



Ugur Kirbac

Image Labeling and Classification by Semantic Tag Analysis

Master of Science Thesis

Supervisors: Esin Guldogan and Moncef Gabbouj

Supervisors and topic were approved by Faculty of Computing and Electrical Engineering Council on Aug 15, 2012.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY

Master's Degree Program in Information Technology

KIRBAC, Ugur: Image Labeling and Classification by Semantic Tag Analysis

Master of Science Thesis, 52 pages

May 2013

Major Subject: Signal Processing

Supervisors: Prof. Moncef Gabbouj and Dr. Esin Guldogan

Keywords: Image labeling, classification, semantic text analysis, WordNet

Image classification and retrieval plays a significant role in dealing with large multimedia data on the Internet. Social networks, image sharing websites and mobile application require categorizing multimedia items for more efficient search and storage. Therefore, image classification and retrieval methods gained a great importance for researchers and companies.

Image classification can be performed in a supervised and semi-supervised manner and in order to categorize an unknown image, a statistical model created using pre-labeled samples is fed with the numerical representation of the visual features of images.

A supervised approach requires a set of labeled data to create a statistical model, and subsequently classify an unlabeled test set. However, labeling images manually requires a great deal of time and effort. Therefore, a major research activity has gravitated towards finding efficient methods to reduce the time and effort for image labeling. Most images on social websites have associated tags that somewhat describe their content. These tags may provide significant content descriptors if a semantic bridge can be established between image content and tags. In this thesis, we focus on cases where accurate class labels are scarce or even absent while some associated tags are only present. The goal is to analyze and utilize available tags to categorize database images to form a training dataset over which a dedicated classifier is trained and then used for image classification. Our framework contains a semantic text analysis tool based on WordNet to measure the semantic relatedness between the associated image tags and predefined class labels, and a novel method for labeling the corresponding images. The classifier is trained using only low-level visual image features. The experimental results using 7 classes from MirFlickr dataset demonstrate that semantically analyzing tags attached to images significantly improves the image classification accuracy by providing additional training data.

PREFACE

I have no special talent. I am only passionately curious.
Albert Einstein

This thesis concludes my Master of Science in Information Technology in the department of Signal Processing, Tampere University of Technology, Finland.

I would like to convey my sincere gratitude to my supervisor Dr. Esin Guldogan and examiner Professor Moncef Gabbouj and for giving me the opportunity for this work and their guidance, motivation and support. In addition, I would also like to thank Professor Serkan Kiranyaz for his time spent on technical discussions and teachings. During the process of experiments and writing of the thesis, Stefan Uhlmann was a very good friend who gave his professional comments as well as motivational advice on my work.

Finally, my family that supported me from the beginning to the end of my education in Finland, I would like to thank my family for their concerns, especially my mother for about my health, my father about my expenses and my brother about my social life.

Tampere, FINLAND, May 2013.

Ugur KIRBAC

Table of Contents

Abstract	I
Preface.....	II
1. Introduction	1
2. Content-Based Image Classification and Retrieval.....	4
2.1. Classification and Learning Types	4
2.1.1. Overview on Machine Learning	4
2.1.2. Supervised and Unsupervised Learning and Classification	6
2.1.3. CNBC: Incremental Evolution of Collective Network of Binary Classifier (CNBC)	9
2.2. Content Based Image Analysis.....	11
2.2.1. Visual Descriptors	11
2.2.2. Similarity Models	13
2.2.3. Indexing	14
2.3. Content Based Indexing and Retrieval Framework: MUVIS	15
2.3.1. MUVIS Overview.....	15
2.3.2. MUVIS Applications.....	16
2.3.3. Indexing and Feature Extraction.....	20
2.3.4. M-MUVIS System.....	22
3. Semantic Text Analysis Using Wordnet	25
3.1. Overview on WordNet and Other Semantic Networks	25
3.2. WordNet Based Similarity Measurement.....	25
3.2.1. Semantic Similarity between Sentences	25
3.2.2. Semantic Similarity between Two Synset and Query Sentences	29
4. The Proposed Framework	34
4.1. Automatic Labeling	36
4.2. Image Classification	39
5. Experimental Results.....	40
5.1. Preprocessing.....	40
1.1 Performance Evaluation of the Automatic Image Labeling	41
1.2 Classification Results	43
5.2. Retrieval Results.....	44
6. Conclusion.....	47
References	48

Table of Tables

Table 2-1 MUVIS multimedia family	16
Table 2-2 MUVIS image types	16
Table 3-1 Pseudo code for scoring algorithm	33
Table 4-1 Pseudo code of the automatic labeling framework.....	38
Table 5-1 The features and parameters used for image classification	40
Table 5-2 Predefined class sentences	41
Table 5-3 Precision of images labeled per class.	42
Table 5-4 Classification performances over the test set using ground truth training datasets with different sizes	43
Table 5-5 Classification performances over the test set using the training datasets automatically created using the proposed framework.....	44
Table 5-6 Retrieval results for ground truth.....	45
Table 5-7 Retrieval results for the NBCs of the training Sets.....	46

Table of Figures

Figure 1-1 The overview of the main framework	2
Figure 2-1 The flowchart of a supervised machine learning application.....	7
Figure 2-2 Left: Binary classification Right: 3-class classification	8
Figure 2-3 A simple coincidence matrix	8
Figure 2-4 Topology of CNBC framework.....	10
Figure 2-5 The original image.....	12
Figure 2-6 3 to 8 Dominant color image.....	12
Figure 2-7 Sample textures	13
Figure 2-8 The main view with all the functionalities of DBS Editor	17
Figure 2-9 Parameter selection for feature extraction.....	18
Figure 2-10 A snapshot of retrieval window of Mbrowse.....	19
Figure 2-11 Interaction of Fex Module with MUVIS Applications	22
Figure 2-12 Demonstration of M-MUVIS system architecture	23
Figure 2-13 Query process in M-MUVIS	24
Figure 3-1 An Example of the syponym saxonomy in WordNet.....	29
Figure 3-2 The Flowchart of semantic similarity between swo sentences.....	32
Figure 4-1 Tags: ford, 1965, mustang, car	34
Figure 4-2 Traditional way of labeling by an expert.....	35
Figure 4-3 Proposed labeling by automatic labeling framework.	35
Figure 4-4 A sample database of six images with associated tags.....	36
Figure 4-5 The illustration of the image labeling mechanism	37
Figure 4-6 The classification scheme with the Random Forest.	39
Figure 5-1 A visual representation of a class vector, as the red tone gets lighter the confidence of labeling decreases. This heterogenic representation facilitates the training set creation.	42

Abbreviations and Acronyms

AFeX	Audio Feature Extraction
ANMRR	Average Normalised Modified Retrieval Rank
API	Application Programming Interface
AVR	Average Rank
CBIR	Content Based Image Retrieval
CEO	Chief Executive Officer
CPU	Central Processing Unit
DC	Dominant color
DCD	Dominant Color Descriptor
DLL	Dynamic Linked Library
FeX	Feature Extraction
FV	Feature Vector
GPS	Global Positioning System
GUI	Graphical User Interface
GTD	Ground Truth Data
HCT	Hierarchical Cellular Tree
HSV	Hue Saturation Value
HTTP	Hypertext Transfer Protocol
MAM	Metric Access Method
ML	Machine Learning
MPEG	Moving Picture Experts Group
NMRR	Normalised Modified Retrieval Rank
NQ	Normal Query
PAM	Point Access Method
PQ	Progressive Query
QbE	Query by Example
QbR	Query by Region
QbS	Query by Subject
RAM	Random-access Memory
SAM	Spatial Access Method
SEG	Segmentation
SSL	Semi-supervised learning
YUV	Luminance-Bandwidth-Chrominance
WSD	Word Sense Disambiguation

1. INTRODUCTION

Pattern recognition is a collection of tools, algorithms and methods used for predicting the actual identity of a given unknown input such as image, video or text. Classification is an instance of pattern recognition that determines to which predefined class a given unknown input belong. For example, it can answer whether a given fruit is apple or banana using the statistical data of fruits.

Classification narrows down towards image classification that deals with only images whose content is predicted using a statistical model formed with the numerical representation of the visual features of images. This kind of approach is known as supervised approach that requires a certain set of labeled samples. In supervised learning, a classifier is trained with the manually labeled (training) dataset and the aim is then to classify the unlabeled examples in the test set. One can expect a higher classification performance as the size of the training dataset increases. In real cases, obtaining a certain set of training data is cumbersome process and takes much time of experts. Our motivation is to reduce the cost of the image labeling process for content-based image classification and retrieval.

Nowadays, user-created tags are available on social media websites such as Flickr. These tags are a useful data source without any expense for researchers. The research conducted on the use of Flickr show that users are eager to provide this semantic context through manual annotations [1]. In addition to describing the content of an image, tags might contain irrelevant words. Moreover, associated metadata attached to the image can also be used as the assistive textual information.

In this thesis, we have presented an approach to utilize tags associated with images for content-based image classification. Fundamentally, visual features and tags are two different but tightly related image descriptors, and in order to utilize both the visual information and user-created tags for image classification, we need to deal with two main challenges. The first challenge is to analyze tags semantically with an efficient and accurate text analysis algorithm to extract the accurate content labels and the second challenge is to establish a robust and effective bridge to use the semantic relationship for image classification. Figure 1-1 shows the general overview of the proposed framework. The mechanism starts with splitting the tag sets from the images in the database. The tags are analyzed and labeled with the categories in a predefined vocabulary. As a result, a training set is created for learning the image classifier. This training set is formed with no expense, in other words, no manual work has been used to label the images.

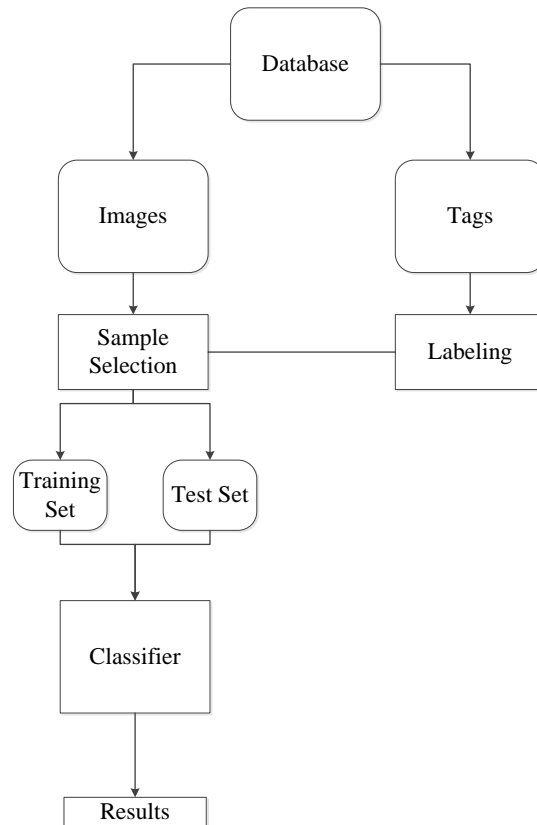


Figure 1-1 The overview of the main framework

Various approaches based on textual information have been proposed for visual classification tasks. For example, Jin et al [2] employed WordNet [3] ontology for removing irrelevant keywords produced during the process of image annotation. They investigate various semantic similarity measures between keywords and determine the correlation between associated keywords (tags) and the visual segments in images. Srikanth et al. [4] performed automatic image annotation by generating a visual vocabulary using WordNet ontology. Cho et al. [5] examined the conceptual relationship between keywords associated with images. They utilize WordNet hierarchy to find the semantic relationship between keywords in annotated images. After measuring the relationship, they removed irrelevant keywords from the whole keyword set and bridged the semantic gap between image content and the tags. The most related prior works are the two recent papers, [6] [7] both of which use the tags as the assistive information to improve the performance of the content-based image classification.

Wang et al. [7] formed a textual representation of the untagged images in a dataset that contains around one million tagged and untagged images. Their approach is using the visual features to obtain the textual data and they perform object-based image classification. Our approach differs in that we do not construct a new textual image representation. However, we both expect that textual features to capture the semantics of images and help to the image content analysis. In addition, we use only associated tags to obtain examples to train a classifier that uses only low-level visual features.

The work in [6] used a semi-supervised technique, which exploits the textual information by fusing with visual features to train a classifier. Their system contains two different classifiers. The first one was trained with both visual content and tags of the images and is used to label the unlabeled training set. Then the output of the first classifier was added to the existing training set for learning the second classifier.

The rest of the thesis is organized as follows. Section 2 describes content-based image classification and retrieval. Semantic analysis of sentences is explained in Section 3. The proposed framework is described in Section 4; we explain how image labeling is performed by semantic analysis of tags attached to images and the use of the labeled images for image classification. In Section 5, we demonstrate the experimental results of the proposed method. Finally, conclusions and future work are given in Section 6.

2. CONTENT-BASED IMAGE CLASSIFICATION AND RETRIEVAL

Content-based image classification is a significant step in image indexing and retrieval area. Content-based image retrieval (CBIR) methods were first proposed in the early 1990s [8] and researchers have studied on various methods to improve the accuracy of both classification and retrieval. These content-based methods have become more popular than text-based image retrieval methods, which are very subjective and noisy because of their use of human-created keywords, and very expensive because of manual processing. The goal of CBIR is to produce the best retrieval results corresponding to human concepts.

2.1. Classification and Learning Types

2.1.1. Overview on Machine Learning

Learning and intelligence are hard to define as they consist of complicated and multiple processes. Merriam-Webster [9] defines “learn” as follows: “To gain knowledge, or understanding of, or skill, by study, instruction or experience / and modification of a behavioral tendency by experience”. Zoologists and psychologists have studied learning in animals and people but here leaning in machines is more important, although there are some similarities between learning in animals and machines. As we know, psychologists have made computational models of their theories on animal and human learning and these techniques have then been transferred and used for machine learning. Some of the techniques and concepts researchers are looking at in the area of machine learning could also highlight forms of biological learning [10] .

The process of programming computers to learn is known as Machine Learning [10]. Computers are utilized for a wide set of tasks. For programmers designing and implementing the correct software is not overly challenging, although there are various tasks and these tasks can be organized into four categories.

First, no human experts exist for certain problems such as in modern automated manufacturing facilities where it does no good to study sensor readings in order to predict machine failures before they happen. This is due to the machines being new, so no knowledge can be communicated to a programmer to build a computer system. Whereas a machine learning system could analyze data and problems and learn to predict what causes the problems. In addition, there are problems where human experts exist especially in many areas of perceptual tasks where human experts exist such as speech and

handwriting recognition as well as natural language comprehension. Almost everyone has expert-level abilities in these areas but cannot explain the route they follow when undertaking the tasks. Luckily, machines can be given examples of the inputs and correct outputs for these tasks, so machine learning algorithms can learn to map the inputs to the outputs.

On the other hand, problems also exist where there is fast changing phenomena. For example, people would like to be able to predict the future behavior of the stock market, consumer purchases, or of exchange rates. These financial fields change so often that despite hopes of constructing a program that is able to predict these changes is almost impossible, as it would also require rewriting. A program that learns can help by continually modifying and tuning rules it has learnt to predict.

Furthermore, certain applications require separate customization for each user. A good example of this is a filter program to distinguish unwanted emails from useful ones. Each user will require their own different filters as it would be ludicrous to ask each user to program their own rules. It would also be impractical to supply each user with a software engineer to continually update the latest rules. A machine learning system, which would recognize which mails, is rejected and which are stored can decipher the filtering rules.

Research questions in the fields of statistics, data mining, psychology as well as machine learning deals with the same questions albeit with a different emphasis /focal point [11].

Statistics concentrates on understanding the phenomena that have generated the data, often with the goal of testing various hypotheses about the phenomena in question. Data mining seeks to locate comprehensible patterns in the data.

Psychological studies of human learning seek to comprehend the mechanisms that are the basis of the various learning behaviors exhibited by people (concept learning, skill acquisition, strategy change, etc.) [12].

