ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Automatic Video Scene Annotation and Summarization Framework (AVSAS)

Mekuanent Birara Ashagrie

A THESIS SUBMITTED TO THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE IN COMPUTER SCIENCE

February 2015

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

Automatic Video Scene Annotation and Summarization Framework

Mekuanent Birara Ashagrie

Advisor: Fekade Getahun (PhD)

Signature of the Board of Examiners for Approval:

Name	**Signature**
1. Fekade Getahun (PhD)	_____
2. Mulugeta Libsie (PhD)	_____
3. Yaregal Assabie (PhD)	_____

February 2015

Dedicated to:

My mother (እማ‘ኒ)

Acknowledgements

First and foremost, I thank God who endowed me with the strength to tackle problems in life and also to finish my work. I must also thank Saint Virgin Mary, Holy Mother of God, for Her intercession and all the Saints in the Kingdom of God for their blessings.

I offer my sincere gratitude to my advisor, Dr. Fekade Getahun, who has supported me throughout my thesis with his patience, motivation, enthusiasm and immense knowledge whilst allowing me the room to work in my own way. His guidance and encouragement helped me through the learning process of the thesis work. Moreover, his advice in those challenging times put me on the right track.

My utmost gratitude goes to my aunt, Tigist Tessema, for the love, care and lessons she gave me in life. Without her cooperation, patience, understanding, support and encouragement, none of these would have happened.

Finally, I would like to thank my colleagues, families, teachers, friends and others whom in any way possible had contributed to the successful accomplishment of this thesis work.

Contents

List of Figures

List of Algorithms

List of Tables

List of Definitions

List of Abbreviations and Acronyms

API	Application Program Interface
AVSAS	Automatic Video Scene Annotation and Summarization
BVW	Bag of Visual Words
CMU	Carnegie Mellon University
DOG	Difference of Gaussian
ELAN	EUDICO Linguistic Annotator
EUDICO	European Distributed Corpora Project
GMM	Gaussian Mixture Model
HOG	Histogram of Oriented Gradients
HSV	Hue, Saturation, Value Color Model
HTK	Hidden Markove Model Toolkit
MSER	Maximally Stable External Region
NLP	Natural Language Processing
OCR	Optical Character Recognition
OpenCV	Open Source Computer Vision
POST	Part Of Speech Tagger
RDF	Resource Definition Framework
RF	Round Forest
RGB	Red, Green, Blue color space
SIFT	Scale Invariant Feature Transform
SURF	Speeded Up Robust Feature
SVCAT	Semi-automatic MPEG-7 Standard Compliant Annotation Tool
SVM	Support Vector Machine
ANVIL	Annotation of Video and Language Data
VATIC	Video Annotation Tool from Irvine, California
VideoANT	Video Annotation Tool
VIRAT	Video and Image Retrieval and Analysis Tool
XML	Extended Markup Language

Abstract

With huge amount of video data being kept on the web, efficient utilization of these resources can be achieved with content based video indexing, grouping, searching and retrieval. For this content based video processing, automatic video scene annotation is required for the identification and labeling of events and objects in a video with a descriptive statement. Besides, annotations are important to provide information to viewers about content of a video.

Video annotation requires large knowledge to define semantic meaning of events and objects in the video. With this regard both manual and semi-supervised video annotations fail as they require expertise for correct identification and labeling of video concepts. Individuals may also be biased with their interest while annotating a video which will put the video into an incorrect class while grouping or indexing. Annotation requires great deal of concept dependency and relatedness processing to give a descriptive statement about a scene in a video.

This thesis introduces a novel scene level automatic video annotation and summarization framework where annotation provides scene level semantic description of videos. The framework uses speech content of a video to support event and object identification process with proper filtering and normalization process. It addresses the problem of concept relatedness and concept formulation process with new event-object affinity matrix, and delivers shot and scene level video annotations. It provides different algorithms and components starting from the video pre-processing stage to the final video summarization process for an efficient annotation and summarization result.

A prototype showing a scene level video annotation is developed and evaluated according to standard video processing evaluation datasets for its accuracy in event and object prediction and overall efficiency of the system is evaluated using ratings of individuals from different professions. The analysis on the evaluation result shows that the system provides an efficient scene level annotation of video contents with 81% accuracy in event prediction and an average user rating of 3.47 out of 4 in overall system evaluation.

Keywords: Video annotation, Scene level video annotation, Concept formulation, Concept normalization, Video summarization, Event prediction and Shot boundary detection.

CHAPTER 1
INTRODUCTION

1.1 Background

Though video collection grows rapidly, description of this multimedia objects using Meta tags cannot efficiently allow analysis and retrieval of video contents. Video content annotation is required to semantically label each object and event in video so that it will be easier to provide information to users as well as make machines understand content of the video [1, 5, 18].

One application of video annotation is to provide additional information to users. These users may be impaired hearing individuals or those who cannot understand the speaker‟s language in the video. While the video is playing, these users can read annotated text so that they can get more information with that. For such application, in addition to the usual image based description, it is better to have scene level description of objects and events.

Most important applications of video annotation are content based video indexing, retrieval and classification. Video retrieval approaches in large multimedia search engines use video descriptions embedded on the Meta tags and titles of video documents. However, such description of a video cannot give accurate and clear information about contents of the videos. User‟s query will be compared with the limited information (that may not even be related to the video content) kept in those components and mostly lead to irrelevant videos. Annotating video contents allows indexing, classifying and retrieving videos with their contents.

A lot has been done on video annotation but most of the works did not go beyond annotating objects and events in a shot [16, 21, 23]. Video annotation is different from image annotation as users want not only object description but also events, object motions which have to be annotated and presented at scene level rather than key frame level. While indexing video content, it is also required to know the context of an object, thus it is better to use scene rather than key frame for the annotation purpose. The challenging task in video annotation is giving descriptive object and event annotations within a scene through scene level frame dependencies and spatiotemporal

correlations of objects. Most works did not consider components in video (text, voice and graphic objects) which are good sources of information to find names and relationships among objects and events.

This thesis addresses problems towards current video annotation techniques and provides a framework for video annotation which can give description of video content at scene level with text based summarization of video considering speech in video as an additional source of information.

1.2 Motivation

YouTube is one of the biggest multimedia search engines which allow millions of people upload or download videos which are of different domains. In 2014, 300 hours of videos are uploaded every minute and everyday people watch hundreds of millions of hours on YouTube [34]. Though YouTube is providing a way which allows users to annotate video documents, it will be tiresome to make manual annotation of the full video content and no one can make sure that those annotations are not biased to the interest of the annotator and even their correctness. Beside the correctness of the annotation, high level knowledge is required from annotators to identify every object as well as relationships between objects in a video.

Consider an educational video that contains different objects and events happening with a narrative speech in it. Annotating this video is giving description to different objects and events so that students can easily understand the concept which the video is transferring. Through annotation, it may be giving labels to different objects and events in the video so that the free text can be displayed while playing the video. Getting this one step higher annotation can be done with user‟s preferred language in cases where the videos are language study types.

From the course organizer‟s angle, this annotation can be used to organize videos based on teaching materials in a way videos can be grouped depending on their content and they can be made available accordingly. For instance, consider e-learning where teaching materials are sent any time when required; if those videos are organized having content based annotations, users can send simple queries to web servers to fetch those video materials. This is possible as

annotations allow indexing web videos and defining formal query structures to search for videos using text.

Data can be easily linked to annotated videos from different sources calculating their similarity. These data can be of any format, i.e., it can be text, audio or video so that users can be provided with related set of information while looking for videos. Existing image level annotations cannot answer the requirements of the above applications of video annotation. The content of a scene should be properly analyzed and concept dependencies between frames should be processed for better results while indexing and classification of video documents.

Let us also consider documentary videos. Such videos contain huge amount of information with text, graphic features and voice incorporated with video segments. For example a video about a museum contains tags on objects and narration about events within the video. These narrations are highly related to the events in the video. Processing audio for event and object identification can support image based event processing minimizing the amount of required knowledge and computation. As a result, there should be a framework that can address such problems as those which are done on these areas could not completely solve the current video annotation requirements.

1.3 Statement of the Problem

The basic requirement in automatic video annotation is to provide full, clear, timely description of objects and events within the video without any supervision. Most video annotation works fail in providing the required description of videos. We can classify problems which those works could not address into four main categories.

***Scene annotation*:** it is more meaningful to provide scene level description of video rather than tagging images with labels which cannot give clear and understandable information. Most video annotation works give description of objects in a key frame level rather than considering the context of the object in that video. Key frame dependency has to be analyzed efficiently in order to define the context of an object in video and give a clear description of a scene; otherwise it will be similar to image annotation. This scene annotation may further be used to make video summarization easier than frame based annotation.

Concept relatedness: as a result of key frame based annotations, existing works provide annotations which may or may not be related to previous events and next events in a scene. As a result, unwanted events are extracted and used during annotating.

Additional video components: video is not only a collection of frames of images but there are also different components with huge amount of information regarding object names and event descriptions. Such components are text, graphic features and voice within the video. It might not be enough to consider low-level and high-level features in key frames of a video as there are complex object relationships and events. So considering these components will improve the accuracy of the annotation.

Annotation information: the information to be delivered to users should go beyond names and properties of objects in videos. A statement describing objects and their interaction has to be clearly built and given as annotation. Videos can be seen by any user with any language as they have visual features that can be analyzed. When it comes to annotation as the aim is to provide object description, tagging with a language which users are not familiar with, will make it useless so a language component is required for translation purpose.

1.4 Objectives

General Objective

The main aim of this thesis is to provide a framework for scene based automatic video annotation and text based summarization of video content.

Specific Objectives

In order to achieve the above general objective, the following specific objectives are identified:-

- Review related literature in the area of video processing, video annotation and video summarization
- Study specific requirements of automatic video annotation and summarization problem
- Adopt existing tools, techniques and approaches in video features extraction and visual similarity

- Study about properties and components of a scene in a video
- Identify components of video annotator
- Propose automatic video scene annotation and summarization framework with its components
- Prepare corpus - pre-annotated image corpus
- Develop a prototype that shows the viability of the proposal
- Test and evaluate capability of the proposed system

1.5 Methodology

In order to achieve the objectives of this study, different methodologies will be employed.

Literature Review

Detail review and assessment will be made on works which are done on the area of video annotation and related issues to know the actual problems with current annotation strategies. Such areas include video pre-processing (video shot boundary detection, key frame selection and feature extraction), object detection and the actual annotation process. Image annotation, image feature extraction, and speech synthesis will also be reviewed as they are building blocks of a video. Different sentence construction and document summarization techniques will also be exploited. From the review of these works, best approaches will be used for video pre-processing, speech synthesis and language modules.

Data Sources

Video annotation requires a huge knowledge source for a clear and meaningful object and event description. To have such training and test sets, samples of annotated videos and images will be collected from different multimedia sources over the web. Sampling will be done depending on their content, domain and graphic clarity. WordNet [30], a semantic knowledge will be used while formulating a concept as well as constructing a sentence.

Tools and Development Environments

Different free and open source tools will be used during image pre-processing and annotation process. Java programming language[1] will be used to implement the proposed solution while C# programing language is used for dataset preparation. Protégé[2] tool will be used to construct ontology with Jena[3] inference tool and a NetBeans IDE[4] with Java programming language will be used to develop the prototype.

Testing and Evaluation

Proper testing will be made and the newly proposed solution will be evaluated in terms of its goals and contributions. A questionnaire will be prepared and distributed to different users and their rating will be used to evaluate the system. Accuracy of components will be evaluated using standard content based video processing evaluation video datasets.

1.6 Scope and Limitations

This research is going to focus on the following aspects

- Shot and scene boundary detection
- Key frame identification
- Concept dependency analysis
- Similarity analysis of voice in video with visual features in a video
- Scene level concept fusion and formulation
- Scene level annotation sentence construction

Annotating a video using key frame as principal element focuses mainly on giving names to objects and events identified in the key frames. However, scene based video annotation provides a meaningful description to a part of a video dealing with different objects and relationships within a number of related frames. While giving description to such a long set of frames, it will

[1] https://www.oracle.com/java/index.html
[2] http://protege.stanford.edu/
[3] https://jena.apache.org/
[4] https://www.oracle.com/technetwork/developer-tools/netbeans/overview//index.html

be more informative and useful if it is in the viewer‟s own language. Though the framework will be generic, in this research we are not going to focus on the following specific components.

- Image segmentation
- Machine translation
- Text extraction

1.7 Application of Results

Video annotation has a broad area of application especially in video retrieval and analysis. Most large scale multimedia search engines like YouTube have huge amount of video resources which are not fully utilized as a result of poor video description which is embedded with metadata component of the videos. Efficiently annotating these videos allows those search engines to index, classify and organize these videos. User queries can be processed based on the context of the videos and relevant results can be generated in a way achieving efficient web multimedia resource usage.

When it comes to scene level video annotation, a video can be considered as a structural database where a video is a table and a scene is record so that a video query language can be formulated and passed to the video to look for a specific scene. This can be applicable especially in news agencies where there are huge amount of news videos and selecting one specific scene segment can be possible with a proper scene annotation.

Apart from the indexing, retrieval and classification of videos, automatic video annotation may help users who need extra video-content descriptions while just watching a video. Therefore, annotation allows having textual description of video segments giving more meaningful and timely information about video events. It allows users clearly understand the video content. Video documents of educational domain can be automatically linked with related videos and textual data based on the annotation text on them. This will be used to give more information about the concept which is kept on the video data.

1.8 Thesis Outline

The rest of this thesis report is organized as follows. Chapter 2 reviews concepts relevant to the realm of the proposed approach. Chapter 3 presents related works. Chapter 4 presents the proposed automatic scene level video annotation and summarization framework. Prototype development and experimental results are presented in Chapter 5. Finally, Chapter 6 concludes the overall work presented in this research work, presents contributions of the study and draws future directions.

CHAPTER 2
LITRATURE REVIEW

2.1 Introduction

This Chapter presents a review of concepts relevant to obtain basic understanding of the ideas of the proposed research work and concepts related to research works towards automatic video scene annotation and summarization. The review presented in this section deals on concepts important to the realm of the proposed approach. Specifically, concepts on video annotations, image annotation techniques, video annotation data sources and their properties, machine learning techniques, and other related issues are presented in detail.

2.2 Video Annotation

Video annotation is a process of giving semantic description to video contents [3, 13]. The data to be used as a description will be extracted mostly from the visual component of the video as in all annotation works, in addition to the audio information in some works like [9]. Recently, huge amount of videos have been made available to users in different ways but the need for searching and retrieving for the efficient utilization of those resources is not answered. What we have is video titles and metadata which cannot clearly and completely represent what is within the video. Annotating videos provides a way to index and structure video documents in a way to make them easily accessible by users [1, 5, 61].

2.2.1 Techniques of Video Annotation

Manual: In manual video annotation users are supposed to understand the video content and give descriptions to objects and events of interests [60]. Considering the huge amount of video resources available, annotation would be costly and limited to the knowledge of the user. The main functionalities of these manual tools are allowing users to specify annotation cuts and post descriptions. A lot of tools have been developed which allow users to annotate videos manually and some of them are:

- ANVIL video annotation research tool[5]
- VATIC video annotation tool[6]
- YouTube video annotation[7]
- VideoANT [62]

Manual annotation of videos requires a great deal of humans effort, time and the annotation result is subjected to humans error [14].

Semi-automatic: The large gap between high-level semantics and low-level features is challenging to be bridged by full-automatic content analysis mechanisms. To narrow down this gap, users are supposed to give relevance feedback which will accelerate the converging speed of the learning process by labeling the most informative samples. Here in semi-automatic fashion, the user interaction to the annotator is limited in comparison with that of manual annotation. For example, a user might be required to choose initial regions of objects of interest as in SVCAT [13] or the user might be also required to check the result while learning the descriptor.

A semi-automatic video annotation approach in [37] uses user interactions in order to track and estimate high-level features of unknown structures and then the automatic estimation of these complex landmarks by the system relieves the user from the burden of manually specifying the full 3D pose of annotations while improving accuracy. Such approaches are good while working with limited knowledge sources as video annotation requires large set of knowledge bases. In addition, as the result of the annotator highly depends on the result of the pre-processor and feature extractors, having a supervised way of cut detections and object of interest identification may provide a great result as compared to the current algorithms.

Automatic: Automatic video annotation techniques are best suit with the current video annotation requirements and they are active research areas. Such techniques are free from manual interactions on specifying key frames, shot and scene boundaries, object of interests and free from annotation feedbacks. With efficient algorithms and large training datasets automatic video annotation techniques provide a better description of video contents.

[5] http://www.anvil-software.org/
[6] http://web.mit.edu/vondrick/vatic/
[7] https://www.youtube.com/yt/playbook/annotations.html

2.2.2 Depth of Annotation

Video annotation can be done at scene level, shot level or object level depending on the domain and application area.

Object Annotation

Object annotation is tagging an object in an image which requires simple object identification and learning process. Most object level annotations are done to identify a specific object from videos. An example of such annotation is the work in [14, 24] which identifies products from videos having individual concept classifiers for each object.

Image or Frame Annotation

Image annotation is a process of assigning labels to an image that describes the content of the image [17, 41, 43]. Given an image, its annotation is carried out by identifying set of objects in that frame and classifying their concept classes accordingly. It can also be extended into an image level concept formulation where an event can be generated from an image and will be used as an annotation of that image. It is through automatic learning of semantic concept models from large number of image samples, where these samples can be collected from unlimited number of images on the web as it is done in [40], and uses the concept models to label new images. Once images are annotated with semantic labels, images can be retrieved by keywords, which is similar to text document retrieval [42].

Shot Annotation

A shot level video annotation contains a process of shot boundary detection as well as key frame selection as a concept to be defined is generated from set of key frames representing the shot [23]. As we go from object to image and image to shot level annotation the number of concepts to be identified and correlated increases, and concept dependencies and relatedness processing between key frames exist.

Scene Annotation

Scene annotation is a process of giving description to content of a video scene. The length of the annotation text increases and gets to be formal statement. A scene in a video is annotated correlating and formulating new concepts from annotations of shots in the scene. There is a need of proper shot clustering and scene timing for a good result of scene concept formulation and annotation.

2.2.3 Video Annotation Process

Video annotation requires different set of operations to generate and provide semantically correct concepts to the user. These processes include video pre-processing, feature extractions, learning and annotation presentation.

2.2.3.1 Video Pre-processing

The accuracy of the annotator is highly dependent on video pre-processing tasks [45, 55]. How well the video is pre-processed determines the accuracy of concept identification and the whole annotator. Video pre-processing in the context of annotation contains different tasks such as segmenting a given video into a set of frames, identifying shot boundary, selecting candidate key frames which can represent shots, and finally as our annotation is of a scene, scene boundary detection. As we consider the audio in a given video, scene level segmentation of the speech is also part of the pre-processing task.

Shot Boundary Detection

Video shot boundary detection is an initial and important video processing task while performing any content based video operations. Shots are continuous set of frames which are taken from a single camera in a single continuous action in time and space [45, 49]. Normally, it is a group of frames that have constant visual attributes, such as color, texture, and motion. Shot boundary detection is a process of analyzing a video sequence to identify cut points of video shots. There are different types of transitions between video shots [49, 50]. The first one is a hard cut which is also called an abrupt change. Such a transition happens when the entropy between two consecutive frames is very high. The other one is a soft cut or a gradual transition which happens

where the change in entropy increases or decreases gradually through a set of consecutive frames. Such transitions are fade-in, fade-out and dissolve. The hard cuts are easier to identify than that of the gradual transitions.

Most of shot detection techniques so far use low level image features like color histograms, edge, texture and their combination to identify any difference between the sequential set of frames in a video with some pre-defined threshold value. Pixel wise comparison is also used though it is very sensitive to camera and object motion. Though it is rotation invariant and a commonly used technique, images which are totally dissimilar may have similar overall histogram because of their color information. To solve this problem, [48] adopts an improved histogram algorithm which computes sum of square histogram difference at each block of frame x and x+1.

Most of the low level image features are not efficient to identify gradual transitions. Block based processing are proposed in [48, 52] where the first one segments frames into blocks and compares every block in frame f_i with every block in frame f_j using region based histogram similarity. This will be very useful to identify set of frames that represent a moving object. The second one computes three histograms (vertical, horizontal and global) over the segmented 8X8 blocks for every frame and compares it with that of the next frame. This improves the efficiency of identifying gradual transitions in a way if majority of the histograms are above the threshold, the transition is a hard cut, and otherwise it is a gradual change.

In [52] the blocks in the first frame are compared to the blocks in the second frame. Motion Vectors can then be calculated for each block to see where each block from the first frame ends up in the second frame. Cuts are then detected by comparing the difference between the intensities of successive motion to a specified threshold.

An approach in [53] minimizes the effect of change in brightness during shot fade in and out transitions. It converts the RGB image into an XYZ color space each having its own color combination which will normalize the cut points. Here histograms are computed once the conversion is done. While performing shot boundary detection, setting the threshold is the challenging task. Automatic threshold computation is proposed in [55, 56] using mean and standard deviation variances from the histogram difference of consecutive frames out of all

frames in the video. A pixel intensity entropy based approach is also proposed in [47], which analyses the change in entropy between consecutive frames. For gradual shot changes the entropy decreases or increases for longer time but for that of hard cut the entropy shows a sudden change.

Almost all shot boundary detection techniques reviewed compute similarity between two consecutive frames which will not result in a clear boundary cut as a result of gradual transition effects between shots.

Key Frame Identification

Video annotation basically starts with identifying objects from sequential set of frames in the video. Considering every frame in a video while doing so is computationally costly and it results in redundancy. Key frames representing set of similar frames must be extracted in order to overcome the above problem. Key frame is the frame which can represent the salient content and information of the shot. The key frames extracted must summarize the characteristics of the video, and the image characteristics of a video can be tracked by all the key frames in time sequence [54]. We may have one or more key frames for a given shot.

Different approaches have been followed to select a key frame. An easy but inefficient way is using the first, middle, last or combination of frames in a shot. In this case a frame which can represent a shot may not be in those static locations; thus, false key frames may be selected. Most of the techniques used [55, 59] rely on clustering frames of a shot and identifying the centroid point by computing the similarity difference between frames in the cluster using different low level and high level features such as color and motion vectors. Dianting Liu *et al*. in [59] split each frames into a 16X16 blocks of pixels and represent each block with single average gray scale or color value. In order to select the key frame, Euclidean distance is calculated between adjacent frames and a frame with the largest sum of Euclidean distance is used as a key frame.

Li Zaho *et al*. [58] use a simple iterative comparison of frame distances with a given threshold in a way if the distance between the current frame and the next frame is greater than the threshold, the current frame will be set as a key frame. This step will iteratively continue until the last

frame in the shot. A luminance based pixel wise difference was also used by [56]. Sandip T. Dhagdi and Deshmukh [55] proposed a new approach which uses three different descriptors; color, edge and wavelet statistics. Frame differences are computed using all the three features and the sum of combinational products of these three results will be used to draw the statistics across all frames with the difference values. Then frames at high curative points will be selected as key frames.

Scene Identification

A scene is a set of contiguous shots having a common semantic significance [49]. Most of the video scene identification approaches use techniques similar with identifying a shot such as block matching of key frames and histogram similarity. An approach in [63] uses a graph partitioning technique in order to cluster shots. Color and motion similarity techniques are used to identify the similarity between shots and a similarity graph is then constructed to find the cut points of similar shots.

There are also techniques which use audio features of the video to identify scene boundaries. Works in [46, 51] use visual features in addition to audio features in order to identify scene cuts. Low level audio features like short-time energy and zero crossing rates are extracted through audio segmentation and classification according to background conditions and speaker diarization. Based on the two conditions the audio stream can then be clustered into audio segments. Finally a scene transition graph can be constructed to find cut points.

Generally, most of the existing scene cut detection approaches relay on histogram differences between shots in the video.

2.2.3.2 Image Segmentation

Segmentation is the first step to extract a region based image representation. It divides an image into different components based on feature homogeneity [35]. This process is used to identify the region of interest which define a concept in the image so the better the segmentation algorithm, the better the annotation will be. The performance of segmentation process is dependent on the type of application.

If a video annotation process starts from low level object identification process where annotation starts with object clustering and relationship analysis, it is very important to define and use an appropriate domain independent segmentation algorithm as a video is not always of a single domain or a simple combination of known objects. For an image segmentation process, it can be easy to choose domain dependent algorithm as we are working with a single image. Some of image segmentation techniques are described below [35].

Thresholding: For an image with homogeneous regions a gray level thresholding can be used to divide foreground and background objects. Binarization can be used to identify a single foreground object whereas multi-thresholding can be used to identify multiple objects in a given image. The result of such approach is not always correct for an image with multiple set of shapes.

