the most uncertain example is Y. However example X is more representative than Y.

\[ \Omega(x) = w \cdot x + b \]  \hspace{1cm} (3-4)

where \( x \) is presented as an example.

The most uncertain positive and negative examples are respectively the minimum positive score and the maximum negative score among all the scores. In this work, we have implemented the uncertainty sampling in two ways, 1) taking the closest sample to the hyperplane and 2) taking the sample based on probability sampling, by converting the scores to probabilities as defined in Equation 3-5:

\[ P(\Omega(x)) = e^{-\Omega^2(x)} \]  \hspace{1cm} (3-5)

where \( \Omega(x) \) is the uncertainty score.

3-3 Representativeness Measure for Queries

Another important measure in active learning is the measure of representativeness. A query that is selected by uncertainty measure can be very informative but it may not be representative to the distribution of other examples in the pool. As can be seen in Figure 3-2, example Y is closer to the hyperplane however example X (which is not as close as Y to the hyperplane) is more similar to the distribution of data. Therefore, making the queries just based on the uncertainty of the classifier is prone to make queries on the outliers. Therefore, to deal with the effect of outliers, we combine the example’s measure of representativeness with the corresponding uncertainty measure.

We measure the representativeness of each example, \( \Psi(x) \), similar to the information density (Settles and Craven, 2008) by taking the average distance of each example to all examples in the pool which is defined in Equation 3-6:

\[ \Psi(x) = \frac{1}{N} \sum_{x' \in U} \text{similarity}(x, x') \]  \hspace{1cm} (3-6)
Where \( N \) is the number of images in the pool, \( U \) is a set of unlabeled examples, and similarity\((x, x')\) is a function to find the distance between \( x \) and \( x' \) which is described in the following section.

Computing the representativeness score may seem to be costly, since it finds the pairwise distance for all the unlabeled points in the feature space. This does not change over time, however, and is therefore computed only once, off-line, before starting the active learning process.

### 3-3-1 Similarity Measure

A variety of similarity measures have been proposed for finding the distance between two points in high dimensional feature space. In order to find the most appropriate similarity metric for the current feature space, this work investigates five similarity functions: Euclidean, Manhattan, correlation, histogram intersection kernel (HIK), and chi-squared distance.

#### Euclidean Distance

The Euclidean distance, i.e. the same kernel as linear SVMs, is the length of line segment that connects two points of \( x \) and \( x' \). The points are feature vectors with \( m \) bins in high dimensional feature space. The Euclidean distance is defined in Equation 3-7.

\[
\text{Euc-dist}(x, x') = \sqrt{\sum_{i=1}^{n} (x_i - x'_i)^2} \tag{3-7}
\]

#### Manhattan Distance

As shown in Equation 3-8, Manhattan distance is the sum of the lengths of the line segment between \( x \) and \( x' \) points when it is projected onto the coordinate axes.

\[
\text{Man-dist}(x, x') = \sum_{i=1}^{n} |x_i - x'_i| \tag{3-8}
\]

#### Correlation Distance

The idea of the correlation metric is to measure the dependence between two points and the common measure is Pearson product-moment correlation coefficient which takes into account the distribution of the feature points and is calculated as following in Equation 3-9:

\[
\text{cor-dist}(x, x') = \frac{\text{cov}(x, x')}{\sigma_x \sigma_{x'}} = \frac{E[(x - \mu_x)(x' - \mu_{x'})]}{\sigma_x \sigma_{x'}} \tag{3-9}
\]

where \( \text{cov}(x, x') \) is the covariance, \( E \) is the expected value and \( \mu \) is the mean.
Histogram Intersection Kernel Distance

The histogram intersection kernel, HIK, is a similarity measure between two histograms of $x$ and $x'$ with $n$ bins in which it takes the minimum of bins between two histograms. As Chapter 4 explains, our features are represented as histograms and this prompted us to investigate the HIK distance. HIK distance is defined in Equation 3-10:

$$\text{HIK-dist}(x, x') = \sum_{i=1}^{n} \min(x_i, x'_i) $$  \hspace{1cm} (3-10)

Chi-squared Distance

Chi-squared distance is the distance between two histograms of $x$ and $x'$ with $m$ bins which each of the histograms is $L1$ normalized, i.e. their entries sum up to one. Chi-squared distance is defined in Equation 3-11.

$$\text{Chi-dist}(x, x') = \sum_{i=1}^{n} \frac{(x_i - x'_i)^2}{(x_i + x'_i)} \frac{1}{2} $$  \hspace{1cm} (3-11)

3-4 Clustering and Active Learning

In each iteration of our active learning system, the uncertainty scores need to be computed for all the images in the pool. The computational time of computing the scores depends linearly on two factors: the length of feature vector (or feature dimension) and the size of the data set. For small data sets computing the scores is quite fast (computational complexity explained in Chapter 5). However, many of the real-world applications involve millions or billions of data records which is a more realistic scenario. Therefore, in this work we propose an efficient way to compute the scores only for the collection of images which are relevant to the visual concept, based on the k-means algorithm. The idea is to identify large number of images, which are clearly not related to the visual concept, before doing active learning. We can then ignore these images in the subsequent steps, leading to considerable speed-ups.

