Draw, Utter and Search: A Multi-modal Video Search Engine

by

Ozan Can Altıok

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Computer Engineering

KOÇ
UNIVERSITY

January 22, 2019

Draw, Utter and Search: A Multi-modal Video Search Engine

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Ozan Can Altıok

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Assoc. Prof. Tevfik Metin Sezgin

_____

Prof. Yücel Yemez

_____

Assoc. Prof. Yusuf Sahillioğlu

Date: _____

For my family

ABSTRACT

With the increasing amount of multimedia content available on the web, the focus on video retrieval engines has been shifting from text-based systems to content-based methods that allow indexing and retrieval based on video contents. This trend has sparked a quest for efficient and effective video retrieval systems on large video collections. Most video retrieval systems rely only on hand-crafted features and manual annotations. Motion of the individual objects, the most decisive information conveyed in videos, is usually overlooked in video retrieval. From a user interaction perspective, motion can be given as a query using speech and sketch simultaneously. Speech allows easy specification of content, events and relationships, while sketching brings in spatial expressiveness. Unfortunately, we have insufficient knowledge of how sketching and speech can be used for video retrieval, because there are no existing retrieval systems that support such interaction. In this paper, we describe a Wizard-of-Oz protocol and a set of tools that we have developed to engage users in a sketch- and speech- based video retrieval task. We report how the protocol and the tools fit together to establish an ecologically valid testbed using *retrieval of soccer videos* as a use case scenario. Using the data collected in the studies, we developed a model capable of interpreting simultaneous speech and sketching to infer the sequence of motions described by a user. The performance results of the model suggest that the protocol and the tools together have the potential to serve as effective means for studying a wide range of multi-modal use cases.

Moreover, a video retrieval system was built by integrating the multimodal interpretation model to a database back-end designed for big multimedia collections. The retrieval system was assessed through user evaluation studies. The evaluation results demonstrate that the given query interpretation mechanism and the database system

make a good couple for motion-based video retrieval on big video collections.

ÖZETÇE

Web üzerinde çoklu ortam içeriğinin artışına paralel olarak, video aramaları, yazı tabanlı arama yöntemleri yerine, video içeriğine göre dizinleme olanağı sağlayan içerik tabanlı video arama yöntemleri kullanılarak gerçekleştirilmeye başlanmıştır. Bu eğilim, büyük video kümeleri üzerinde etkili ve verimli arama gerçekleştirebilecek video arama sistemleri üzerine bir araştırma sürecinin başlangıcı olmuştur. Birçok video arama sistemi, sadece el yordamıyla oluşturulan özniteliklere ve etiketlemelere bağlı olarak arama gerçekleştirmektedir. Video gibi devimsel içerikleri birbirinden ayıran en önemli özellik olan nesnelerin hareket bilgisi görmezden gelinmektedir. Hareket, çizim ve konuşmanın eş zamanlı olarak kullanılmasıyla belirtilebilecek bir bilgidir. Konuşma, içeriğin, olayların ve nesnelerin birbirleriyle olan ilişkilerinin kolaylıkla belirtilebilmesine olanak tanırken, çizim uzamsal ifade kabiliyeti sunmaktadır. Fakat, söz konusu etkileşim yapısına sahip bir video arama sistemi bulunmadığından, bu kiplerin video aramalarında nasıl kullanılabileceğine dair bir bilgi eksikliği mevcuttur. Bu çalışmada, kullanıcıların çizim ve konuşma tabanlı video arama görevlerine aktif katılımlarını sağlayacak bir Oz Büyücüsü yönergesi ve bazı araçlar geliştirilmiştir. Söz konusu araçların ve arama yönergesinin birbirleriyle olan uyumu, bir kullanım alanı üzerinde (*futbol maçlarının aranması*) değerlendirilmiştir. Ardından, toplanan kullanıcı etkileşim verileri kullanılarak, eş zamanlı olarak verilen çizim ve konuşma girdilerinden kullanıcının bahsetmiş olduğu hareket olaylarının sıralamasının elde edilebildiği bir makine öğrenmesi modeli geliştirilmiştir. Bu modelin performans sonuçları, video arama yönergesinin ve araçların farklı türlerde videoların aranmasında çoklu etkileşim mekanizmalarının irdelenmesi için uygun olduğunu göstermektedir.