Clustering: With most commonly used clustering techniques like k-means, set of pixels will be clustered depending on their color and texture values to create a region from features belonging to a single cluster. One basic problem here is defining the number of clusters.

Region merging and splitting: can be one solution for the above drawback where region descriptors are compared with region descriptors of an adjacent region. If they are similar they will be merged, if not, they will be split. The problem with this is the type of descriptor to use. For example, if a color descriptor is used in a region growing, a person with different cloth color may be split into different regions.

Contour based segmentation: is also used which works by evolving a curve around an object. This could be good for clearly isolated shapes in an image but for others with overlapping objects the contour will merge both objects as a single shape. Structural techniques like edge based segmentation perform segmentation by detecting the edge in an image using commonly used sobel and canny edge detection algorithms. Detecting every edge in an image cannot identify regions of interest in an image and they are highly prone to noise.

2.2.3.3 Video Concept Detection

Video annotation requires in depth acquisition and analysis of objects and events within the video. Video concept detection is a process by which different set of features are extracted from the input video and analyzed in a way to find the semantic description behind each video scene. Concept detection within a video scene starts with extracting features from key frames representing each shot in the scene and performing a concept fusion and analysis to come up with a concept description to the scene.

Visual Features and Feature Extraction Techniques

Different visual features [28, 29] representing a given key frame can be extracted in a way to perform visual matching and classification to identify concepts. Such visual features include:

Color: images can be represented with different color spaces like RGB and HSV where, if an image is in an RGB color space, it is represented in the form of Red, Green and Blue values of pixels while HSV represents a pixel as hue, saturation and value dimensions. Having these color spaces, there are different color features that can be extracted from images.

*Color histogram**:*** These features are extracted by calculating the amount of color present in the image. A color histogram H for a given image is defined as a vector H = {h[1], h[2], . . . h[i], . . . , h[N]} where *i* represents a color in the color histogram, h[i] is the number of pixels in color i in that image, and N is the number of bins in the color histogram, i.e., the number of colors in the adopted color model. One advantage of color histograms is, they are invariant to rotation, translation and scaling (this scaling has to be known while performing similarity between images) of an object but the histogram does not contain semantic information, and two images with similar color histograms can possess different contents.

Color moments: Color moments have been successfully used in many retrieval systems. The first order (mean), the second (variance) and the third order (skewness) color moments have been proved to be efficient and effective in representing color distributions of images. The first color moment of the *k*-th color component (k = 1, 2, 3) is defined by:

$$M_{k=}^{1} \frac{1}{XY}\sum_{x=1}^{X}\sum_{y=1}^{Y} f_k\ (x,y) \qquad \text{.[2-1]}$$

Where $f_k(x, y)$ is the color value of the *k*-th color component of the image pixel (x, y) and XY is the total number of pixels in the image. The *h*-th moment, h = 2, 3, . . of *k*-th color component is then defined as:

$$M_{k=}^{h} \left(\frac{1}{XY}\sum_{x=1}^{X}\sum_{y=1}^{Y}(f_k \ ((x,y) - M_k^1)^h\right)^{\frac{1}{h}} \text{.......... [2-2]}$$

Since only 9 (three moments for each of the three color components) numbers are used to represent the color content of each image, color moments are a very compact representation compared to other color features. The similarity function used for retrieval is a weighted sum of the absolute differences between the suitable moments.

Color correlogram: Color correlogram characterizes color distributions of pixels and spatial correlation of pairs of colors. Let I be an image that comprises of pixels f(i, j). Each pixel has certain color or gray level. Let [G] be a set of G levels g1, g2. . . gn that can occur in the image. For a pixel f let I(f) denote its level g, and let Ig correspond to a pixel f, for which I(f) = g. Histogram for level gx is defined as:

$$h_{g_x}(I) \equiv \underset{f \in I}{Pr} \mid f \in I_{g_i} \mid \text{.......... [2-3]}$$

Second order statistical measures are correlogram and autocorrelogram. Let [D] denote a set of D fixed distancesd1, d2, . ., dD. Then the correlogram of the image I is defined for level pair (gx, gy) at a distance d

$$\gamma_{g_x,g_y}^{(d)}(I) \equiv \underset{f_1 \in g_x, f_2 \in g_y}{Pr} [\, f_2 \in I_{g_x} \mid \mid f_1 - f_2 = d] \text{.......... [2-4]}$$

Which gives the probability that given any pixel f1 of level gx, a pixel f2 at a distance d in certain direction from the given pixel f1 is of level gx. Auto-correlogram captures the spatial correlation of identical levels only $\alpha_g^{(d)}(I) = \gamma_{g,g}^{(d)}(I)$.

Shape: is used to encode simple geometrical forms which will be used to identify and represent real world objects. Shape feature can be extracted using contour based or region based methods. The former calculates shape features only from the boundary of the shape, while the latter method extracts features from the entire region.

Edge: shows change in an intensity corresponding to discontinuity of an image. There are different edge detection techniques such as canny edge detection and Sobel edge detection.

Texture: Color is usually a pixel property while texture is measured from a group of pixels. Based on the domain from which the texture feature is extracted, they can be broadly classified into spatial texture feature extraction methods and spectral texture feature extraction methods. For the former approach, texture features are extracted by computing the pixel statistics or finding the local pixel structures in the original image domain, whereas the latter transforms an image into frequency domain and then calculates feature from the transformed image. Gabor filter is the most commonly used texture feature extraction technique which is designed to sample the entire frequency domain of an image by characterizing the center frequency and orientation parameters.

The problem with the above visual features is they are not scale and rotation invariant, considering the color feature, different images with similar colors can have the same color histogram.

SIFT: Though it is computationally costly in comparison with the above visual features, SIFT [27], a scale and rotation invariant visual feature can represent visual objects efficiently. SIFT generates key points representing a given image performing a scale space Gaussian smoothing at a different scale to find scale invariant points. The difference of Gaussian is used to find these initial key points in an assumption that DOG (Difference of Gaussian) is a scale invariant function. In order to illuminate points extracted around edges, a hessian matrix is used. It finds gradient direction and magnitude in order to construct gradient direction histogram. The gradient direction histogram is required to make it rotation invariant. Such features are most commonly used during object identification and recognition process from datasets where there is a possibility of having similar images.

SURF: This feature is somehow similar to SIFT with some improvements which make it computationally fast and robust to image transformations. Here the DOG is computed on rescaled images than those filtered at different scale [57]. Such feature descriptions can be used

for fast and robust point matching between two images under scale, rotation, noise, illumination changes and changes in cluttered background [47].

Bag of Visual Words: an image can be treated as a text which can then be represented using bag of words. It is based on extracting local features from the images, e.g., SIFT, and then clustering them into "*visual words"* which is called a codebook. Images are then represented as histogram counts of these visual words [39].

2.2.3.4 Learning Techniques

There are different machine learning techniques which can be applied to different application domains. In an image and video processing these machine learning techniques can be used to analyze the type and structure of an object in order to classify it to some classes of concepts.

Support Vector Machine: are binary classifiers which will separate a class looking for the optimal separating hyper plane between the two classes by maximizing the margin between the classes" closest points. Data points on the "wrong" side of the discriminant margin are weighted down to reduce their influence. If a linear separator is not found, data points are projected into higher-dimensional space where the data points effectively become linearly separable. It is the most commonly used learning technique in video and image annotation tasks.

Individual classifiers like SVM require large training dataset for efficient classification process. They can be enhanced with collaborative learning techniques [8] to incorporate cross-concept collaborations into the joint learning of similar detectors over related concepts. Images and videos contain concepts which are related to each other so that this collaborative learning can be used to identify concepts which are related to concepts identified by individual classifiers.

Active Learning: is a machine learning technique that selects the most informative samples for labeling and uses them as training data. It has been widely explored in the multimedia research community for its capability of reducing human annotation effort [38]. A typical active learning system is composed of two parts, that is, a learning engine and a sample selection engine. The learning engine trains a model based on the current training set. Then, the sample selection engine selects the most informative unlabeled samples for manual labeling, and these samples

are added to training set. In this way, the training set obtained is more informative than that gathered by random sampling. Samples are collected with the following criteria;

- A sample should be consistent with the aim of the learner, which is, minimizing the expected risk.
- Samples should be uncertain.
- Samples should be diversified; there should be batch of samples.
- Samples should be from regions of high density.
- Samples should be relevant.

2.2.4 Formats of Video Annotation

Once the concept is formulated the next step is putting the annotation in machine as well as human understandable formats. These results can be tagged on a moving video or they can be kept as a separate file with the query video. In the former case, the annotation text can be kept as a script file with timing information so that it can be called in players which can integrate scripts. The annotation can be written into xml of RDF files so that it can be further used for other content based video operations. An example annotation result is shown in *Figure 2-1* [19].

Figure 2-1: Sample Annotation Results in XML Format

2.2.5 Video Annotation Knowledgebase

Ontology: consists of entities and their relationships, which are organized hierarchically [19]. It may be in the form of classes and subclasses where each class may consist of one or more instances. Ontology can be defined as an explicit specification of a conceptualization. For example, "cat" is a subclass of class "animal". It helps to associate semantic to any object or its image and provides a better understanding in proper context.

WordNet: is an on-line lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory [30]. English nouns, verbs, and adjectives are organized into synonym sets, each representing one underlying lexical concept. Different relations link the synonym sets. WordNet can be used to compute semantic similarities between concepts as well as sentence analysis as in [1].

ConceptNet: organizes a wide range of commonsense concepts and relations, such as those found in the Cyc knowledgebase, yet this knowledge is structured not as a complex and intricate logical framework, but rather as a simple, easy-to-use semantic network, like WordNet [31]. While it supports many of the same applications as WordNet such as query expansion and determining semantic similarity, its focus is on concepts rather than words, it is more diverse relational ontology, and its emphasis on informal conceptual connectedness over formal linguistic rigor allows it to go beyond WordNet to make it practical, context oriented commonsense inferences over real world texts. This can be used to define concept relatedness while working with different concepts in a shot or a scene.

ImageNet: is an image database organized according to the WordNet hierarchy of nouns, in which each node of the hierarchy is depicted by hundreds and thousands of images. These images are used to identify objects and the concept class of each object. Currently it has an average of over five hundred images per node. These images are collected from the web and labeled by human labelers using Amazon‟s Mechanical Turk crowd-sourcing tool [32].

Video Datasets: There are also different video datasets that can be used for different content based video processing for training and evaluation purpose. Some of them are listed below:

- TRECVID[8] video dataset
- VIRAT[9] video dataset
- CALVIN[10] dataset
- YouTube-Objects[11] dataset

2.3 Video Summarization

Video summarization is a process of shortening a video extracting most important key frames which helps in efficient storage, quick browsing, and retrieval of large collection of video data without losing important aspects. According to a survey in [33], there are different techniques of video summarization. Feature based summarization uses color, motion and audio features where key frames are extracted based on these features. Clustering based approaches group frames with similar characteristics and eliminates those frames with irregular trends. There are also event based summarization techniques where interesting events are extracted to select frames that are important. Given a video, it can be segmented into shots where a representing key frame can be extracted and used to summarize the video. The main aim of video summarization in this work is providing a text based summary of videos.

2.4 Audio

In addition to the visual component, we have audio within the video which can be used in the different parts of video annotation. These features can be used to identify scene cuts; additionally, a speech can be analyzed to identify objects and events.

2.4.1 Speech Recognition

Speech recognition is a process of recognizing words from a given dictionary uttered by a speaker, relying only on the information contained in the uttered speech signal and on prior knowledge of the problem domain [36]. Basically speech recognition is used for two main purposes. One is for dictation which is translation of spoken words into text, and second

[8] http://trecvid.nist.gov/
[9] http://www.viratdata.org/
[10] http://groups.inf.ed.ac.uk/calvin/datasets.html
[11] http://people.ee.ethz.ch/~presta/youtube-objects/website/youtube-objects.html

controlling the computer, that is capable of authorizing a user to operate different application by voice.

Speech recognition processes: Having the audio signal from the input source the following are the two core operations in speech recognition [64].

- *Acoustic processing:* Provides a method of calculating the likelihood of any feature vector sequence Y given a word W so that it transforms the pressure wav form into symbols.
- *Language Modeling:* The purpose of the language model is to take advantage of linguistic constraints to compute the probability of different word sequences. Assuming a sequence of Kwords, $W = \{w1, w2,\ldots, k\}$, the probability $P(W)$ can be expanded as $P(W)=(Pw1,Pw2,\ldots,Pwk)$ where any word wk depends only on the previous $N-1$ words in the sequence which is known as an N-gram model.

Speech recognition tools

HTK[12]: The HTK is a software toolkit for building speech recognition systems. It can perform either continuous density, semi-continuous density or discrete probability HMM based tasks. It is developed by the Cambridge University Speech Group.

CMUSphinix[13]: is an open Source Toolkit for Speech Recognition developed by Carnegie Mellon University.

Google Speech API[14]*:* Google‟s Web Speech API aims to enable web developers to provide, in a web browser, speech-input and text-to-speech output features that are typically not available when using standard speech-recognition or screen-reader software. The API itself is agnostic of the underlying speech recognition and synthesis implementation and can support both server-based and client-based/embedded recognition and synthesis. The API is designed to enable both brief (one-shot) speech input and continuous speech input. Speech recognition results are provided to the web page as a list of hypotheses, along with other relevant information for each hypothesis.

[12] http://htk.eng.cam.ac.uk/
[13] http://cmusphinx.sourceforge.net/
[14] http://www.google.com/intl/en/chrome/demos/speech.html

2.4.2 Part of Speech Tagging (POST)

POST is a process of tagging words in a sentence with the correct part of speech and grammatical behavior of the word. There are rule based and stochastic methods of POS tagging which attain accuracies of 96-97% [7]. Knowing the lexical category of a word provides a way to define events and objects that are found in a given text. This text can be from speeches in a video or from scripts of videos.

The problem with POS tagging is ambiguity of a word as well as unknown words. The ambiguity is as a result of multiple tags that can be given to a word in a sentence and there may be an unknown term in the corpus where the tags are derived. Unlike the highly edited genres that conventional NLP tools have been developed for, conversational text contains many nonstandard lexical items and syntactic patterns. These are the results of unintentional errors, dialectal variations, conversational ellipsis, topic diversity, and creative use of language and orthography [44]. Not only this but in the case of video annotation, if there is audio usage specially a speech, there is a high probability of having informal words. *Stanford Log-linear Part-Of-Speech Tagger*[15] is a Java implementation of the log-linear part-of-speech taggers which is one of the most known POS tagger software.

2.5 Text Summarization

Text Summarization is condensing the source text into a shorter version preserving its information content and overall meaning. It is very difficult for human beings to manually summarize large documents of text. Text Summarization methods can be classified into extractive and abstractive summarization [26]. An extractive summarization method consists of selecting important sentences, paragraphs, etc. from the original document and concatenating them into shorter form. The importance of sentences is decided based on statistical and linguistic features of sentences.

Abstractive text summarization attempts to develop an understanding of the main concepts in a document and then express those concepts in clear natural language. It uses linguistic methods to examine and interpret the text and then to find the new concepts and expressions to best describe

[15] http://nlp.stanford.edu/software/tagger.shtml

it by generating a new shorter text that conveys the most important information from the original text document.

Extractive summaries are formulated by extracting key text segments (sentences or passages) from the text, based on statistical analysis of individual or mixed surface level features such as word/phrase frequency, location or cue words to locate the sentences to be extracted. The "most important" content is treated as the "most frequent" or the "most favorably positioned" content. Such an approach thus avoids any efforts on deep text understanding. They are conceptually simple and easy to implement.

2.6 Summary

Video annotation can be done manually, with supervision or automatically to give description to objects, shots, and scenes. This process involves different tasks from various disciplines such as image processing, natural language processing, audio processing, etc. Basic operations like shot and scene boundary detection, key frame selection and object and event identification are done using image frames through different image processing activities.

Different low level and high level image features represent an image while performing different operations on the image. These features can be used in selecting the appropriate one depending on the application and domain of the video. Though current approaches cannot efficiently identify shapes of objects, there are also various segmentation techniques for object identification.

Concepts behind objects can be extracted from class labels defined in an image or video dataset in addition to existing semantic knowledgebase. Semantic networks of concepts like WordNet and ConceptNet can be used with different NLP operations like sentence construction, summarization, speech recognition, part of speech tagging, etc., to formulate concepts and annotation statements.

The results from a video annotation process can be delivered in different formats. One is using XML/RDF which can be used for further content based video processing and the other is using script like texts which can be displayed on video media players while watching the video.

CHAPTER 3
RELATED WORK

3.1 Introduction

In this Chapter, we present research works which are related to our work. There are a number of research works in the area of video annotation focusing on improving a single component of a video annotation system [4, 8, 10] or dealing with generic architecture [1, 9, 19]. We reviewed works proposed to provide video annotations along with different levels of video annotation process.

Video annotation can be done at scene level or shot level [3, 20]. The main task in a scene level video annotation is to formulate concept which can be used to give informative description of the video scene. This concept formulation requires image processing activities as well as concept construction techniques. Scene annotation requires shot dependency analysis to formulate a concept extracted from different objects found in different shots of a single scene. A lot has been done on the area of video annotation relying on annotating shots. In almost all works, concept formulation depends on objects which are found on a key frame representing a shot which is almost similar to annotating an image [40].

The video annotation practice can be fully automatic [9, 12] or semi-automatic [37]. Most video annotation researches are domain specific and rely on a single object [20, 24] while some are domain independent [6, 11]. These research works use different methods of video pre-processing, concept acquisition and concept formulation techniques. Most existing works are done specifically to improve one of the components of video annotation framework focusing mainly on either learning techniques or pre-processing tasks.

In this Chapter we present the most related video annotation works grouped based on domain. We have evaluated these works according to the module they aim to improve as well as the overall requirements of automatic video annotation systems.

3.2 Domain Independent Video Annotation

A generic video annotation framework in [1] considers each shot separately while annotating a video. This framework contains two layers of similarity analysis; the first layer computes similarity value between videos in a dataset and the query video, whereas the second one is used to differentiate and merge similar annotation texts from similar videos. Video signature, a compact representation of spatio-temporal features, is generated in order to perform video similarity and later to find free-text annotations from similar videos. This video signature is generated using SIFT features [27] generated at some trajectory points in a video. A video matching is done by performing a similarity comparison between each and every trajectory point of the query video and a video in a dataset normalized with the minimum number of trajectories found in either of the two videos. Once the similarity is computed, Stanford NLP Log-linear Part-Of-Speech Tagger is used in order to identify objects using nouns, and actions using verbs. As the approach uses free-text annotations from multiple similar videos, the authors used WordNet [30] in bridging word difference and compute similarities between terms found in different videos. Finally, ConceptNet [31] is used to derive the actual concept relationship between objects trajectory of key frame.

This generic framework considers similarity between single shot videos where there is no need for scene analysis and concept dependency. The ultimate result annotating in such videos is more or less similar to annotating an image. Hence, this generic framework relies on a shot annotation rather than a full video annotation.

A fast semantic diffusion approach in [4], aims to improve annotation accuracy and adopting concept affinities in a large scale domain independent images and videos. This work focuses on exploring context knowledge, modeling concept relationships using weighted graph where nodes are concepts generated by concept classifiers and edges represent weights reflecting concept correlation. The weighted graph is applied in semantic diffusion to generate concept annotation resulting from function level diffusion process with respect to the concept affinities. Given a training set, this approach constructs a semantic graph by the relationship between concepts which is based on an affinity metrics computation. The concept affinity is calculated by computing the probability of finding two concepts X and Y together in a given training dataset.

This work can be good for predicting a correlated concept in the presence of already known concepts but the main drawback is the possibility of missing an object which is not in the same domain with that of the concept which is already known. This is because lower values are found as a result of computing concept affinity between those concepts. For example, a ball is a concept in sport domain. However, if a scene shows an animal holding a ball, the concept affinity matrix may result *person*, *field,* etc. as a concept that is correlated with a *ball* than that of showing an animal. In this case, an object "*Animal*" will be missed. Nevertheless, this work has a merit in event affinity construction, event correlation definition and event prediction rule construction, as events are highly related to each other.

An automatic video annotation through search and mining is proposed in [5] to exploit overlap in the content of news video for automatically annotating similar videos. An image feature is combined with text, and SVM concept features are used to search for related videos. This work is based on a single shot video because with a single text query it may not be possible to describe a large video. Once related set of videos are generated transcripts of these videos are mined as an input for the annotation process. A term frequency vector is created for each video transcript to select those with maximal number of occurrence to be used for annotation. The problem here is how similar those searched videos would be and, how many of the events which are extracted from video transcripts of those 50% of retrieved videos would be there in the video which is to be annotated.

The result of annotation is highly dependent on the similarity between videos and, it is unlikely to get full length annotated videos very similar to the input video. In addition, another problem would be unavailability of transcripts in almost all videos which is most of the time there only on news videos. However, they are not available with videos. In comparison with transcripts, subtitles are more available in documentaries and movies.

In [6], an approach similar to [5], using graph reinforcement technique is proposed. In this case a query is issued to a video dataset. Different visual features like edge histogram, color histogram, autocorrelogram, etc. are extracted for each individual visual feature and a stable graph is constructed among results of similar features. Then their user tags will be analyzed and union of the graphs will be used to determine annotation for the query video. In comparison to [5], this

work uses user tags of similar videos collected from YouTube and provide better accuracy due to refinement. It has a similar drawback with [5], that is, getting full length annotated video is highly questionable.

A framework for group based image retrieval and video annotation is proposed in [2] which computes a region based Wavelet Transform similarity of query video with pre-annotated videos to find set of similar image frames. They used this approach in an assumption of improving the speed of the algorithm as it is robust than SIFT. A video frame is split into 4X4 region and a feature vector is considered using the centroid point of the wavelet transformed data at the region. Once the feature vector is constructed, similarity is calculated using earth movement distance measure between the same blocks of frames of videos.

Once weighted list of text entries that accompanied the top similar matched videos is found, a sentence analysis will be done. This sentence is a video level text which will be taken into the standard NLP log linear part of speech tagger in order to identify objects, events and location triplet. This will then be followed by sentence analysis and simple sentence merging of these sentences using ConceptNet and WordNet. The basic drawback in this research work is, it takes the whole annotation text of similarly video into the POS tagger and directly uses those identified set of objects and events in the sentence merging module. In this case, not all identified events and objects could possibly be in the query video and the timing of those events and objects is not known. Though a video is split into frames and annotation is based on key frames of similar videos, the approach disregards context; i.e., an event found in one video frame may not always have a similar context in another video frame. As a result, unrelated set of events may be generated from such annotation works.

Ontology and rule learning based video annotation and retrieval is proposed in [3]. The ontology contains concepts, concept instances and their linguistic relationship from WordNet and rule learning is done from the ontology. Once a rule is learned, it is applied to the ontology that contains instances, obtained using semantic classifiers, to automatically extend the video annotation. For example, ontology contains the instances, „„airplane‟‟, „„sky‟‟ and „„ground‟‟, detectors which is an “airplane take-off event”. These detectors have been created using the Viola and Jones algorithm (provided by OpenCV) and color-based pixel classification with a

support vector machine, to detect and localize objects. Then, the spatio-temporal evolution of the appearance of concepts is determined using a tracker, based on an improved version of the particle filter.

Concept instances are associated with color and luminance histograms, which are used by the tracker to identify each instance in a video sequence. This work focuses on event extraction using rule based inference. A correlative multilevel video annotation is also proposed in [10] to address the issue of modeling inherent correlation between concepts and maximize the performance of concept fusion technique which is dependent on the validity of concepts generated by individual concept detectors. In this work, concept correlation is calculated between concepts identified by individual classifiers, using their mutual information.

Aiming to improve the quality of learning in an assumption of over-fitting existence while taking individual concept classification in SVM, a collaborative learning is proposed in [8]. There are different objects having similar shape, color and texture. For example: “cat”, “dog”, “tiger” and others. For such concepts, individual classification of these concepts may lead to a wrong result as a cat may be classified to be in a dog class or the reverse. For this reason, they have proposed to train a single classifier for related concepts. Concept affinities are estimated from visual information of different concepts for joint learning of different concept detectors. Later in the boosting stage, all the detectors gradually focus on independent learning for discrimination of related concepts. Under such a strategy, classifiers are learned and combined sequentially with gradually decreasing collaborations. Rather than making it a two stage object classification process, this work can only consider maximizing the efficiency of the discriminating algorithm than looking for related ones, and finding the specific object class.