In this section, first we briefly explain k-means algorithm. Then, we describe how we employed k-means in our active learning system.

3-4-1 K-Means Clustering

K-means is one of the unsupervised learning methods in which the procedure uses only the unlabeled examples (MacQueen, 1967). Given the unlabeled data, $U = \{x_1, x_2, ..., x_n\}$, the main idea of k-means algorithm is to estimate the unknown $k$ cluster centers (means), i.e. $M = \{\mu_1, \mu_2, ..., \mu_j\}$, with the aim to minimize the following objective function:

$$J(M) = \sum_{i=1}^{N} ||x_i - \mu_j||^2 $$  \hspace{1cm} (3-12)
3-4 Clustering and Active Learning

Figure 3-3: Five examples of “potted plant” concept. The pink circle is drawn to highlight the concept. In some of the images “potted plant” is quite small part of the image.

where \( \mu_j \) is the closest cluster center to \( x_i \). The k-means algorithm constructed in the Algorithm 3-2.

**Algorithm 3-2** K-means Algorithm

1: place \( k \) points as initial centroids. {usually randomly}
2: repeat
3: assign each of the data, \( x_i \in U \), to the nearest \( \mu_j \), i.e. the position of \( k \) centroids.
4: recomputes \( \mu_j \)
5: until until the centroids no longer move

3-4-2 Proposed Method

In order to find the applicable set of images to be used in active learning process, first we apply k-means in which we set \( k \) as twice the total number of concepts, because some of the concepts, e.g. the “potted plant” concept shown in Figure 3-3, are quite difficult in a sense that they refer to only a minor part of the image.

After clustering in order to detect the most relevant clusters to be used in the active learning of a visual concept, we apply the initial model, i.e. built with only few positive and negative images, on each cluster and then compute the uncertainty (confidence) score of the classifier. Next, we compute the precision of each cluster where precision is described in Chapter 5.

Here, the question that comes up is “how to compute the precision of each cluster without having the ground truth?” Our solution is to hypothesize the labels of one cluster at a time as positive and the remaining clusters as negative. Figure 3-4 gives an illustration of this idea with five clusters. The intuition behind this approach is that the cluster that contains more positive images or images that are similar to positive has higher precision.
After computing the precision of each cluster, we take the top $L$ ranked clusters. Depending on the difficulty of each concept, $L$ can be different for each of visual concept. In order to determine $L$ we have done an experiment by computing the mean precisions over all concepts in the Section 5-4-3. By defining $L$, then, we use the collection of top $L$ clusters for active learning of the visual concept. The summary of the idea how to identify the relevant clusters to be used in active learning is shown in Algorithm 3-3.

**Algorithm 3-3 Identifying Relevant Clusters**

**Input:** $U = \{u_1, ..., u_n\}$, $L = \{l_1, ..., l_n\}$

1: $\theta \leftarrow$ initial SVM model
2: apply $\theta$ on all data and compute the confidence score
3: apply K-means to find $K$ clusters
4: **for all** $K$ **do**
5: label the cluster as positive and the remaining clusters as negative
6: compute corresponding precision of the cluster
7: **end for**
8: Sort clusters according to the highest precision
9: Relevant-clusters $\leftarrow$ top $L$ clusters
10: **return** Relevant-clusters

We now describe the pseudo code of our proposed active learning approach in Algorithm 3-4. The algorithm first trains the model based on the few initial labeled images. Next, it computes the combinational score of uncertainty and representativeness measures. Then, based on the minimum score the query is made from the unlabeled pool of images. And in the end the labeled image will be removed from pool. The parameter $\alpha$ has a value in a range of $[0, ..., 1]$ and is a balanced weight between the uncertainty and representative scores. In
order to find the right $\alpha$, we have done some experiment in Section 5-4-1.

**Algorithm 3-4** The Proposed Active Learning Method

**Input**: Relevant-clusters, $L = \{l_1, ..., l_n\}$, $\alpha \leftarrow 0.9$ 

1: $Q$: query
2: $\Omega(x)$: uncertainty Score
3: $\Psi(x)$: Representativeness Score
4: repeat
5: for $i = 1$ to size($U$) do
6: $S_{x_i} = [\alpha \Omega(x_i) + (1 - \alpha)\Psi(x_i)]$
7: end for
8: $Q = \text{arg min}(S_{x_i})$
9: $L \leftarrow L \cup (Q, \text{Label}(Q))$
10: $U = U \setminus Q$
11: retrain Model and compute average precision
12: until Stopping Criteria
In this chapter, first we describe the standard data set employed for our active learning system. Next, we explain the feature extraction methods employed by Euvision.