As we can determine from the discussion on applications, the range of learning problems is extensive. However, researchers have identified multiple templates that can be applied in numerous situations. These templates make deployment of machine learning in practice easy and our discussion will largely focus on a choice set of such problems. We now give a by no means complete list of templates.

Machine Learning (ML) presents a number of applications, most importantly in data mining field. Machine learning can be used where multiple discover the relationship between multiple features [13]. Databases are created with the items that have the same kind of features. Considering pattern recognition systems, two types of learning mechanisms are very important: Supervised and Unsupervised Learning (instead of learning, classification can be used interchangeably in pattern recognition field). Unsupervised learning uses unlabeled items in a database whereas supervised learning is carried out if the items are labeled. Unsupervised algorithms result unknown but beneficial classes of instances whereas supervised learning requires predefined classes before classification [14].

2.1.2. Supervised and Unsupervised Learning and Classification

Unsupervised Learning algorithms do not require any labeled data that, in contrast, is the prerequisite for supervised learning algorithms. It seeks the hidden information of a bunch of unlabeled data. In theory, this type of learning does not have evaluation criteria since the input is unknown. One of the most commonly used types of unsupervised algorithms is clustering [14]. Clustering methods simply compute the similarity between instances to collect them into different groups. Various distance metrics exist in the literature and Euclidean is one of the most commonly used metric. Euclidean distance [15] between two n-dimensional feature spaces gives the numerical similarity measure of two patterns [16]. Researchers gravitate towards clustering because of several reasons:

- The collection and manually categorizing the training data set can be costly and time consuming.
- In some cases for data mining, natural clusters are chosen over manually created ones.
- The properties of feature vectors can vary as the database grows. Especially, in medical image classification area, testing data set might be different from the training data used for the classifier in the beginning.

Theoretically speaking, a set of feature vectors can be defined as

$$D = \{\mathbf{x}_1 \dots \mathbf{x}_n\} \quad (1)$$

Clustering problem is that grouping each feature vector into a cluster with a fixed and predefined size as c .

$$\bigcup_{i=1}^c D_i = D \quad (2)$$

$$D_i \cap D_j = \emptyset \quad (3)$$

For all $i \neq j$, we need some further assumptions for the problem to be sensible. (An arbitrary division of D into different classes is not likely to be useful.) Here, we assume that we can measure the similarity of any two feature vectors somehow. Then, the task is to maximize the similarity of feature vectors within a class. For this, we need to be able to define how similar are the feature vectors belonging to some set D_i .

Supervised learning is the process of learning a set of rules from manually labeled data so called training set. The purpose is creating a classifier that uses the small portion of a database as training set and uses the big portion of the database as test set. The flowchart of supervised ML application for a real-world problem is demonstrated in Figure 2-1. The first step is collecting the dataset, which is very expensive in some cas-

es. Mostly, an expert in the field suggests which fields (attributes, features) are the most informative [13].

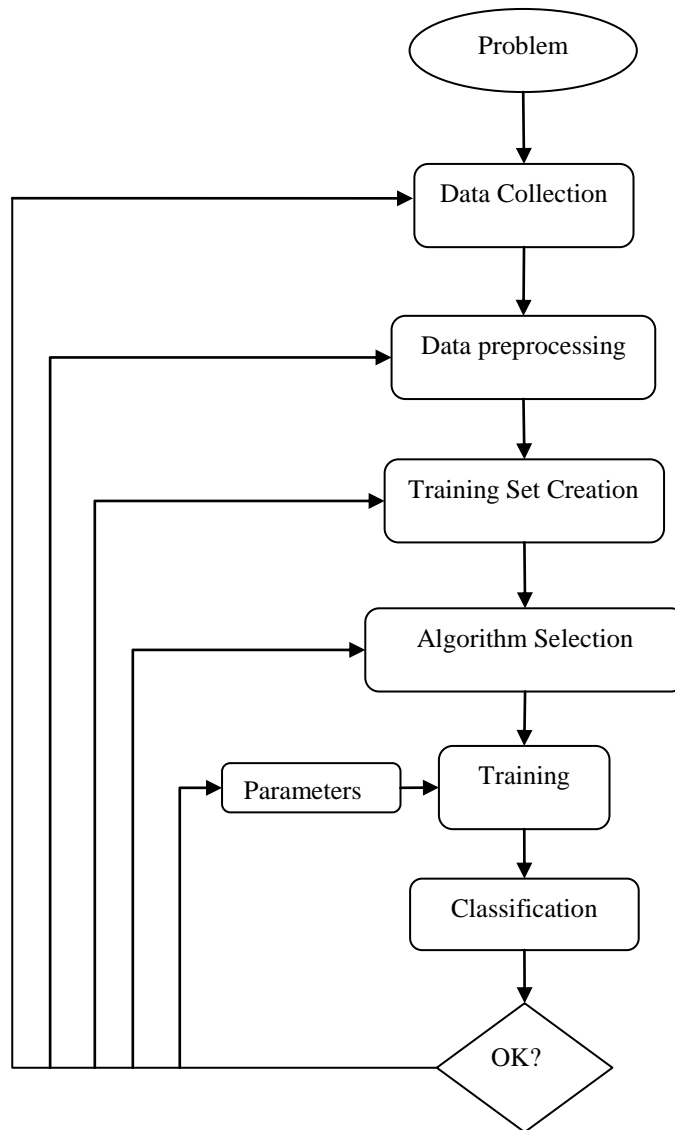


Figure 2-1 The flowchart of a supervised machine learning application

Semi-supervised learning (SSL) stands between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is given some supervision information, mostly for labeled examples. In this case, the dataset $X = (x_i) \in [n]$ can be divided into two parts: the points $X_l := (x)$, for which labels $Y_l := (y_1, \dots, y_n)$ are provided, and the points $X_u := (x)$, the labels of which are not known [13].

One of the most commonly studied issues in the machine learning field is possibly **Binary Classification** [17]. It has been used for a great deal of significant developments for a long time. Actually, the basic question is to which random variable $y \in \{\pm 1\}$ a pattern

x in X domain will be assigned. For example, given samples of cards on which are images of cherries and bananas, we want to categorize if the object is cherry or banana. Examples can be derived, however understanding the basic problem will help us to figure out most of the practical issues.

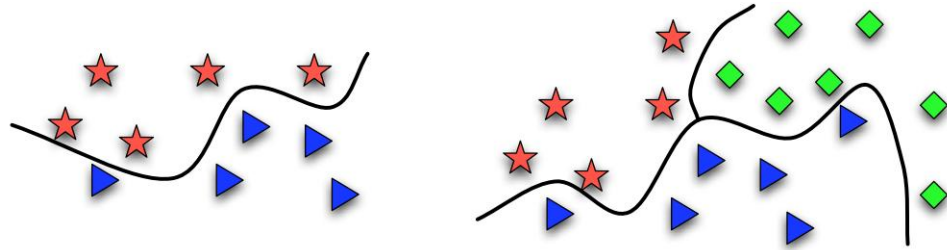


Figure 2-2 Left: Binary classification Right: 3-class classification

In the 3-class classification case that is illustrated in Figure 2-2, vagueness is higher. For example, separating the diamonds from triangles is not enough alone to categorize the objects accurately because we also have to separate the diamonds from the stars.

Multiclass classification deals with categorization of more than two classes. The fundamental distinction is that $y \in \{1, \dots, n\}$ can assume multiple values. For example, music can be divided into different genres such as art music, traditional music, and popular music based on the composers. The critical level of the error depends on the possible consequences. For example, in medical field, the significance of the accuracy is higher than e-mail spam classification [10].

In classification problems, performance measurements are carried out with the aid of coincidence matrix. Figure 2-3 demonstrates a generic coincidence matrix for a binary classification problem [18].

		True Class	
		Positive	Negative
Predicted Class	Positive	True Positive Count (TP)	False Positive Count (FP)
	Negative	False Negative Count (FN)	True Negative Count (TN)

Figure 2-3 A simple coincidence matrix

True outputs are demonstrated by lighter color while false decisions (errors) are dimmed. The true positive rate of a classifier is calculated by dividing the number of accurately categorized positives (the true positive count) by the total number of positives. The false positive rate of the classifier is calculated by dividing

the inaccurately categorized negatives (the false negative count) by the total number of negatives.

The overall accuracy of a classifier is calculated by dividing the total accurately categorized positives and negatives by the total number of patterns [18]. Below are the performance evaluation formulas.

$$\text{True Positive Rate} = \frac{TP}{TP+TN} \quad (4)$$

$$\text{True Negative Rate} = \frac{TN}{TN+FP} \quad (5)$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad (6)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (7)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (8)$$

- True positive (tp) – a pattern classified as class n that really was.
- True negative (tn) – a pattern classified as **not** class n , and really was not.
- False negative (fn) – a pattern classified as **not** class n , though it really was.
- False positive (fp) – a pattern classified as class n , though it was not.

2.1.3. CNBC: Incremental Evolution of Collective Network of Binary Classifier (CNBC)

A number of image classifiers have been studied and the collective network of binary classifier developed by Serkan et al. has been used in our experiment [19]. The unique characteristic of this classifier is that it does not need a complete training data in the beginning of the training. It creates a number of networks of binary classifiers (NBCs) to discriminate each category and optimal binary classifier is chosen in each of the NBCs by evolutionary search. Visual and digital performance measurements of the framework proved that this system is accurate and efficient for scalable CBIR and classification. In order to increase classification accuracy that leads to an improvement of the CBIR performance, a global framework that represents a collective network of evolutionary classifiers is used. This approach creates an assigned classifier to classify a class based on a particular feature. Each incremental session will “learn” from the current best classifier configurations and improve them [15]. Furthermore, new classes or features can be introduced with each incremental evolution to trigger the CNBC to create new corresponding NBCs and BCs within to *adapt* to the change dynamically.

The topology of CNBC is also worth mentioning. Figure 2-4 shows the topology of CNBC framework. In order to accomplish the scalability regarding to a varying number of classes and features, the CNBC framework accommodating a network of binary classifiers (NBCs) is created. In this case, NBCs can evolve with the current evolution sessions; it is performed using the training dataset that is created by collecting a set of data

(GTD) from some relevance feedback sessions. Each NBC stands for a specific image category and the number of evolutionary binary classifiers (BCs) in the input layer varies. Each BC conducts binary classification using a single (sub-) feature in the input layers. As soon as a new feature is appended, a new BC will be created in each NBC and evolved with the new set of training data. Thus, re-evaluations are prevented and scalability regarding to varying number of features is maintained.

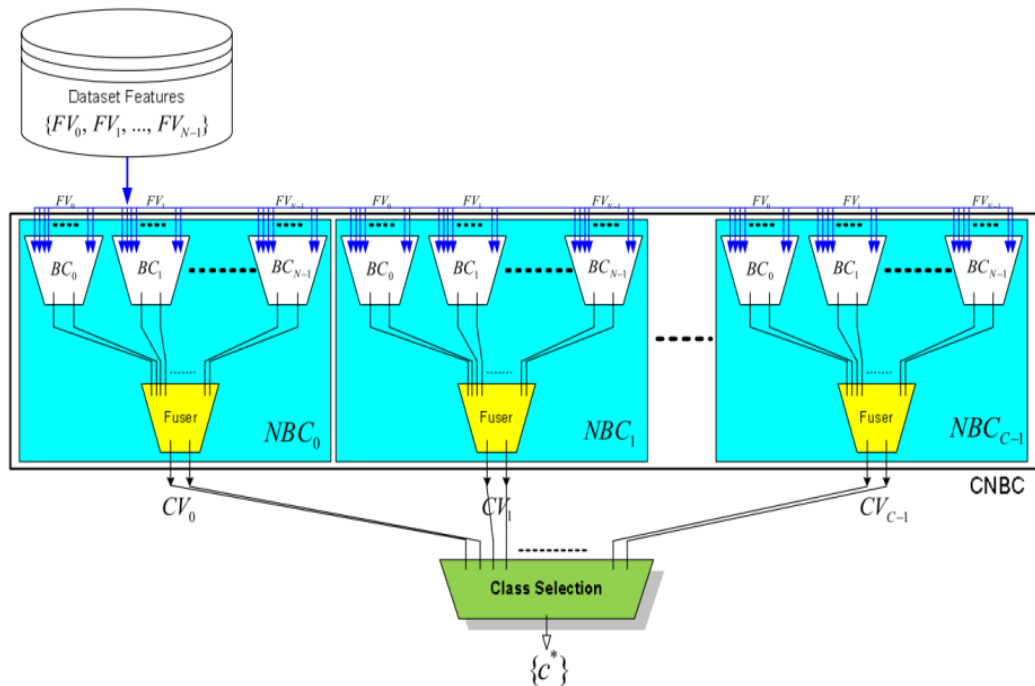


Figure 2-4 Topology of CNBC framework

Each NBC contains a “fuser” BC in the output layer, which produces a single binary output by collecting and fusing the binary outputs of all BCs in the input layer. These fusers indicate the relevancy of each feature vector (FV) to the NBC’s corresponding class. Furthermore, CNBC is also scalable to any number of classes because as soon as a new class is declared by the user, a new NBC can be created (and evolved) only for this class without any need of any modification or update of the other NBCs as long as they can classify its GTD above a certain accuracy required. In this way, the overall system dynamically adapts to varying number of classes. As shown in Figure 2-4, using as many classifiers as necessary is the fundamental idea of this approach. Therefore, we break a massive learning problem into many NBC units along with several BCs. Thus, we prevent the need of using complex classifiers as the performance of both training and evolution processes degrades significantly as the complexity rises due to the dimensionality problem.

2.2. Content Based Image Analysis

With the new era, image databases expand dramatically through social networks and applications. As a result, efficient storing and searching algorithms have become a hot topic for the researchers [8]. Mostly, images are indexed with the associated textual information. On the other hand, textual information is manually and subjectively created.

Images with various contents cannot always be described by a few words, therefore content based image indexing and retrieval has become more important in the new era. In addition to textual features, Content-Based Image Retrieval (CBIR) utilizes the visual features of images such as color, texture, and shape [20].

The origin of the use of the term CBIR in the literature is by Kato in 1992, which used this term for his research [21]. Query by Example (QbE) is one of the most common methods that inputs an example image whose features are extracted and compared with the features of other images in the database to compute the similarity between each other.

2.2.1. Visual Descriptors

Visual descriptors can be divided into two groups. One of them is low-level features (color, shape and texture) extracted from images by a feature extractor program on a computer and high-level features that are defined by humans. One of the motivations of CBIR research is to reduce the semantic gap between low-level and high-level features [22]. On the other hand, similarity distance measurement takes an important place in the recent research field.

MPEG-7 [23] defines a standard set of multimedia content. Visual descriptors are the heart of CBIR systems and they are categorized based on the features of content, such as color, texture and shape.

Color is a significant property, which defines objects and gives a distinctive perception to the humans [24]. Several color descriptors are specified in MPEG-7. In addition, several different color spaces such as YUV, HSV can be used for different purposes. In this section, a very short overview will be given about color descriptors.

Dominant Color Descriptor provides numerical information about an image. The information can be color values, distribution or variance. A comparison between an original image and 3 to 8 dominant color image is demonstrated in Figure 2-5 and Figure 2-6.

Color Structure Descriptor provides color distribution and some information about local color structure in an image by means of a structuring element.

Color Layout Descriptor provides the spatial distribution of color in an image.