A semantic video annotation with the use of BVW (i.e., SIFT features extracted from key frame) is proposed in [13]. To classify images RF (Round Forest) classifier, which uses group of unpruned decision trees whose leaf nodes are labeled with estimates of the posterior distribution over the image classes is used. These trees are built on randomly selected subspaces of the training data. RF classifiers show better accuracy in classifying multi-class high dimensional data with lesser computational requirement compared to SVM. The authors have done a spatio-temporal refinement over concept fusion approaches as the later ones do not give a good result.

This system provides shot level and object level video annotation allowing users to select objects of interest to be annotated in case there is a need by the user, or it will provide a shot level annotation. However, the annotation is on key frame with no consideration of shot dependencies and scene concept integrations.

Improvement on different manual annotation approaches has been done through different annotation tools like ANVIL[16], ELAN[17], etc. by adding a public netbook to a web annotation tool for any user to log and collaboratively annotate a video [22]. This is done for the purpose of improving browsing and searching capabilities especially in digital archives. SemiTube [22] is a client server annotation system where a client can select fragments and regions from video sequence and provide annotations, whereas the server module is responsible for user management, annotation authorizing, and annotation search. This work aims to link different annotations and create a semantic annotation network among annotated media objects.

Aiming to define co-occurrence relationship among multiple concepts, a Bayesian network based concept inference is proposed in [12]. For example, given a key frame with "sky", "road" and "car", it is more likely that a "car" comes with "road" than with a "sky" considering their dependency relationship. This network of object dependencies is constructed from large dataset so that an inference can easily be done. This is similar to semantic diffusion tree in [11] and used to annotate a single key frame.

Local invariant region descriptor based video segmentation and content extraction algorithms are used in [15] aiming to have robust video annotation and retrieval. While identifying a shot, local region descriptors are identified using maximally stable external regions (MSER) and SIFT. Using a greedy algorithm the difference between those regional features of consecutive frames is computed and, when the difference in consistency of local features is above a given threshold value, a shot is identified. This approach is computationally costly as SIFT feature vectors are dense. It can be enough to consider color and edge histograms as there is a small change between frames of a single shot.

[16] http://www.anvil-software.org/
[17] http://www.mpi.nl/corpus/html/elan/

Here, considering the whole set of frames for content extraction assuming the presence of sudden changes that can occur in camera movement and illumination will be costly. They also assumed that features extracted from one or more key frames are not robust enough to adequately represent concepts in a shot. They tracked extracted local regions throughout the shot while it is enough to work with a key frame. Though this approach is assumed to be good for scene and shot cut detection, it would not be feasible to work with every frame in a shot for concept acquisition.

3.3 Domain Specific Video Annotation Works

In transport domain, an ontology based approach that uses SIFT feature of key frame images is proposed in [19]. A linear SVM classifier with histogram of visual words is used. Ontology is constructed in an assumption of supporting the annotation containing objects identified by the classifier as root nodes.

The ontology is only compulsory to know the high level class of identified objects. It can be done using classifier with a proper class labeling. The annotation is restricted to identifying only labeling transport domain objects in the video.

In sport domain, Assfalg *et al*. [20] proposed video annotation approach that uses both visual features and graphical features like text captions on video frame. The authors assumed that two types of video scenes exist in a sport domain video, where one is a studio or an interview scene, and another one is the actual sport event scene. These scenes are identified considering repetitive shots with some intervals of variable length throughout the video sequence so that similarity of these shots will be computed to ignore them in further processes.

An object that remains stable for certain amount of time is a graphic object; and the approach looks for corners that exist among consecutive frames. In order to classify visual shot features, they have used edge detection and segmented the image frame. They have also used color features to increase robustness to edge based classification.

The main aim of this research work is to identify players, playing filed, audience and sport type. While identifying sport type, the playing field is used assuming that it is specific for a specific sport type and used a color histogram over the playing field. This work focuses on object tagging

where objects are classified and tagged rather that a scene based concept processing. Even event prediction is not done except identifying the sport type which is from the field type. The use of color histograms for a playing field representation and defining a sport type is not sufficient to differentiate sports which can be played on similar fields. For example: a “badminton” court is somehow similar to that of a “volleyball” court.

A semi-automatic annotation of human actions in a video is proposed in [21], which uses movie scripts as an input for the annotation to minimize the cost of finding manually annotated datasets for classifier learning. They have used OpenNLP toolbox of speech tagging to identify instances of nouns, verbs, and particles, and named entity recognition to identify peoples‟ names. Having events and their orders in the script of a video, it would not be easy to find the exact place of the event in the video sequence because scripts do not have time information. As a solution for this they used a temporal localization of dialogues in the script matching script text with the corresponding subtitle using dynamic programming. They have claimed that still with dialogue time support, a precise localization of actions will not be gained in a script based annotation and they proposed a discriminative clustering approach to find correct segment locations.

The availability of dialogues and scripts is limited to movies. If dialogues are found, inferring events from dialogues can be done rather than matching scripts with dialogues. Even dialogs and scripts may not be similar. For example, the script “*He closed the door while talking about his day . . . and he says . . . .*” and the dialogue that he speaks is about his day “I had been consulting my customers . . .”, it is true that the dialogue statement can be found in the script but action filtering should be done as it is difficult to find the event happening in that video frame is the one before the dialogue or after the dialogue.

An approach to annotate video products is proposed in [24] which aim to use annotation of similar web images in the video frames. These products are mined from Google image and amazon using BVW for product similarity and SVM classifiers to identify the product on the key frame from the training dataset.

Ontology reasoning is considered to annotate soccer video events and objects in [25]. In this work, they have done face detections using color autocorrelogram with k-nearest neighbor

classifier and the recognition is done with SIFT visual features around eyes region. For text captions, MSER features are used to detect actual regions and OCR is applied to identify the actual text from those regions. Playgrounds are detected using color histogram information over different regions. Finally, an event is inferred with spatio-temporal reasoning using the soccer ontology constructed. An example of the inference is shown below.

> *IF a shot-on-goal is followed by crowd AND at least two player-close-up, AND all this is followed by a score-change event AND the time interval from shot-on-goal to score-change is between 40 and 80 seconds THEN a scored-goal event is asserted for the interval from the shot-on-goal to score-change (included)*

The main drawback of this approach is number of rules to define as well as concepts that are going to be defined by the ontology require great deal of motion analysis. For example player-close-up may require isolating intersection of two players while running with that of two players having a hug.

3.4 Video Annotation Systems Considering Audio Features

In [9], an automatic video annotation method that detects and label foreground region of interest especially rigid moving objects using appearance and motion information is proposed. In each video sequence, scale invariant feature vector SIFT is used to extract feature points of extracted objects. For background objects, motion vector is calculated for each pair of SIFT feature points in adjacent frames. Then quantization process is applied on the motion vector to categorize the motion vectors into small number of classes. For each class, bin of histogram is calculated and a class with the maximum bin value is selected to be the foreground.

In order to find the boundaries from those SIFT foreground interest points, spatial distribution of those points will be defined with a Gaussian distribution. If (X, Y) denotes the centroid point of the foreground SIFT feature, and ∂x, ∂Y denote standard derivations with respect to X and Y respectively, the upper left corner and right lower corner of the boundary regions are ($X+2\partial X$, $Y-2\partial Y$) and ($X+2\partial X$, $Y+2\partial Y$) respectively. In this work, a consensus foreground object template that looks similar image patterns from the video frames and identifies the probability of a pixel to be part of that foreground object is used.

Audio features are used while finding the class level of the foreground objects training a GMM (Gaussian Mixture Model) classifier and the visual classification is conducted using HOG (Histogram of oriented gradients). The audio signal in the video is used to compute similarity between audio in a given foreground object and the training dataset. This approach is limited to foreground object identification and labeling rather than identifying events that those foreground objects are creating. It also requires a set of audios recorded from different objects as to identify the right object type. Two objects with similar audios may result an invalid classification.

Another approach in [18] exploits multi-contents of videos, visual features and speech features through parallel processing in a way it extracts scalable color and homogeneous texture from both the un-segmented shots and the segmented regions of the key frame, where the speech is automatically recognized and stem words are taken to be important words for the annotation. They have used two kinds of prediction models which are the rule based and the statistical models in order to predict frequent item sets and keywords of annotations. They have used a speech association rule to check the relatedness of the keyword and the speeches.

3.5 Summary of Related Works

The concept of video annotation is to give a meaningful description to the video content organized to a set of objects and events defining of the relationship between objects. Most existing works on video annotation [2, 5, 13] do not isolate the concept of image annotation with that of video annotation. A video is split into frames and annotation relies only on a key frame which relies on image annotation. Performing a single key frame based annotation results in unrelated set of annotation concepts as a result of independent processing of the key frames in the video sequence for event extraction and concept formulation.

In addition, some of the related works [24] are even specific to a single object type in the video and annotation is identifying and tagging an object in a video. Processing a single key frame also requires in depth analysis of object co-occurrence in order to identify an event that happens in the key frame.

In annotating a video, a strong concept dependency and relatedness has to be done among key frames representing shots in a scene so that a fine annotation result will be gained. In any video,

shots in a scene are dependent on one another in a way something that happens in previous shots will highly define what is happening in the current shot. As a result, individual annotation of these key frames cannot give a good result in video scene annotation.

All the works in video annotation provide object or event names as annotation result in either text format or XML format. From the importance of a video annotation, a formal statement should be built to describe a scene in a video. This formal statement can easily specify what is really happening with what and it can state cause and effect relationship between objects and events clearly which will then be understandable by viewers of the query video. In addition, it can be easier to work with these statements for a text based video summarization process.

Some works try to incorporate different knowledge sources than that of visual datasets while identifying events and objects. Such sources like scripts and dialogues are not available in all videos. Not only may the unavailability of these scripts, but similar full length videos that they consider to extract these scripts may not be found. A copy of a video may not always be found; even though it is found, timing information is highly required. Computational cost of full video similarity computation between a query video and set of videos in the video dataset is also high. Similarity of two videos may be most of the time on a single scene or a set of shots, as a result, looking for other video will be time taking and costly. This can be easily done using an image dataset where the similarity is not as such complex. Audio components are available in almost all kinds of videos as compared to scripts and subtitles and they can be good inputs for annotation.

Generally, existing works are key frame based annotations with no inter shot concept relatedness and dependency processing which results in unrelated annotation tags. This can be applied to image annotations as annotation depends only on a single image frame. Some of them are done selecting a specific approach for specific object identification and labeling process which cannot be applied to video annotation as there are different types of objects in a video.

CHAPTER 4
AUTOMATIC VIDEO SCENE ANNOTATION AND SUMMARIZATION FRAMEWORK

4.1 Overview

In this Chapter, we present our scene based video annotation and summarization approach. Different components of the proposed AVSAS (Automatic Video Scene Annotation and Summarization Framework) architecture are described along with relevant techniques and proposed algorithm.

Video annotation is the process of giving semantic description to an input video. The semantic description should clearly describe objects and events of the video. Apart from object identification and labeling, the annotation should be in a formal sentential form which describes objects and object interactions in a clear way. In this work, the annotation process start with annotating objects and events in a key frame which represents shots followed by annotating a concept within a scene hierarchically. The concepts in a scene could then be analyzed in a way to provide a textual summary of the video.

4.2 Video Annotation Requirements

The following are list of video annotation requirements that we are following while designing and developing the proposed AVSAS.

Automatic*:* The task of video annotation should be automatic so that user‟s interaction is minimal. Basic sub-tasks in annotation that are required to be automatic are listed as follows.

- *Video Scene, shot and key frame identification:* A user is not required to manually identify and point where a scene, a shot and a key frame starts and ends as it is too tiresome and time consuming. This gets harder if we consider the motivating scenario of our research which is, giving these tasks to users while annotating web videos like those in YouTube could be difficult. It is not computationally feasible to consider the whole set of frames in a video so

that a representative key frame which carries a content of a shot should be compared from set of frames and identified automatically. As the annotation is for a video scene, a scene has to be clearly identified and kept with its set of time intervals. The AVSAS system should be capable of analyzing each key frame representing shots in a video to cluster them into different classes of scenes. It has to compute dependencies and relatedness between shots throughout the video as shots of a scene could be distributed across the video at a different time frame.

- *Object identification:* The task of object identification should also be automatic in a way objects of interest have to be segmented and isolated from a given key frame. A key frame may contain a number of objects but the annotation should use those foreground objects which constitute the contents of the video.
- *Learning:* Automatic similarity matching has to be performed as to identify concepts behind objects and their spatio-temporal relationship. Given a training image or video dataset, the system should automatically find out those which are similar to the query identified set of objects.
- *Audio segmentation:* Given the time frame of each video scene, the system should be capable of segmenting the audio so that an associated audio scene would be generated and processed to use its text equivalent as one source of the annotation.
- *Annotation tagging:* A video annotator should automatically tag annotation free text into a playing video so that users can view annotations while watching a video.

Hierarchical: Video annotation shall be done at different granularity level. The process starts with identifying and annotating objects and events found in a single key frame. The system tags concepts in key frames and merges these concepts to come up with shot level annotations. The fusion of these concepts shall be processed to give a scene level annotation. Video summary which can be in the form of text could also be generated applying some language processing.

Accurate and informative: Apart from the concept of event and object annotation, the result of scene based annotation should be descriptive. In previous works, key frame based annotation is in the form of words or phrases but a scene requires construction of formal sentence. For this reason the system should give a grammatically correct statement that gives informative description about the content of a scene.

Annotation format: The system should provide annotations in a format which can provide two basic classes of importance. One is for further processing of video documents like indexing, classification and retrieval of video scene. The other one is allowing users to view the annotation in a playing video. In both cases the format can be RDF/XML where the semantic description of the video content can be defined with such structures so that it can be further processed.

4.3 Preliminaries

In this Section, we have defined and presented basic concepts relevant for this work.

Definition 4-1: [Video]:

A video is a sequential set of image frames that are clustered within different set of scenes to give meaningful information. Beside a set of image frames, a video also may contain audio and text. Our approach considers visual scenes and audio features of a video while performing annotation processes. For this purpose, we can define a video formally as:

$$V = \{\{V_{s1}, \dots, V_{sn}\}, M\} \quad [4\text{-}1]$$

where:

V_{si}: video scene

M: Metadata descriptor of the video: such as name, title, subtitle, rating, length, frame-width, frame-height, frame-rate, bit-rate, channel, sample-rate, artists, etc.

Definition 4-2: [Scene]

A scene VS in a Video V is a sequential set of shots that holds meaningful information which is a combination of concepts that each shot carries. The scene S_i in V is located at a specific time frame T_i and has a textual annotation. It is formalized as follows:

$$VS = \{\langle S_1, T_1, A_1\rangle \dots, \langle S_m, T_m, T_m\rangle\}, \quad [4\text{-}2]$$

where $\langle S_i, T_i, A_i\rangle$: represent shot S_i at time frame T_i having annotation A_i

Definition 4-3: [Shot]

A shot S of a scene is a sequence of frames captured at a single point of camera move. This sequence of video frames can be represented with a single key frame or set of frames while used for annotation process.

$$S = \{F_1, F_2, \ldots, F_p\} \quad [4\text{-}3]$$

where F_i: key frame

Definition 4-4: [Ontology]

Ontology in this work defines a set of objects and their relationships that can be found in a video. These object relationships are modeled as object properties and further they will be inferred to identify an event which is happening in a given shot.

$$Ont = \{Obj_i, Obj_j, R\}, \quad [4\text{-}4]$$

where Obj_i: object and

R: relationship between the two objects

4.4 System Architecture

As shown in Figure 2-1, video scene annotation and summarization system takes a query video as input and generates a free text annotation of the video. Once this is done, a high level summary of the video would be generated to be part of the final annotation result; which can be used for content based video processing.

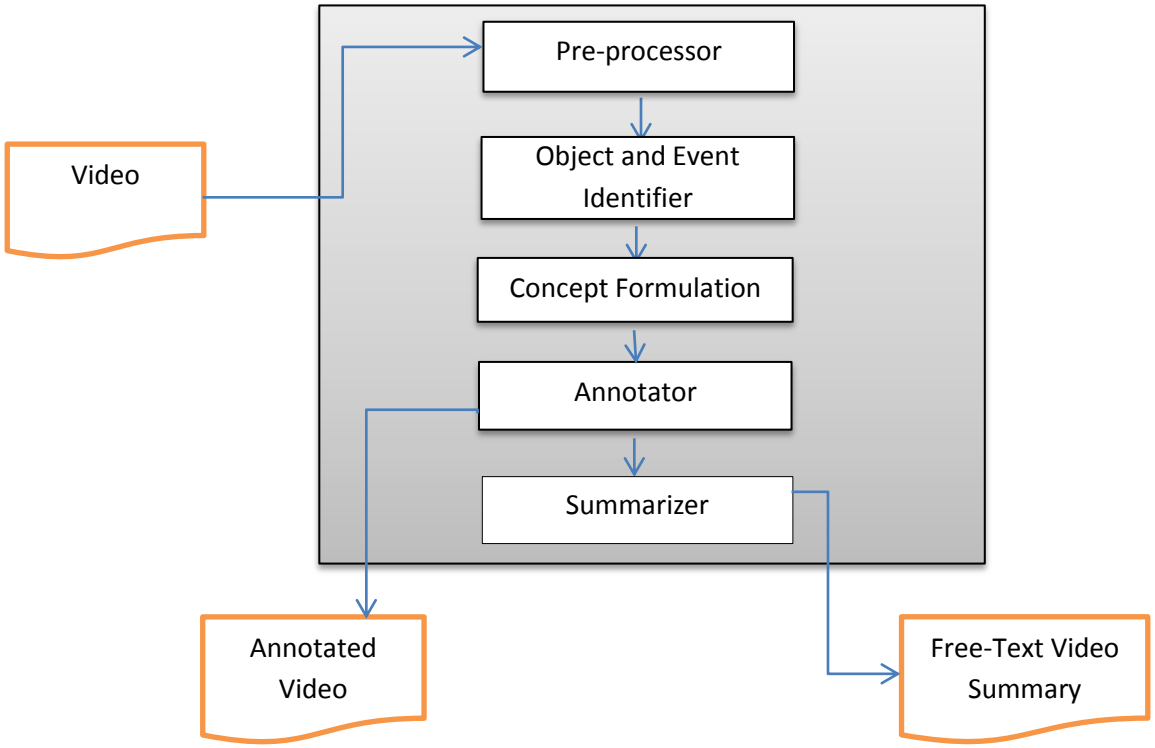

Figure 4-1: High Level AVSAS Architecture

Initially, the video is split into visual and audio streams so that its corresponding features will be extracted and processed separately. Each component of the video will be pre-processed and used as input for object and event identifier. The object and event identifier analyzes and extracts concepts representing objects and events in the audio and video scene. Concepts associated with each object and event in the scene is passed to the concept formulator along with their corresponding time frame. A formal annotation statement will be generated after a recursive concept fusion process in the concept formulation module. This component uses the result of audio object identifier to improve annotation quality. The annotator module generates a file that contains free-text annotation of scenes in the video. In addition to the scene level annotation, a summary of the annotations is generated with the summarizer component as high level free text summary of the video.

Video annotation basically relies on key frames starting with identifying objects of interest within those frames and defining the spatio-temporal correlation between those objects. Objects of interest are those which can clearly define a concept within a shot. For example, given a video

shot of football game with visual objects person, ball, net and sky, a sky object is not a point of interest.

Considering the audio of the video, objects of interest are extracted. For example, in a football game, the narration of a commentator is very important; a speech recognizer with a good part-of-speech-tagger can be used, to identify objects and actions which can be used in relation with the visual feature objects for a fine annotation result. While processing the audio signal, time stamps (time of occurrence) of the visual objects may not be exactly similar with time stamps of the audio feature objects. For this reason, a point in a visual data where objects and events are extracted will be identified from the audio signal. The result of video annotator is highly dependent on the quality of the pre-processing activities which include proper identification of key frames, shots and scenes. This component is detailed in the next sub-section.

4.4.1 Pre-processor

The pre-processor module is responsible to make the video ready for annotation processes. It accepts a video and sends it to both the audio and visual processor for initial tasks on the query video as shown in Figure 4-2. The result of the pre-processor module is a cut point of a scene, representative key frames of each shot, shot cut points and audio segments depending on the scene time information from the visual pre-processor.

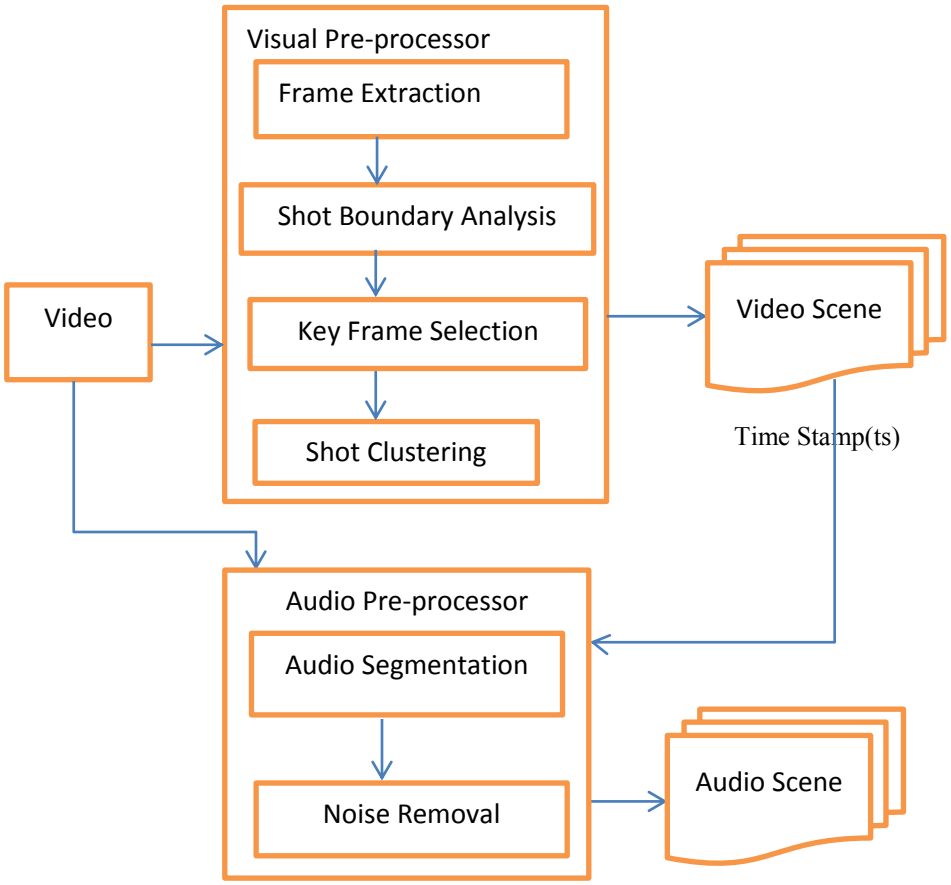

Figure 4-2: Architecture of Video Pre-processor Component

4.4.1.1 Visual Pre-processor

Given a video, the first task in the visual pre-processing is frame extraction in a video. Frame extraction is the process of splitting the video into set of image frames taking into account the frame rate of the video. Frame rate is the number of image frames that will be displayed per second. Mostly, the frame rate of videos range between 25 and 30. *Algorithm 4-1*shows how the visual pre-processor segments a video into set of frames using frame rate automatically extracted from the video metadata or default value of 25.

The result of the visual processor is clusters of shots in a class of different scenes with the time stamp which is the starting and ending time of each shot in a given video. It is not always correct to consider the starting and ending time of the first and last shot of a scene as its time stamp

because shots of similar scene may be located at different locations in a video. Considering a movie as an example, a scene might be partitioned and edited to be displayed at different time frames within the video. This distribution of shots of a single scene will be detected using shot clustering and their dependency will be analyzed in detail while formulating a concept that a scene carries.

Algorithm 4-1:Frame Extraction Algorithm

```
  Input: V: Video
Variables:
  FrameRate: Int
Output: VF: List  // List of Video frames
Begin
  FrameRate = V.GetFrameRate()
  If FrameRate = Null THEN //if FrameRate is not set
    FrameRate = 25
  END IF
  VF = V.GetVideoFrame(1/FrameRate)
  Return VF
END
```

The task of identifying shot boundaries requires a detail analysis of similarities between frames. Low level features like color and edge are used to compute histogram difference between frames. Considering these low level features, computing histogram over the whole region of a frame and differentiating it with other frames results in a poor boundary. This is because even though there is no camera move, new objects can be part of the shot at any point which will create a change on edge and intensity values; resulting in a big difference among frames of similar shots. As it is indicated in *Algorithm 4-2*, we use a region based bins of histogram, especially corner based histogram, as the change in intensity at corners is minimal.