4-1 Dataset

The evaluation of the active learning methods has been done on Pascal Visual Object Classes 2007 (VOC)\(^1\) dataset. The Pascal VOC challenge is an annual benchmark in visual object category recognition and detection, in which it provides a standard dataset of images and annotations collected by the consumer photographs from the Flickr\(^2\) photo-sharing website. The two main goals of this competition is the classification and detection of objects/concepts from 20 visual concept classes in realistic scenes (Everingham et al., 2010).

The whole data collection consists of 9,963 images and 24,640 annotated objects. The collection has been split into 50% for training/validation and 50% for testing in which the train/validation set consists of 5011 images and the test set consists of 4952 images. Table 4-1 indicates the number of positive examples for each concept in the dataset.

\(^1\)http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/
\(^2\)http://www.flickr.com/
Table 4-1: Visual concepts and the number of positives images per each concept in Pascal VOC2007 dataset

<table>
<thead>
<tr>
<th>Concepts</th>
<th>Train/Validation set</th>
<th>Test set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aeroplane</td>
<td>238</td>
<td>204</td>
</tr>
<tr>
<td>Bicycle</td>
<td>243</td>
<td>239</td>
</tr>
<tr>
<td>Bird</td>
<td>330</td>
<td>282</td>
</tr>
<tr>
<td>Boat</td>
<td>181</td>
<td>172</td>
</tr>
<tr>
<td>Bottle</td>
<td>244</td>
<td>212</td>
</tr>
<tr>
<td>Bus</td>
<td>186</td>
<td>174</td>
</tr>
<tr>
<td>Car</td>
<td>713</td>
<td>721</td>
</tr>
<tr>
<td>Cat</td>
<td>337</td>
<td>322</td>
</tr>
<tr>
<td>Chair</td>
<td>445</td>
<td>417</td>
</tr>
<tr>
<td>Cow</td>
<td>141</td>
<td>127</td>
</tr>
<tr>
<td>Diningtable</td>
<td>200</td>
<td>190</td>
</tr>
<tr>
<td>Dog</td>
<td>421</td>
<td>418</td>
</tr>
<tr>
<td>Horse</td>
<td>287</td>
<td>274</td>
</tr>
<tr>
<td>Motorbike</td>
<td>245</td>
<td>222</td>
</tr>
<tr>
<td>Person</td>
<td>2008</td>
<td>2007</td>
</tr>
<tr>
<td>Pottedplant</td>
<td>245</td>
<td>224</td>
</tr>
<tr>
<td>Sheep</td>
<td>96</td>
<td>97</td>
</tr>
<tr>
<td>Sofa</td>
<td>229</td>
<td>223</td>
</tr>
<tr>
<td>Train</td>
<td>261</td>
<td>259</td>
</tr>
<tr>
<td>Tvmonitor</td>
<td>256</td>
<td>229</td>
</tr>
</tbody>
</table>

According to the Pascal VOC annotation guideline, concepts are labeled as positive if they are more than 10-20% visible (there are some concepts in the data set that are more than 50% occluded). Moreover, concepts which placed in mirrors also labeled as positive.

4-2 Data Foundation

For the data foundation we have used one the most popular feature representation methods which is Bag-of-Words (BoW) model. The features that have been employed are bag-of-words (BoW) implemented by Snoek et al. (2012). A BoW representation for an image is a histogram vector in which each bin of histogram denotes a visual word while the corresponding value to the bin is the number of occurrences of the particular word. In order to obtain BoW features the following steps needs to be done respectively: keypoint detection, visual description and word encoding (Csurka et al., 2004).

1. **Keypoint Detection**: Keypoints are points/corners in an image which are highly distinctive and invariant to the noise, scaling, rotation, viewing direction and changes in illumination. For this reason we have employed Harris-Laplace point detector (Tuytel-laars and Mikolajczyk, 2008). Harris corner detector is not invariant to scale. Hence, in order to make it scale-invariant, Harris-Laplace detector applies the Harris corner detector (Harris and Stephens, 1988) at multiple scales then automatically choose the characteristic scale by applying Laplacian operator.
2. **Visual Description:** A visual descriptor describes the primary characteristics of the interest points of the image. They describe the shape, color, gradient, location and texture of the local region around the interest points. In this work we have employed Scale-Invariant Feature Transform (SIFT) descriptors by Lowe (2004). To create a SIFT descriptor first the gradient magnitude and orientation is computed for each point in the region ($16 \times 16$) around the keypoint location. Then the histogram of these orientation multiplied by the magnitude is created which summarize the contents over 4x4 spatial bins (16 bins total). The total dimension of the feature vector is $1 \times 128$.