Figure 2-5 The original image

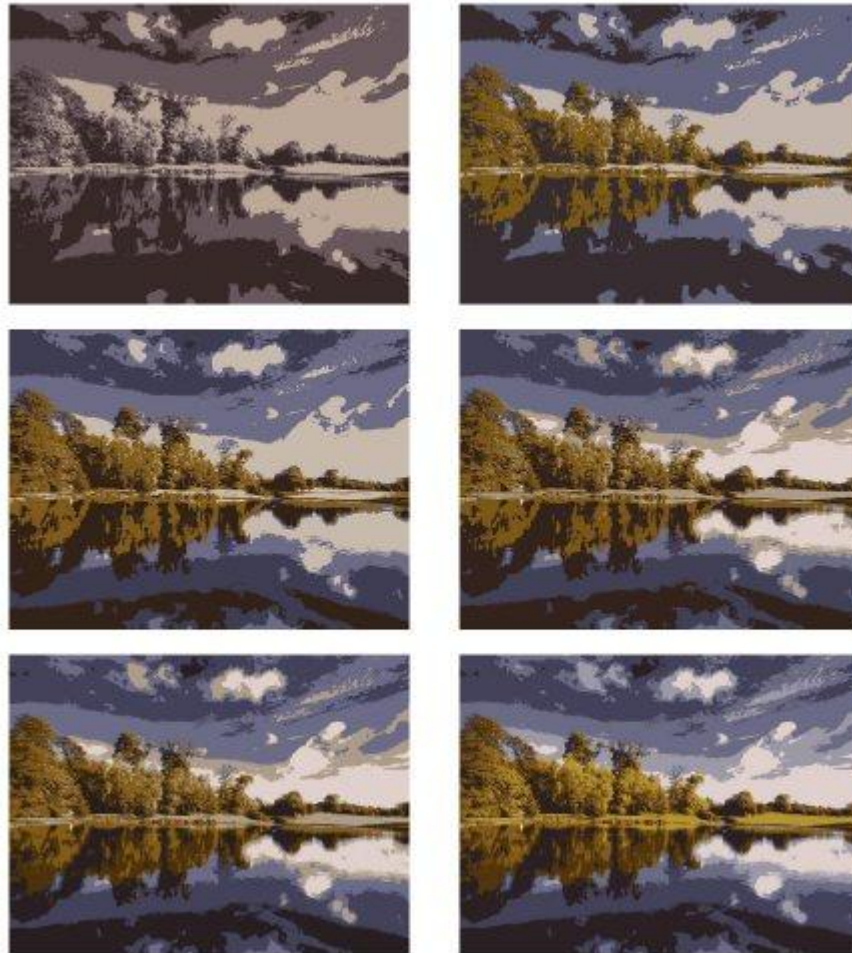


Figure 2-6 3 to 8 Dominant color image

Texture is an important visual feature for searching and browsing through large collections of homogenous patterns. Even though texture can be understood and associated with easily, no universally accepted formal definition exists in the literature. An image

texture stands for a set of metrics determined in image processing designed to measure the quality of the perceived texture of an image. It also provides a clue about the spatial arrangement of color or intensities in an image or selected region of an image. Image textures can be artificially formed or found in images and can be used for image segmentation or classification. Figure 2-7 demonstrates 4 sample textures.

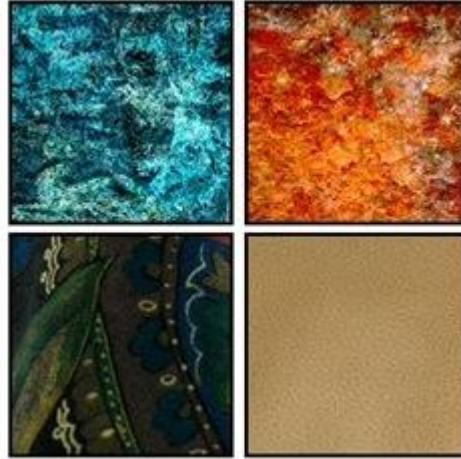


Figure 2-7 Sample textures

Shape of the objects in an image gives significant information about the content of the image. In terms of human perception, shapes alone can have semantic data and is a slightly more versatile concept than other low-level features such as color and texture. Two fundamental shape descriptors are used in CBIR systems. The Region Shape [25] descriptor captures the distribution of objects within a particular region whereas the contour shape descriptor captures specific shape features of the contour of region. Geometric Moments [26], Zernike Moments [27], [28], and Grid Representation [29] exploit the region-based approach. MPEG-7 also mentions in its standards that Zernike moments can be used for region-based description and Curvature Scale Space Descriptors for contour-based description [26], [27], [28].

2.2.2. Similarity Models

Similarity between multimedia items should be represented with a numerical model. Similarity distance is calculated with the aid of feature vectors extracted from multimedia items.

A few metric axioms must be confirmed to make m (distance function) valid and a_1, a_2, a_3 generic stimuli, the metric axioms are as follows:

$$m(\mathbf{a}_i, \mathbf{a}_i) = \mathbf{0} \text{ for } i = 1, 2, 3 \dots n \text{ (constancy of self-similarity)} \quad (9)$$

$$m(\mathbf{a}_i, \mathbf{a}_i) \geq \mathbf{0} \text{ for } i \neq j \text{ (minimality or non-negativity)} \quad (10)$$

$$m(\mathbf{a}_i, \mathbf{a}_j) = m(\mathbf{a}_j, \mathbf{a}_i) \text{ for } i, j = 1, 2, 3 \dots n \text{ (symmetry)} \quad (11)$$

$$m(\mathbf{a}_i, \mathbf{a}_j) + m(\mathbf{a}_j, \mathbf{a}_k) \geq m(\mathbf{a}_i, \mathbf{a}_k) \text{ for } i, j, k = 1, 2, 3 \dots n \text{ (triangle inequality)} \quad (12)$$

Most commonly used distance functions are:

$$m(\mathbf{a}_1, \mathbf{a}_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \text{ Euclidean distance} \quad (13)$$

$$m(\mathbf{a}_1, \mathbf{a}_2) = |x_2 - x_1| + |y_2 - y_1| \text{ City-block distance} \quad (14)$$

The metric model is the most commonly used model for computing similarity distance because of its advantages. For example, it helps indexing as well as providing consistency with feature-based description. Nevertheless, similarity metrics with feature vectors has some discrepancies with human perception of similarity.

Transformational distances are used to capture the similarity between shapes, which are transformed by deformation techniques. The quantity of deformation that enables two shapes coincide determines the similarity. This approach is based on the idea that, in order to evaluate the similarity between shapes, one shape is transformed into the other through a deformation process. Similarity between shapes is then measured through the amount of deformation needed to make the two shapes coincide. Elastic models use either a discrete set of parameters to model the deformation, or a continuous contour undergoing a continuous deformation and are used in Photobook [30].

2.2.3. Indexing

Indexing is one of the crucial components of CBIR systems. Dealing with huge collection of images, indexing decrease the amount of time spent in file access throughout query operation. Partitioning methods used for accessing the large image collections are very important to discuss for the sake of indexing. Three essential partition methods: Point Access Methods (PAMs) that partition the feature space, Spatial Access Methods (SAMs) that partition the database and Metric Access Methods (MAMs) [31] that partition the feature space by means of similarity distances. The examples of above methods are k-d Trees, R-Trees and SS-Trees [32] [33] For PAMs, k-d trees are binary trees. Each node is considered a k-dimensional point and every non-leaf node is considered a hyper plane that partitions the space into two half-spaces. R-tree divides the feature space into high dimensional sub-parts therefore, they are more appropriate for high dimensional feature vectors than K-d trees. In addition, SS-tree performs partitioning by means of minimum bounding spheres. One example of MAMs is the M-tree, which captures a number of points and associates each data point with its nearest representative. The Pyramid Technique [30] can be efficiently used for the higher-dimensional feature vectors

2.3. Content Based Indexing and Retrieval Framework: MUVIS

2.3.1. MUVIS Overview

MUVIS framework manages image retrieval related processes (indexing, browsing, querying, summarization) of the multimedia collections and accommodates applications for real-time audio and video capturing, encoding, database creation, multimedia conversion, indexing and retrieval [34].

MUVIS provides an interface that incorporates visual/aural feature extraction (FeX/AFeX) algorithms, SEGmentation (SEG) and Shot Boundary Detection (SBD) functions. It is based upon three applications, each of which has different responsibilities and functionalities. The first component, AVDatabase, is mainly responsible for real-time audio/video database creation with which audio/video clips are captured, (possibly) encoded and recorded in real-time from any peripheral audio and video devices connected to a computer. The second one, DbsEditor, performs the indexing of the multimedia databases and therefore; offline feature extraction over the multimedia collections is its main task. The last component, MBrowser, is the primary media browser and retrieval application into which PQ technique is integrated as the primary retrieval (QBE) scheme. NQ is the alternative query scheme within MBrowser. Both PQ (Sequential and over HCT) and NQ can be used for retrieval of the multimedia primitives with respect to their similarity to a queried media item (an audio/video clip, a video frame or an image). Due to their unknown duration, which might cause impractical indexing times for an online query process, in order to query an (external) audio/video clip, it should first be appended (offline operation) to a MUVIS database upon which a query can then be performed. There is no such necessity for images; any digital image (inclusive or exclusive to the active database) can be queried within the active database. The similarity distances will be calculated by the particular functions, each of which is implemented in the corresponding visual/aural feature extraction (FeX or AFeX) modules.

MUVIS databases are formed using the variety of multimedia types belonging to MUVIS multimedia family as given in Table 2-1. The associated MUVIS application will allow the user to create an audio/video MUVIS database in real time via capturing or by converting into any of the specified format within MUVIS multimedia family. Since both audio and video formats are the most popular and widely used formats, a native clip with the supported format can be directly inserted into a MUVIS database without any conversion. This is also true for the images but if the conversion is required by the user anyway, any image can be converted into one of the “Convertible” image types presented in Table 2-2.

Table 2-1 MUVIS multimedia family

MUVIS Audio				MUVIS Video			
Codecs	Sampling Freq.	Channel Number	File Formats	Codecs	Frame Rate	Frame Size	File Formats
MP3	16, 22.050,	Mono	MP3	H263+	1..25 fps	Any	AVI
AAC	24, 32, 44.1 KHz	Stereo	AAC	MPEG-4			MP4
G721 G723	Any		AVI	YUV 4:2:0			
PCM	Any		MP4	RGB 24			

Table 2-2 MUVIS image types

MUVIS Image Types							
Convertible Formats							
JPEG	JPEG 2K	BMP	TIFF	PNG			
Non-convertible Formats							
PCX	GIF	PCT	TGA	PCX	EPS	WMF	PGM

2.3.2. MUVIS Applications

MUVIS applications were firstly developed for Windows OS with specific Windows libraries however; those libraries have been integrated to Linux as well. Main features of the application are presented in the following sections.

DbEditor deals with indexing and other kind of editing tasks for the MUVIS databases. Audio/video clips can be created by a database application as well as available clips could be added to the MUVIS database.

Table 2-2 shows supported formats in the system. On the other hand, different formats can be converted and added to the MUVIS database. The fundamental task of DbEditor is feature extraction. The low-level features are extracted from the image and appended to any given MUVIS database. In addition, DbEditor is capable of modifying the existing features in a MUVIS database. All the functionalities are presented below.

- Appending and removing multimedia items such as audio/video clips and images
- Dynamic integration and modification of feature extraction (FeX and AFeX) modules.
- Extracting and removing features of multimedia items of a database by using available FeX and AFeX modules.
- Converting of various audio/video files into any MUVIS format
- Preview of multimedia items in a database.
- Display statistical information of a database and/or items in a database.

Figure 2-8 demonstrates the interface of DBS Editor. Figure 2-9 is a view of parameter selection window. Parameters control the type of feature vectors.