Another problem in shot boundary detection is transition effect. To handle this, bin of histogram difference between frames in a range of 2 or 2.5 seconds is used as any transition will not last

more than two seconds. As shown in *Algorithm 4-2* the similarity computation is between a frame and its 50th equivalent. If the two frames are similar it will look for the next 50th frame or it recursively looks those frames before the 50th frame.

Algorithm 4-2: Shot Boundary Detection Algorithm

```
Input:
  F Frame, // Frame
  timestamp TimeStamp,
  FrameSimTreshold Float //Frame similarity threshold
Variable:  Temp_Frame Frame
Output:
  Shot time log//shot boundary
Begin
    Boolean Flag=True;
    For every frame F_i in F
     If(exist(F_i+50)) then
      Temp_Frame = F_i+50
       Sim = ComputeSimilarity(F_i.RegionHistogram,
       Temp_Frame.RegionHistogram)
       If (Sim >= FrameSimTreshold)
         If (Flag==false)
           Timestamp = Temp_Frame.timestamp
           F_i=F_i+51
           Flag = True
         Else
           Temp_Frame = F_(IndexOf(Temp_Frame)  + 50)
           Flag = true
       Else
         Temp_Frame = F_(IndexOf(Temp_Frame)  - 1)
         Flag=false
       End if
```

End if
Next
End

Considering entire frames within a shot to identify concepts is costly. Thus, representative key frames are selected for each shot and further processing will be on those key frames. Considering the definition of a shot in *Definition 4-3*, a shot may have different frames and it is likely that a single key frame may not contain all concepts in a shot though the shot is taken from a single camera move. This can be more understandable if we consider a surveillance single point camera where we have different moving objects which can pass by that point. For this reason it could not be enough to consider only a single frame as it is not easy to find a single centroid frame from such shots. A key frame selection shall be conducted by organizing frames in a shot into different clusters as shown in *Algorithm 4-3* and selecting the centroid frame of each cluster to be a member of the key frame set depending on its content similarity value using *Algorithm 4-4*. A frame with the minimal difference with all the others is selected to be a representative key frame.

Algorithm 4-3: Frame Clustering Algorithm

Input: Set of frames in a Shot *Framesimilaritythreshold*
Output: *FrameClusters*
Variables: *Cluster NewCluster*
Begin:
$F_{initial} = F_0$ *of the Shot*
For each frame F_j *in the Shot*
Initialize NewCluster with $F_{initial}$
If(Similarity($F_{initial}$*,*F_j*))==1*
NewCluster.add(F_j*)*
Shot.remove(F_j*)*
End if

```
    Next
    If(Shot.length!=0)
      Goto 5
    End if
End
```

Algorithm 4-4: Key Frame Selection Algorithm

```
Input:
  FrameClusters
Output:
  Key frame set
Variables:
  ArrayList SimilarityResult(Sr)
Begin:
    Foreach cluster Ci in FrameCluster
     Foreach frame Fi in Ci
      Foreach frame Fj in Ci
       Sr[i] = Sr[i]+ Similarity(Fi,Fj)
      Next
     Keyframeset.add(F((IndexOf(Min(Sr))))
    Next
   Next
End
```

Identification of scene requires detail analysis of concept dependencies between shots to cluster similar shots into a single class of scene. Referring to Definition 4-2, a scene is a sequential set of shots distributed over the video. In identifying a scene, it is a necessity to consider the whole set of shots in a video. Identifying a scene needs computing similarity between key frames using low level features of key frames of different shots. Scene boundary detection is supported with object matching mechanism as shown in *Algorithm 4-5* to improve the false negative result of the

similarity computation. When a new object becomes a member of a shot, low level image similarity results become poor as a result of a change in intensities and color. Thus, set of objects in the shot can be compared to identify whether a key frame is part of that scene or not.

The object matching for scene boundary detection can be done by calculating the difference in the number of similar objects between consecutive key frames as shown in *Algorithm 4-6*. The change in the number of similar objects and intensity difference is minimal while comparing consecutive frames than between the first frame and every other frame. This will maximize the efficiency of the scene cut identifier.

Algorithm 4-5: Object Matching Supported Scene Identification Algorithm

```
Inputs:
   S List, //Shot
   St timestamp, // shot time stamp,
   Threshold float, // Similarity threshold
Output:
  Scene_time_log
Begin
   Foreach ki in S
     X = new cluster()   // create a new cluster
     X.TimeStamp = Ki.timestamp // associate time stamp
     Foreach Kn in S
      If(((HistogramSimilarity(Ki,Kn))>     Threshold)||
       (ObjectTracking(OKi,OKn)==True)) then
                /*bjectTracking(OKi,OKn)     is     the
                similarity of two frames in terms of
                the number of object occurrence in Ki
                and Kn*/
           X.TimeStamp = Kn.timestamp
           ShotSet.Remove(Ki)
```

```
            Ki =Kn
        End IF
      Next
   Next
End
```

In object matching, key frames of each shot can be segmented to get set of objects where this can be also an input for object identification process. In this work we have done a manual segmentation of interesting regions from key frames. For video annotation process, identifying objects at scene boundary detection level cannot have any effect as those objects are required for further processes.

Algorithm 4-6: Object Matching Algorithm

```
Input:
      Object sets of key frame (OKi) and (OKj)
Output:
      Result: Boolean
Begin
      If count(Intersection(Oki,Okj)<
         Average(count(SetdifferenceOfObjects(OKi,OKj)),
            Count(SetdifferenceOfObjects(OKi,OKj))) then
            Result = true
      Else
            Result = false
      End IF
End
```

Techniques of Visual Similarity Analysis

In video pre-processing, frame similarity analysis can be performed at different stages: identifying scene boundary, shot boundary detection and key frame extraction. The similarity

analysis is used to measure how much objects (for instance frames) are similar and used later to classify them into a cluster of shots and select a centroid frame of cluster to be a key frame representing shot. This centroid frame could be one or more frames according to the length and concept distribution in the shot.

While performing similarity computation between frames, especially those within the same shot cluster, it is enough to consider edge and color histogram values as frames within a shot contain highly similar edge and color distributions unless there is a change in motion of some objects. This change in motion makes a difference highly on the edge values and to some level on the color distribution.

Given two image frames, first bins of color histograms of each frame can be computed. For each frame, 64 bins of histograms will be used and their similarity is processed depending on those bins. The bins are defined on the basis of the number of pixels with the RGB color ranging from 0 to 255. The sum of pixels having an RGB value in a range of 0-3 will be kept in a first bin and it continues grouping pixels with the next four consecutive RGB color values until it reaches 255.

Having these bins of histograms for both frames, then sum of differences between the 64 bin values will be calculated and the result of this computation will be compared with the threshold value t. In this work we have manually set the thresholds while performing similarity between images at different stages. If the result is above t, then the two frames are considered to be similar; else not. The number of bins to use determines the result of the similarity computation. If the number of bins is large, the result will be much better than having a minimal number of bins as having small bins makes pixels with large RGB value gap to be grouped under one bin cluster. Therefore, in one frame we may have all pixel values in a given cluster range with a single RGB value but in the other frame those pixels in that cluster may be of cluttered RGB value but the result will come to be similar which is an invalid conclusion.

Computing bins of edge histogram is done in a similar way to that of color histogram. Two frames are similar if both edge and color histogram differences of the two frames are above their respective threshold. The threshold might not be similar for both edge and color based similarity

in a way it has to be a little lower for edge based similarity than that of color based similarity. This is because, edges are highly prone to noise and they are not invariant to motion. It is a challenging task to set these thresholds automatically as it depends on the quality of these videos and it is also dependent on the content where videos with many color combinations may require a higher threshold and for those with minimal number of color or intensity changes, it is best to use lower threshold values.

Generally, video annotation is highly dependent on the results of the pre-processing steps. Well generated results of this step fit best to succeeding processes and give a better result. For example, a poor key frame or a misplaced scene cut results in invalid annotation. The first result of video pre-processing module is the time signature of each scene which will be used as an input for audio pre-processing module in order to split the audio signal accordingly. The time signature is used to put annotation results per scene.

Not only scene time frame but also time stamps of each shot are extracted in the video pre-processing step. As annotation starts by defining concepts in each and every key frame of a shot, ordering of concepts generated from these shots will be required while formulating a scene level formal statement. This is because during concept fusion we shall take into account the temporal ordering of concepts in the video. For example, a plane concept coming before a land concept is highly probable to refer to "*plane is landing"* than "*plane is taking off"*.

Finally, visual key frames will be extracted with their time stamps as in the case of the scenes and shots. These key frames are going to be used in later steps in order to extract concepts that the video is referring to.

4.4.1.2 Audio Pre-processor

Audio segmentation is a process of splitting audio signals as shown in *Algorithm 4-7* depending on the time signature of each scene which is derived from the visual pre-processing module. Audio in a scene can be used to identify those objects and actions in a scene. This can be more effective than shot based splitting as objects and events that are derived from shots may not fully define visual features within that given shot as an audio in a single shot is short. It is known that a speech within the video may be unrelated to visual objects thus there is object filtering in later

stages such as concept formulation. This module uses the time stamp of each scene extracted during pre-processing of video and extracts the audio signal per scene.

It may not always be possible to have a clear cutting point of audio scene only with the time stamp of the visual scene. The audio scene may start and end with a time which is somehow earlier or later. Here the audio segmentation algorithm should identify silence points near to the visual scene time frames so that point of audio scene cut could be identified. Silence points may not always be ending points of a narrator‟s speech or statement. However, this point will not affect the result as the audio scene is to be used only to identify objects and events in the video that the speaker specifies rather than being used to annotate the video directly translated to text as a whole.

Without a video feature processing, audio scene can be used to annotate the video. However the annotation result would be very poor due to the following reasons:

- *Noise*: audio signal in most videos is subjected to background noises, sounds from moving objects, animals, etc. Thus there is a need to apply intensive noise removal technique to clearly detect what the speaker says.
- *Relatedness*: Most of the time a speaker narrates about events in more descriptive manner by adding some related facts and events to clarify and give more information to the viewer (for instance, in a news video a journalist narrates about events while the camera focus is on the narrator/journalist). In doing so, new objects and events might be added to take part in the video.

Having the narrative audio scene with minimal noise, the main question is relatedness and this is why we could not use the whole audio scene statement as is for annotation. In this work, we extract objects and events in audio scene and further process and check their presence in the visual content in order to use them accordingly as shown in the object ranking algorithm.

Algorithm 4-7: Audio Segmentation Algorithm

Input: Audio signal, Visual scene cluster time log
Output: *AudioSegments*
Begin:
For every scene S_i in the video scene cluster
Create a new audio scene AS_i
ASt = FindNearBySilencePoint(Audiosignal, FindStartingTime(S_i))
AEt = FindNearBySilencePoint(Audiosignal, FindEndingTime(S_i))
AudioSegments.add(SplitAudio(Audiosignal,ASt,AEt))
Next
End

Generally, the aim of having the audio scene is to support the visual object extraction and identification as it is easier; relationship between objects and events can easily be identified. For example a speech from commentators in a football event can be used to identify events happening in a video more easily than using video visual features. It will also be easier to build the final scene annotation statement directly from the translated speech with a minimal concept normalization process than building it by fusing objects identified from visual processor.

The nearby silence point of an audio signal will be calculated depending on the time frame given and the algorithm looks backward to find the nearby background silence point as well as forward to find the nearby forward silence point. Finally, the one which is nearer to the given cut point will be used to split the audio signal. If a silence point which is exactly between statements rather than words is identified, the process of event and object index construction can be efficient.

Though it is not implemented, once the audio signal is segmented, a noise removal should be done to disregard background sounds so that event and object identification process results in a good output. Generally, the task of the pre-processor is identifying scenes, shots, key frames and scene level audio signals which are used as an input to the object recognizer module.

4.4.2 Object and Event Identifier

Identifying objects and events, representing the relationship between objects at given time and location, from the input video is a precondition for annotation. Object recognition is a process of identifying different objects in a frame. In this work, objects are extracted from visual features of key frames and the speech as explained in the next-sub-sections.

4.4.2.1 Object Class Name Identification using Visual Features

Given a key frame, a visual object class name identification process segments the image and identifies the class name of segmented key frame region by searching similar image from ImageNet image dataset computing similarity using low level visual features. From the object identification architecture shown in Figure 4-3 the process involves three steps. First it segments the visual key frame, extract visual features and perform classification process to identify set of objects and events.

Segmentation

Given a key frame, object identification starts with segmenting the key frame into n-regions of any shape where each region has a meaningful visual content. Though video is a sequence of image frames having different set of objects, not all objects are directly important while defining the video content. A meaningful content will be defined from the correlation of some objects called objects of interest. The final result of the segmentation process is identifying these objects of interest which highly define the content of the video.

Objects in an image can be categorized into foreground and background objects. Foreground objects are those that carry more information whereas background objects contribute very limited information to the semantic content of the video.

Though there are different image segmentation techniques [35], in this work we have used a manual segmentation of key frames for proper identification of objects that are found in the key frame. The segmentation is done by cropping regions in the key frame which are assumed to be objects of interests.

The result of the segmentation process is a set of shapes where their class label will be identified during the classification process.

Feature Extraction

Once segmentation is completed, a feature vector will be computed for each segment of the image frame so that the type of object will be searched from objects dataset. There is a requirement for these features to be scale and rotation invariant as to find similar object from the dataset. Scale and rotation invariant SIFT features can be extracted from segments as they are very efficient to find similar images from the dataset. In a condition where we cannot have absolute match of an object in a dataset, these features are not appropriate. Key points from such feature vectors can be more useful when searching for identical objects rather than finding list of similar objects even with a minimum difference.

It is possible to use either image or video dataset while learning different concept detectors. In our case we prefer to use an image dataset with the following reasons.

- *Image dataset availability*: there is a limited source of annotated videos which are kept for research purposes in comparison to image dataset. Most of the annotated videos are very short which are about 5-10 seconds representing a single event in a shot. Not only their length but most of them are domain specific sets. There are objects and events defining image dataset which are available in mass and in different domains as compared to that of video dataset.
- *Computational cost*: video annotation starts from low level object tagging which will further go to scene concept formulation. Storing and computing similarities between frames of a query video with each and every video in the video set will be costlier than image dataset.
- *Annotation format*: format of annotation is another factor in which annotation of videos specifies only the event which is on the video without listing set of objects in those sequential set of image frames. In images we can find shapes of objects with their annotations or event annotation texts. This will be highly required to define new concepts in relation to objects and events in other frames of a scene.

Classification

The classification process uses SVM classifiers in order to recognize an object computing the similarity between the segmented key frame regions and every element in the dataset. The dataset has different set of images with their class label and while the classification process, the segmented region from the key frame will be compared with those images in the dataset and when most similar image is found the class label of that image will be given to the segmented region image as a concept. To optimize classification, objects already found in previous frames are not going to be searched from the dataset again. In looking for an object, the object will be searched in local repositories that keep known objects. Once a similar image is found the free-text annotation of that image will be used as a new concept.

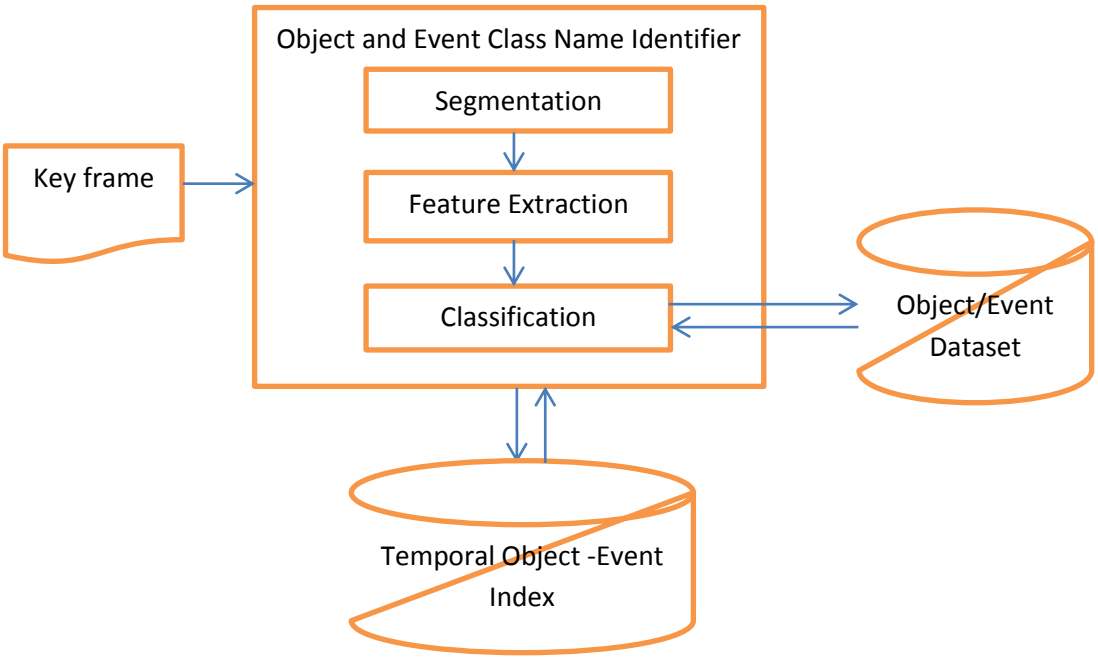

Figure 4-3: Object and Event Class Name Identification using Visual Features

Object class name identification as shown in *Algorithm 4-8* is a two way process where the system initially looks for an event considering the whole key frame and then it goes for segmenting the frame into different regions to identify objects. Both processes are required to construct a fine object/event matrix where it can be easier to build the final annotation result.

Algorithm 4-8: An Algorithm for Object and Event Class Name Identification using Visual Features

Input: *Shot number SID*
Variables:
Key frame Kf
Object dataset ODS
Event dataset EDS
Output: *Set of objects*
Begin:
Features = ExtractFeatures(Kf)
For every DatasetFeatureVector DFV(i) in the EDS

```
    If similarity(Features, DFV(i))==1
      Update event in an Event/Object Index table(SID,
      ExtractClassLabel(DFV(i)))
     Goto 15
     End if
    Next
    ListOfShapes.add(ToShape(Segments(Kf)))
    For each shape Si in ListOfShapes
     ObjectFeature=ExtractFeatures(Si)
     For each ObjectDatasetFeatureVector ODFV(i) in ODS
      If similarity(ObjectFeature, ODFV(i))==1
        Update object in an Event/Object Index
        table(SID,  ExtractClassLabel(ODFV(i)))
      End if
     Next
   End
```

4.4.2.2 Object and Event Identification using Audio Signals

An object can be recognized using audio signals in two ways. The first one is using the audio signal from inherent sound property of an object. For example, an object animal may be detected by processing its sound when it barks. The second one is using the narrator"s speech which will be translated to text and further analyzed by a speech tagger to identify nouns and verbs. By doing so, objects and actions within a given audio signal can be easily fetched. In this work, we use the second approach as it can easily detect actions than that of the first one in the presence of a speaker. One drawback of this approach is the fact that a speaker may talk about an object which does not exist in that video scene. Actions generated by the audio object recognizer can give correct concept relationships which will be basic while formulating a concept at scene level. These concept relationships are stronger and unambiguous than that of the concept which can be constructed from rules which are built from concept ontology.

Referring to Figure 4-4, audio scene which is extracted at video pre-processing stage will be given to the speech recognizer in order to get its text equivalent. There are different speech recognition systems though their output is kept in question. One of the works which has a good result is Google‟s speech recognition API which records the speaker voice and returns its text equivalent. It identifies a word considering silence as its cut point. This API can be configured to work with an existing audio signal which has to be in flac format and then be used to return the whole statement as a web response which can easily be read by any application. Another approach, Java speech recognition library, SPHINX, requires a formal definition of the language model. We used Google‟s speech recognition API as it has already defined language model though it is only in English.

One thing which is very important here is identifying the end of a statement. In an audio scene we may have a phrase, a single statement, a composite statement or even a set of paragraphs. Identifying these points will be very useful in order to identify an event which belongs to a set of objects so that object relationship can be easily extracted.

Taking the importance of an audio scene processing to the annotation process and the availability of videos in different languages, this audio scene object identification process requires a language processing module in two different ways.

- *Language identification*: the language with which the audio is recorded is identified before translating it into text. This will help to minimize unknown terms from the speech recognition module.
- *Language translation*: once the language is known, depending on the language model translation to English will be done and if the language is not English then a bi-lingual text to text translation will be done to make it converted into English. This translation to English is required because the result of the visual processing is in English and for further processing while concept formulation both the results of the audio and video processing should be in identical language. But this does not mean the result is always in English, it can be finally translated back to the user‟s preference.

Stanford"s standard NLP part of speech tagger is used for the purpose of extracting objects and events from the row text resulting from the speech recognition component. Nouns and Noun phrases are mapped into object. Verb and verb phrases represent event.

To optimize sentence construction processes while formulating concept, which is also another very important advantage of an audio scene processing, object and event occurrence will be logged specifying which event is happening between which objects with the statement number. It will be like building a statement indexing table having list of objects with their respective events and statements to which they belong to. This will be used to build a formal, well defined annotation statement.

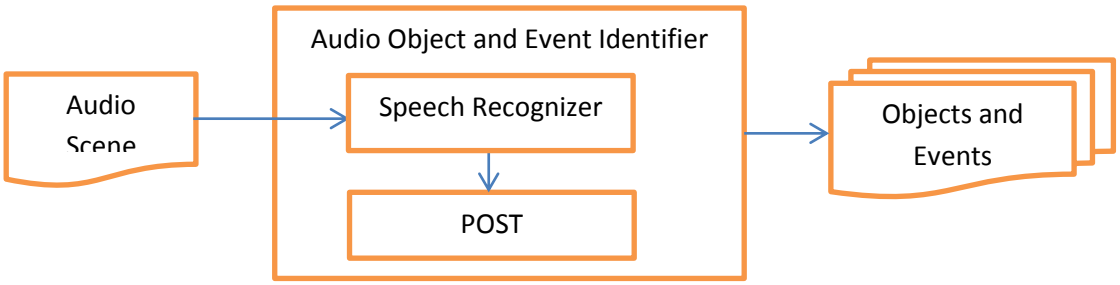

Figure 4-4: Object and Event Identification from Speech in a Video

Algorithm 4-9: An Algorithm for Object and Event Identification from Speech in a Video

Input:
Audio scene cluster
Output: *Object and event index*
Begin:
For each Audio fragment AF_i in the Audio scene cluster
Text= SpeechRecognizer.recognize(AF_i)
TaggedText=POST(Text)
For each statement s in the TaggedText
objectSet.add(ReadNouns(s))
eventSet.add(ReadVerbs(s))

Update event and object index(objectSet,eventSet)
Next
Next
End

The object and event index as a result of the audio processing shown in *Algorithm 4-9* is used to keep events and respective set of objects participating in the event and will then be used in a ranking module for an exact event extraction or object ranking. The result of the POS tagger is read statement by statement so that a clear object/event index table can be constructed. This allows an easy statement construction in addition to having a clear identification of objects and events. The result of the audio object identifier is highly dependent on the efficiency of the speech recognizer, the clarity of the audio signal and the accuracy of statement cut point finder.

4.4.3 Concept Formulation

The process of concept formulation starts from key frames and finally ends with merging concepts in a scene into a formal statement. In order to merge these concepts a language model will be used to summarize those key frame level statements or phrases into a general final annotation text. This will be done for the entire key frame in a scene and the result will be from the combination of the sentences of each key frame.

Referring to *Figure 4-5*, concept formulation process takes object and event class labels, rank them according to their relatedness and importance, fuse concepts in order to identify new events, define shot level annotations, compute concept relatedness and construct a scene level video description statement. Once objects are identified, the next step is concept formulation in which concepts representing objects are going to be fused together in order to give a meaningful description of the event which is happening in a given scene. Concept formulation in a video is done in a hierarchical manner starting at a frame level. Frame level concept formulation will be carried out by fusing concepts of identified set of objects of the key frame in two ways.

The first one is by considering event sets of the visual or audio scene in order to lookup the event or the relationship that exists between objects of that specific key frame. Here, the audio event will be checked and used as connector of objects of that visual key frame which can be identified

from the event and object index table, which in turn will be generated at the end of audio object and event identification stage.

One basic advantage of processing an audio data which is the speech is to minimize the cost of processing motion of an object or object interactions in order to define an event which is really happening in a given shot or scene. For instance, in a football domain video scene, a ball in the hands of a goal keeper is reported by a commentator as a keeper saved the goal. Processing this speech can easily identify the set of objects and events which are happening on the video scene and hence we can have a *goal keeper* which is of type *person* and a *goal* into the *object* set and a *saved* action in an event set. In contrary, consider processing the visual key frame; this requires intensive image processing techniques with a large image dataset. In the absence of event dataset, it will also require a domain specific object relationship network in order to define the actual concept behind the interaction of those objects.