3. **Word Encoding:** The final step of a BoW model is to encode feature vectors into visual words in which each image can be represented by a histogram of visual words. For this purpose, K-means clustering (Hartigan and Wong, 1979) is used to generate the visual words and a hard codeword assignment (Snoek et al., 2012) with maximum of 3950 words is employed. Figure 4-1 demonstrates each step of the Bag-of-Word foundation.
Chapter 5

Experiments and Results

In order to compare the active learning query methods, we consider three aspects: higher performance, fewer training examples, and less computational time. Therefore, the evaluation task can be of a challenge. This chapter first considers formally how to evaluate the methods discussed in the previous chapter. Next, the evaluation methods and the implementation details of active learning approaches are described. Finally, we discuss the result of each experiment.

5-1 Evaluation

In this section we describe how we performed the evaluation of the query strategies. In the following, first we explain how we built the evaluation sets, i.e. train and test sets. Then, we discuss the evaluation metrics and the one which is more suitable for our work.

5-1-1 Building the Evaluation Sets

To evaluate the performance of the implemented methods we use special case of $k$-fold cross-validation method which is a statistical method of evaluating and comparing algorithms (Kohavi, 1995). In $k$-fold cross-validation, the data is randomly dividing into $k$ equal size subsamples. Then the cross-validation process repeats taking one of the $k$ subsamples for testing the model and the remaining $k-1$ for training.

In active learning, special case is required because in each iteration the selected query needs to be taken out from the pool. The flowchart of our evaluation is shown in Figure 5-1. We have used 2-fold cross validation, because with more folds the overlapping data of each fold is particularly problematic in active learning. For example, when the query-making strategy is based on selecting the example that has the closest distance to the decision boundary, the same example will be selected in the folds with the data overlap. The training sets are frequently identical (rather than just similar) and the variance on the performance is under-estimated.
5-1-2 Evaluation Metric

There are many different evaluation metrics proposed for evaluating the performance of the information retrieval systems. First, we explain the basic measures which are *precision* and *recall*. A true positive, TP, is predicted positive example (or relevant document) that is correctly a real positive. And, false positive, FP, is predicted positive example that is not a real positive (see Table 5-1).

<table>
<thead>
<tr>
<th>Retrieved Documents</th>
<th>Relevant Documents</th>
<th>Non Relevant Documents</th>
</tr>
</thead>
<tbody>
<tr>
<td>true positives (TP)</td>
<td>false positives (FP)</td>
<td></td>
</tr>
<tr>
<td>false negatives (FN)</td>
<td>true negatives (TN)</td>
<td></td>
</tr>
</tbody>
</table>

Therefore with these definitions, the precision of a system, $S$, is the fraction of positive predictions which are correct. As shown in Equation 5-1:

$$\text{Precision}(S) = \frac{TP(S)}{TP(S) + FP(S)}$$  \hspace{1cm} (5-1)

and recall is the fraction of positive examples that have been correctly classified (recalled) which is shown in Equation 5-2:
Recall(S) = \frac{TP(S)}{(TP(S) + FN(S))} \quad (5-2)

In our system in order to measure how well the learning process of the active learner is performing we compute the Mean Average Precision (MAP), which is the mean of average precision over 20 concepts. In order to compute MAP, first we need to compute the average precision. The advantage of the average precision over the precision is that it also considers the order (ranking) of the retrieved images. Average precision is defined in Equation 5-3 and mean average precision (MAP), which from now we call it performance graph is defined in Equation 5-4:

\[
\text{Average Precision} = \frac{\sum_{j=1}^{n} \text{precision}(j) \times \text{rel}(j)}{\#\text{relevant documents}} \quad (5-3)
\]

where \( \text{rel}(j) \) is equal to 1 if the retrieved example at rank \( j \) is a relevant document otherwise zero.

\[
\text{MAP} = \frac{\sum_{c=1}^{C} \text{Average precision}(c)}{N} \quad (5-4)
\]

where \( C \) is the number of concepts which in this work is 20.

Another metric that we have considered in this work is the Area Under the Curve (AUC) of the performance-time graph. As shown in Figure 5-2, AUC is the area of the region under the graph of performance, shaded blue.

![Figure 5-2: Area under the curve (AUC). To compute AUC, we sum the area of all the rectangles.](image)

We compute AUC by summing the average of the green rectangle, obtained by taking the mean height between MAP(\( t_i \)) and MAP(\( t_{i-1} \)) and base is the distance between current time and next time. AUC is defined in Equation 5-5:
Experiments and Results

\[ \text{AUC} = \sum_{i=2}^{N} \left[ \frac{\text{MAP}(t_i)}{2} + \frac{\text{MAP}(t_{i-1})}{2} \right] \times (t_i - t_{i-1}) \]  

(5-5)

where \( N \) is the number of active learning iterations and \( t_i \) is the computational time per each iteration.