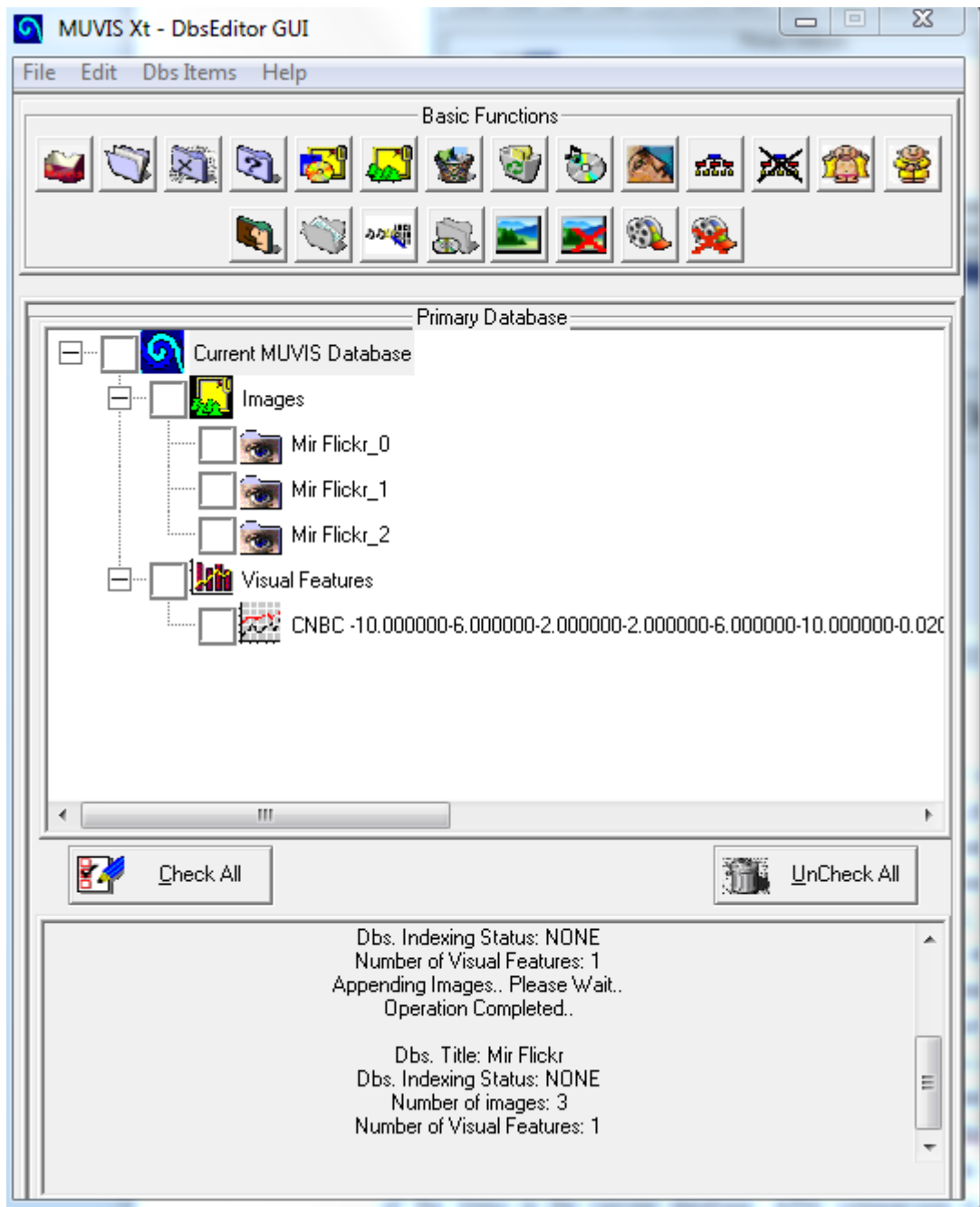


Figure 2-8 The main view with all the functionalities of DBS Editor

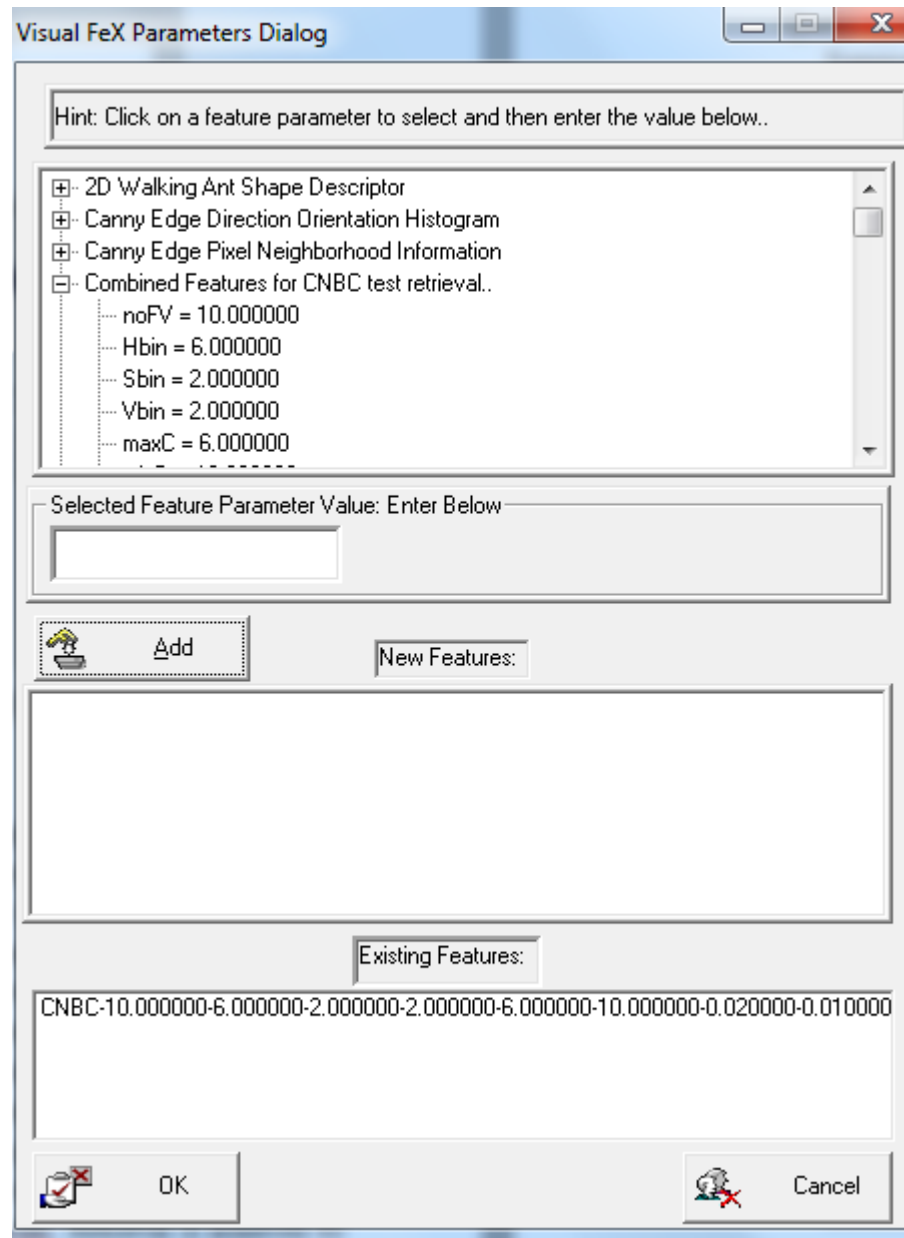


Figure 2-9 Parameter selection for feature extraction

MBrowser is the skeletal of the application has all the functionalities of a multimedia player and a robust multimedia database browser. In addition, users are able to access any kind of multimedia items in various hierarchic stages. Video display hierarchy is composed of five different levels. These levels are single frame, shot frames (key-frames), scene frames, a video segment and full video clip. MBrowser is implemented with a search and query engine that is able to perform query operation. Query operation is conducted to find the similar multimedia items to the query items. Query image does not need to be in the active database, any kind of external digital image can be used as a query image. The application first appends the query image to the MUVIS database and

achieves the query operation. Query retrieval is another important component of MBrowser. Retrieval is achieved by comparing the feature vector(s) of the query multimedia item with the feature vector(s) of the items in the current database. After comparison is accomplished, ranking process of similarity distances takes place and program returns retrieval results for the query primitive. Similarity distance used for ranking is measured with special functions implemented separate modules. Progressive Query (PQ) is another attractive method for query operation in MBrowser. Normal Query (NQ) is the simplest form of query operation that retrieves the total number of matched items. Compared to NQ, PQ is a novel approach for query retrieval. It returns instant query results and allows the user check the preliminary results. Users can stop the query process if they are satisfied with the provided query results. An example PQ approach is shown in Figure 2-10. Progressive Query approach will be elaborated in the next subsection.

Summarizing the functionalities of MBrowser, we can use some bullet points.

- Video summarization via scene detection and key-frame browsing,
- Random access support for audio/video clips,
- Displaying any crucial information (i.e. database features, parameters, status, etc.) related with the active database and user commands,
- Visualizations of feature vectors of the images and video key-frames.
- Various browsing options: random, forward/backward and aural or visual HCT (if database is indexed via HCT).

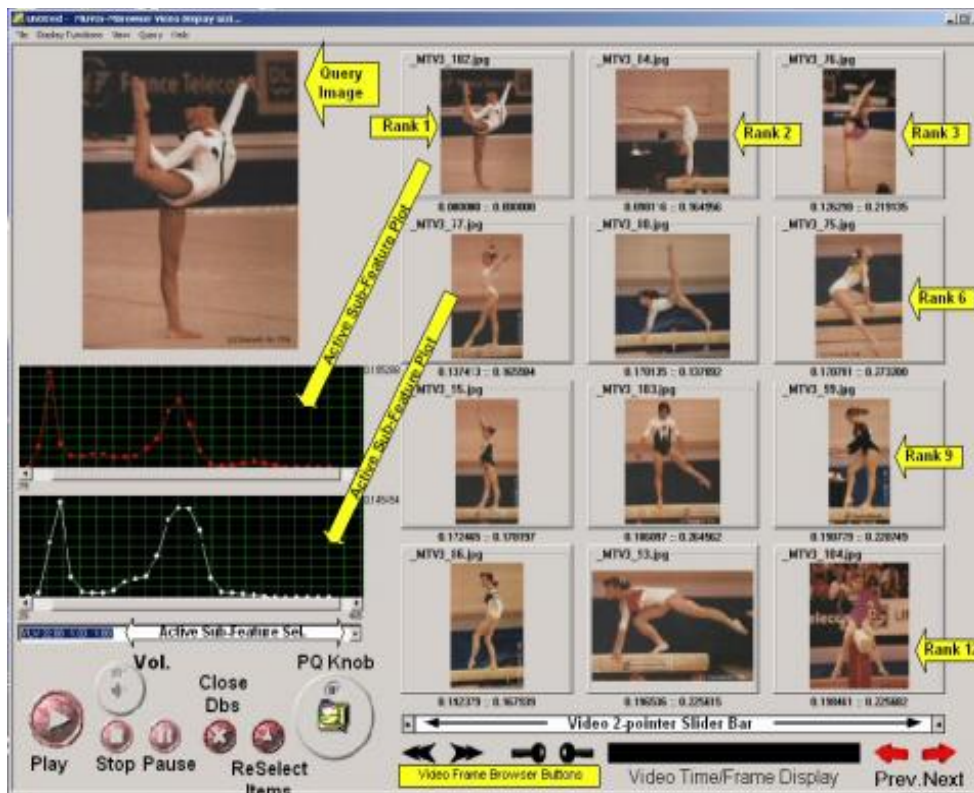


Figure 2-10 A snapshot of retrieval window of Mbrowsers

2.3.3. Indexing and Feature Extraction

DbEditor, as briefly mentioned in the previous section, performs the indexing of MUVIS databases. The process is achieved in three steps. Database creation is a compulsory and first step of the process. This stage deals with sequential indexing that indexes (gives a number to each item) the multimedia items in the database. Two optional steps that are used for fast query and Hierarchical Cellular Tree (HCT) browsing functionalities follow the first step. The second step deals with feature extraction by means of FeX and AFeX modules. Once features are extracted, the third step performs HCT indexing. Unique characteristics of HCT indexing are as follows.

- Dynamic (Incremental) indexing scheme.
- Parameter invariant (None or minimum parameter dependency)
- Dynamic cell size formation.
- Hierarchic structure with fast indexing (i.e. $\sim O(n \log n)$) formation.
- Similar items are grouped into cells via Mitosis operation(s).
- Optimized for PQ.

Indexing a MUVIS database has a speed advantage. When HCT is used to index a MUVIS database, the similar items can be retrieved faster through “PQ over HCT”. In addition to speed advantage, HCT browsing scheme that is the advanced browsing scheme is activated in MBrowser interface. HCT indexing is not the main requirement for PQ, it is possible to use progressive query with the aid of sequential indexing. Progressive query with sequential indexing is called Sequential Progressive Query.

MBrowser accommodates two main retrieval schemes for the multimedia items in a MUVIS database: browsing and query-by-example (QBE). In addition to those retrieval schemes, MBrowser provides three different browsing methods: sequential, random and HCT. Indexing is compulsory only for the first two methods. Based on the features in the database as well as the type of the database, visual and aural browsing can be performed with HCT browsing. In the cases where both visual and aural features exist in the database, which means the database is hybrid or video database, both of the browsing methods can be performed. Nevertheless, for the databases that contain only visual features (i.e. images), only visual HCT browsing is possible.

As shortly mentioned above, two QBE methods are available: Normal Query (NQ) and Progressive Query (PQ). NQ is the basic QBE operation and it utilizes the aural or visual features (or both) of the queried multimedia item (i.e. an image, a video clip, an audio clip, etc.) and all the database items. The algorithm computes the similarity distances between feature vectors and then merges them to get a particular similarity distance for each database item to the query item. All the items are ranked according to their similarity distances and the list of the ranked items is the result of the query. NQ has some drawbacks. It is computationally expensive, uses much of the system re-

sources such as CPU and RAM especially for huge databases. These drawbacks have led us to implement more efficient and robust algorithm for query operation. Thus, Progressive Query (PQ) was born. It is an alternative retrieval approach provides instant sub-results of the query. Therefore, users can interact with the immediate results through MBrowser. MBrowser allows users to browse and control the query operation after the first set of results. Users can stop the query operation if the first set of results is very satisfactory. Eventually, PQ and NQ will return the same set of retrieval results however, PQ is faster than NQ especially if HCT is used to index the database and PQ is performed with HCT.

A number of techniques can collect multimedia items. For example, video samples can be captured on real-time and converted to a format that MUVIS recognizes. Once multimedia items are appended to a MUVIS database, their features are extracted and stored in order to accomplish sequential indexing scheme for the database. Visual and aural indexing schemes are achieved with the aid of visual and aural feature extraction frameworks. These modules can be separately implemented as dlls and dynamically integrated to MUVIS system. This mechanism allows developers to integrate third party libraries to the system. Next two paragraphs describe the details of visual and aural feature extraction systems.

Video clips and images provide visual features for a MUVIS database. Features of video clips are extracted from the key-frames of the video clips. During real-time recording phase, AVDatabase may optionally and separately store the uncompressed (original) key-frames of a video clip along with the video bit-stream. If the original key-frames exist, they are utilized feature extraction process. If not, DbsEditor can extract the key-frames from the video bit-stream and use them instead. The key-frames are the INTRA frames in MPEG-4 or H.263 bit-stream. In most cases, a shot detection algorithm is used to select the INTRA frames during the encoding stage but sometimes a forced-intra scheme might be present in order to prevent possible degradations. Image features on the other hand are simply extracted from their 24-bit RGB frame buffer, which is obtained by decoding the image.

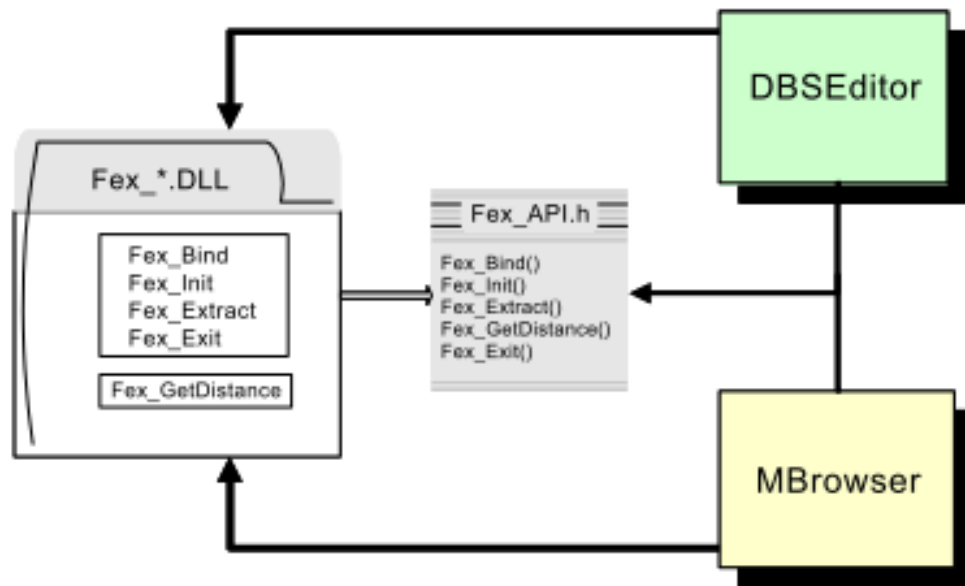


Figure 2-11 Interaction of Fex Module with MUVIS applications

The rest of the implementation details of FeX structure are similar to AFeX: each visual FeX module should be implemented as a Dynamic Link Library (DLL) with respect to FeX API, and stored in a suitable directory. FeX API establishes the communication and handshaking between a MUVIS application and the feature extraction module. Figure 2-11 demonstrates the API functions and the basic interaction between MUVIS applications and an illustrative FeX module.

2.3.4. M-MUVIS System

Social media gravitates towards to mobile environment. Google's CEO Eric Schmidt noted that mobile world is growing faster than their expectations [35]. Nowadays, mobile phones are faster and more powerful than it used to be. This huge growth renders all PC applications executable in mobile environments. M-MUVIS is a content-based image retrieval system implemented with both Java and C++ [36]. The Image query started by the client application in a mobile device is performed in the server side by means of native C++ code. Query results are sent to the mobile device over network and screened by the user.

M-MUVIS can be divided into two main parts that are client and server side applications. Since the M-MUVIS server have been created using both d native C++ code we can take advantage of scalability, and portability of Java and fast execution of C++ native code. The communication protocol satisfies the criteria of information retrieval most of wireless devices [37].

The client application consists of three components/packages.

- a) Query Information Engine: Various data about query are utilized in this engine. This module handles the user activities prepared by user interface in the client device.
- b) User Interface: This module handles user interaction with the device such as query operations.

c) Communication: HTTP [34] protocol is used for the communication over (GPRS) [38] between device and the server.