These concept fusion and event dataset based approaches can also lead to multiple results for a given set of objects. For the above case having a person and a ball in an object set the resulting event may be “a person playing”, ”a person throwing”, which is somehow related but not similar to the one which is happening in that shot. With proper time handling and object filtering, events that can be extracted from the speech are accurate and most of the time represent what is really happening in the scene and hence can be used to give a better annotation statement.

The second way of formulating key frame level concept is by using object relationship definition networks which will be defined by using ontology. Building object relationship for generic domain object set can be done in an open way where any domain expert can participate. It can also be updated while annotation which can be generated from audio data. It is not feasible to be specific while constructing objects relationship hierarchy in a video annotation process as a video contains different objects and events of different domain. Most of video annotation works rely on single shot video where objects are of a single domain. In this case it is not required to build generic domain ontology.

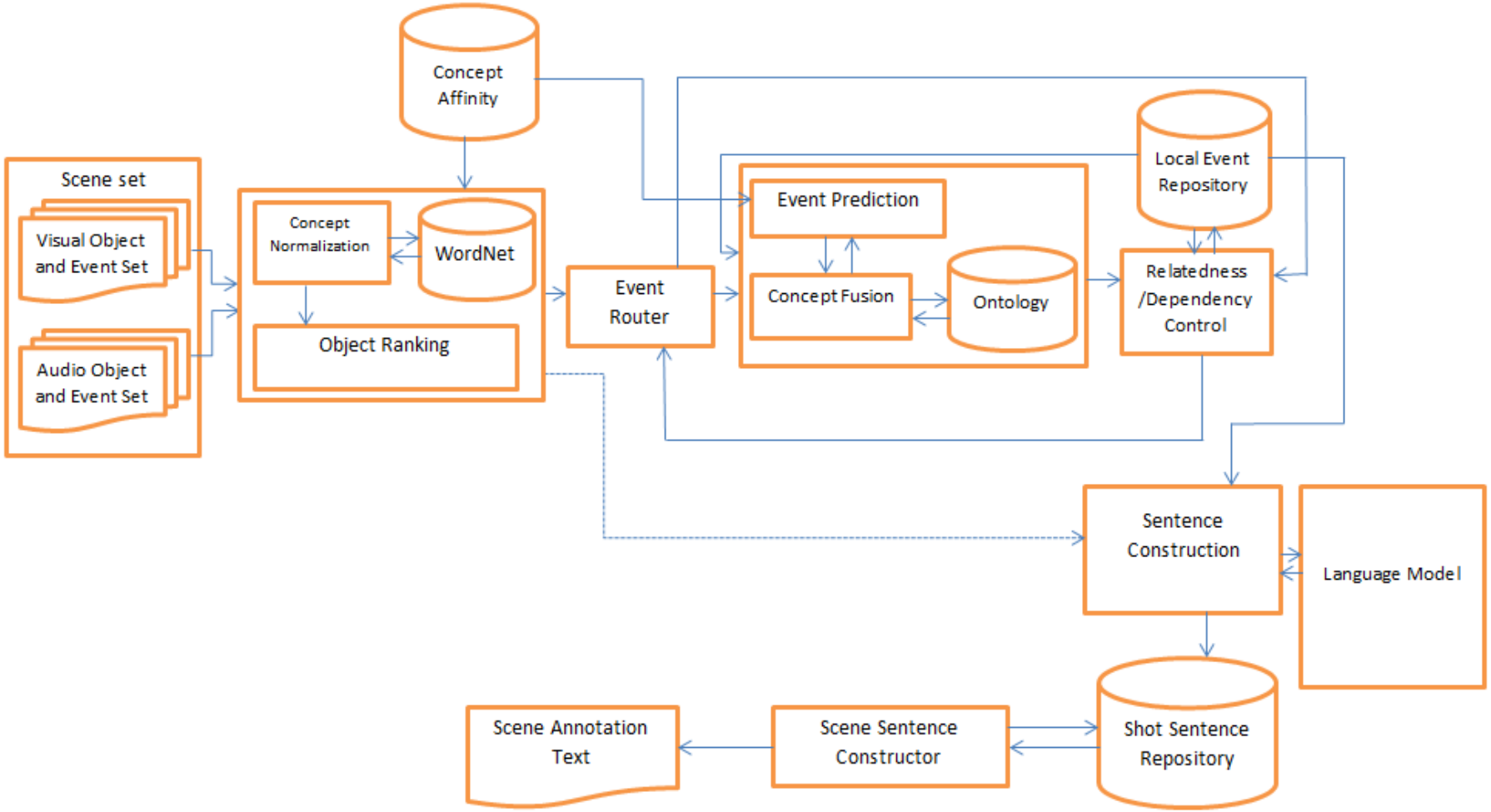

Figure 4-5: Concept Formulation Architecture

4.4.3.1 Concept Normalization and Instance Passing

Concept normalization is a process of computing the similarity between concept names of those in visual set and audio set in order to avoid redundancies. Concepts extracted by the visual object identifier and audio object identifier may have different names but could be semantically similar; this could be caused due to differences in naming of an object class in a visual processors dataset and the speakers naming for a similar object in the audio fragment. The concept normalizer handles such difference using WordNet concept network as in *Algorithm 4-10*. For example, a visual classifier might identify a "*person*" class object but that of an audio object identifier may have a "*man*" class object where both are similar concepts.

Higher class concepts are always better for concept formulation while lower class concepts or instances are good in an informative sentence construction. Here, instances are not going to be used in concept fusion stage as there are a number of instances for a single concept and trying to build ontology of those instances for object relationship inference would be complex. While constructing a sentence an instance of a given object can be fetched back and be used.

Similar object labels in different names in the audio and video object set could be merged and given a single name in the object/event index and if one of the names is an instance of the other it will be kept in the instance set of that object.

Algorithm 4-10: Concept Normalization Algorithm

```
Input:
 Video Concept Set VCS // both objects and events
 Audio Concept Set ACS // both objects and events
Begin:
 For every Concept Ci in VCS
   For every Concept Cj in ACS
    If LookUpSimilarity(Ci,Cj,WordNet)==Synonyms
      ChangeName(Cj,Ci)
    End if
    If LookUpSimilarity(Cj,Ci,WordNet)==Hyponyms
```

```
        If Cj and Ci are both Events
          ChangeName(Ci,Cj)
        Else If  Cj and Ci are both objects
          InstanceMatrix.Add(Ci,Cj)
          ChangeName(Cj,Ci)
        End If
    End If
  Next
 Next
End
```

4.4.3.2 Object Ranking

There could be more audio object (some of the objects are not related to the scene) sets having similar time stamp to the visual object set. Thus there is a need to filter out the unrelated object sets by assigning weight to each as shown in *Algorithm 4-11*. For every object which is found in both audio and visual object set, higher weight is given; otherwise lower weights is given. While fusing objects, all the objects will be considered depending on their weights. These low weight objects can later be used to find higher level concept definitions while working with ontologies. The following issues are the basic reasons for objects to be given weights:

- *Relatedness*: objects and events extracted from a single video scene are highly related and removing these objects and events might be ignoring the advantage of having additional information about something that is happening in the video. Their relatedness can be taken as an input for event prediction; in a way low weighted events and objects can be inputs to predict what can happen next taking high weight objects into consideration. They can also be used for dynamic updating of event affinity in semantic graph which can be done once their relatedness is checked.
- *Missing object while segmentation*: with current segmentation approaches, it is unlikely to find a clear boundary of a single object in an image and there is a possibility of missing an object during segmentation. This may be due to overlapping of shapes or image quality. Having this minimal object shape segmentation or false positive object

identification, the resulting formulated concept may not be accurate enough. Not only from the segmentation processes but also object classifiers might not be capable of identifying the class label of a segmented area of an image given an object dataset. Efficiency of object formulation can be maximized using these low weight concepts into consideration.

- *Event over fitting*: an event defined in an audio event set might not work for the visual object sets. The concept of assigning low weights to objects only in an audio object set is very important in order to filter out events defined using only in audio. Thus, weighting is important to minimize such over fitting errors which exist as a result of unrelated or unwanted concepts.

Object ranking module will give ranks to objects according to their occurrence and affinities and provide an event or set of objects to the concept formulator as shown in *Algorithm 4-10*. While ranking objects, for those which are found in both sets (audio and video object set) we give high weights. For those which are in the visual object set but not in audio object set we give middle level weights. For those which are related to high weight objects, but not found in the visual set, low weight is given and finally for others a zero weight value will be given and this objects are not going to be considered while object fusion.

The resulting structure of the object ranking module would be an event and object temporal index containing an object with its priority level, instance if any and event class as shown below.

$$OTI = \{\langle Obj_1, P, E, I\rangle \ldots, \langle Obj_n, P, E, I\rangle\}, \quad [4\text{-}5]$$

where $\langle Obj_1, P, E, I\rangle$: represent object Obj_i with priority P, event class E and instance I.

Algorithm 4-11: Object Ranking Algorithm

Input:
Visual object set VOS
Audio object set AOS
Output: RankedObjects
Begin:
For every object V_i in VOS

For every object A_i in VOS
If similarity(V_i,A_i)==1//check whether a visual object is found in an audio object set
SetObjectPresence==True
End if
Next
If SetObjectPresence==True //
UpdateIndex.SetWeight(V_i,High)
Else
UpdateIndex.SetWeight(V_i,Mid)
End if
Next
For every object A_i in AOS
If search(A_i,VOS)==0
If ConceptAffinity(A_i, Fetch(HighWeightedConcepts))==1
UpdateIndex.SetWeight(A_i,Low)
Else
UpdateIndex.SetWeight(A_i,Zero)
End If
End If
Next
End

4.4.3.3 Event Router

Event router module is used to route events and objects depending on their hierarchy to the concept fusion process according to *Algorithm 4-12*. For a single shot, if there is an identified event it will send that event with the set of objects to the next phase or it will look for the set of top priority objects and send them to the concept fusion process. When the concept fusion process fail to get an event with those high priority objects or if the event is unrelated to the

previous event list, the event router sends those objects with lower priority to look for other events in the concept fusion process.

Algorithm 4-12: Event Routing Algorithm

```
Input: Shot ID, Ranked Index Table, Visual Event Set
Output:
       An object set or an event
Begin:
 Event= SearchEvent(ShotID,VisualEventSet)
 If Event!=empty
   Return Event
 Else
   ObjectSet=SearchObjectIn(ShotID,RankedIndexTable)
   If max(ObjectSet).hasWeight>=Mid
     Event=SearchEvent(ObjectSet,AudioEventSet)
     Return Event
     Instantiate DependencyController()
    Else
     Return ObjectSet
     Instantiate EventPrediction()
     If EventPrediction.RequestMoreObject=True
       Return(Fetch(LowWeightObjects,ShotID))
     End if
   End if
 End if
End
```

4.4.3.4 Concept Affinity

Event prediction can be done from event prediction rules that are statistically built or inferred from affinity matrices. Event affinity can be computed in two ways, one is considering event with event co-occurrence and the other one is event co-occurrence with objects. Web mining can be done to be an input for a statistical approach of computing affinities and constructing the affinity network as shown in *Figure 4-6*. Given the training data set, affinity of events can be calculated as follows:

Event to Event affinity: shows how likely an event would appear with another event.

$$Aff(E1, E2) = \frac{number\ of\ occurrence\ of\ E1\ with\ E2}{Number\ of\ occurrence\ of\ E1\ in\ the\ dataset} \quad \text{[4-6]}$$

Event to object affinity: shows how likely an event E comes with an object O.

$$Aff(E, O) = \frac{number\ of\ occurrence\ of\ E\ with\ O}{Number\ of\ occurrence\ of\ E} \quad \text{[4-7]}$$

Calculating both affinities in combination with object to object affinity, it would maximize the efficiency of concept formulator improving the event prediction and object ranking processes. Having these values, a semantic graph whose nodes are events or objects and edge weights are values of the affinity between those nodes, can be constructed in a structure shown in *Figure 4-6*.

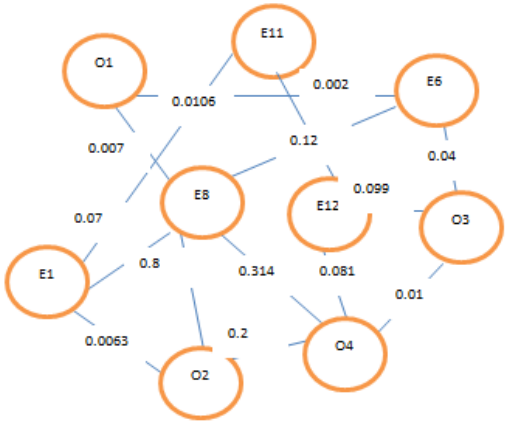

Figure 4-6: Semantic Affinity Graph Structure

4.4.3.5 Event Prediction

Object relationship ontologies are not efficient as they require large set of rules to be defined in order to predict the next event except defining an event given set of objects. For this reason a prediction rule should be defined from concept affinity matrix shown in the previous section so that in addition to the one we have from the concept fusion, a smooth event flow can be generated.

The order of event occurrence is represented in concept affinity graph so calculating the affinity between previous event fetched from the local event repository and those events in the concept affinity graph as shown in an *Algorithm 4-13* gives an event that can come with the previous event that is fetched from the local event repository.

Algorithm 4-13: Event prediction algorithm

Input:
Previous event Pe from local event repository
Set of objects O from object router
Affinity threshold T
Output:
Event
Begin:
For every event node En in the concept affinity graph
If affinity(En, Pe) > threshold
EventList.Add(En)
Flag=true
End if
Next
If (flag==true)then
For each Event E_i in the EventList
For each Object O_i in O
$EiAff_i$= $EiAff_i$ + Affinity(E_i,O_i)

```
    Next
    EOAffinityList.Add(EiAff_i,E_i)
  Next
  For each EOAffinityi in EOAffinityList
    If (EOAffinity_i.EiAff_i == Max(EOAffinityList))
     Return EOAffinityi.Ei
    End if
  Next
 Else
  Return SearchRelationshipsFromOntology(O)
 End if
End
```

Event prediction approach will maximize the accuracy of the annotation process as a scene is a set of related events. While making event prediction, if no result is found a simple concept fusion approach will be followed to define an event which is happening in that frame.

4.4.3.6 Concept Fusion

Another way of identifying an event is through concept fusion. This module will identify the event that can be inferred from the coexistence of objects. These object and event relationships are modeled in the ontology in a way nodes of the ontology are objects and data properties are events. These data properties have different domains and ranges where these domains and ranges are objects. For example, having a person and ball object given from the event router, a data property "isHolding" with domain "person" and range "ball" can be inferred from the ontology

Once an event is generated in either of the event prediction of concept fusion module, the result should be checked whether it is semantically related to the previous one as it is highly required for events of scene to be related.

4.4.3.7 Relatedness/Dependency control

Dependency analysis is a process of identifying whether an event at frame t+1 is dependent on the event which happens at frame t. The aim of having this process is to maintain the smooth flow of events throughout the video scene. A scene in a video is a sequence of frames where this frames are of similar information. In a key frame based annotation, there is no need to process what has been in the previous frames to come up with one generic description of a scene. But in scene based annotation, the information to be extracted from those consecutive frames should be related and the event that happens in frame i+1 should be initiated by the one which precedes it.

As a result of multiple objects properties that can be defined in ontology for two objects, we may find set of events defined within a single key frame having more than two objects. The relatedness will be checked in a way if those concepts can exist together for the sake of good sentence construction with related concepts.

Relatedness control is basically required because sometimes individual processing of key frames for event prediction would not work. Events which are extracted using a single key frame set of objects may not necessarily be dependent on an event which had happened in previous frames. This can work in the case of key frame based annotation where concepts rely only on objects of that individual frame but for a scene where relatedness of concepts is highly required, it is good to compute this relatedness. Relatedness control is used in two ways; the first one is to check whether set of concepts that are generated by the concept fusion process are really related to the concept in the previous event set, and, the second one is to check whether an event which is directly sent from the event router is related to previous events in the local event set.

Relatedness between concepts can simply be checked by taking the affinity between concepts in a local repository and the one fetched by the concept formulator as in *Algorithm 4-14*. If this event is not related to the one in the previous set it will be discarded in an assumption that it may be an outlier or an over fitting problem so that it will not be updated into the local event repository and then a recursive event prediction process will be done until all the low weighted objects are used.

Algorithm 4-14: Concept Relatedness Algorithm

```
Input:
  Event E
Variables: Flag Boolean
Output: Update status
Begin:
 For every event Ei in the local event repository
  If affinity(E,Ei) > threshold
   Flag==true
  End if
 Next
 If flag==true
  UpdateLocalEventRepository(E)
  Update==1
 Else
  If (Local event repository == Empty)
   Goto 12
  Else
    Instantiate EventPrediction (EventRouter.
    SendLowWeightedObjects())
    Return(Update==0)
  End if
 End if
End
```

Proper organization of these concepts will be done and will be kept finally as a phrase or a statement. For the whole scene, an approach for proper aggregation of phrases and sentences will be done. While aggregation of these key frame level annotations, a dependency analysis will be done in order to arrange the concept flow and control the proper arrangement of events. This is related to looking for an event which comes at frame t+1, the initial task is to look for the event

in frame t and predict the event which will come next in the presence of a given set of objects as explained in the next sub-section.

4.4.4 Shot Sentence construction

Having the related set of events with their object sets, the next step is to build a formal language dependent sentence for a given shot. This sentence construction will be defined based on the language model where the syntax and semantic rules of the language are defined. As in the previous works, event tagging or object tagging would not define the real events in the video scene and it can also be applied for a shot as we can find the objects and events which define set of actions in the video sequence. For example, given a shot with extracted objects of a "ball" and a "person" with a generated event while concept formulation which is "throwing", a formal statement which is "A person throwing a ball" can be generated which is a simple noun verb object combination. A statement which is going to be generated for video scene description may be a simple sentence in the case of shot level sentence construction and it can be a complex sentence especially in a scene level sentence construction as a result of multiple objects and multiple events.

With two or more objects in an object set, it is a challenging task to identify the cause and the result in a given defined event which is happening between these two. Even two or more objects may be reasons for a given event to happen on two or more other objects. As far as the researcher‟s knowledge, there are no works on sentence construction from set of terms. As a result we will consider following the grammatical rule of targets language to order terms accordingly. To optimize this, if all the terms are found in the audio event-object index, we can use the statement generated from the audio scene.

4.4.5 The Annotator

The result of the concept formulation process will be used into two different ways depending on the application areas of video annotation process. One is to give information for the viewer tagging the result on a playing video. In this case the statement will be used to provide a textual description of what is really happening in that video which can further be extended in a way adding related information to an event to be used as a reference to the one which is identified in

the query video. This additional information can be in the form of a link or free text which is related to the annotation.

The second one is a separated file which can be in the form of free text or RDF/XML which can be used for further video processing activities like context based video searching, classification, scene extraction, etc. In this case the result will be kept in a file putting the time frame and annotation statements of each scene. The former one can also be taken from this file and tagged to a playing video according to its time frame. Both are considered in this research where the annotation can be available in an extensible RDF format where it can also be placed on a playing video.

The one to be made available on the playing video can be tagged with user‟s preferred language so that the user can easily understand what is really happening in the video. This is more important on educational videos where students can easily understand events and objects that are recorded in a video for educating them.

For local language video indexing and retrieval, especially for local search engines, it is also better for the annotation to be in user‟s preferred language. This can be done in two ways. One is by asking users for their preferences or by taking the speaker‟s language extracted from the audio speech. A video with a language L is mostly intended to those who speak that language. For this end, while language identification which is done while audio processing, there may be a possibility of having multiple set of languages in that video so one of those languages should be selected in case the user preference is not given specially for web videos. This language filtering can be done by choosing a language L with a large air time than others in the identified language set.

Once the annotation is formally kept on the video, depending on the time stamp of each scene, then a text based video summarization can be followed as shown in Figure 4-7.

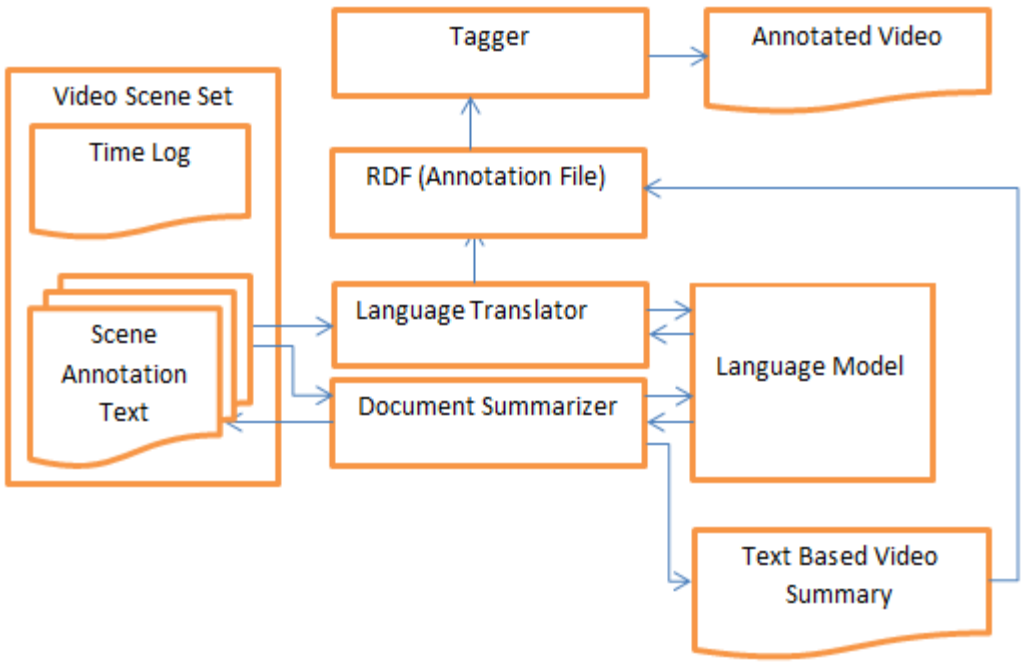

Figure 4-7: Annotation and Summarization Architecture

Annotation should be performed at scene level with object tagging whenever it is required. Shot level annotation can also be performed tagging the shot with shot level concept formulation result. The format of the annotation will be RDF statement containing the annotation free text and time frame of each annotation text. For a user it can be displayed as a subtitle to give some description about the scene. It may be very helpful to annotate the video with the preference of the user where a language module can be incorporated to translate the default annotation language to the one which the user favors.

4.4.5.1 Language Translator

This module is responsible to translate the annotation text into users preferred language so that users can easily understand annotations. Local language web content indexers can also easily understand and process the annotated video. In this research work we have implemented the system only using English language.

4.4.5.2 Tagger:

Annotation tagger is used to put the annotation text in to the video. We have created a subscript file so that it will be displayed while the user is watching the annotated video.

4.4.5.3 RDF (Annotation File):

This is the annotation file in the form of RDF so that it can be semantically processed by semantic web crawlers for efficient video utilization.

4.4.6 Video Summarization

Content based video categorization and other video processing tasks can be done by using a video summary; which will be generated from the final result of the annotation process. Such a video summarization is a process of summarizing the annotation result to give a precise description of a video which can then be indexed and used to search a video based on its content. A metadata or title of a video is most of the time generic and may not describe what is really happening inside. A good example here can be news videos which are titled with the date, host and little about the event.

The approach that we are following is an extractive summarization method which is by selecting important sentences from the scene statement set and concatenating them into shorter form. The importance of the sentences is decided by checking whether the class levels of the events in that sentence represent the others or checking whether an event is a hyponym of set of events in other statements. One thing which makes it different from the usual document text summarization is that all statements are set of events which are related to each other and dependent on one another. A term frequency based summarization may not be a good solution as the concepts are not usually identical.

The whole summarization result in combination with video titles and metadata can be used to make a better video indexing as to classify and make a fine result for video searching. Video summarization will be done by using concepts at each scene, merging them and finding out a generic representation of the entire video. One challenge in video summarization is if the scenes

are of different concepts like that of a news video, it would not be possible to look for text based summarization as summarizing different unrelated events is difficult.

In the usual video summarization concept of removing low weight frames which carry a minimal content, it would be possible for every video but in case of text based/annotation based summarization, it is highly required for every scene to be somehow related. In any case, a higher level text summarization will be done in combination with whatever the video has, especially for web videos.

4.5 Summary

Video annotation is a process of giving semantic description to a video content. This semantic description is generated from different image and audio processes. With the basic video annotation requirements we have proposed generic automatic video scene annotation and summarization architecture.