5-2 Experimental Setup

There are different sorts of software packages implementing Support Vector Machines (SVMs) which most of them are free for academic usage. Since active learning involves training the SVM classifier in each iteration, before implementing the active learning cycle we have made some experiments on various implementation of SVMs to evaluate their capabilities and speed. Among the LIBSVM (Joachims, 1999), SVMLight (Chang and Lin, 2011), and VLFeat (Vedaldi and Fulkerson, 2010) implementations, we decided to continue our work with VLFeat implementation of SVM as it appears to be faster.

This active learning system is implemented using Matlab-R2010a language and environment \(^1\) and it uses the VLFeat \(^2\) (version 0.9.13) implementation of SVMs. The implementation is done on a dual quad core Intel processor at 2.93 GHz and with 72 GB RAM.

5-3 Initial Seeds

To the best of our knowledge, the number of initial seeds or finding a way how to select them is more empirical. Having too many initial seeds conflict with the goal of active learning; since active learning is supposed to find the best examples so that it can reduce the number of training set. On the other hand having too few initial seeds may overfit the model only on those few training data. To get an estimation for the number of initial seeds we have done two experiments by randomly selecting 0.25% and 0.5% of the data with a uniform distribution. Since by running these experiment we did not notice any impact on the active learning curve and the goal is to have less training examples therefore we have followed our active learning experiments with 0.25% of data that has been selected randomly.

Moreover, it is worth noting that as an alternative solution to randomly selecting initial seeds, we have performed an informal experiment to select the cluster centroids as initial seeds for our active learning. However, we did not note any improvement effect on the active learning curve.

5-4 Results and Analysis

The goal is to investigate which query strategy achieves no harm to the performance while reducing the number of training examples. As a baseline, we already have the performance

\(^1\)http://www.mathworks.nl/products/matlab/  
\(^2\)http://www.vlfeat.org/
5-4 Results and Analysis

Figure 5-3: Uncertainty Sampling. The green plot is the uncertainty sampling with the closest sample, and the blue plot is the uncertainty sampling with probability distribution.

of the classifier trained on the whole pool of images. Therefore, we can evaluate our strategy whether it can get the same or higher performance of the classifier while having less training images. To get a better understanding of the results, first we report the results of the two margin based approaches. Next, we show the effect of adding a representativeness measure to the selected query method. Finally, we discuss the result of our active learning query approach continued with k-means.

5-4-1 Experiment 1: Query Strategies

As described in Section 3-2-2, we have implemented the uncertainty sampling in two ways. Figure 5-3 compares the two implemented approaches. The green plot is the uncertainty sampling which takes the closest sample to the hyperplane, and the blue plot is the probability sampling. As can be seen, the closest sample to the hyperplane outperform the probability sampling. We empirically determined that it made little sense to spend much time exploring this further. As the rest of this work, what we mean by uncertainty sampling is the closest sample to the hyperplane.

Figure 5-4 compares of the performance of the uncertainty sampling (green plot), MaxMin margin (blue plot), and the random baseline (red plot) in each iteration of active learning. The dashed line is the passive learning trained on fully labeled training data. The evaluation is done with 2-fold cross validation and in each iteration of active learning only one image is
Figure 5-4: The Mean Average Precision (MAP) in each iteration of active learning over 20 concepts. The green plot is uncertainty sampling, the blue plot is MaxMin margin, the red plot is random sampling, and the dash line is the baseline performance achieved by passive learning. Since, MaxMin is computationally very expensive, we did not continue it further.

The mean average precision of the model in passive learning (when it is trained on whole dataset) with 2-fold cross validation is 0.3183 MAP. However, by employing active learning after only 480 iterations, we reach that performance. As can be seen, at iteration 1030, we get the highest performance of the model which is 0.3334 MAP. This performance, is even higher than the performance of the model trained on the whole dataset. This means that by employing active learning, we can eliminate atypical images that negatively affect the performance of the classifier.

In our standard dataset, we have a hold-out testset to test our evaluated approaches. However, in order to check our active learning method on the hold-out testset, we need a stopping criterion. In this work, the stopping criterion, is a parameter defined by 2-fold cross validation which we set it to 500. Figure 5-5, demonstrates the result of employing uncertainty sampling and random sampling on the hold-out testset. As can be seen, uncertainty sampling,
i.e. the green curve, reaches the performance of passive learning at iteration 300. We conclude that by employing active learning, we can achieve reduction of the amount of training data of the order of 90%. Note that, since in our evaluation experiments, MaxMin margin query approach did not achieve a high performance, and is very time consuming, we have not consider it further.

**Adding Representativeness**

As described in Section 3-3, we consider finding the best distance measure to compute the representativeness score. Table 5-2 compares the area under the curve (AUC) of the distance metrics which has been described earlier. The aim is to select the distance metric which achieves the highest AUC. On our dataset using 2-fold cross validation, correlation measure yields the highest AUC of all distance metrics considered.