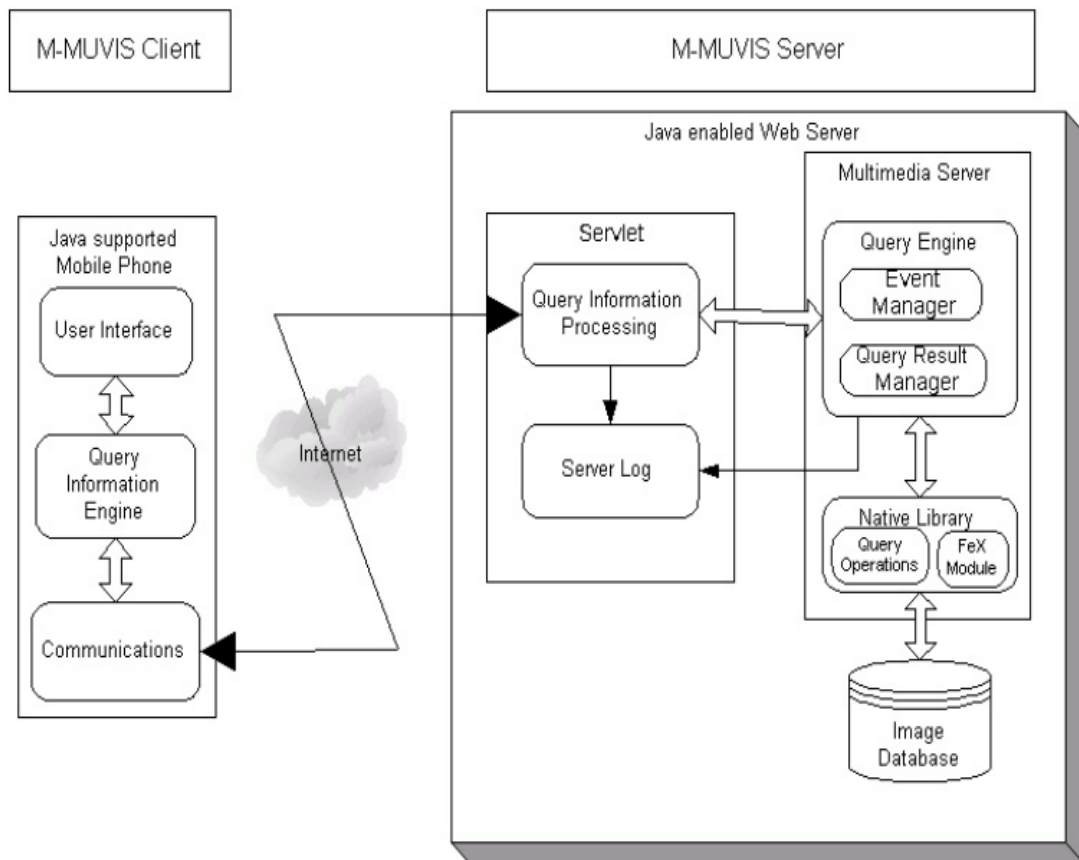


Figure 2-12 Demonstration of M-MUVIS system architecture

The essential components of the M-MUVIS server are as follows:

- Native Library
- Query Information Processing
- Server Log
- Query Engine
- Event Manager
- Query Result Manager

The M-MUVIS server accommodates a servlet running inside Java enabled Tomcat web server that contains a database with images that are scaled down to sizes appropriate mobile device screens. After query is received, servlet parses and processes it. Query Information Processing module passes the information of the query to Query Engine and query operation is carried out in Native Library. The query operation can use one or more combination of low-level features. The similarity search is carried out by comparing the feature vectors of query image and the images in the database. After all the similarity distances are computed and ranked, the first 12 images (the most simi-

lar images to the query image) are retrieved. The similarity distances are calculated by using the feature vector of queried image with feature vector of the images in a database. Ranking operation is performed afterwards and the retrieval result is formed using the best-12 ranked images. Event Manager invokes retrieval result event and Query Engine sends the retrieval results to Query Information Processing module. The client receives the results as HTTP format and retrieves the images one by one from the server. The query process is demonstrated in Figure 2-13.

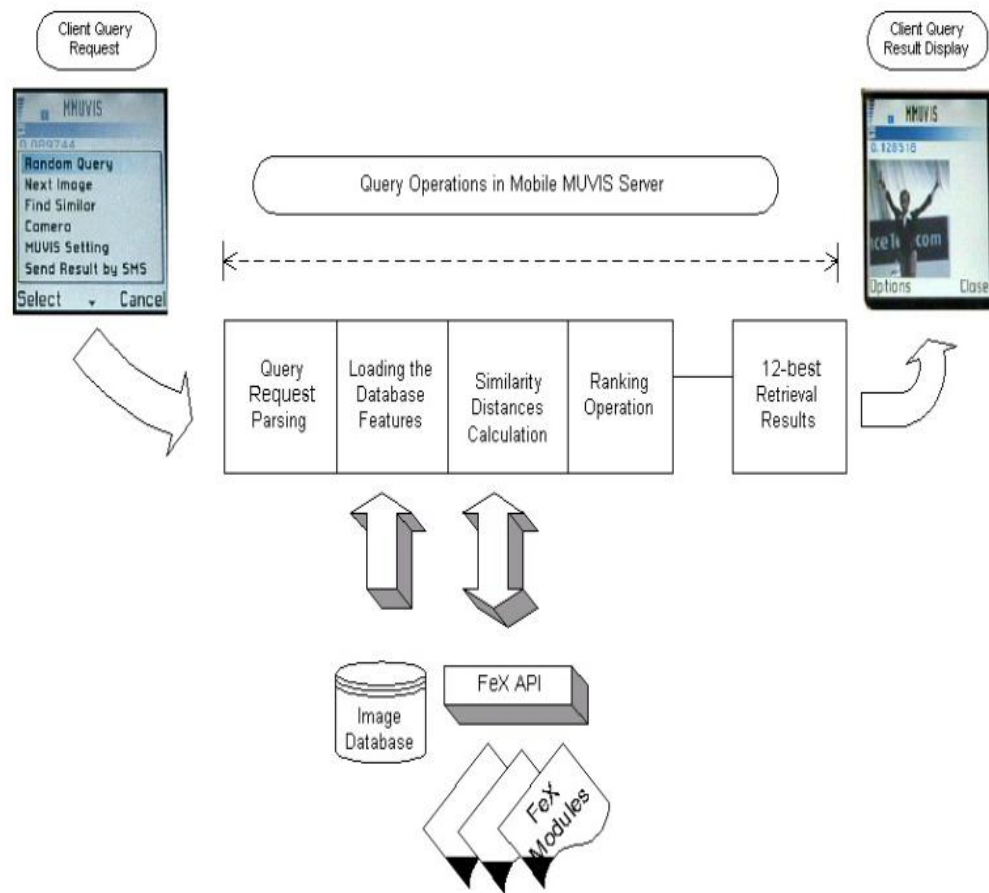


Figure 2-13 Query process in M-MUVIS

Even though, M-MUVIS system showed good results, it had some software and hardware limitations, but they are to be solved with the new technological improvements such as 3G and smart phones.

3. SEMANTIC TEXT ANALYSIS USING WORDNET

3.1. Overview on WordNet and Other Semantic Networks

WordNet is a taxonomy that provides a huge lexical database of English language. Nouns, verbs, adjectives and adverbs are collected into groups of cognitive synonyms (synsets) [3](Miller, 1995)(Miller, 1995)(Miller, 1995). Each synset in the taxonomy corresponds to a gloss that explains the concept of its words. For example the words *baby*, *infant* form a synset, this is defined with this gloss in WordNet: *a very young child (birth to 1 year) who has not yet begun to walk or talk*. Synsets are represented as nodes in WordNet taxonomy and the nodes are linked to each other with a particular relationship. Hyponym indicates that two synsets have *is-a-kind-of* relation. Meronymy represents *is-a-part-of* relationship. For example, *retriever* is a kind of *dog* and *retriever* is a hyponym of *dog*. Antonymy is a opposition relationship such as *long-short*, *female-male*.

In our work, we have used WordNet.Net module for semantic similarity measurement by Simpson and Dao [39].

3.2. WordNet Based Similarity Measurement

3.2.1. Semantic Similarity between Sentences

Semantic relatedness is a more generic concept than semantic similarity because it also covers antonymy and meronymy relationships. Related concepts are not necessarily similar. For example, *female-male* are not similar entities but they related in antonymy manner. Relatedness is a more necessary component for most of the applications compared to similarity [40].

Semantic similarity measurements calculate the semantic distance between two sentences. The output is the confidence score which indicates how similar two documents are, meaning as the score increases the semantic relation increases.

The essential steps for semantic measurement are described in five sections.

- Tokenization
- POS tagging
- Stemming words
- Finding which sense of a word is active in a specific context (Word Sense Disambiguation)

- Computing the similarity between sentences

Tokenization is the process of breaking a set of text up into words, which are called tokens. In addition, stop words that are unimportant words such as article, web pages are eliminated.

Part of speech tagging (POS tagging or POST) is the process of assigning a part of speech to the words in a text based on its context and definition. POS can be noun, verb, pronoun and adverb. For example, a sentence “John eats an apple” can be decomposed as John-noun, eats-verb, an-determiner, apple-noun. The tagger algorithm performs the tagging with a sentence as input and a specified tag set (a finite list of POS tags) and gives an output, which is a single best POS tag for each word.

It is worth mentioning two kinds of taggers. Rule-based taggers use hand written rules to disambiguate tag ambiguity, for example Brill's tagger [41]. Stochastic taggers resolve tagging ambiguities by using a training corpus to compute the probability of a given word having a given tag in a given context.

Stemming words is performed with Porter stemming algorithm in order to remove suffixes from words. Terms or words with a common stem mostly have a similar meaning, for example:

DEVELOP
DEVELOPED
DEVELOPING
DEVELOPMENT
DEVELOPMENTS

Frequently, the performance of an information retrieval system will be improved if term groups such as this are conflated into a single term. This can be achieved by simply removing the suffixes, -ED, -ING, -MENT, -MENTS, to leave the root term DEVELOP. In addition, the suffix stripping process will reduce the total number of terms in the information retrieval system, and hence reduce the size and complexity of the data in the system, which is always advantageous.

Word Sense Disambiguation identifies the meaning (sense) of a word in a particular sentence. The lexical ambiguity of a word refers to the fact that one word having more than one meaning in the language [42]. Anything can be ambiguous if it is open to more than one interpretation. For instance, consider two instances of the different senses of written form of word "*spring*":

1. *Springtime, the season of growth*
2. *fountain, outflow*

and the sentences:

1. *I want to stay here until the spring of this year.*
2. *I like spring water.*

Such situations can lead to some problems while finding the similarity of two words. Humans can distinguish the meaning of words by looking at the context. However, in computational linguistics, word-sense disambiguation (WSD) is a problem of language processing, while finding the meaning used in a sentence, and when the word has multiple meanings, which is known as polysemy [42]. A number of supervised approaches have been studied [43], [44], [45]. In addition, a model for WSD is designed based on decision trees using a corpus that consists of 22 million tokens, after manually sense-tagging around 2000 harmonic lines for five test words [46].

Another most commonly used method is the Lesk algorithm [47], which is a dictionary-based method. The algorithm is based on the theory that words used in a text stream have semantic relatedness and the relatedness can be determined with the aid of dictionary definitions, so called gloss, of the words. The definitions can also be used to compute the semantic similarity of each pair of word senses in a lexical network such as WordNet.

The main goal is to find the number of words used in common in both glosses. Words overlapping indicate the semantic relatedness of two glosses. For example, [47] performs disambiguation algorithm for the *pine cone* word pair, the word *pine* has two senses in the Oxford Advanced Learner's Dictionary,

Sense 1: kind of **evergreen tree** with needle-shaped leaves,

Sense 2: waste away through sorrow or illness.

The word *cone* has three senses:

Sense 1: solid body which narrows to a point

Sense 2: something of this shape, whether solid or hollow

Sense 3: fruit of a certain **evergreen tree**

Comparing the two senses of the word *pine* with the three senses of the word *cone*, *evergreen tree* can be observed as the most encountered sense for both words. Thus, for the *pine cone* word pair, Sense 1 of *pine* is distinguished from the Sense 2 and Sense 3 of *cone* is distinguished from Sense 1 and Sense 2.

In our framework, we used the extended gloss overlap measure algorithm [48] because of the certain constraints of Lesk algorithm.

This algorithm can access a dictionary where adverbs, nouns and adjectives are linked through semantic relations in taxonomy such as WordNet. In this algorithm a better scoring mechanism compared to bag of words counter is used. In the disambiguation process, each word in a sentence is considered a target word. The total number of words in a sentence is represented with N. The steps of the algorithm are given below.

1. *Context**Selection*

Context selection is the first step of the algorithm. To reduce the processing time in the cases where N is big, K number of context which is placed in a k-nearest neighbor fashion both at the left and right side of the target word is defined. By doing so, we are aiming to reduce the computational space and decrease the processing time accordingly. For example: If k is six, there will be three words to the left of the target word and three words to the right.

2. *Searching**words*

Searching words is performed to find all possible senses of verbs and nouns in a context.

3. *Finding**Conceptual**Relations**between**Words*

WordNet provides brief definition for words. These definitions are called gloss. The algorithm seeks the gloss of synonym sets (synset) which are interlinked by means of hyponym, hypernym, meronym and tropym.

4. *Gloss**Pair**Combination*

In order to measure the relatedness between gloss pairs, overlap of words between two pairs are sought and the overall relatedness score is computed using the separate scores of each pair. The abbreviations stand for comparisons between conceptual relations. For example, measuring the relatedness between two particular synsets s1 and s2, the $Score(hypo(s1)-hype(s2))$ means that the gloss of hyponym of s1 is compared with the gloss of hypernym of s2.

$$TotalScore(s1,s2)=Score(hype(s1)-hypo(s2))+Score(gloss(s1)-hypo(s2))\dots \quad (15)$$

In addition, the order of the input does not change the overall score, in other words,

$$TotalScore(s1,s2) = TotalScore(s2,s1) \quad (16)$$

The *pine cone* example above has eighteen possible combinations of senses, six senses for the cone and three for pine. As the scoring mechanism, a new algorithm which distinguishes N-single words from N-consecutive words is used. It also uses glosses as a bag of words. The task is to examine the overlaps between two text streams. The scoring mechanism can be described such that M sequentially overlapping words results M^2 units of score, meaning that an overlapping set of words "W X Y Z" results $4^2 = 16$ units of score whereas "X Y" is $2^2 = 4$ and "X Y" + "W Z" is $2^2 + 2^2 = 8$.

5. After each pair has been successfully scored, the sense which has the highest score is chosen as the most appropriate sense of the target word.

Computing the similarity between sentences is based on the path length between the word senses that are distinguished at the previous step (WSD). Every word sense in WordNet has its own synset, subsequently own node. Similarity between two given sentences is computed based on the path-length distance between the synsets.

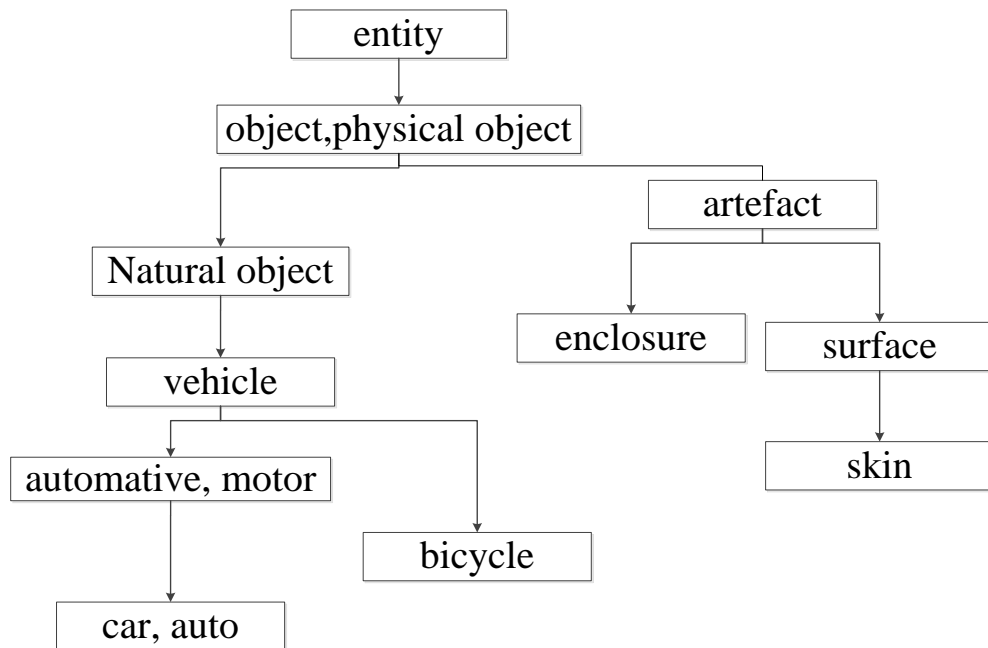


Figure 3-1 An example of the hyponym taxonomy in WordNet

In Figure 3-1 the distance between {car, auto} synset and {automotive, motor} synset is 2 while the distance between the concepts in the same synset is 1. The similarity score is calculated as follows:

$$\mathit{SimScore}(A, B) = 1/\mathit{distance}(A, B) \quad (15)$$

For example, if we are given two sentences A and B where A = {babies drink milk}, B = {Milk is produced from cow}, the algorithm first tokenizes the sentences, thus each word can be processed separately. Then, it performs word stemming and POS tagging to use the words in a dictionary to later perform word sense disambiguation. After the most appropriate word sense, subsequently synset, is computed by WSD a semantic similarity matrix is formed.