Video annotation starts with video pre-processing where there is an audio pre-processing and visual pre-processing. In the visual pre-processing module, the process starts with video segmentation which splits the video into frames and these frames are then clustered to give a set of shots. For each shot a representative key frame is selected and scene is identified based on those set of key frames. These key frames are checked if they have similar set of objects in order to classify them as members of a single scene or not. The scene time will be given to the audio pre-processor in order to split the audio in to fragments of audio scenes. The result of the pre-processor module is set of key frames, shot time frames, scene time frames and audio scene fragments.

With the results from the pre-processor, object identification follows in two ways. One is using the key frames generated at the visual pre-processor and the other is using the audio fragments. Key frames are manually segmented and similarity is computed with images in the dataset using low level image features in order to extract class labels of the segmented regions. Class labels of those images are taken to be object and event names for the segments image regions of the key frames. The audio fragments are recognized and tagged to identify verbs and nouns where nouns are objects and verbs are events.

These object and event class labels or names extracted from audio and visual object identification process will be normalized to avoid redundancies and set of objects are ranked to identify high priority objects that can be used to determine the event happening at a specific shot. Event router is used to send high priority objects as well as events to the event prediction and concept fusion modules where events are extracted either considering previous set of events or using ontology. The ontology maps set of objects with their relationships. Once events are extracted their relatedness will be checked by the dependency control module to remove unrelated events from the local event repository list where events of a scene are kept. This relatedness control is required because of the requirement that concepts in a scene should be related.

Depending on the languages grammatical structure, a shot level sentence is constructed using its set of objects and extracted event. These shot level sentences are then summarized to give a scene level sentence. These scene level texts can be processed with the document summarizer module in order to give a video level text summary. Finally, the scene level text will be translated to users preferred language and tagged to the video. An RDF format annotation file will be generated from the system which can be used for further video processing applications.

CHAPTER 5
IMPLEMENTATION AND EXPERIMENTAL RESULTS

5.1 Overview

A scene based video annotation and video summarization system is developed according to the functionalities defined in the previous Chapter. In this Chapter, we have presented different tools and development environments used to develop the prototype of the system. In addition, screen shots are presented to demonstrate the user interface and the outputs of the system. Finally, we have evaluated accuracy of the system using pre-annotated single shot videos from TRECVID and user reviews are collected to evaluate the overall systems accuracy.

5.2 Development Environment

Starting from the initial video pre-processing task up to the end of video annotation and tagging, we have used different tools and programming environments.

Java programming language with NetBeans development environment is used to work with every visual and audio processing activity as shown in *Appendix B*. Java provides different libraries which make image and audio processing activities easier and some libraries have also been used while working with these tasks. Java is an efficient programming tool in multimedia processing and it can be integrated with different programming environments. The integrity makes it best for an easy rule inference processing from ontologies and others. It can also be integrated with MatLab[18] to work with advanced image processing tasks. Basic libraries used in this work are

- *Xuggle-xuggler-5.4*[19]: allows Java programs to decode, encode, and experience (almost) any video format. It is used to split the video into frames.

[18] http://www.mathworks.com/products/matlab/
[19] http://www.xuggle.com/xuggler

- *Stanford-postagger*[20]: is used to assign parts of speech to words extracted from the audio data.
- *Jaudiotagger-2.0.2*[21]: is used to split the audio data into different fragments.
- *Ws4j-1.0.1*[22]: is used to normalize concepts performing different operations on WordNet.

Though it is limited, we have also used C# programming language for the purpose of sending HTTP Web requests and to read HTTP Web responses in the presence of a proxy server which is not supported by java. This is used while preparing image dataset from ImageNet which is specified in the dataset preparation section. Protégé is used to construct ontology for an object relationship network definition and Jena to infer concepts from the ontology. XML is used to store temporal data which is an event repository and temporal sentence index.

5.3 Dataset Preparation

Video annotation process requires a large training dataset for event and object classification tasks. For this reason, we have used ImageNet which contains hierarchically organized collection of images with their class labels. While doing so, first we have downloaded synsets and their WordNet IDs, which are available in the ImageNet page[23]. Figure 5-1 (a) shows sample Synset ID of extracted from ImageNet and Figure 5.1 b) shows the synset of each image having a specific synset ID. Having this file, we write a C# HttpRequest program in order to extract URLs of those synsets, as shown in Figure 5-2, sending an HttpWebRequest and reading the HttpWebResponce line by line. We have modified this program a bit and used it to read each image found in every URL extracted for a single synset. Figure 5-3 shows sample images that are fetched. Finally, we have kept these images with their synset class label names. We have extracted 151,700 Synset IDs where some of them are not indexed yet, with their WordNet labels, set of URLs and their images. For the indexed Synset IDs, there are around 500 images and we have downloaded images of 161 Synset IDs which are 67,137 images. Some of the Synsets are selected based on their WordNet label, those related to the test video data.

[20] http://nlp.stanford.edu/software/tagger.shtml
[21] http://www.jthink.net/jaudiotagger/
[22] https://code.google.com/p/ws4j/
[23] www.imageNet.com

n02118333 n02119789	n02118333
n02471300 n02478875	fox
n02471762 n02473983	n02119789
n02100399 n02100735	kit fox
n02374149 n02390258	Vulpes macrotis
n02109811 n02110185	n02471300
n02329401 n02338449	hominoid
n02430045 n02431976	n02478875
(a)	(b)

Figure 5-1: (a) List of Sample SynsetIDs from ImageNet and (b) Sample SynsetID along with Synonymous Words from WordNet

http://www.david.element.ukgateway.net/mammals7redfoxes4_files/image002.jpg
http://images.jupiterimages.com/common/detail/12/83/22578312.jpg
http://farm1.static.flickr.com/230/505000131_0e5801fb77.jpg
http://static.flickr.com/182/469209369_2cf02e5633.jpg
http://www.kodiak.org/images/fox2.jpg
http://static.flickr.com/37/107833993_a2d5f68c83.jpg
http://www.secondchancewildlife.com/photogallery/2005%20Photo%20Gallery/Images/5-15-05BFoxF.JPG
http://www.iws.org/images/fox1.jpg
http://farm2.static.flickr.com/1334/862766435_dc36a6867a.jpg
http://www.newsday.com/media/photo/2003-12/10444588.jpg
http://farm1.static.flickr.com/141/317311968_9f5df24c5c.jpg
http://farm1.static.flickr.com/193/505853827_4753eb8597.jpg

Figure 5-2: Sample URLs Fetched for the SynsetID n02118333

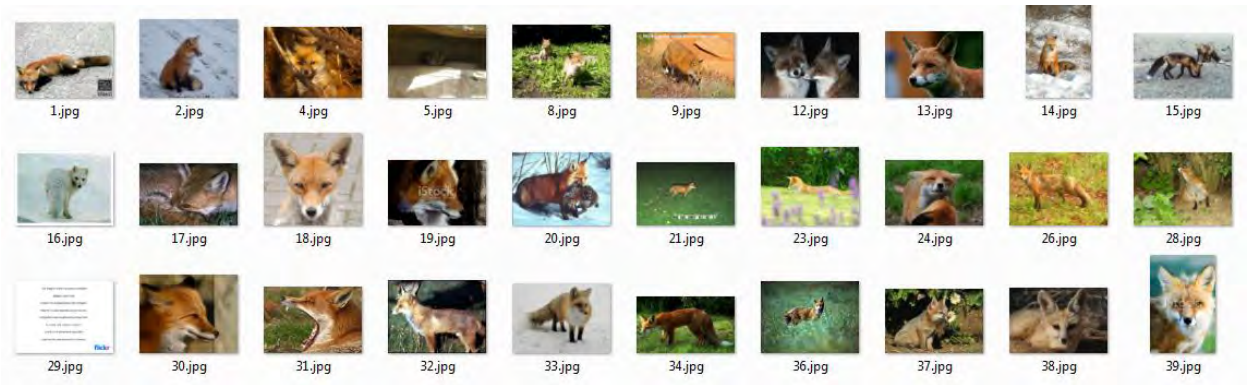

Figure 5-3: Sample Images Fetched from the Set of URLs for the SynsetID n02118333

5.4 Ontology Construction

As part of the prototype, we have constructed sport domain ontology to show how concept formulation is done through ontology based concept fusion. Referring to Figure 5-5, nodes of the ontology show set of objects in sport domain event that can be related using object properties. The object properties that we define focus on a basketball sport as we are using a sport video of this domain to demonstrate the prototype. As shown in Figure 5-4, the relationship between objects (Event) is defined using object properties, where the domain and range of this properties show the objects to be related. This relationship allows defining subjects and verbs which can be used while sentence construction in a way domain value of the object property is a subject and range value is an object. In our case we have defined object properties that can show the relationship between ball and a player as shown in Figure 5-4.

```
<?xml version="1.0"?>
……
<Ontology xmlns="http://www.w3.org/2002/07/owl#"
    ……
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#isDribbling"/>
        <ObjectProperty IRI="#isPlaying"/>
    </SubObjectPropertyOf>
    <SubObjectPropertyOf>
        <ObjectProperty IRI="#isPassing"/>
        <ObjectProperty IRI="#isPlaying"/>
    </SubObjectPropertyOf>
    <ObjectPropertyDomain>
```

```
        <ObjectProperty IRI="#isDribbling"/>
        <Class IRI="#Player"/>
        </ObjectPropertyDomain>
        <ObjectPropertyDomain>
        <ObjectProperty IRI="#isPassing"/>
        <Class IRI="#Player"/>
    </ObjectPropertyDomain>
    ……
    <ObjectPropertyRange>
    <ObjectProperty IRI="#isDribbling"/>
        <Class IRI="#Ball"/>
        </ObjectPropertyRange>
        <ObjectPropertyRange>
        <ObjectProperty IRI="#isDribbling"/>
        <Class IRI="#PlayingField"/>
    </ObjectPropertyRange>
    ……

</Ontology>
```

Figure 5-4: Sample Object Properties from Sport Domain Ontology

Figure 5.4 shows sample ontology developed for this prototype, it shows object properties and the corresponding graphical representation is shown in Figure 5.5 in portage format.

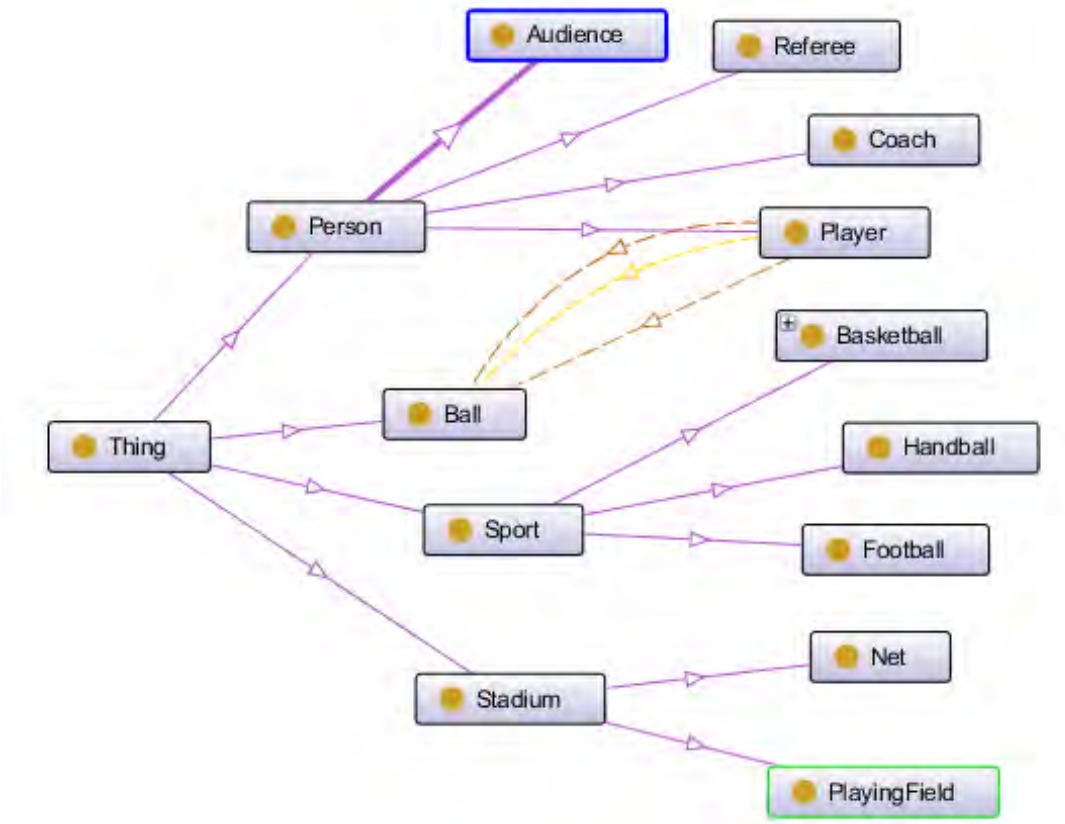

Figure 5-5: Sample Sport Domain Ontology

5.5 System Prototype

A prototype is developed to show the validity of the proposed video scene annotation and summarization framework. It is capable to perform the annotation process automatically and the number of interaction with users is very limited. Thus, the system has limited user interfaces associated with uploading video, setting thresholds when required, and shows results while computation.

System User Interface

User interface allows a user to upload the video to be annotated as shown in *Figure 5-6*. Once the video is uploaded, the pre-processing module starts to extract audio component of the video and then goes to a separate audio and visual data processing. The result of the visual pre-processor module is a set of key frames and an XML file holding information about scene cuts and shot cuts of the video. Initially the video is split into frames based on the frame rate of the

video, kept in some directory and then read to find a shot boundary. Those in a similar shot are clustered in one directory and from that directory a key frame is selected. A scene boundary is then checked by processing key frames of each shot. To demonstrate the system, we have used a YouTube video of 2 minutes and 46 seconds length with 9 different shots of similar scene.

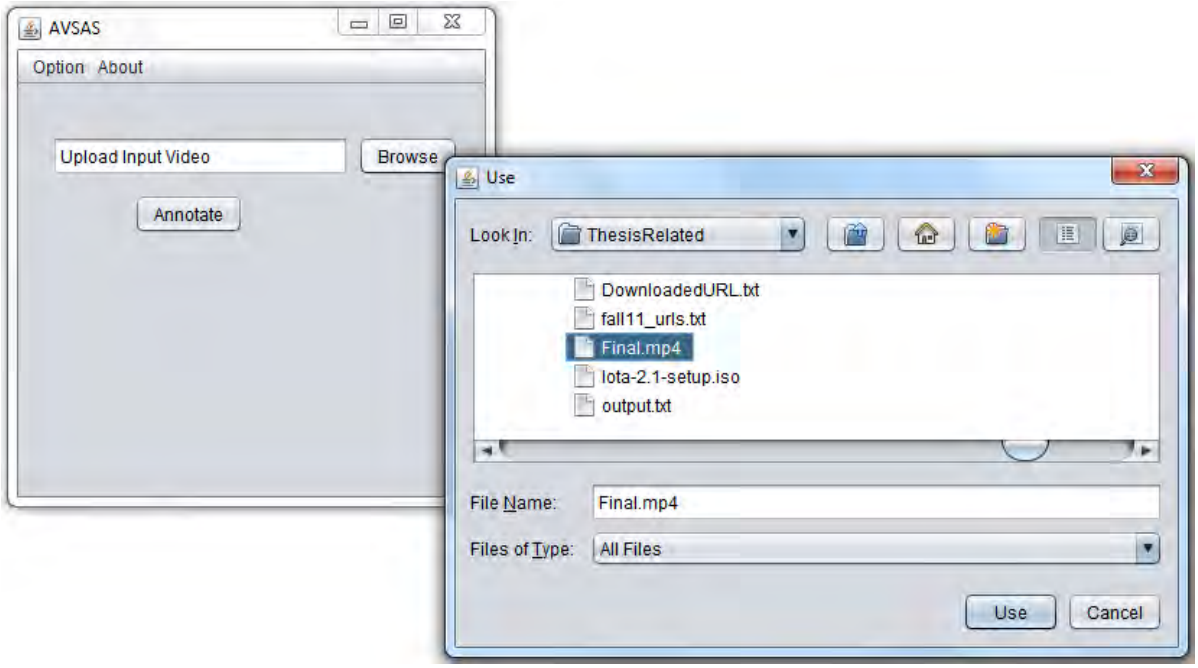

Figure 5-6: Automatic Video Scene Annotation and Summarization System User Interface

For the example video, 4,986 frames are extracted, as shown in Figure 5-7 depending on the frame rate of the video extracted automatically from its property and grouped into 10 shots then the system identifies key frames and the final result of the visual pre-processor is an XML file shown in *Figure 5-8*, indicating time frames of each scene and shot with the frame ID that is going to be a representative key frame for each shot.

Figure 5-7: Sample Set of Frames Extracted from the Query Video

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Video frameRate="25" Length="00:02:46" Format="MP4" Location="D:\ThesisRelated\Final.MP4">
    <Scene ScID="SC1" StartTime="00:00" EndTime="02:46">
        <Shot ShID="Shot0"   StartTime="00:00:00" EndTime="00:00:08">
            <KeyFrame Fid="Frame1422255034869" format="png"/>
        </Shot>
        <Shot ShID="Shot1"   StartTime="00:00:08" EndTime="00:00:17">
            <KeyFrame Fid="Frame1422255100550" format="png"/>
        </Shot>
        <Shot ShID="Shot2"   StartTime="00:00:17" EndTime="00:00:28">
            <KeyFrame Fid="Frame1422255167714" format="png"/>
        </Shot>
        <Shot ShID="Shot3"   StartTime="00:00:28" EndTime="00:00:35">
        <Shot ShID="Shot4"   StartTime="00:00:35" EndTime="00:01:10">
        <Shot ShID="Shot5"   StartTime="00:01:10" EndTime="00:01:14">
        <Shot ShID="Shot6"   StartTime="00:01:14" EndTime="00:02:04">
        <Shot ShID="Shot7"   StartTime="00:02:04" EndTime="00:02:20">
        <Shot ShID="Shot8"   StartTime="00:00:20" EndTime="00:00:24">
            <KeyFrame Fid="Frame1422255926842" format="png"/>
        </Shot>
        <Shot ShID="Shot9"   StartTime="00:02:24" EndTime="00:02:46">
            <KeyFrame Fid="Frame1422255975392" format="png"/>
            <KeyFrame Fid="Frame1422256056257" format="png"/>
        </Shot>
```

Figure 5-8: Clusters of Shots in Scene-1

Audio Pre-processing

Given the query video, the audio data is encoded using Jaudiotagger java library for audio encoding. For the implementation purpose we split the audio into pieces of no longer than 20 seconds to take into Google speech recognition API in flac format. We then merge phrases and took the recognized text to the POS tagger defined using Java "Stanford-POS-tagger" library to have a tagged text as shown in Figure 5-9.

```
He_PRP is_VBZ gon_VBG na_TO show_VB us_PRP the_DT basic_JJ fundamentals_NNS of_IN
how_WRB to_TO become_VB a_DT better_JJR shooter_NN First_NNP you_PRP wan_VBP na_TO
get_VB the_DT ball_NN ,_, you_PRP wan_VBP na_TO get_VB good_JJ hand_NN placement_NN ,_,
you_PRP wan_VBP na_TO form_VB a_DT tea_NN which_WDT your_PRP$ hands_NNS right_RB
here_RB which_WDT your_PRP$ thumbs_NNS and_CC this_DT is_VBZ your_PRP$ hand_NN
and_CC this_DT is_VBZ your_PRP$ shooting_JJ hand_NN ._. So_RB ,_, you_PRP wan_VBP
na_TO have_VB your_PRP$ hands_NNS here_RB you_PRP got_VBD good_JJ leg_NN placement_NN
,_, good_JJ knee_NN bend_VB right_RB here_RB his_PRP$ right_JJ foot_NN is_VBZ slightly_RB
in_IN front_NN of_IN his_PRP$ left_NN for_IN balance_NN Okay_UH I_PRP should_MD be_VB
able_JJ to_TO get_VB that_DT ball_NN right_NN over_IN the_DT front_NN ._. keep_VB
your_PRP$ elbow_NN in_IN the_DT right_JJ elbow_NN here_RB in_IN this_DT case_NN
he_PRP is_VBZ right_JJ handed_NN and_CC you_PRP wan_VBP na_TO get_VB a_DT good_JJ
follow_VB through_IN on_IN the_DT shot_NN ._. So_IN he_PRP is_VBZ gon_VBG na_TO show_VB
us_PRP a_DT couple_NN of_IN repetitions_NNS on_IN great_JJ technique_NN repetition_
NN and_CC muscle_NN memory_NN ._. Here_RB we_PRP go_VBP ,_, the_DT follow_VB through_IN
follow_VB through_IN every_DT time_NN Here_RB we_PRP go_VBP there_RB he_PRP is_VBZ
Notice_NNP the_DT follow_VB through_IN ,_, notice_VB the_DT balance_NN ,_, every_DT
time_NN shoots_VBZ the_DT same_JJ every_DT time_NN and_CC he_PRP gets_VBZ shots_NNS
up_IN there_EX every_DT shot_NN you_PRP wan_VBP na_TO it_PRP to_TO look_VB the_DT
same_JJ you_PRP want_VBP every_DT shot_NN looks_VBZ the_DT same_JJ Follow_VB through_IN
good_JJ muscle_NN memory_NN ,_, follow_VB through_IN good_JJ muscle_NN memory_NN Move_VB
back_RB a_DT little_JJ bit_NN when_WRB he_PRP moves_VBZ back_RB the_DT shot_NN still_RB
looks_VBZ the_DT same_JJ it_PRP is_VBZ a_DT natural_JJ flow_NN natural_JJ shot_NN
everyone_NN goes_VBZ in_IN when_WRB you_PRP use_VBP the_DT proper_JJ technique_NN
action_NN Not_RB that_IN far_RB up_IN you_PRP are_VBP looking_VBG at_IN the_DT ceiling_NN
you_PRP need_VBP to_TO be_VB able_JJ to_TO see_VB the_DT court_NN Show_NN how_WRB to_TO
do_VB it_PRP see_VB that_DT fingertip_NN control_NN able_JJ to_TO see_VB the_DT whole_JJ
court_NN Hold_VB the_DT ball_NN hold_VBP the_DT ball_NN hold_VBP the_DT ball_NN right_NN
Right_RB You_PRP wan_VBP na_TO keep_VB that_DT dribble_NN low_JJ Low_JJ Low_JJ get_VB
lower_JJR Lower_JJR Get_VB low_JJ Get_VB low_JJ Get_VB low_JJ get_VB low_JJ Not_RB
that_IN low_JJ then_RB you_PRP gon_VBG na_TO come_VB up_RP see_VB the_DT court_NN
fingertip_NN control_NN low_JJ dribble_NN so_IN you_PRP be_VB able_JJ to_TO control_VB
the_DT ball_NN I_PRP got_VBD it_PRP I_PRP got_VBD it_PRP I_PRP got_VBD it_PRP Now_RB
if_IN you_PRP wan_VBP na_TO move_VB faster_RBR and_CC go_VB up_RP the_DT court_NN
You_PRP can_MD still_RB see_VB it_PRP
```

Figure 5-9: Encoded Audio Tagged with Stanford Part of Speech Tagger

Verbs and nouns are identified from the tagged text as shown in Figure 5-9 for further identification of objects and events. A statement node in Figure 5-10 shows a single statement with its text value followed by each and every verb and noun text which is extracted from the tagged text. A node verb contains those words tagged as verb with their type where type indicates the type of the verb or the type of the noun. Nouns are then taken to be objects and verbs are taken to be events ones they are normalized.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Event_Object_Index>
<Statement ID="1" Text=" He is gon na show us the basic fundamentals of h
<Statement ID="2" Text=" So , you wan na have your hands here you got goo
<Statement ID="3" Text=" keep your elbow in the right elbow here in this
    <Verb ID="0" Text="keep" Type=""/>
    <Verb ID="1" Text="is" Type="Z"/>
    <Verb ID="2" Text="wan" Type="P"/>
    <Verb ID="3" Text="get" Type=""/>
    <Verb ID="4" Text="follow" Type=""/>
    <Noun ID="0" Text="elbow" Type=""/>
    <Noun ID="1" Text="elbow" Type=""/>
    <Noun ID="2" Text="case" Type=""/>
    <Noun ID="3" Text="handed" Type=""/>
    <Noun ID="4" Text="shot" Type=""/>
</Statement>
<Statement ID="4" Text=" So he is gon na show us a couple of repetitions
    <Verb ID="0" Text="is" Type="Z"/>
    <Verb ID="1" Text="gon" Type="G"/>
    <Verb ID="2" Text="show" Type=""/>
    <Noun ID="0" Text="couple" Type=""/>
    <Noun ID="1" Text="repetitions" Type="S"/>
    <Noun ID="2" Text="technique" Type=""/>
    <Noun ID="3" Text="repetition" Type=""/>
    <Noun ID="4" Text="muscle" Type=""/>
    <Noun ID="5" Text="memory" Type=""/>
</Statement>
<Statement ID="5" Text=" Here we go , the follow through follow through e
</Event_Object_Index>
```

Figure 5-10: Result of Audio Pre-processor for a Video Scene-1

Objects are segmented from the visual key frame and concept classifiers identify labels of each object as shown in Figure 5-11. Here, we have used a manual segmentation of objects as existing approaches that we adopt have inadequate segmentation results.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<VisualObjectSet>
    <Keyframe ID="Frame1422255034869">
        <Object Oid="Ob1" Label="Person"/>
        <Object Oid="Ob2" Label="Person"/>
        <Object Oid="Ob3" Label="Person"/>
    </Keyframe>
    <Keyframe ID="Frame1422255100550">
        <Object Oid="Ob1" Label="Ball"/>
        <Object Oid="Ob2" Label="Person"/>
        <Object Oid="Ob3" Label="Person"/>
    </Keyframe>
```

Figure 5-11: Result of the Learning Phase with Manual Image Segmentation

Having the above two results, the next step is to normalize this concepts and create a concept hierarchy. This concept normalization is done to merge similar concepts which are specified in different naming. For this, object and event labels from the audio set will be checked for their similarity with object and event labels found in the video set; if they are similar, they are merged as one concept. For example, the object "couple" and "person" from the audio object set have a similarity value *sim('couple', 'person') = 0.274,* which in our case is taken to be not similar as we took a similarity threshold of 0.4. There is no standard similarity threshold as per the knowledge of the researcher but we have checked relatedness between sample terms and we have found 0.4 as an average value. Object ranking is done to check cross object relatedness so that unrelated objects are removed from the final object and event set. Here concept occurrence is checked using event-object matrix and the hierarchy shows most related objects according to their affinity value.

Ontology inference is used to identify the relationship between objects and events as shown in *Table 5-1*. For example, given object set of shot one which is "*person*", an object property having a domain "*person*" is an event "*IsStanding*" which will then be used to annotate an event in shot one.

Table 5-1: Set of Events Inferred from Ontology

Objects	*Event*	*Shot*
Person, Person, Person	*IsStanding*	*Shot 0,1,2*
Person,Ball,Net	*IsShooting*	*Shot 3,4,5,6*
Person,Ball,Field	*IsDribbling*	*Shot 7,8,9*

Here the system sends objects rather than events to the concept formulation module where events are extracted from ontology. This is because the objects fetched while audio processing and those in the visual set are not similar. When the system gets multiple results from ontology relationship definition, it looks back to the audio event set which is extracted from set of verbs in the tagged text and selects the event which is found both in the audio and visual event sets according to the time frame. For instance, if the concept fusion results concepts like "IsShooting" and "IsThrowing" where there is only a verb "shot" in the tagged encoded text, then the selection goes to "IsShooting". The challenge here is the timing information of audio and videos may not be similar.

Higher level object property of these events from ontology is assigned for the description of the scene. For example, the higher level object property for the events defined for the shots are "*IsStanding*" and "*IsPlaying*" so finally we have a playing event assigned to the scene.

Shot level annotation sentences are constructed by merging objects and verbs according to the grammatical definition of the English language. In the above video, we have got the following three sentences from list of objects and events extracted shown in *Table 5-1*. Figure 5-12 shows sample annotation text and corresponding automatic annotation results associated to scenes and shots of the input video.