After discovering the most appropriate distance metric for computing the representativeness score, we have done a parameter search, in a range of \([0, ..., 1]\) by using 2-fold cross validation, to define the best value for \(\alpha\). We empirically determined that the best weight for \(\alpha\) is 0.9.

As can be seen in Figure 5-6, we compare the uncertainty score with the combination of uncertainty and representativeness score. In the first 200 iterations of active learning employing the combinational score steadily yields higher performance compared to just using uncertainty score. However, after almost 270 iterations employing just uncertainty score


Table 5-2: Comparing the area under the curve (AUC) of five distance metrics where used to compute the representativeness score.

<table>
<thead>
<tr>
<th>Measures</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manhattan</td>
<td>136.3617</td>
</tr>
<tr>
<td>HIK</td>
<td>136.4441</td>
</tr>
<tr>
<td>Chi-squared</td>
<td>137.1862</td>
</tr>
<tr>
<td>Euclidean</td>
<td>138.3468</td>
</tr>
<tr>
<td>Correlation</td>
<td>139.0975</td>
</tr>
</tbody>
</table>

archives slightly higher performance than combining it with representativeness. The reason can be explained as the exploration versus exploitation trade-off in reinforcement learning (Sutton and Barto, 1998). We can think of representativeness as exploitation of the images with similar distribution and uncertainty as exploring the unknown uncertain images. And, our goal is to make a balance between exploration and exploitation so for this reason we combine the two scores.

Figure 5-6: Comparing combination of representativeness and uncertainty scores with uncertainty score. The green curve is the performance with uncertainty score, the blue curve is the performance with the combinational score, and the dash line is the baseline performance by passive learning.

Our experimental results show that employing the combined score will lead the system to a faster increase in performance. However, at some iterations for more exploring we also need to obtain only uncertainty score. It is worth mentioning that the previous performance graph is an average over all the available concepts. In order to understand more details
about performance of combinational score, we illustrate the performance graph for some of the concepts individually in Figure 5-7. As can be clearly seen, for the concepts in the first row, i.e. “airplane”, “horse”, and “cat”, employing the combined score yield significant improvements. However, for the first two concepts of the second row, i.e. “car” and “tv-monitor”, there is almost no difference between the two curves. And, the last graph which belongs to the concept of “motorbike” is the failure of the combinational score.

![Figure 5-7](image)

**Figure 5-7**: Comparing combination of representativeness and uncertainty scores with uncertainty score for the concepts of (from left to right) “airplane”, “horse”, “cat”, “car”, “tv-monitor”, and “motorbike”. The first row is the success of combinational score; the first two plots of the second row shows almost no difference between the two methods and the last plot is the failure of the combinational score.

We tried to understand why employing combined score fails on the concept of “motorbike”. As shown in Figure 5-8, one reason might be the very different distributions of the motorbikes. Images with “motorbikes” are very dissimilar, possibly more than for other concepts. Adding a measure of similarity is therefore damaging to learning.

In Figure 5-9, we show the experiment of adding representativeness on the hold-out testset. In this experiment, adding the representativeness does not achieve a faster performance as the experiment of Figure 5-6. Since adding the representativeness measure did not achieve
a significant improvement and carries a cost in terms of training time, we did not explore it further.

5-4-2 Experiment 2: Training Time

One of the very important aspects of active learning cycle is the computational time per each iteration. Considering a human annotator who is waiting for an image to be annotated per iteration, the goal is to reduce the waiting time per each iteration. In each iteration we deal with the time of computing score of all the images, making the query, and retraining the model. The complexity of computing score is $O(Ncd)$, where $N$ represents the number of images in the pool, $c$ represents the number of classes which here is 2, and $d$ represents the number of feature dimensions. In addition, the complexity of uncertainty query approach is $O(N)$, where $N$ is the number of examples. The worst case computational complexity of retraining the SVM model is quadratic (Bottou and Jen Lin, 2006); however, as shown in Figure 5-10 in our experiment we observed almost a linear curve for retraining the SVM.
5-4 Results and Analysis

Figure 5-9: Adding the representativeness on the hold-out set. The green and blue curves are respectively, uncertainty and uncertainty combined with representativeness. The dash line is the passive learning baseline on the hold-out testset.

model.

Figure 5-10: SVM time curve for retraining the model.
To make this work more applicable for the Euvision company, we calculated the time needed for each segment of active learning. This makes it possible to easily estimate how different feature vector lengths, different SVM kernels, and larger dataset will affect the training time. Table 5-3 illustrates the time in milliseconds for each segments of active learning.