Bunun yanında, oluşturulmuş çok kipli yorumlayıcı, çoklu ortamlar için hazırlanmış ölçeklenebilir ve hızlı bir veritabanı sistemi ile birleştirilmiş ve bir video arama sis-

temi meydana getirilmiştir. Söz konusu video arama sistemi, kullanıcı değerlendirme çalışamaları ile değerlendirilmiştir. Çalışmalardan elde edilen sonuçlar, oluşturulan çok kipli yorumlama mekanizmasının ve veritabanı sisteminin büyük video kümeleri üzerinde hareket tabanlı video araması için iyi bir ikili olduğunu göstermektedir.

ACKNOWLEDGMENTS

This work was supported by the CHIST-ERA project iMotion with contributions from the Scientific and Technological Research Council of Turkey (TÜBİTAK), under Grant No.: 113E325.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

Chapter 1

INTRODUCTION

Video has become an influential tool for capturing and delivering information in personal, professional and educational applications. This trend has raised the importance of efficient and effective video retrieval systems on large video collections. Most of the video retrieval systems rely only on hand-crafted features and manual annotations. Motion of the individual objects, the acutest information carried on by videos, is usually ignored in video retrieval. In the human-computer interaction front, sketching and speech are the two modalities that complement each other to describe motion in a video. Sketching indicates placement of objects together with their motion path. Through speech, users can indicate events and their relationships. For instance, in a soccer match, one can mark coordinates of a player by drawing a circle, and specify a goal scored by the player by drawing an arrow heading from the player to the goalpost. However, drawing an arrow would not itself suffice to describe a goal since the arrow can also indicate a forward pass, or a free kick. Users can explicitly indicate that the motion is a goal shot through speech.

However, we do not have adequate information about how speech and sketching can be used for video retrieval since there are no existing video retrieval systems based on these modalities. To gather information regarding the joint use of these modalities, a protocol must be designed to capture participants to talk and draw simultaneously. Also needed are some applications to collect speech and sketching data and to process these data for later use. Moreover, no adequate research has been conducted on the co-interpretation of these modalities and thus generating a plain-text search query.

This paper presents a protocol together with a set of software tools to stimulate

participants to draw and talk in a video retrieval task. We tested the protocol and the software tools on the soccer match retrieval use case scenario. We ensured ecologically validity by involving participants with relevant expertise in this area. These software tools are also customizable and the protocol is easy to follow. Next, we developed a model which is capable of interpreting simultaneously given sketching and speech inputs, and understanding the motion sequence described by a user. The model has a remarkable performance on the data we collected. Finally, a video retrieval system was built by integrating the model into a database back-end that is designed for big multimedia collections. Several user evaluation studies were conducted on the retrieval system. The results of the studies imply that the data collection software, the data collection protocol and the multimodal interpretation model serve as a powerful trio for building multimodal motion-based video retrieval systems on big multimedia collections.

The rest of this paper is structured as follows. Section 2 presents the related work on developing sketch and speech-based multi-modal systems. Section 3 briefly gives information on selecting a relevant video search use case. Section 4 describes the software tools and the protocol used in the data collection studies with some insights on the data collected. Section 5 defines the multimodal interpretation model and presents the performance results of the model. In Section 6, we illustrate the design of the database system used in the video retrieval system. Section 7 presents the user interface design of the video retrieval system, illustrates the protocol followed in the user evaluation studies, and reports the results of these studies. Section 8 concludes the article.

Chapter 2

RELATED WORK

A bulk of work has been done on the multimodal representation of motion in the human computer interaction literature. In the light of all the works, our research was planned to boost the natural use of sketching and speech on motion scenarios in video clips. In this context, sketch interpretation, natural language understanding and video retrieval systems are within the scope of our research.

Sketching is the first modality that has drawn human computer interaction researcher's attention to developing motion-based video retrieval systems. The spatial information conveyed in sketching has been treated as the primary information to retrieve in video clips. A motion-based video retrieval system on soccer videos uses sketching for describing the motion of the ball and players [Al Kabary and Schuldt, 2013]. Users can also limit the area by drawing a boundary shape. To specify the events in addition to the motions drawn, a user-interface with selection buttons is provided. In the system designed by [Collomosse et al., 2009], users are allowed to illustrate much more complex scenarios without limiting the area. The system classifies all the arrows as motion cues and rest of the symbols as objects. Each motion cue is linked to the object that is closest within a certain distance. Both systems use only sketching as the input method, limiting the users' ability to articulate spatial information of objects and their motions. Speech, which enables specifying events and relationships of motions comfortably, is overlooked in these systems.