3.2.2. Semantic Similarity between Two Synset and Query Sentences

The method explained in the previous section can be considered *divide and conquer* algorithm since we break down the problem into more sub problems of the same type therefore, it becomes simple enough to be solved. It is used to find the best sense of each word in a given sentence. The semantic similarity between words senses is the main component to find semantic similarity between sentences. Finding the semantic similarity between word senses allows us to find the similarity between sentences.

The most appropriate senses for the words in a sentence can be extracted using the method explained above. In order to compute the similarity between two sentences, the semantic similarity between word senses is taken into account. Path length is used to measure the similarity between the word senses.

In WordNet hierarchy, if a word has multiple senses, it will be represented in multiple synsets in the different parts of the taxonomy. WordNet records the various semantic relations between synsets as well as the relations between word senses. The distinction between semantic and lexical relations is crucial. The difference between the semantic relation and lexical relation is that semantic relation describes the relation between two whole synsets while lexical relation describes the relation between words of two different synsets. For example, some semantic relations are hypernym, hyponym, holonym whereas antonym is lexical relations. In WordNet hierarchy, the antonym of the second sense of the noun *girl* (*girl#n#2*) is the first sense of the noun *boy* (*boy#n#1*). The synset of the noun *girl* is {*girl#n#2*, *female child#n#1*, *little girl#n#1*}. The antonym of *female child#n#1* is *male child#n#1*.

We have examined three different measurements methods. The first one is path length-based similarity measurement. Each similarity measurement has unique characteristics and can be used for different applications.

The path length-based similarity measurement is a popular way of measuring the similarity between nodes. The similarity between synsets is measured based on hyponym-hypernym relation, which is also known as is-a relation. However, some limitations exist in is-a relation therefore, only two types of POS, *noun-noun* and *verb-verb* are used in the framework. Taxonomy is treated as an indirect graph to measure the distance between synsets in WordNet.

In the WordNet noun hierarchy, the term subsumer refers to a specific shared ancestor of two concepts (or synsets). The least common subsumer (LCS) of two concepts is the closest node to them [49]. More particularly, the LCS of two synsets is the most specific subsumer of the two synsets, which is at the lowest level. In the example above, the LCS of {*auto*, *car*} and {*truck*} is {*automotive*, *motor*}, because the {*automotive*, *motor*} node is the closer node than the common subsumer which {*wheeled vehicle*}.

The path length is used to determine the relatedness and the relatedness between concepts is inversely proportional to the path length between two synsets. In other words, the shorter the path length is the more related the words are.

Two synsets of the same part of speech does not need to have a common subsumer. However, if a particular ancestor node is used; two synsets will have a connection between each other. In WordNet, synsets are allowed to have more than one ancestor. However, in the cases where a number of paths exist between two synsets, the shortest path is chosen. Lemmatization is done before looking up the word in WordNet. Therefore, the distance between *pen* and *pens* is 0, because they are identical. The word *worse* has *bad* as its lemma. This link is missed by stemming, as it requires a dictionary lookup. In addition, the word *talk* is the base form for word *talking* and henceforth it is

matched in lemmatization. Different POS are not compared since they are in different taxonomies. Therefore, verbs are compared only with verbs as well as nouns are compared only with nouns.

This measurement technique is called **MS1** and it has the advantage of being simple to implement and have sufficient performance compared to the other similarity measures such as P. Resnik, Wu and Palmer, Leacock and Chodorow, [50] [51]. Our method can be demonstrated as follows:

$$\text{Sim}(s1,s2) = 1/\text{distance}(s1,s2); \quad (16)$$

where distance is the path length from s1 to s2 using node counting.

The second one is called MS2 and in the method below, the order of the sense is utilized in addition to the path.

$$\text{Sim}(s1,s2) = \text{SenseWeight}(s1) * \text{SenseWeight}(s2)/\text{PathLength} \quad (17)$$

where s1 and s2 are the target words that similarity between them is measured. *SenseWeight* function indicates the ratio of the frequency of the use of input (sense s1) to the frequency of the use of all senses. *PathLength*, as indicated before, is the node count from s1 to s2.

For the given inputs, the flowchart is illustrated below. For the sentences X and Y, with the lengths m and n respectively, the major steps of finding the similarity can be described as follows:

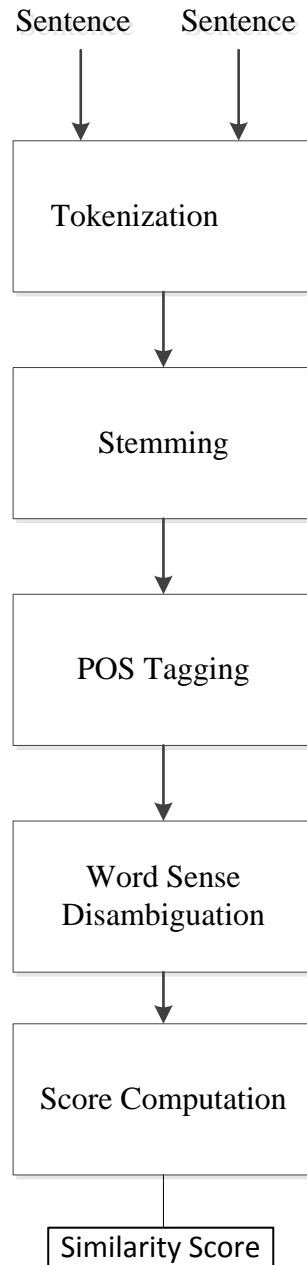


Figure 3-2 The flowchart of semantic similarity between two sentences

Forming a Semantic Similarity Relative Matrix $R [m, n]$ of each pair of word senses at fifth step is quite straightforward. The semantic similarity between the best sense of the word at position i of X and j of Y is represented by $R [i, j]$. Thus, $R [i, j]$ is also the weight of the edge connecting from i to j . For the words that do not have any gloss in the dictionary, edit-distance similarity is used and it results a lower associated weight. For example, an acronym OSL (Open Source Lab) edit-distance similarity is used. The formula used to capture semantic similarity is a problem of calculating a maximum total matching weight of a bipartite graph. Disjoint nodes are defined as X and Y . In the cases where computational time is important, fast heuristic method is used. The pseudo code is given in Table 3-1.

Table 3-1 Pseudo code for scoring algorithm

```

ScoreSum<-0

foreach (X[i] inX){
  bestCandidate <- -1;
  bestScore <- -maxInt;
  foreach (Y[j] in Y){
    if (Y[j] is still free && r[i, j] > bestScore){
      bestScore <- R[i, j];
      bestCandidate <- j;
    }
  }
  if (bestCandidate != -1){
    mark the bestCandidate as matched item.
    scoreSum <- scoreSum + bestScore;
  }
}

```

The final similarity score is computed through the formulas which are MS1 and MS2. These formulas are used to determine the semantic similarity between two word senses. However, to find the similarity between two sentences, MS1 and MS2 are utilized in different manners. The first method used for MS1 is matching average and the second method that used for MS2 is Dice coefficient [52]. Matching average can be illustrated as $Score = \frac{Match(X,Y)}{|X|+|Y|}$ and Match(arg1,arg2) function matches the tokens of *arg1* and *arg2*. In practice arg1 and arg2 are sentences. The similarity is found by dividing the scores of Match function for each matching results by the total number of tokens. For example, given the sentences X with the 4 tokens and Y with 3 tokens, The bipartite matcher [53] has returned the matched pair X[1] and Y[1] with the score of 0.8, X[2] and Y[2] with the score of 0.6 and the overall score is calculated $2*(0.8+0.6)/7 = 0.4$. On the other hand, Dice coefficient method is used for the second formula (i.e. $MS2 = \frac{2 \times |X \cap Y|}{|X| + |Y|}$) to obtain a ratio between the tokens which can be matched and the total number of tokens. A threshold should be defined to eliminate the less possible match candidates. Considering the same example above, the overall score would be, using the Dice coefficient, $2*(1+1)/7 = 0.57$.

4. THE PROPOSED FRAMEWORK

The proposed framework in this thesis consists of two parts, the first part deals with textual data to perform an accurate image labeling and the second part uses the images labeled for training the classifier

We utilize tags associated with images by the semantic similarity measurement module, WordNet.Net. Finding the semantic similarity between tag sets and predefined words that represent class labels allows us to build a bridge between tags and image content.

Tags are also textual data as well as annotations, metadata or subjective observation of the image content. Typically, non-expert users choose tags subjectively and without any set of rules. In other words, a tag directly reflects the user's view about the image content. Therefore, such subjective tags can be misleading if they are analyzed as individual words and therefore should first be semantically analyzed. For this purpose a framework that is based on WordNet ontology semantically analyses tags to capture whole meaning of the content. As soon as the semantics of the tag set is captured, the corresponding image is categorized with (a) class label(s) in the predefined vocabulary. Then we train the classifier over the training set based on these labels using only low-level visual features. The classifier is then tested over the unlabeled test dataset.



Figure 4-1 Tags: ford, 1965, mustang, car

In Figure 4-1 , a sample image with its associated tags (keywords) is shown. The tag set of this image consists of four words that partially describe the content of the image. In our semantic relatedness algorithm, a set of tags is considered a *sentence*. We are interested in the meaning of the all tags together, in other words, we seek the meaning of the *sentence*.

The key concept consists of finding the semantic relationship between sentences. Therefore, tag analysis framework has a great importance for an accurate labeling. This framework is the unique part of this work and it accommodates a particular algorithm for analyzing tags semantically to capture a clue about the content of images.

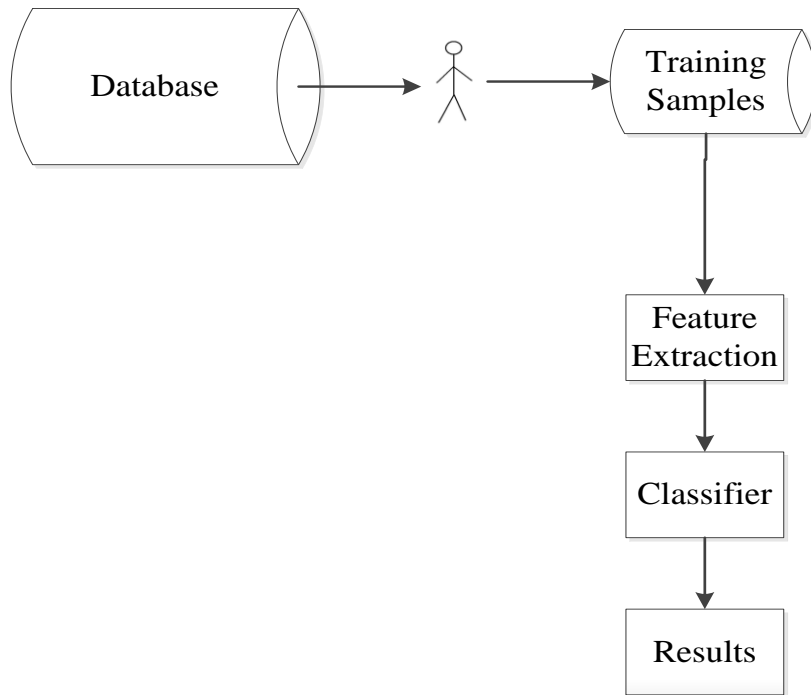


Figure 4-2 Traditional way of labeling by an expert.

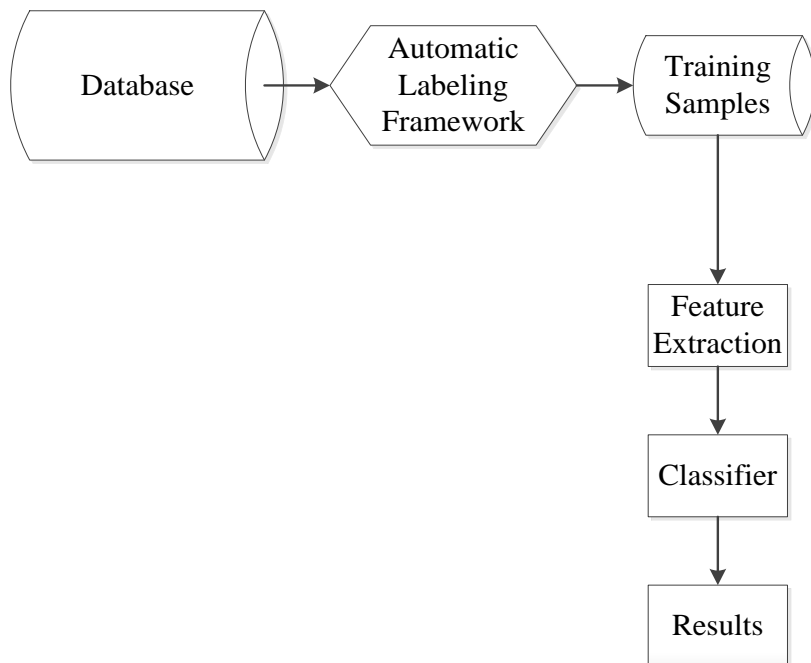


Figure 4-3 Proposed labeling by the automatic labeling framework.

Figure 4-2 shows the traditional way of manual labeling performed by an expert who spends time and effort to create a training dataset for image classification. Figure 4-3 shows the proposed system, which labels the images using the proposed automatic labeling framework which does not require any human intervention during the labeling process. The proposed labeling framework uses a fixed number of class words (also called class sentences) that best represent predefined classes, For example, the

class word for the class *car* is the word, *car*, and the class sentence for the class *cloud* is the word, *cloud*. Our aim is to find the semantic similarity between the class sentences and the tag sets of each image in the database.

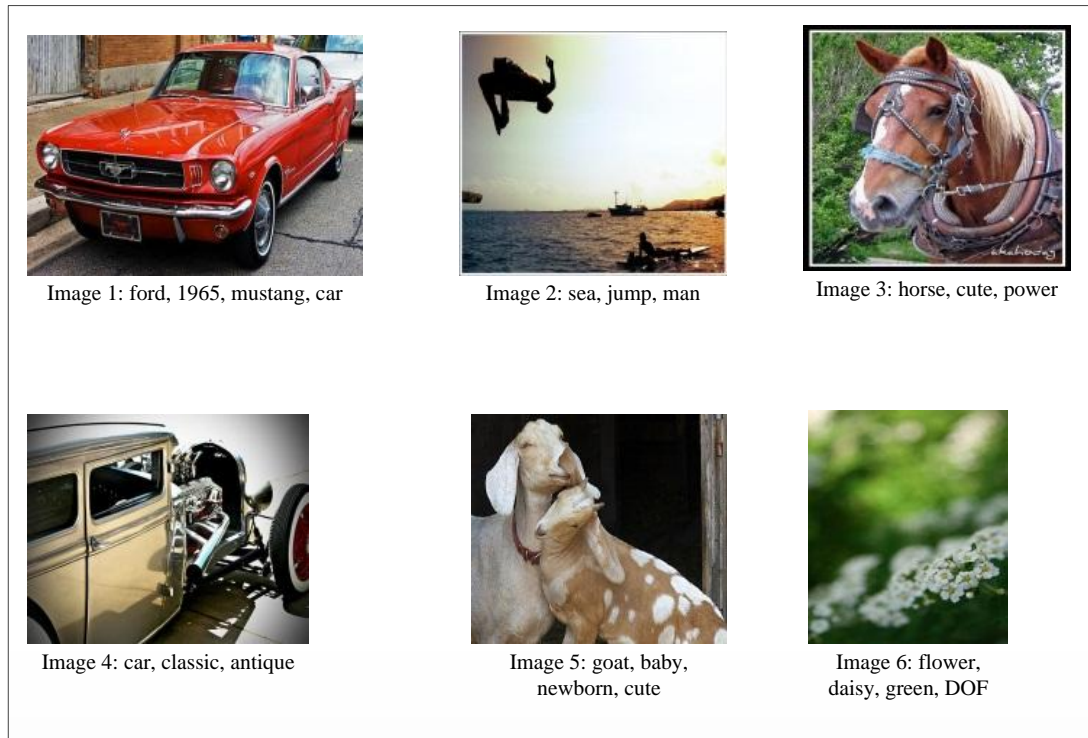


Figure 4-4 A sample database of six images with associated tags

Nevertheless, as shown in Figure 4-4 some tags unavoidably contain the labeling noise injected during the manual labeling process. The figure shows a mini sample database that consists of six images and their user-created tags. The user who uploaded images 1, 5, and 6 did not think that tags are not noisy since those images probably were taken when the owner had multiple ideas in his/her mind. However, in terms of content-based image classification, the tags *1965*, *ford*, *cute* are noise, since they are general words that do not particularly reflect the content. Therefore, using tags directly without any semantic analysis in image classification tasks is not a consistent or reliable solution. In addition to tags, the visual features of images should also be taken into account to improve the image classification since 80% of human cognition comes from visual information [54]. For this reason, we utilize tags to label the images by capturing the semantics and use the labeled items for further visual classification.