<table>
<thead>
<tr>
<th>Sections</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computing score</td>
<td>44.1</td>
</tr>
<tr>
<td>Uncertainty Query</td>
<td>45.4</td>
</tr>
<tr>
<td>MaxMin Query</td>
<td>22951</td>
</tr>
<tr>
<td>Random Query</td>
<td>1.1</td>
</tr>
<tr>
<td>Retraining the model (average time)</td>
<td>54.9</td>
</tr>
</tbody>
</table>

In Figure 5-11 we compare the performance of margin based results considering the time demanded by each approach on hold out set. Since the MaxMin query approach is computationally so expensive (see Table 5-3), we omit it from the performance-time graph. Figure 5-11 compares uncertainty sampling with the random baseline in the function of time. This gives a slightly different picture than Figure 5-4. In the first 50 iterations, random query is performing better because it is faster at querying. However, after that there is a big advantage in using the uncertainty sampling. It is worth mentioning that in our time computation, we did not take into account the time for removing the query from unlabeled pool. This time, has a constant value for all the query approaches. However, in the implementation, due to the system memory management, we observed a large variation of the value.

![Figure 5-11: The comparison of performance-time graph between uncertainty and random sampling.](image)
5-4-3 Experiment 3: K-means

Our goal is to reduce the time spent labeling instances. The time required for performing a query is, therefore, of great importance and depends on the size of the unlabeled pool. If we could reduce the size of this pool without discarding relevant datapoints, we could improve our performance on this metric. In the following, we use k-means clustering in combination with a minimal initial model in an attempt to perform a form of semi-supervised learning where we discard a large portion of the training data up front without negative impact on the final model. This investigates the two definition of “cost”: the time per iteration of active learning, and the time for whole process of active learning. In this section, first we show the experiment for finding a value for $L$, the number of spent clusters. Next, we demonstrate the result of employing k-means in active learning.

Defining $L$

Depending on the difficulty of each concept, $L$ can be different for each of visual concept. Some concepts, such as “airplane”, have a good performance with low value of $L$, which other concepts, such as “person” which is presented in many images of our dataset, requires a far higher value of $L$. We determine $L$ by computing the mean precisions over all concepts in the experiment shown in Figure 5-12. We obtain the trade-off between training time and classifier performance for different values of $L$, the number of best clusters. As shown in Figure 5-12, the average precision is high for the first clusters and after 20th cluster it almost goes to zero. Hence, just to be sure we are not losing any informative images we set $K$ to 20 where it means we are almost using only 50% whole data.

![Figure 5-12](image)

**Figure 5-12:** On the $x$ axis we have the clusters, and on the $y$ axis we have ranked mean precision of the 20 visual concepts.
**K-means in Active Learning**

The result of employing k-means before starting the active learning cycle is shown in Figure 5-13. As can be seen, in the first 120 iterations of active learning we did not harm the performance. However, after that, active learning needs more exploration and the sub category of images clearly does not reach the same performance of the whole collection.

![Figure 5-13](image)

**Figure 5-13**: Comparing the performance of active learning with and without pre-clustering respectively the pink and blue graph. The dash line is the performance baseline.

It is worth remembering that the result of the previous graph is the average performance over 20 concepts. In order to get more insight on employing pre-clustering in active learning, we have investigated the performance curve for each of the concepts individually. Figure 5-14, illustrates the result of 3 success and 3 failure of employing pre-clustering for active learning. For some of the concepts, e.g. “airplane”, “train”, “bird”, not only the pre-clustering does not harm the performance but also it slightly improves the results, however for the other concepts, e.g. “motorbike”, “person”, and “bottle”, it seems that the active learner needs more exploration.

In order to overcome this problem and achieve the same performance of active learning while using only half of the images in the pool, we update k-means every 150 iteration of active learning. Figure 5-15, compares the performance of active learning without clustering technique (blue graph), active learning with pre-clustering (pink graph), and active learning with 3 times updated clustering (green plots). As can be seen, with updating the k-means clustering we almost get the same performance of active learning by using only half of data in each iteration.
Figure 5-14: Comparing the performance of employing pre-clustering for uncertainty sampling on some of the concepts: the top row which is the success (from left to right) shows “airplane”, “train”, “bird” and the bottom row which is the failure shows “motorbike”, “person”, and “bottle”. The blue curve is the active learning without pre-clustering and the pink curve is the active learning with pre-clustering.
Employing k-means clustering during active learning process is computationally expensive. The time complexity of the k-means clustering is of order $O(IKnm)$, where $K$ is the number of clusters, $I$ is the number of k-means iterations, $n$ is the number of data that needs to be clustered, and $m$ is the number of feature dimensions. This means that by employing k-means clustering during the active learning cycle, the “whole time” of the active learning process will take longer. Figure 5-16, illustrates the performance of the active learning approach with and without employing k-means with respect to time. As can be seen, in the green curve we take into account the k-means computation time, therefore, the whole process takes longer time.