```
1
00:00:01,000 --> 00:00:28,000
Person is standing
Person is playing

2
00:00:29,000 --> 00:02:04,000
Person is shooting a ball
Person is playing

3
00:02:04,000 --> 00:02:46,000
Person is dribbling a ball
Person is playing
```

a)

b)

c)

d)

Figure 5-12: (a) Sample Annotation Generated for the Scene of the Video Input; b),c) and d) Scene Annotated with Generated Annotation Text

The annotation result can be represented in XML format for further content based video processing as shown in Figure 5-13. The XML file defines annotation having scene along with its time boundary and description, its associated set of fragments each having time boundary and description. The description is associated to the annotation of the scene or its fragments.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Annotation>
    <Video>
        <Scene ScID="SC1" StartTime="00:00:00" EndTime="00:02:46" Description = "Person is playing">
            <Fragment StartTime="00:00:00" EndTime="00:00:28" Description = "Person is standing"/>
            <Fragment StartTime="00:00:28" EndTime="00:02:04" Description = "Person is shooting a ball"/>
            <Fragment StartTime="00:02:04" EndTime="00:02:46" Description = "Person is dribbling a ball"/>
        </Scene>
    </video>
</Annotation>
```

Figure 5-13: Annotation Result in the form of XML

The result of the system is highly dependent on the following issues:

- Segmentation result.
- Clarity and quality of the audio as well as the visual data.
- Ontology relationship result: this is due to the problem of having multiple results from ontology.
- Thresholding: similar threshold values may not be used while pre-processing, object identification and learning phase.
- Sentence construction techniques.

5.6 Evaluation

Once the prototype is developed, it is evaluated in two ways. One is evaluating the accuracy of the implemented algorithms and evaluating the quality of the entire system. We have evaluated the performance of the shot detection processes using TRECVID evaluation standard video dataset and the accuracy of the system with users review.

Event detection accuracy is evaluated according to TRECVID standard evaluation video data sets where 1600 videos from 11 different sport categories with an average of 10 second length single shot videos are collected. We have selected one good quality shot from each category to test the system as shown in Figure 5-14. The resulting event from the proposed system is compared to the one extracted as annotation of those videos. One thing which is missing here is we haven't found standard videos having a speech data in order to evaluate the full system incorporating the audio event prediction module. The comparison relies only on the visual object and event processor component.

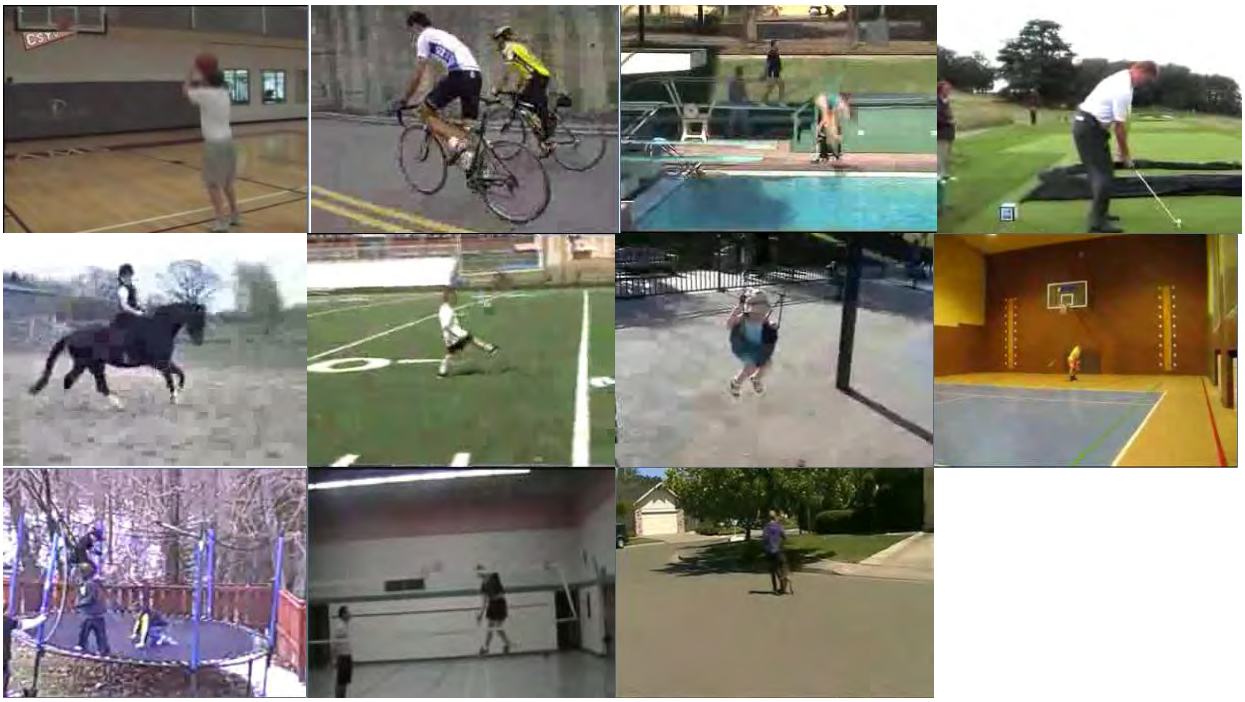

Figure 5-14: Screen Shots of Selected Test Video Shots

The annotation that the proposed system generates is compared with the annotation of the above video shots which is given in a single XML file as shown in *Figure 5-15*. This video dataset is annotated specifying the object of interest with the action that the object is performing. For example, video shot one is annotated with description object *"person"* with an attribute value of "*basketball_shooting*".

```
<?xml version="1.0" encoding="UTF-8"?>
<viper xmlns="http://lamp.cfar.umd.edu/viper#" xmlns:data="http://lamp.cfar.umd.edu/viperdata#">
  <config>
    <descriptor name="Information" type="FILE">
      <attribute dynamic="false" name="SOURCETYPE" type="http://lamp.cfar.umd.edu/viperdata#lvalue">
        <data:lvalue-possibles>
          <data:lvalue-enum value="SEQUENCE"/>
          <data:lvalue-enum value="FRAMES"/>
        </data:lvalue-possibles>
      </attribute>
      <attribute dynamic="false" name="NUMFRAMES" type="http://lamp.cfar.umd.edu/viperdata#dvalue"/>
      <attribute dynamic="false" name="FRAMERATE" type="http://lamp.cfar.umd.edu/viperdata#fvalue"/>
      <attribute dynamic="false" name="H-FRAME-SIZE" type="http://lamp.cfar.umd.edu/viperdata#dvalue"/>
```

```
    <attribute dynamic="false" name="V-FRAME-SIZE" type="http://lamp.cfar.umd.edu/viperdata#dvalue"/>
    <attribute dynamic="false" name="START TIME" type="http://lamp.cfar.umd.edu/viperdata#fvalue"/>
  </descriptor>
  <descriptor name="PERSON" type="OBJECT">
    <attribute dynamic="true" name="Location" type="http://lamp.cfar.umd.edu/viperdata#bbox"/>
    <attribute dynamic="true" name="basketball_shooting" type="http://lamp.cfar.umd.edu/viperdata#bvalue">
      <default>
        <data:bvalue value="false"/>
      </default>
    </attribute>
    <attribute dynamic="true" name="biking" type="http://lamp.cfar.umd.edu/viperdata#bvalue">
      <default>
        <data:bvalue value="false"/>
      </default>
    </attribute>
```

Figure 5-15: Annotation Sample of Test Videos

The result of the system is compared with the annotation provided on the video dataset for their similarity using word similarity measure. This is done to avoid false negative conclusions as a result of having two words representing the same event one from the proposed system and the other in the annotation text of the test video dataset.

Among those 11 video shots, the proposed system identifies events and generates similar annotation result for the nine video shots and two of the video shots are falsely annotated. The accuracy is calculated based on the number of truly annotated shots over the whole video shots in the evaluation dataset. 81% shot level event identification accuracy is gain with our approach given these sport domain single event evaluation video dataset.

While evaluating the entire system, we gave the result of the annotator to 25 people that we select randomly from different profession. 15 of them are postgraduate students at Addis Ababa University who have basic knowledge on video related concepts, and the rest are individuals from different profession who watch movies. The selection is based on the assumption that those with video related knowledge can see and evaluate technical issues such as shot and scene timing while others may evaluate the applicability, accuracy and importance of the system.

The evaluation matrix is prepared and given to them in the form of questionnaire as shown in an *Appendix A*, in which they put the weight of each criterion according to their view. Weights indicate how good the annotator is in an incremental way where 1 indicates lower value and 4 is the higher weight. User‟s evaluation is recorded as shown in *Table 5-2* and analyzed in detail. Rows of *Table 5-2* show evaluation values of each individual and columns show evaluation criteria listed on the questionnaire where:

- Cr1: the annotation is given at the right time frame of scenes and shots
- Cr2: concepts in the video and annotation tags are similar
- Cr3: objects stated in the annotation are found in the video
- Cr4: all the objects in the video are stated in the annotation
- Cr5: all the events in the video are stated in the annotation file
- Cr6: events stated in the annotation file are similar to those in the video
- Cr7: unrelated events that are found in the speech data are excluded in the annotation file
- Cr8: the annotation sentence is clear, understandable and related to the actual content of the video
- C9: scene level annotation statement gives a description of the scene

Table 5-2: Records of User Evaluations

	Cr1	**Cr2**	**Cr3**	**Cr4**	**Cr5**	**Cr6**	**Cr7**	**Cr8**	**Cr9**
1	3	4	4	3	2	4	4	4	3
2	4	4	4	3	3	4	4	4	3
3	3	3	4	3	2	4	3	4	2
4	4	4	4	3	3	4	4	4	4
5	3	4	4	3	2	4	3	4	3
6	4	3	4	3	3	4	4	4	3
7	3	4	4	4	3	4	3	4	3
8	3	4	4	3	2	3	3	4	4
9	3	4	4	3	3	4	4	4	2
10	4	3	4	3	3	4	3	4	4
11	3	4	4	3	3	4	3	4	3
12	4	3	4	4	3	3	4	4	3
13	4	4	3	3	2	4	4	4	3
14	4	4	4	3	2	4	4	4	3
15	4	3	4	3	3	4	4	4	3

16	4	4	4	3	3	4	4	4	3
17	4	4	4	4	3	3	4	4	3
18	3	4	4	3	3	4	4	4	4
19	3	4	4	3	2	4	4	4	2
20	4	4	4	3	2	4	4	4	4
21	4	4	4	3	3	4	3	4	3
22	3	4	4	3	3	4	3	4	3
23	4	4	3	3	3	4	4	4	3
24	4	4	4	3	2	4	4	4	3
25	3	4	4	3	2	3	4	4	3

With Cr1, the evaluation result shows that concepts that are in the video are identified and properly tagged according to their time frame. This good timing of annotations indicates the quality of the pre-processing tasks where shots, key frames and scenes are properly identified.

As a result of proper object and event identification process, concepts in the annotation file are seen to be similar with those found in the video. This result is shown in Cr2 in a way it shows that number of false positive events identified is very limited. This is again shown in Cr3, where object identification process has been rated to be good as all the objects in the annotation are found in the video.

Cr4 shows lower results as all the objects are not found in the annotation file. This is due to different reasons; one is interesting objects are selected manually while implementing the prototype as we cannot get good segmentation algorithm. Second, object filtering is done while normalization process to give a lower rank to unrelated objects. The other is due to the knowledge of the evaluators about objects of interests lowers the result.

We have a number of events in a given video, where some of them are from object interactions and others from motion of an object. For example "*A person getting lower*" is an event of a single object "*Person*" which can be identified with motion processing. The system provides identification of events as a result of object interactions with a limitation of motion processing. This results in missing those events which are found in the video scene as shown from the evaluation result of Cr5.

Events listed in the annotation result are seen to be found in the video though they are minimal as indicated in the above paragraph and have achieved a good event filtering as a result of the normalizer in a way events from the speech processor which are not relevant are excluded from the annotation.

Finally, the result of Cr8 and Cr9 shows that statements of the annotation are rated to be clear and understandable where scene level annotations need some work to make it descriptive than being generic. This is due to sentence limitation of the construction technique while constructing a sentence in the presence of multiple set events and objects.

CHAPTER 6
CONCLUSION AND FUTURE WORKS

6.1 Conclusion

Though a lot has been done in the area of video annotation, almost all works focus on key frame which is very close in technique to image annotation. Results from such annotations are inadequate in describing the contents of a video as they are limited to labeling objects or identifying a single event with a very limited concept correlation and relatedness processing. As a video is a collection of related concepts represented in a sequential set of image frames, it needs a great deal of concept dependency processing to give a semantic description on the video content.

A novel scene level video annotation and summarization framework is proposed to give annotations to scenes in a video and provide high level description of them. Annotation starts with identification of objects in a video scene which is done in two ways; the first is processing the encoded audio speech and the other is from visual data which is processing each key frame. In order to obtain these objects, the video is first pre-processed to extract the audio speech from it and split it into shots and scenes. Key frames from these shots are selected which will then be inputs for the object identification process. The audio speech is split into parts according to the visual scene time frame and encoded to have the text equivalent which is tagged with proper word class to identify objects and events.

SVM classifiers are used to identify objects from features extracted from segmented key frames and once the class label of these objects are identified from the training set, normalization of objects with those from the audio set is performed to remove unnecessary objects from the audio set and give ranks to objects based on their occurrence and relatedness. This helps to use events generated in an audio set if the set of objects in both the audio and visual set are similar which in turn, will minimize the cost of event identification process.

Concept formulation is accomplished in two ways; one is using the event prediction process from event-object-net which is a network of events, objects and their occurrence matrices, and the other is inferring from object relationship ontology. Event-object index is constructed to keep set of events finally extracted from the concept fusion phase. These events are further analyzed to check their relatedness to construct a statement combining these concepts. Finally, sentences are constructed for each shot and a scene level sentence is constructed.

We have used different tools and development environments while developing the prototype. Evaluation of the system is carried out using different evaluation criteria. The evaluation result shows that the proposed framework provides a good shot level event prediction and well description of a scene.

Automatic video annotation system requires varying threshold for different videos based on their quality. Segmentation is a big issue in object identification process where a good segmentation algorithm results better shapes that can be easily mapped to an object. In addition to segmentation, a good object of interest filtering technique is also required. Finally, if a complex sentence can be constructed using set of nouns and verbs, a better, well formulated description of a video scene can be achieved.

6.2 Contributions

The contributions of this research work are:

- Scene level video annotation and summarization architecture
- Shot boundary detection algorithm
- Object matching supported scene detection algorithm
- Audio supported event and object detection
- Structure of Web based ObjectEventNet for event prediction
- Concept normalization and ranking
- Prototype of shot and scene level video annotation

6.3 Future Works

This research work explores different areas that can be further improved as well as some components that should be implemented and integrated for better functioning of the system. Some of the future works include:

- Implementing image segmentation for fine object identification.
- Implementing and deploying the ObjectEventNet which can be an input for other research works also.
- Designing and implementing complex sentence construction techniques given set of object and event concepts.
- Implementing the text based summarizer for the entire video.
- Designing automatic ontology building from EventObjectNet
- Implementing the language translator
- Developing audio noise removal module
- Implementing the whole system with full functionalities

References

[1] Amjad Altadmri and Amr Ahmed, "A framework for Automatic Semantic Video Annotation Utilizing Similarity and Commonsense Knowledge Bases", School of Computer Science, University of Lincoln, Lincoln, UK, March 2013.

[2] V. Radha and K. Tamil Selvi, "A Framework for Group Based Image Retrieval and Video Annotation", Journal of Global Research in Computer Science, Vol. 4, No. 12, India, December 2013.

[3] Lamberto Ballan, Marco Bertini, Alberto Del Bimbo, and Giuseppe Serra, "Video Annotation and Retrieval Using Ontologies and Rule Learning", University of Florence, Italy, 2010.

[4] Yu-Gang Jiang, Qi Dai, Jun Wang, Chong-Wah, Xiangyang Xue, and Shih-Fu Chang, "Fast Semantic Diffusion for Large-Scale Context-Based Image and Video Annotation", IEEE Transactions on Image Processing, Vol. 21, No. 6, June 2012.

[5] Emily Moxley, Tao Mei, Xian-Sheng Hua, Wei-Ying Ma, and B. S. Manjunath, "Automatic Video Annotation Through Search and Mining", Vision Research Lab, University of California, Santa Barbara, 2008.

[6] Emily Moxley, Tao Mei, and B. S. Manjunath, "Video Annotation Through Search and Graph Reinforcement Mining", IEEE Transactions on Multimedia, Vol. 12, No. 3, April 2010.

[7] Martinez Angel R.,"Part-of-Speech Tagging",Wiley Interdisciplinary Reviews: Computational Statistics, John Wiley & Sons Inc., 2012.

[8] Bo Geng, Linjun Yang, Chao Xu, and Xian-Sheng Hua, "Collaborative Learning for Image and Video Annotation", Peking University, Beijing. China, pp. 443-450, 2008.

[9] Shih-Wei Suna, Yu-Chiang Frank Wanga, Yao-Ling Hunga, Chia-Ling Chang, Kuan-Chieh Chen, Shih-Sian Cheng, Hsin-Min Wanga, and Hong-Yuan Mark Liao, "Automatic Annotation of Web Videos", Institute of Information Science and Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan, 2011.

[10] Guo-Jun Qi, Xian-Sheng Hua, and Yong Rui, "Correlative Multi-Label Video Annotation", 2007.

[11] Yu-Gang Jiang, Jun Wang, Shih-Fu Chang, and Chong-Wah, "Domain Adaptive Semantic Diffusion for Large Scale Context-Based Video Annotation", Proceedings of IEEE International Conference on Computer Vision (ICCV), Kyoto, Japan, Sept. 29-Oct. 2, 2009.

[12] Fangshi Wan, De Xu, Wei Lu, and Weixin Wu, "Automatic Video Annotation and Retrieval Based on Bayesian Inference", Jiaotong University, Beijing, China, 2007.

[13] Vanessa El-Khoury, Martin Jergler, Getnet Abebe Bayou, David Coquil and Harald Kosch, "Fine-Granularity Semantic Video Annotation: An Approach Based on Automatic Shot Level Concept Detection and Object Recognition", Chair of Distributed Information Systems, University of Passau, Passau, Germany, Dec 2012.

[14] Wei Ren and Sameer Singh, "An Automated Video Annotation System", ICAPR, LNCS 3687, pp. 693 – 700, 2005.

[15] Arasanathan Anjulan and Nishan Canagarajah, "A Novel Framework for Robust Annotation and Retrieval in Video Sequences", University of Bristol, Bristol, UK, 2006.

[16] Daisuke Yamamoto, Tomoki Masuda, Shigeki Ohira, and Katashi Nagao, "Collaborative Video Scene Annotation Based on Tag Cloud", Nagoya University, Japan, 2008.

[17] Bin Cui1, Bei Pan, Heng Tao Shen, Ying Wang, and Ce Zhang, "Video Annotation System Based on Categorizing and Keyword Labelling", Peking University, DASFAA 2009, LNCS 5463, pp.764–767, 2009.

[18] Vincent S. Tseng, Ja-Hwung Su, Jhih-Hong Huang, and Chih-Jen Chen, "Semantic Video Annotation by Mining Association Patterns from Visual and Speech Features", National Cheng Kung University, Tainan, Taiwan, 2008.

[19] Khushboo Khurana and M. B. Chandak, "Video Annotation Methodology Based on Ontology for Transportation Domain", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, Issue 6, June 2013.

[20] ürgen Assfalg, Marco Bertini, Carlo Colombo, and Alberto Del Bimbo, "Semantic Annotation of Sports Videos", IEEE International Workshop, 2012.

[21] Olivier Duchenne, Ivan Laptev, Josef Sivic, Francis Bach, and Jean Ponce, "Automatic Annotation of Human Actions in Video", INRIA Ecole Normale Sup´erieure, Paris, France, 2009.

[22] Marco Grassi, Christian Morbidoni and, Michele Nucci, “A Collaborative Video Annotation System Based on Semantic Web Technologies”, University of Marche, Via Brecce Bianche, Ancona, Italy, July 2012.

[23] Wan-Lei Zhao, Xiao Wu, and Chong-Wah Ngo, “On the Annotation of Web Videos by Efficient Near-duplicate Search”, IEEE, 2010.

[24] Guangda Li, Meng Wang, Zheng Lu, Richang Hong, and Tat-Seng Chua, “In-Video Product Annotation With Web Information Mining”, ACM Trans. Multimedia Comput. Commun. Appl. 8, Nov 2012.

[25] Lamberto Ballan, Marco Bertini, Alberto Del Bimbo, and Giuseppe Serra, “Semantic Annotation of Soccer Videos by Visual Instance Clustering and Spatial/Temporal Reasoning in Ontologies”, University of Florence, Florence, Italy, Aug 2009.

[26] Vishal Guptaa and Gurpreet Singh Lehal, “A Survey of Text Summarization Extractive Techniques”, Journal of Emerging Technologies in Web Intelligence, Vol. 2, No. 3, August 2010.

[27] David G. Lowe, “Distinctive Image Features From Scale-Invariant Keypoints”, International Journal of Computer Vision 60(2), 91–110, January 2004.