4.1. Automatic Labeling

The goal of the tag analysis is to find a semantic relationship between tags attached to images and label the tag sets. In order to achieve this goal, first we implement a func-

tion that takes two sentences as input and determines the relatedness between them. WordNet.NET module serves well to this purpose. It has a unique architecture that measures the similarity between two sentences. For example, the semantic similarity score between the sentences, “*baby newborn infant*” and “*brand new car nascar ford steer*”, is 0.413 , whereas the score for another sentence, “*cute baby honey jack born 2011*”, is 0.6877 . As the scores indicate, the first sentence is semantically more similar to the last sentence. However, the current version of WordNet.Net module alone was not enough for the proposed automatic labeling framework. To remedy this, we implemented a number of modules on top this framework Figure 4-5 demonstrates the WordNet.Net module as a black box, which outputs only the similarity score between the class sentences and the tag sets in the database.

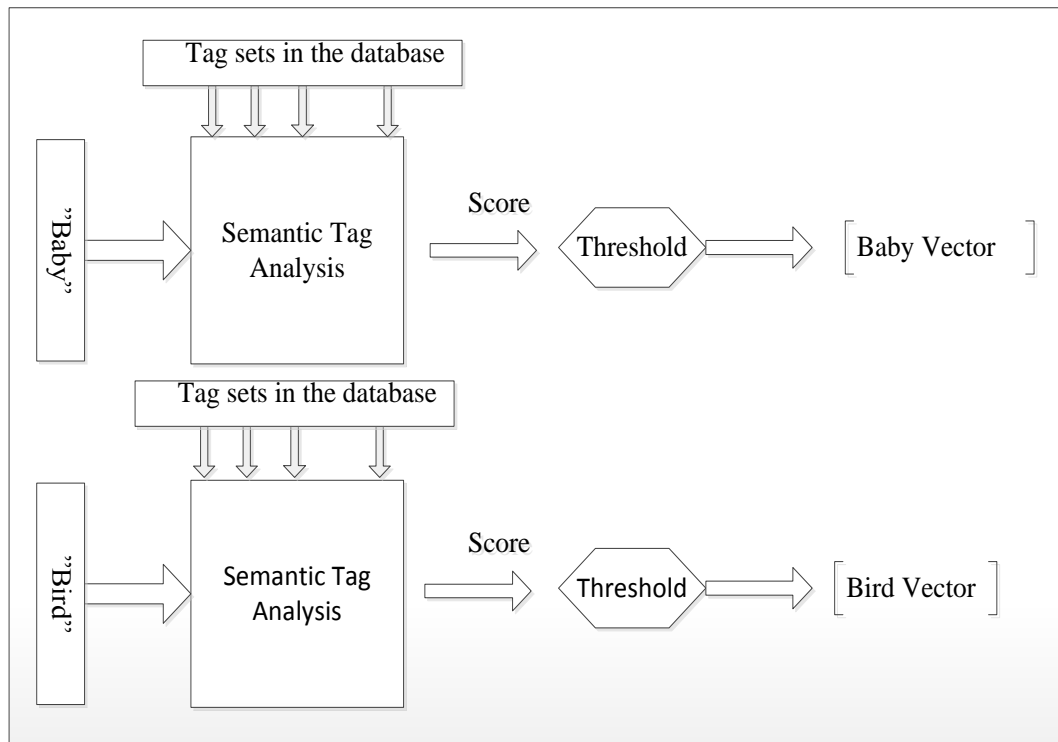


Figure 4-5 The illustration of the image labeling mechanism

The proposed labeling framework works in five consecutive steps, as follows:

1. Find the similarity score between all class sentences and all tag sets.
2. Apply a threshold for each score.
3. Label the images corresponding to the tag sets.
4. Store the image indices in a vector.
5. Rank the indices based their similarity scores.

Table 4-1 Pseudo code of the automatic labeling framework.

```

STORE all the tag sets as target sentences in a target vector
INITIALIZE class sentences as source sentences
DECLARE a class vector for each class
FOR each source sentence
  FOR each sentence in target vector
    COMPUTE the score between source and target sentence
    IF the score < threshold
      Add index of target sentence to the class vector
    ENDIF
  END LOOP
RANK the indices in class vector based on their scores
END LOOP

```

As presented in Table 4-1, two nested loops compare the sentences and find a similarity score. The first loop is for the class sentences and the second one is for the sentences (tag sets) in the database. Each class sentence is compared with the tags sets in the database, and the output (similarity score) is stored in a vector for the score ranking process. Recall that the higher the score, the more semantically related the sentences are. A tag set is labeled as the class of class sentences if its score is greater than the threshold value that we set empirically to 0.55. For example,

```

class sentence "baby"
sentence1 "brand new car nascar ford steer" (tag set1)
sentence2 "cute baby honey jack born 2011" (tag set2)
SimScore("baby", sentence1) = 0.25 < threshold
SimScore("baby", sentence2) = 0.69 > threshold
Result: sentence2 (tag set2) is labeled as baby.

```

After the labeling is performed, the indices of tag sets are stored in a vector (i.e. class vector) that corresponds to the indices of images. The indices are ranked according to their similarity score, by doing so we aim to aggregate the highest scores and make a heterogenic observation about the labels. For example, the class vectors below contains the indices ranked based on the similarity scores.

```

class vector = [IndiceSimilarity Score]
baby = [820.98, 360.86, 1070.75 ... 8370.56]
bird = [6401.0, 160.99, 7100.88 ... 9980.65, 340.56]
car = [40.76, 60.68, 13400.59, 1490.59 ... 250.57]

```

Take, for instance the *baby* class where the images *im82.jpg*, *im36.jpg*, *im107.jpg* have higher similarity scores than the image *im837.jpg*. Therefore, the confidence decreases as we go towards the end of the vector. We have the same number of vectors filled with image indices as predefined classes. For example, for a database organized into seven classes, seven vectors are obtained from the tag analysis framework. Each vector contains the labeled images of a specific class.

4.2. Image Classification

We used a Random Forest classifier for our experiment. Random forest (RF) is a coordinated group of classifiers that contain multiple decision trees such that each classifier is trained with the random distribution of input samples [55]. A test sample is fed from the top of the tree and labeled with its category when it reaches the final node. Each tree outputs its individual decision (vote) and the classification is performed according to the majority of the votes.

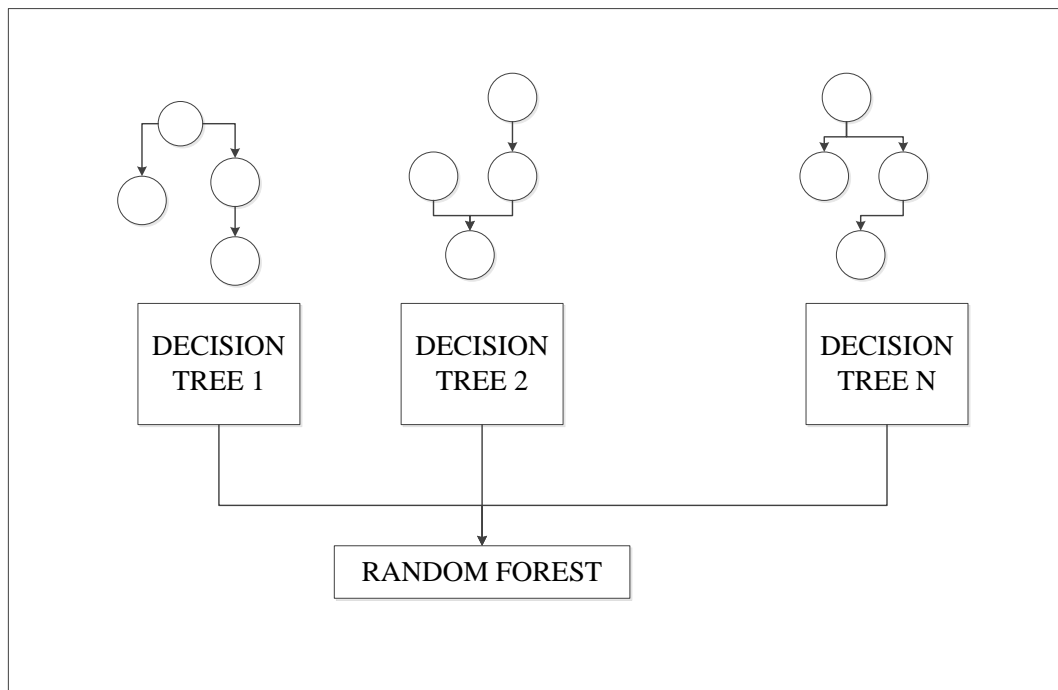


Figure 4-6 The classification scheme with the Random Forest.

The RF classifier promises several important properties as enlisted below, all of which constitute the reason of its use within the proposed classification system:

- Robust for large databases
- Fast
- Variable handling
- Reuse of forest
- Scalable number of trees

5. EXPERIMENTAL RESULTS

5.1. Preprocessing

We used the publicly available MirFlickr [56] database, which is commonly used for the visual detection and annotation task. The Ground Truth Data (GTD) of this database consists of multi-label 1400 images and organized into separate class folders, each of which provides the ground-truth and the associated keywords. Each class folder contains the same number of images (i.e. 200) carrying various contents (i.e. *baby, car, flower, food, bird, dog, and cloud*). As enlisted in Table 5-1, 10 low-level features (i.e. HSV Color Histogram, Dominant Color, Color Structure, Local Binary Pattern, Gabor, Ordinal Co-occurrence and Edge Histogram [57]) are extracted by the MUVIS framework [34].

Table 5-1 The features and parameters used for image classification

FV	Feature	Parameters	Dim.
1	HSV Color	H=6, S=2, V=2	24
2	Histogram	H=8, S=4, V=4	128
3	Dominant Color	$N_{DC}^{\max} = 6, T_A = 2\%, T_S = 15$	27
4		$N_{DC}^{\max} = 8, T_A = 2\%, T_S = 15$	35
5	Color Structure	32 bins	32
6		64 bins	64
7	Local Binary Pattern		16
8	Gabor	scale=4, orient.=6	48
9	Ordinal Co-occurrence	d=3, o=4	36
10	Edge Histogram Dir.		5

In MirFlickr database, each image comes with its associated tag set stored in a text file enumerated with the index of the image. It helps the users to process the text files separately and keep track of the image corresponds to the tag sets. We read a text file into a string buffer and represent each document as a sentence into the automatic labeling framework. Semantic analysis of tags is performed with WordNet.Net module that takes two sentences as input and outputs the similarity score between them. The automatic labeling framework then categorizes the images with the right labels that will constitute the training dataset. The RF classifier is then trained using the low-level fea-

tures and image classification is performed over the test (unlabeled) set. The next subsection evaluates the performance of the proposed system.

1.1 Performance Evaluation of the Automatic Image Labeling

In this section, we evaluate the performance of our labeling framework by semantic tag analysis with the traditional Precision and Recall hit-miss counters. Note that *Recall* and *Precision* are defined as:

$$\text{Recall} = \frac{tp}{tp+fn} \quad \text{Precision} = \frac{tp}{tp+fp} \quad (18)$$

where,

t_p is true positive

f_p is false positive

f_n is false negative

Precision is the fraction of the number of relevant items to number of all classified items, and *Recall* is the fraction of the number of relevant items to the total number of relevant items in the database.

The predefined class sentences to compute the similarity scores with all the tag sets in the database are shown in Table 5-2. The distance (relatedness) between the class sentence and the tag set indicates the relationship between class of interest and the content of the corresponding image.

Table 5-2 Predefined class sentences

Class	Class Sentence
baby	“baby”
car	“car”
flower	“flower”
food	“food”
bird	“bird”
dog	“dog”
cloud	“cloud”

We performed automatic labeling for the threshold value of 0.55 and the precision of each class is given in Table 5-3. In this table, we evaluated the accuracy of various percentages of each class vector. For example, Figure 5-1 illustrates the distribution of indices based on their confidence (similarity) scores. The percentages of the class vector represent the proportion to the total number of items in a vector. For example, if a vector contains 100 items in total, 10% corresponds to 10.



Figure 5-1 A visual representation of a class vector, as the red tone gets lighter the confidence of labeling decreases. This heterogenic representation facilitates the training set creation.

We evaluated the accuracy of each portion separately per class. For example, the accuracy of the first 10% of the *baby* class vector is 1.0 indicating that the first 10% of the images that are labeled as *baby* have all correct labels. However, the accuracy of whole class vector that labeled as *car* is 0.37.

Table 5-3 Precision of images labeled per class.

Percentage of total labeled items per class	baby	bird	car	cloud	dog	lower	food
First 10%	1.0	1.0	0.92	0.75	1.0	1.0	1.0
First 20%	1.0	0.92	0.90	0.81	0.96	1.0	1.0
First 30%	1.0	0.92	0.88	0.83	0.97	1.0	1.0
First 40%	1.0	0.92	0.77	0.78	0.98	1.0	1.0
First 50%	0.96	0.81	0.65	0.84	0.98	1.0	1.0
First 60%	0.91	0.71	0.56	0.75	0.99	0.97	0.94
First 70%	0.8	0.64	0.48	0.73	0.91	0.97	0.82
First 80%	0.72	0.57	0.43	0.71	0.83	0.93	0.8
First 90%	0.66	0.51	0.41	0.68	0.74	0.86	0.91
100%	0.6	0.48	0.37	0.64	0.7	0.81	0.74

Accordingly, we have created six different training set out of the vectors above.

Training Set 1: The first 10% of all classes.

Training Set 2: The first 40% of all classes.

Training Set 3: The first 70% of all classes.

Training Set 4: The first 100% of all classes.

Training Set 5: Empirically selected for experimental purpose.

Training Set 6: Empirically selected for experimental purpose.

The cells showed in blue are used for Training Set 5 and the cells in red are used for Training Set 6. Training Set 5 and 6 are created by observing the precision values of various percentages of each class. A tradeoff arises between the precision values and the number of training samples. The portions for Training Set 5 are selected according to

the highest precision values. Training Set 6 is selected sacrificing precision values but increasing the number of items.