However, the advantage of employing k-means is the reduction of the time “per each iteration” of active learning. Considering a scenario where a human annotator is waiting for an image to annotate; if for example, each query takes 2 seconds and then after 150 annotations the annotator takes 2-3 minutes break it would be less annoying than if each query takes 10 seconds. Therefore, if we consider the computational time of k-means as a short break that the annotator needs to take after labeling, for example after 200 continuous images, then the time spent clustering has no associated cost, and employing k-means in active learning will be an efficient approach. Figure 5-17 illustrates the performance with respect to the time, of the active learning approach (uncertainty sampling) with and without employing k-means. Here, we followed our scenario and did not take into account the cost of computation of k-means.

As shown in Table 5-4, we conclude that by employing this novel idea based on k-means, we can make the time two times faster per iteration of active learning.
Figure 5-16: Performance of active learning with respect to time 500 iterations. The green curve is the active learning (uncertainty sampling) with updated k-means and the blue curve is the active learning (uncertainty sampling) on whole dataset. The green curve takes into account the k-means computation time.

Table 5-4: The computational time (in milliseconds) while employing k-means.

<table>
<thead>
<tr>
<th>Sections</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computing score</td>
<td>22.01</td>
</tr>
<tr>
<td>Uncertainty Query</td>
<td>22.6</td>
</tr>
<tr>
<td>Computing K-means</td>
<td>109113</td>
</tr>
</tbody>
</table>
Figure 5-17: Performance of active learning with respect to time in 500 iterations. The green curve is the active learning (uncertainty sampling) with updated k-means and the blue curve is the active learning (uncertainty sampling) on whole dataset. The green curve does not take into account the k-means computation time.
Chapter 6

Conclusions and future work

In this thesis we have investigated the problem of labeling large amounts of data for a supervised learning task. We have presented different query approaches for the active learning system and an effective way to reduce the annotation effort while maintaining the SVMs classification performance. We have qualified the cost of employing active learning by 3 definitions: 1) amount of training data, 2) time needed per each iteration of active learning, and 3) time needed for whole process of active learning. In order to reduce the cost of time per each iteration of active learning, we have proposed a novel idea, based on k-means, to use only a subset of relevant images for a concept, per each iteration of active learning. To evaluate the implemented active learning methods we have performed the experiments on the Pascal VOC 2007 dataset.

In order to investigate the active learning system for visual concept detectors, our first research question is as follow:

Research Question 1: “Can we reduce the annotation cost?”

As Figure 5-5 shows, the answer of this question is “yes”. By employing active learning we have reduced the amount training data by up to 90%. We have also proposed a novel method based on k-means shown in Figure 5-17 to make the computation of active learning twice as fast per iteration.

Given the result of research question 1, we move our attention to the second question:

Research Question 2: “What is the best query strategy?”

We evaluate the best query strategy based on the mean average precision (MAP) over 20 concepts with 2-fold cross validation. As discussed in Section 5-4-1, among the implemented query methods, the best query strategy with current features and dataset is the “uncertainty sampling”.

In order to understand whether active learning has improvement on performance of the system, our third research question is:

Research Question 3: “Can active learning remove atypical images to improve the performance over a model trained on a fully-labeled dataset?”
Regarding to our empirical result on the current dataset, the answer of this question is “Yes”. As shown in Figure 5-4, by employing our active learning, not only we did not harm the performance of the system but also we slightly improved the mean average precision of the model.

6-0-4 Future Work

For the future work, it would be interesting to test our algorithm on a larger dataset, like millions of images. However, we believe that our algorithm shows the promising potential.

Moreover, it would be interesting to investigate if we can reduce the cost by combining the active learning with semi-supervised learning. By employing semi-supervised learning we would like to find the decision boundary of the SVM classifier not only by using the labeled data but also by taking into account the unlabeled data. We think this idea worth implementing, however, it has the risk that each iteration of active learning takes longer.

In addition, it is also worth investigating a function as a stopping criteria for the active learning. By employing this function, we can stop the active learning on the highest performance of the model.

In a world where human computer interaction (HCI) is growing fast, due to the effectiveness of active learning, we believe that our contribution has the potential to improve the existing products and we hope it will encourage others to apply active learning for an efficient data construction, to create a visual concept model.
Bibliography


Indu, M. T., Bhuvana, S., Selvy, P. T., and Radhakrishnan, R. (2012). In A New Active 
Learning Technique Using Furthest Nearest Neighbour Criterion for K-NN and SVM Classifiers.

and Smola, A., editors, Advances in Kernel Methods - Support Vector Learning, chapter 11, 


learning. In Proceedings of the Eleventh International Conference on Machine Learning, 
pages 148–156. Morgan Kaufmann.


Vision, 60(2):91–110.


classification.

support vector machines, application to temporal relation classification. In IJCNLP, pages 
56–64.