[28] Ryszard S. Choras´, “Image Feature Extraction Techniques and their Applications for CBIR and Biometrics Systems”, International Journal Of Biology And Biomedical Engineering, 2007.

[29] Dong Ping Tian, “A Review on Image Feature Extraction and Representation Techniques”, International Journal of Multimedia and Ubiquitous Engineering, Vol. 8, No. 4, July, 201 3.

[30] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller, “Introduction to Wordnet: An On-Line Lexical Database”, August 1993.

[31] H. Liu and P. Singh, “ConceptNet - A Practical Commonsense Reasoning Tool-kit”, BT Technology Journal, Vol. 22, No. 4, October 2004.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, 2013.

[33] Muhammad Ajmal, Muhammad Husnain Ashraf, Muhammad Shakir, Yasir Abbas, and Faiz Ali Shah, “Video Summarization: Techniques and Classification”, ICCVG 2012, LNCS 7594, pp.1–13, Lahore, Pakistan, 2012.

[34] "Youtube Statistics", Retrived from: http://www.youtube.com/yt/press/statistics.html, Accessed on: February 14, 2015.

[35] "A Survey on Image Segmentation Techniques for Object Psychology Essay", Retrieved from:http://www.ukessays.com/essays/psychology/a-survey-on-image-segmentation-techniques-for-object-psychology-essay.php?cref=1.,November 2013, Accessed on: February 14, 2015.

[36] Edmondo Trentin and Marco Gori, "A Survey of Hybrid Ann/Hmm Models for Automatic Speech Recognition", Italy, April 2000.

[37] Gerhard Reitmayr, Ethan Eade, and Tom W. Drummond, "Semi-automatic Annotations in Unknown Environments", 6th IEEE and ACM International Symposium, pp. 67-70, 2007.

[38] Meng Wang and Xian-Sheng Hua, "Active Learning in Multimedia Annotation and Retrieval: A Survey", ACM Transactions on Intellegent System Technologies 2, Article 10, February 2011.

[39] Marian Kogler and Mathias Lux, "Bag of Visual Words Revisited-An Exploratory study on Robust Image Retrieval Exploiting Fuzzy Codebooks", University of Klagenfurt, 2008.

[40] Xin-Jing Wang, "Duplicate-Search-Based Image Annotation Using Web-Scale Data", Proceedings of the IEEE, June 1, 2012.

[41] Ameesh Makadia, Vladimir Pavlovic, and Sanjiv Kumar, "Baselines for Image Annotation", European Conference on Computer Vision, 18 May 2010.

[42] Dengsheng Zhang, Monirul Islam, and Guojun Lu, "A Review on Automatic Image Annotation Techniques", School of Information Technology, Monash University, Churchill, Vic.3842, Australia, 2012.

[43] Ankush Gupta and Prashanth Mannem, "From Image Annotation to Image Description", International Institute of Information Technology, Hyderabad - 500032, India, 2013.

[44] Olutobi Owoputi, Brendan O"Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith, "Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters", Proceedings of NAACL-HLT 2013, pp. 380–390, Atlanta, Georgia, 9-14 June 2013.

[45] Purnima. S. Mittalkod and G. N. Srinivasan, "Shot Boundary Detection Algorithms and Techniques: A Review", Research Journal of Computer Systems Engineering-An International Journal, Vol. 02, Issue 02, June 2011.

[46] P. Sidiropoulos, V. Mezaris, I. Kompatsiaris, H. Meinedo, M. Bugalho, and I. Trancoso, "On the Use of Audio Events for Improving Video Scene Segmentation", Analysis, Retrieval and Delivery of Multimedia Content, N. Adami, A. Cavallaro, R. Leonardi, P. Migliorati (Eds.), Springer Lecture Notes in Electrical Engineering, Vol. 158, 2012.

[47] Junaid Baber, Nitin Afzulpurkar, Matthew N. Dailey, and Maheen Bakhtyar, "Shot Boundary Detection from Videos Using Entropy and Local Descriptor", School of Engineering and Technology, Asian Institute of Technology, Pathumthani, Thailand, 2011.

[48] Ganesh I. Rathod and Dipali A. Nikam, "An Algorithm for Shot Boundary Detection and Key Frame Extraction using Histogram Difference", International Journal of Emerging Technology and Advanced Engineering, Vol. 3, Issue 8, August 2013.

[49] Swati D. Bendale and Bijal. J. Talati, "Analysis of Popular Video Shot Boundary Detection Techniques in Uncompressed Domain", International Journal of Computer Applications, Vol. 60, No. 3, December 2012.

[50] P. Swati Sowjanya and Ravi Mishra, "Gesture Interpretation for Video Shot-Boundary Detection", International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE), Vol. 2, Issue 1, January 2013.

[51] Panagiotis Sidiropoulos, Vasileios Mezaris, Ioannis Kompatsiaris, Hugo Meinedo, Miguel Bugalho, and Isabel Trancoso, "Temporal Video Segmentation to Scenes using High-level Audiovisual Features", IEEE Transactions on Circuits and Systems for Video Technology, 2011.

[52] Abdelati Malek Amel, Ben Abdelali Abdessalem, and Mtibaa Abdellatif, "Video Shot Boundary Detection Using Motion Activity Descriptor", Journal of Telecommunications, Vol. 2, Issue 1, April 2010.

[53] Jordi Mas and Gabriel Fernandez, "Video Shot Boundary Detection Based on Color Histogram", La Salle School of Engineering, Ramon Llull University, Barcelona, Spain, 2003.

[54] Guozhu Liu and Junming Zhao, "Key Frame Extraction from MPEG Video Stream", Proceedings of the Second Symposium International Computer Science and Computational Technology, Huangshan, China, 26-28 Dec. 2009.

[55] Sandip T. Dhagdi and P. R. Deshmukh, “Keyframe Based Video Summarization Using Automatic Threshold & Edge Matching Rate", International Journal of Scientific and Research Publications, Vol. 2, Issue 7, July 2012.

[56] Jiawei Rong, Wanjun Jin, and Lide Wu, “Key Frame Extraction using Inter-Shot Information”, Department of Computer Science and Engineering, Fudan University, Shanghai, China, 2004.

[57] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, “SURF: Speeded Up Robust Features”, In Computer vision-ECCV 2006, pp. 404-417. Springer Berlin Heidelberg, 2006.

[58] Li Zhao, Wei Qi, Stan Z. Li, Shi-Qiang Yang, and H. J. Zhang, “Key Frame Extraction and Shot Retrieval using Nearest Feature Line (NFL)”, In Proceedings of the 2000 ACM Workshops on Multimedia, pp. 217-220, 2000.

[59] Dianting Liu, Mei-Ling Shyu, Chao Chen, and Shu-Ching Chen, “Integration of Global and Local Information in Videos for Key Frame Extraction”, In Information Reuse and Integration (IRI), 2010 IEEE International Conference, pp. 171-176, 2010.

[60] Samuel Huron, Petra Isenberg, and Jean Daniel Fekete, “PolemicTweet: Video Annotation and Analysis through Tagged Tweets”, Proceedings of the IFIP TC13 Conference on Human-Computer Interaction (INTERACT), 2013.

[61] Muhammad Usman, Ghani Khan, Rao Muhammad, Adeel Nawab, and Yoshihiko Gotoh, “Natural Language Descriptions of Visual Scenes: Corpus Generation and Analysis”, Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pp. 38–47, Avignon, France, April 23 – 27, 2012.

[62] Bradford Hosack, “VideoANT: Extending Online Video Annotation beyond Content Delivery”, Vol. 54, No. 3, TechTrends, May/June 2010.

[63] Zeeshan Rasheed and Mubarak Shah, “A Graph Theoretic Approach for Scene Detection in Produced Videos”, University of Central Florida, In Multimedia Information Retrieval Workshop, 2003.

[64] André Gustavo Adami, “Automatic Speech Recognition: From the Beginning to the Portuguese Language”, Universidade de Caxias do Sul, Brasil, 2010.

Appendix A: Questionnaire

Investigators:

Mekuanent Birara
Addis Ababa University
Maquintosh20@gmail.com

Dr. Fekade Getahun
Asst. Professor
Addis Ababa University
FekadeGetahun@aau.edu.et

I. Purpose of the questionnaire

The purpose of this questioner is to evaluation a research prototype on video scene annotation and summarization. Your participation will help us understand the performance of this work and find holes for further improvement.

III. Subject's Permission

By completing this questioner, you agree that you have seen the annotated video file provided with this questioner and give your voluntary consent to participate. Should you have any questions about this research or who to contact in the event of evaluation-related injury, please contact by the mentioned addresses.

Thank you very much in advance for giving us your valuable time and evaluations!

Scene based video annotation and summarization

Scene based video annotation is a process of providing description to contents of the video at scene level. This scene level annotation can be visible on video while playing or it can be presented as an XML file.

Please provide your Evaluation points with the following criterion having a look at the annotation XML file or the annotated video. (Please mark only one „X" for each line in the labeled column) **Poor = 1, Satisfactory = 2, Good = 3, Very good = 4**				
Pre-processing	**1**	**2**	**3**	**4**

Please provide your Evaluation points with the following criterion having a look at the annotation XML file or the annotated video. (Please mark only one „X" for each line in the labeled column) **Poor = 1, Satisfactory = 2, Good = 3, Very good = 4**					
Pre-processing		**1**	**2**	**3**	**4**
1	The annotation is given at right time frame of scenes and shots				
2	Are concepts in the video and annotation tags similar				
Concept acquisition		**1**	**2**	**3**	**4**
3	Objects stated in the annotation are found in the video				
4	All the objects in the video are stated in the annotation				
Concept formulation		**1**	**2**	**3**	**4**
5	All the events in the video are stated in the annotation file				
6	Events stated in the annotation file are similar to those in the video				
7	Unrelated events that are found on the speech data are excluded in the annotation file				
Annotator		**1**	**2**	**3**	**4**
8	The annotation sentence is clear, understandable and related to the actual content of the video				
9	Scene level annotation statement gives a description of the scene				

General questions

Do you think the annotator is important? Yes __ No __

Do you think processing speech data improve the annotation? Yes __ No __

Any comments

_____

_____

_____

_____

_____

Appendix B: Sample Code

```
// Shoot boundary detection
package Preprocessing;
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.StandardCopyOption;
import java.util.logging.Level;
import java.util.logging.Logger;
public class ShotBoundary {
   public  File theDir;
   public static int n=50;
   public void DetectShot(String directoryName, int t) {
     try {
        //  System.out.println(t);
        CompareBeans cb= new CompareBeans();
        Colorhistogram ch = new Colorhistogram();
        //EdgeDetector ed= new EdgeDetector();
        EdgeDetection EDe= new EdgeDetection();
        File directory = new File(directoryName);
        //get all the files from a directory
        File[] fList = directory.listFiles();
        String curr = "";
        String prev="";
        System.out.println("the initialized value"+curr);
        int count=0;
        for(int f=0; f<fList.length-2;f++)
        {
            curr=directoryName+"/"+fList[f].getName();
             if(f==0)
            {
            prev=curr;
            }
           else prev=directoryName+"/"+fList[f-1].getName();
           while(f<fList.length)
           {
```

```
            if((f+n)<fList.length)
            {
if((cb.compBeans(ch.Bin(ch.GrayHistogram(directoryName+"/"+fList[f].getName())),ch.Bin(ch.GrayHistogram(dir
ectoryName+"/"+fList[f+n].getName())),t)==true)&&
((cb.compBeans(ch.Bin(ch.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[f].getName()))),ch.Bin(c
h.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[f+n].getName()))),t)==true)))
            {
               add(f,f+n,count,fList);
               f=f+n+1;
            }
            else{
                 for(int rev=f+n;rev>=f;rev--)
                   {
if((cb.compBeans(ch.Bin(ch.GrayHistogram(directoryName+"/"+fList[f].getName())),ch.Bin(ch.GrayHistogram(dir
ectoryName+"/"+fList[rev].getName())),t)==true) &&

((cb.compBeans(ch.Bin(ch.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[f].getName()))),ch.Bin(c
h.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[rev].getName()))),t)==true)))

                      {
                                                                  add(f,rev,count,fList);
                              count=count+1;
                              f=rev+1;
                              break L;
                          }
                      }
                   }
          }
            else
            {
if((cb.compBeans(ch.Bin(ch.GrayHistogram(directoryName+"/"+fList[f].getName())),ch.Bin(ch.GrayHistogram(dir
ectoryName+"/"+fList[fList.length-1].getName())),t)==true)&&
((cb.compBeans(ch.Bin(ch.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[f].getName()))),ch.Bin(c
h.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[fList.length-1].getName()))),t)==true)))
            {
               add(f,fList.length-1,count,fList);
                        }
```

```
                else{
                    for(int rev=fList.length-1;rev>=f;rev--)
                      {
if((cb.compBeans(ch.Bin(ch.GrayHistogram(directoryName+"/"+fList[f].getName())),ch.Bin(ch.GrayHistogram(dir
ectoryName+"/"+fList[rev].getName())),t)==true)                                                    &&
((cb.compBeans(ch.Bin(ch.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[f].getName()))),ch.Bin(c
h.GrayHistogramBuff(EDe.detectEdge(directoryName+"/"+fList[rev].getName()))),t)==true)))
                        {
            add(f,rev,count,fList);
                                count=count+1;
                                f=rev+1;
                                break L;
                            }
                        }
                    }
                }
            }
        }
    } catch (IOException ex) {
        Logger.getLogger(ShotBoundary.class.getName()).log(Level.SEVERE, null, ex);
    }
   }

  public void add(int start, int end,int count, File[] fList)
   {
      for(int add=start;add<=end;add++)
                        {
         try {
            theDir = new File("D:\\ThesisRelated\\Shots\\Shot"+count);
            theDir.mkdir();
            Files.copy(fList[add].toPath(),(new                    File(theDir                    +"\\"+
fList[add].getName())).toPath(),StandardCopyOption.REPLACE_EXISTING);
         } catch (IOException ex)
{              Logger.getLogger(ShotBoundary.class.getName()).log(Level.SEVERE, null, ex);
         }
       }
   }
```

```
   public void addtoprev(int start, int end,int count, File[] fList)
   {
      for(int add=start;add<=end;add++)
                        {
         try {
            theDir = new File("D:\\ThesisRelated\\Shots\\Shot"+count);
            Files.copy(fList[add].toPath(),(new File(theDir +"\\"+
fList[add].getName())).toPath(),StandardCopyOption.REPLACE_EXISTING);
         } catch (IOException ex) {
  Logger.getLogger(ShotBoundary.class.getName()).log(Level.SEVERE, null, ex);
         }
       }
   }
}

//Audio processor
package Speech;
import edu.stanford.nlp.tagger.maxent.MaxentTagger;
.
.
.
import org.xml.sax.SAXException;
public class SpeechProcessor {
   public static void ExtractAudio(String [] args) throws IllegalArgumentException, EncoderException
   {
      File source = new File("D:\\ThesisRelated\\Test feature\\Borat.mp4");
      File target = new File("D:\\ThesisRelated\\Speech\\Result.mp3");
      AudioAttributes audio = new AudioAttributes();
      audio.setCodec("libmp3lame");
      audio.setBitRate(new Integer(128000));
      audio.setChannels(new Integer(2));
      audio.setSamplingRate(new Integer(44100));
      EncodingAttributes attrs = new EncodingAttributes();
      attrs.setFormat("mp3");
      attrs.setAudioAttributes(audio);
      Encoder encoder = new Encoder();
```

```
        encoder.encode(source, target, attrs);
    }
public static void splitFile(int [][]a) throws IOException, UnsupportedAudioFileException, CannotReadException,
TagException, ReadOnlyFileException, InvalidAudioFrameException
    {   File file = new File("D:\\ThesisRelated\\Speech\\Result.mp3");
        //AudioFile audioFile = AudioFileIO.read(file);
        //int duration= audioFile.getAudioHeader().getTrackLength();
        int duration=510;
        FileInputStream fis = null;
        FileOutputStream fos = null;
        long filesize = file.length();
        for(int i=0;i<a.length;i++)
        {
        int splitsize = (int)(filesize*(a[i][1])/ duration);
        byte[] b = new byte[splitsize];
        try {
            fis = new FileInputStream(file);
                        String filecalled = "D:\\ThesisRelated\\Speech\\result" + "_split_final"+i  + ".mp3";
                fos = new FileOutputStream(filecalled);
                int bi = fis.read(b);
                //int i2 = fis.read(c);
                fos.write(b, 0, bi);
                fos.close();
                fos = null;
                //System.out.println(filecalled + "     " + i + " bytes");
                //filesizeActual += i;

            //Assert.assertEquals(filesize, filesizeActual);
           // mergeFileParts(result, splitval);
            //check(filename1, mergeFile);
        } finally {
            if(fis != null) {
                fis.close();
            }
            if(fos != null) {
                fos.close();
            }
```

```
        }
    }
    }

    private static void mergeFileParts(String filename1, int splitval) throws IOException {
        FileInputStream fis = null;
        FileOutputStream fos = null;
        try {
            String name1 = filename1.replaceAll(".mp3", "");
            String mergeFile = name1 + "_merge.mp3";
            fos = new FileOutputStream(mergeFile);
            for (int j = 1; j <= splitval; j++) {
                String filecalled = name1 + "_split_" + j + ".mp3";
                File partFile = new File(filecalled);
                fis = new FileInputStream(partFile);
                int partFilesize = (int) partFile.length();
                byte[] b = new byte[partFilesize];
                int i = fis.read(b, 0, partFilesize);
                fos.write(b, 0, i);
                fis.close();
                fis = null;
            }
        } finally {
            if(fis != null) {
                fis.close();
            }
            if(fos != null) {
                fos.close();
            }
        }
    }
public static void POST() throws IOException,ClassNotFoundException
   {
      String tagged;
       MaxentTagger tagger = new MaxentTagger("taggers/english-left3words-distsim.tagger");
FileInputStream fstream = new FileInputStream("D:\\ThesisRelated\\FinalText.txt");
       DataInputStream in = new DataInputStream(fstream);
```

```
      BufferedReader br = new BufferedReader(new InputStreamReader(in));
       while((sample = br.readLine())!=null)
          {
          tagged = tagger.tagString(sample);
          FileWriter q = new FileWriter("D:\\ThesisRelated\\output.txt",true);
          BufferedWriter out =new BufferedWriter(q);
          //write it to the file output.txt
          out.write(tagged);
          out.newLine();
          out.close();
          }
  }
public static void main(String []args) throws FileNotFoundException, ParserConfigurationException, TransformerException, TransformerConfigurationException, SAXException, IOException, XPathExpressionException, UnsupportedAudioFileException, CannotReadException, TagException, ReadOnlyFileException, InvalidAudioFrameException, ClassNotFoundException
      {
          int [][]aa= new int[3][2];
          POST();
          splitFile(aa);
          int statementcounter=0;
          String statement="";
          Scanner sc = new Scanner(new File("D:\\ThesisRelated\\output.txt"));
          String [] words={""};
            String line="";
          while(sc.hasNext())
          {
             line = line + sc.nextLine();
                    }
          words = line.split(" ");
          for(int i=0;i<words.length;i++)
          {
         if((i==0)||(words[i-1].equals("._."))||(words[i-1].equals("?_.")))
             {
                statementcounter++;
                String [][] verbList = new String[200][2];
                String [][] nounList = new String[200][2];
```

```
int sNList=0;
int sVList=0;// counter for verbs in the sentence
//combine the words and form the statement
for(int j=i;(j<words.length)&&!words[j].equals("._.");j++)
{
  //find the nouns and verbs and write them on the xml file
  if (words[j].matches("(?i).*_VB.*"))
   {

     verbList[sVList][0]=words[j].replaceAll("_.*", "");
     verbList[sVList][1]=words[j].replaceAll(".*_VB", "");
     sVList++;//used to count the number of verbs that are found in the statement
                        }
  else if (words[j].matches("(?i).*_NN.*"))
   {
                nounList[sNList][0]=words[j].replaceAll("_.*", "");
   nounList[sNList][1]=words[j].replaceAll(".*_NN", "");
     sNList++;
   }
  //remove tagges before combining the words
  String s = words[j].replaceAll("_.*", "");
  //end of tag removal
  statement=statement+" "+s;
                }
String si=""+statementcounter;
XML("Statement",si,statement,"","");
statement="";
for(int v=0;v<sVList;v++)
{
  XML("Verb",v+"",verbList[v][0],verbList[v][1],si);
}
for(int n=0;n<sNList;n++)
{
   XML("Noun",n+"",nounList[n][0],nounList[n][1],si);
}
 }
}
```

```
    }
    public static void XML(String node,String attr1, String attr2, String attr3, String attr4) throws ParserConfigurationException, TransformerConfigurationException, TransformerException, SAXException, IOException, XPathExpressionException
    {
        File f= new File("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml");
         if (!f.exists())
         {
            DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
            DocumentBuilder docBuilder = docFactory.newDocumentBuilder();
            Document doc = docBuilder.newDocument();
            Element rootElement = doc.createElement("Event_Object_Index");
            doc.appendChild(rootElement);

            TransformerFactory transformerFactory = TransformerFactory.newInstance();
            Transformer transformer = transformerFactory.newTransformer();
            DOMSource source = new DOMSource(doc);
            StreamResult result = new StreamResult(new File("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml"));

            transformer.transform(source, result);

            System.out.println("File saved!");
         }
         if(node=="Statement")
            {
               DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
               DocumentBuilder db = docFactory.newDocumentBuilder();
               Document doc = db.parse("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml");
               Element dataTag = doc.getDocumentElement();

               Element newStatement = doc.createElement(node);
               newStatement.setAttribute("ID", attr1);
               newStatement.setAttribute("Text", attr2);
               dataTag.appendChild(newStatement);
               TransformerFactory transformerFactory = TransformerFactory.newInstance();
               Transformer transformer = transformerFactory.newTransformer();
```

```
                DOMSource source = new DOMSource(doc);
                StreamResult result = new StreamResult(new
File("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml"));

                transformer.transform(source, result);
                 System.out.println("File saved!");
              }
            else if(node=="Verb")
              {
                DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
                DocumentBuilder db = docFactory.newDocumentBuilder();
                Document doc = db.parse("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml");
                XPathFactory xPathfactory = XPathFactory.newInstance();
                XPath xpath = xPathfactory.newXPath();
                XPathExpression expr = xpath.compile("//Statement[@ID="+attr4+"]");
                NodeList n =  (NodeList) expr.evaluate(doc, XPathConstants.NODESET);

                Element newVerb = doc.createElement(node);
                newVerb.setAttribute("ID", attr1);
                newVerb.setAttribute("Text", attr2);
                newVerb.setAttribute("Type", attr3);

                for(int i=0;i<n.getLength();i++)
                {
                   Element el = (org.w3c.dom.Element) n.item(i);
                   el.appendChild(newVerb);
                }
                TransformerFactory transformerFactory = TransformerFactory.newInstance();
                Transformer transformer = transformerFactory.newTransformer();
                DOMSource source = new DOMSource(doc);
                StreamResult result = new StreamResult(new
File("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml"));

                transformer.transform(source, result);

                System.out.println("File saved!");
              }
```

```
            else
              {
                 DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
                 DocumentBuilder db = docFactory.newDocumentBuilder();
                 Document doc = db.parse("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml");
                 XPathFactory xPathfactory = XPathFactory.newInstance();
                 XPath xpath = xPathfactory.newXPath();
                 XPathExpression expr = xpath.compile("//Statement[@ID=\""+attr4+"\"]");
                 NodeList n = (NodeList) expr.evaluate(doc, XPathConstants.NODESET);

                 Element newNoun = doc.createElement(node);
                 newNoun.setAttribute("ID", attr1);
                 newNoun.setAttribute("Text", attr2);
                 newNoun.setAttribute("Type", attr3);

                  for(int i=0;i<n.getLength();i++)
                 {
                    Element el = (org.w3c.dom.Element) n.item(i);
                    el.appendChild(newNoun);
                 }
  TransformerFactory transformerFactory = TransformerFactory.newInstance();
      Transformer transformer = transformerFactory.newTransformer();
                 DOMSource source = new DOMSource(doc);
                 StreamResult result = new StreamResult(new
File("D:\\ThesisRelated\\XML\\Audio_NV_INDEX.xml"));
                 transformer.transform(source, result);
                 System.out.println("File saved!");
              }
            }
           }
```

Declaration

I, the undersigned, declare that this research is my original work and has not been presented for degree in any other university, and that all sources of materials used for the research have been acknowledged.

Declared by:

Name: Mekuanent Birara

Signature: _____

Date: _____

Confirmed by advisor:

Name: Fekade Getahun (PhD)

Signature: _____

Date: _____

Place and date of submission: Addis Ababa University, February 25, 2015.