1.2 Classification Results

In order to demonstrate the efficiency of the tag sets analysis, we conduct two separate experiments over the sample dataset. The first experiment is the traditional supervised image classification where a distinct training and test sets exist, both of which are manually labeled by the expert. In the second experiment, we assume that at least 1-3% of whole database as initial ground-truth training data exists and is mainly highlighting the image classes of the database and automatic labeling and categorization will be performed by the proposed framework over the image tags to form the training dataset, as presented earlier. The performance evaluation of the image classification experiments in terms of average *Precision* and *Recall* are presented in Table 5-4 and Table 5-5. The first experiment is performed with different percentages of the ground-truth data. %3, %10, %25 and %75 of GTD are used for as the train dataset. In this experiment we aimed to demonstrate classification performance of the classifier over ground-truth training datasets with different sizes.

Table 5-4 Classification performances over the test set using ground truth training datasets with different sizes

Training Rate	Average Precision	Average Recall
3%	0,216	0,207
10%	0,32	0,31
25%	0,378	0,368
50%	0,40	0,39
75%	0,48	0,47

As you can see from the table, the AP (Average Precision) score for all classes increases gradually with increasing training rate.

Table 5-5 Classification performances over the test set using the training datasets automatically created using the proposed framework

Training Sets	Average Precision	Average Recall
Training Set 1	0,235	0,238
Training Set 2	0,34	0,334
Training Set 3	0,35	0,36
Training Set 4	0,29	0,31
Training Set 5	0,33	0,33
Training Set 6	0,37	0,36

Table 5-5 shows that, the classifier yields the maximum performance (i.e. AP = 0.37).

5.2. Retrieval Results

Query by example (QBE) operation is the main the retrieval technique in MUVIS framework. This simply means that the (sub-) features of the query item are compared with the features of other visual items in the database for (dis-) similarity measurement. The comparison is performed by applying a distance metric (i.e. L2 (*Euclidean*) in our case) between the feature vectors of the query and each item in the database. As we mentioned in Section 2, CNBC can also be used for retrieval, in the cases where a CNBC is used for retrieval, the same (L2) distance metric is then applied to the class vectors at the output layer of the CNBC (10x2=20-D for MirFlickr database). The results are ranked based on their similarity distances and it yields the retrieval output. For both Retrieval performances are evaluated using average precision (*AP*) and MPEG-7 average normalized modified retrieval rank (*ANMRR*) measures, both of which are computed by performing a single query operation for all the images in the database (i.e. batch query) as well as a retrieval window that is equal to the number of ground truth images.

$$AVR(q) = \frac{\sum_{k=1}^{N(q)} R(k)}{N(q)} \text{ and } W = 2N(q) \quad (19)$$

$$NMRR(q) = \frac{2AVR(q) - N(q) - 1}{2W - N(q) + 1} \leq 1 \quad (20)$$

$$ANMRR = \frac{\sum_{q=1}^Q NMRR(q)}{Q} \leq 1 \quad (21)$$

where $N(q)$ is the minimum number of relevant (via ground-truth) images in a set of Q retrieval experiments, $R(k)$ is the rank of the k^{th} relevant retrieval within a window of W retrievals that are taken into account during for each query, q . If there are less than $N(q)$ relevant retrievals among W then a rank of $W+1$ is assigned for the remaining (missing) ones. $AVR(q)$ is the average rank formed using the query, q . Since the query item is also in the database, the first item retrieved will be the query item itself and it is removed from the overall ranking because its presence will yield a biased $NMRR(q)$ result. Therefore the first relevant retrieval ($R(1)$) is ranked by counting the number of irrelevant images a priori and note that if all $N(q)$ retrievals are relevant, then $NMRR(q)=0$, the best retrieval performance is thus achieved. Note that $NMRR(q)=1$ in the worst case when no relevant items are able to be retrieved within W . Therefore, the smaller $NMRR(q)$ indicates a better retrieval result for the query, q . The performance criteria are calculated by performing the query operation for all images in the database, which is also called batch query, and within a retrieval window with same the number of ground truth images, $N(q)$ for each query q . Therefore, AP becomes equivalent to average recall and average F1 measures.

Table 5-6 Retrieval results for ground truth

Training Sets	Average Precision	ANMRR
Original Feature Vectors	0.198781	0.796380
3%	0.169064	0.828612
10%	0.203327	0.790941
25%	0.233656	0.753362
50%	0.401113	0.571222
75%	0.607878	0.355970

Table 5-7 Retrieval results for the NBCs of the training sets

Training Sets	Average Precision	ANMRR
Original Feature Vectors		
Training Set 1	0.178379	0.818117
Training Set 2	0.231390	0.756837
Training Set 3	0.261718	0.721754
Training Set 4	0.252324	0.734310
Training Set 5	0.263453	0.720158
Training Set 6	0.249770	0.736041

Table 5-6 and Table 5-7 present the retrieval performances obtained from each batch query operation.

6. CONCLUSION

We have presented a framework to accomplish supervised image classification in the cases where all the images have associated tags and only a little amount of them is labeled.

We have analyzed how tags accompanying images can be used for image classification process. An existing approach analyzes textual data semantically using WordNet, performs word sense disambiguation (WSD), and determines the semantic similarity between tags. Based on the results we have obtained, we conclude that tags give a hint about the content of images, provide necessary information for labeling, which improves content-based image classification. We base the success of the proposed framework on the comparison between two experiments we have conducted. The average precision score of the classifier when it is trained with a very small amount of training data, that is %3, is 0.216. However, when the training set that automatically created by just analyzing the tag sets of the images is used for training the classifier, the performance climbed to 0,37. It also demonstrates that using the database where images have associated tags, we can obtain a labeled training set that is up to %25 percent of the total number of items. The higher percentages, which also contain the labels with less similarity score, produce erroneous classification results because of the noise in the tags and subjective and irrelevant tagging by users.

Using NBCs created for each class using Random Forest classifier, our tag-based semantic analysis framework is applicable to image retrieval as well. Our comprehensive experiments on MirFlickr dataset demonstrate the effectiveness of our framework for improving the image retrieval accuracy. As for the classification, performing query operation with the class vectors created with the labeled images by our proposed framework yield comparable results with the ones with ground-truth data. Numerical results show that the AP and ANMRR values for %25 of GTD are 0.233 and 0.753, respectively. However, the AP and ANMRR values for Training Set 5 are 0.263 and 0.720, respectively.

In the future, we plan to incorporate more information into our proposed framework. Especially, the metadata come with images, such as GPS coordinates, time stamps and focus points can be semantically analyzed for labeling images.

REFERENCES

- [1] S. B. a. V. Z. R., “Flickr tag recommendation based on collective knowledge,” in *Proceeding of the 17th international conference on World Wide Web - WWW '08*, 2008.
- [2] L. K. L. W. M. A. Yohan Jin, “Image annotations by combining multiple evidence & wordNet,” in *Proceedings of the 13th annual ACM international conference on Multimedia - MULTIMEDIA '05*, 2005.
- [3] G. Miller, “WordNet: a lexical database for English.,” in *Communications of the ACM*, 2005.
- [4] M. V. J. B. M. M. D. Srikanth, “Exploiting ontologies for automatic image annotation,” in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '05*, 2005.
- [5] M. C. C. K. P. Cho, “Image Retrieval and Classification Through Conceptualization Based on WordNet.,” in *Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques, Third International Conference on Intelligent Computing.*, Qingdao, China, 2007.
- [6] M. a. V. J. a. S. C. Guillaumin, “Multimodal semi-supervised learning for image classification,” in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, vol., no., pp.902-909, 13-18 June 2010..
- [7] D. H. a. D. F. G. Wang, “Building text features for object image classification,” in *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1367--1374, 2009.
- [8] D. Z. G. L. W.-Y. M. Ying Liu, “A survey of content-based image retrieval with high-level semantics,” *Pattern Recognition*, vol. 40, no. 1, pp. 262-282, 2007.
- [9] “ In Merriam-Webster.com Retrieved Feb 8, 2013, from <http://www.merriam-webster.com/dictionary/learn>”.
- [10] e. S. Mendelson and A. J. Smola, *Machine Learning Proceedings of the Summer School 2002*, Australian National University., Springer, 2003.
- [11] N. C. John Shawe-Taylor, *Kernel Methods for Pattern Analysis*, New York, NY: Cambridge University Press, 2004.
- [12] T. G. Dietterich, “Ensemble Methods in Machine Learning,” in *Proceedings of the First International Workshop on Multiple Classifier Systems*, June 21-23, 2000.

- [13] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," in *Informatica (Slovenia)*, 31(3):249--268, 2007.
- [14] M. N. M. ., P. J. F. A. K. Jain, "Data clustering: a review," in *ACM Computing Surveys (CSUR)*, v.31 n.3, p.264-323, Sept. 1999. .
- [15] T. I. A. Y. a. M. G. S. Kiranyaz, "Evolutionary Artificial Neural Networks by Multi-Dimensional Particle Swarm Optimization," *Neural Networks*, vol. 22, p. 1448–1462, 2009.
- [16] P. E. H. D. G. S. Richard O. Duda, "Unsupervised Learning and Clustering," in *Pattern classification (2nd edition)*, New York, Wiley, 2001, p. p. 571.
- [17] N. C. a. J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods.*, Cambridge University Press, 2000.
- [18] D. L. Olson and D. (. and Delen, *Advanced Data Mining Techniques*, Springer, 1st edition (February 1, 2008).
- [19] T. M. M. G. Serkan Kiranyaz, "Dynamic and scalable audio classification by collective network of binary classifiers framework: An evolutionary approach," *Neural Networks*, vol. 34, pp. 80-95, October, 2012.
- [20] T. Sikora, "The MPEG-7 Visual Standard for Content Description - an Overview," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 696-702, Jun 2001.
- [21] T. K. Kyoji Hirata, "Query by Visual Example - Content based Image Retrieval," in *Proceedings of the 3rd International Conference on Extending Database Technology: Advances in Database Technology.*, March 23-27, 1992..
- [22] W. M. S. S. G. A. J. R. Smeulders A. W. M., "Content-based image retrieval at the end of the early years," *IEEE Transactions on PAMI*, vol. 22, pp. 1349-1380, Dec, 2000.
- [23] "ISO/IEC. Information technology-Multimedia content description interfaces. Part 3: Visual. 15938-3:2002".
- [24] P. S. S. B.S. Manjunath, *Introduction to MPEG-7: Multimedia Content Description Interface.*, New York, NY : John Wiley & Sons, Inc, 2002.
- [25] M. Z. Bober, "MPEG-7 visual shape descriptors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 716- 719 , Jun 2001.
- [26] M. Hu, "Visual pattern recognition by moment invariants," *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 179-187, February, 1962.
- [27] M. Teague, "Image Analysis Via the General theory of Moments," *Journal of Optical Socie-ty of America*, vol. 70, no. 8, pp. 920-930, 1980.
- [28] K. M. S. a. W. F. L. Mehtre B. M., "Shape Measures for Contentbased Image Retrieval: A Comparison," *Information Processing & Management*, vol. 33, no. 3, pp. 319-337 , 1997.
- [29] L. G. J. a. S. A., "Region-based shape representation and similarity measure

- suita-ble for content-based image retrieval,” *Multimedia System*, vol. 7, pp. 165-174, 1999.
- [30] P. R. S. S. Pentland A, “ Photobook: Content-based manipulation of image databases,” *International Journal of Computer Vision*, vol. 18, no. 3, pp. 233-254, 1996.
- [31] O. G. Volker Gaede, “Multidimensional access methods,” *ACM Computing Surveys (CSUR)*, vol. 30, no. 2, pp. 17-231, 1998.
- [32] A. Guttman, “R-trees: a dynamic index structure for spatial searching,” in *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, Boston, Massachusetts , June 18-21, 1984.
- [33] R. J. David A. White, “Similarity Indexing with the SS-tree,” in *Proceeding ICDE '96 Proceedings of the Twelfth International Conference on Data Engineering*, 1996.
- [34] K. C. E. G. O. G. a. M. G. Serkan Kiranyaz, “MUVIS: A Content-Based Multimedia Indexing and Retrieval Framework,,” in *Proc. of the Seventh International Symposium on Signal Processing and its Applications*, Paris, France, 2003.
- [35] M. G. F. T. Predicted, “<http://www.clickz.com/clickz/news/2029405/google-ceo-mobile-growing-faster-predicted>”.
- [36] I. A. S. K. S. G. M. Ahmad, “Content-based image retrieval on mobile devices. In: Creuzburg, R. & Takala, J. (eds.),” in *Proceedings of SPIE-IS&T Electronic Imaging, Multimedia on Mobile Devices*, San Jose, California, USA, 16-20 January 2005.
- [37] I. K. S. & M. Ahmad, “Audiovisual retrieval framework for multimedia archives on Java enabled mobile devices. In: Milic, L. et al. (Eds.). ,,” in *Proceedings of EU-ROCON 2005 - The International Conference on "Computer as a Tool"*, Belgrade, 2005.
- [38] M. M. J. Lempiäinen, *Radio Interface System Planning for GSM/GPRS/UMTS*, Kluwer Academic Publishers, 2001.
- [39] C. M. Simpson T., “WordNet.Net <http://opensource.ebswift.com/WordNet.Net>,” 2005.
- [40] A. a. G. H. Budanitsky, “Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures,” in *In Workshop on WordNet and Other Lexical Resources, Second Meeting of the North American Chapter of the Association for Computational Linguistic*, Pittsburgh, PA, 2001.
- [41] E. Brill, “ A simple rule-based part of speech tagger,,” in *Proceedings of the third conference on Applied natural language processing*, Trento, Italy, March 31-April 03, 1992.
- [42] Z. a. N. H. Zhong, “Word Sense Disambiguation Improves Information

- Retrieval,” in *In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers*, 2012.
- [43] U. Zernik and G. E. C. R. a. D. Center, “Tagging word senses in corpus: The needle in the haystack revisited,” in *In Proceedings, AAAI Symposium on Text-Based Intelligent Systems*, Stanford, CA, 1990.
- [44] M. A. Hearst, “Noun homograph disambiguation using local context in large corpora,” in *Proceedings of the 7th Annual Conf. of the University of Waterloo Centre for the New OED and Text Research*, Oxford, United Kingdom, 1991.
- [45] J. F. Lehman, “Toward the essential nature of statistical knowledge in sense resolution,” in *Proceedings of the 12th International Conference on Artificial Intelligence*, Seattle, Washington, 31 July - 4 August 1994.
- [46] E. W. Black, “An experiment in computational discrimination of English word senses,” *IBM Journal of Research and Development*, vol. 32, no. 2, pp. 185-194, March 1988.
- [47] M. E. Lesk, “Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone,” in *In Proceedings of SIGDOC '86*, 1986.
- [48] S. a. P. T. Banerjee, “Extended gloss overlaps as a measure of semantic relatedness,” in *In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 2003.
- [49] S. P. a. J. M. Ted Pedersen, “WordNet::Similarity-measuring the relatedness of concepts,” in *In Proceedings of NAACL*, 2004.
- [50] P. Resnik, “Semantic similarity in a taxonomy: An information based measure and its application to problems of ambiguity in natural language,” *Journal of Artificial Intelligence Research*, vol. 11, pp. 95-130, 1999.
- [51] Z. Wu and M. Palmer, “Verb semantics and lexical selection,” in *In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics*, 1994..
- [52] L. R. Dice, “Measures of the amount of ecologic association between species,” *Ecology*, vol. 26, no. 3, 1945.
- [53] P. K. A. Kasturi R. Varadarajan, “Approximation algorithms for bipartite and non-bipartite matching in the plane,” in *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, Baltimore, Maryland, January 17-19, 1999.
- [54] X.-S. H. N. Y. W.-Y. M. S. L. Lei Wu, “Flickr distance,” in *Proceedings of the 16th ACM international conference on Multimedia*, 2008.
- [55] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.
- [56] M. a. L. M. Huiskes, “The MIR flickr retrieval evaluation,” in *Proceeding of the*

1st ACM international conference on Multimedia information retrieval, British Columbia, Canada , 2008.

- [57] G. E. a. G. M., “Feature selection for content-based image retrieval.” *Signal, Image and Video Processing*, vol. 2, no. 3, p. 241–250, 2008.