The works mentioned above imply that the symbols used to describe motions are different than others. In this context, usage patterns of depicting motion have become a hot topic in the human-computer interaction research. The user interface developed by [Thorne et al., 2004] defines a set of sketched symbols for basic body

gestures. For instance, one needs to draw a bowl-like shape to describe jumping, whilst a small semi-ellipse must be drawn for walking. The motion to animate is composed of such basic motion drawings. Primitive motion symbols are recognized by expressing these symbols in terms of arcs and lines [Hammond and Davis, 2006a]. Another interface for controlling robots leverages sketching to articulate the path of a robot [Skubic et al., 2007]. Moreover, Kitty, which was developed for cartoonists, also uses free-form sketching to construct paths for objects [Kazi et al., 2014].

In addition to these works, a research has been done on expressing motion of articulated objects in video clips. In the work by [Mauceri et al., 2015], 3 different query modes have been proposed for describing motion of objects: appending arrows to sub-parts of a formerly drawn object, appending arrows to sub-parts of an articulated object template and sorting out key-frames of the motion by moving joints of an articulated object template. Through the user evaluation studies, it has been concluded that the query modes using articulated object templates are more accurate than drawing an object and appending arrows to sub-parts. However, it must be noted that using object templates reduces naturalness of a given motion query.

It is clear that most of the works presented above take the advantage of free-form strokes and arrows to illustrate motion. Comparing these two query modes, arrows have more expressive power than free-form strokes because of their arrow head which reveal direction. This has motivated us to use arrows rather than free-form strokes for depicting motion in the development of our video retrieval system.

There are some multimodal systems that support the joint use of speech and pen without performing video retrieval. For instance, QuickSet [Cohen et al., 1997] enables users to locate objects on a map and define movements of them by using pen gestures and speech. The system employs hidden Markov models and neural networks to recognize pen gestures. To interpret pen gestures and speech together, it extracts features of each modality, and tries to find the best match between them through statistical methods on the features. Once the joint interpretation with the highest probability was found, the objects are manipulated through the instructions inferred

from multimodal interpretation. Additionally, IBM has developed a text editor that uses speech and pen gestures together for manipulating texts [Vergo, 1998]. Users can type a text through speech and do some pen gestures supported by some utterance to manipulate a particular text. The system utilizes feature-based statistical machine translation methods [Papineni et al., 1997]. These methods append the features extracted from pen gestures into the natural language features obtained from speech, and applies a statistical machine translation on the features in order to infer the instructions given by a user.

Empirical studies on the cooperative use of speech and sketching have been another direction in human-computer interaction research. There are a few studies that have investigated the use of speech and sketching. [Adler and Davis, 2007a] have designed a study to observe how people use speech and sketching simultaneously and to finally implement a multi-modal white-board application based on these observations. They asked students from circuit design class to describe and draw a floor plan for a room, design an AC/DC transformer, design a full adder, and describe the final project they had developed for their circuit design class. In each session, experimenter and participant sit across a table. On either sides of tables, there are tablets each running an application that replicates whatever gets drawn in one tablet in another. The participant describes each of the 4 scenes above via speech and sketching. Throughout the sessions, the experimenter also adds to the participant's drawing and directs questions, leading to a bi-directional conversation. Later, [Adler and Davis, 2007b] designed another study for collecting speech and sketched symbols to develop a multi-modal mechanical system design interface. They asked six participants to talk about and draw a pendulum system on a white board. Although these studies investigate the joint use of speech and sketching, they did not have multimedia retrieval in focus. Our protocol and a set of tools are focusing on the investigation of collaboratively using speech and sketching for multimedia retrieval.

Moreover, [Oviatt et al., 2000] have drawn attention to *the unavailability of multi-modal corpora* for developing new methods for multi-modal language processing. They

have underlined the need for methods and tools for obtaining multi-modal inputs. Our protocol and software tools also help human-computer interaction researchers to solve this issue by enabling collection of speech and sketching simultaneously.

The long progression in natural language understanding has also motivated us to focus on multimodal interpretation for motion-based video retrieval. [Yu et al., 2013] developed LibShortText, which is a library leveraging support vector machines and logistic regression on bag-of-words features. This library is useful when the dictionary of a natural language dataset is limited. Later, neural networks became popular as the natural language datasets grow in size. The advantage of these networks is that they are also capable of learning how to extract features from a short text segment at the character or the word level (or both levels). The model designed by [dos Santos and Gatti, 2014] constructs features of a short text (namely *embeddings*) through the long short term memory (LSTM) structures. Another work done by [Joulin et al., 2016] is a mixture of old-style feature extractors and feed-forward neural networks. The advantage of that architecture is its runtime, that is, the text classification model is constructed quicker than the other two. We designed our multimodal interpretation framework by leveraging these state-of-the-art systems, and the classification performance results suggest that the given data-collection protocol and the multimodal interpretation framework can be used end-to-end for building sketch-and-speech based video retrieval systems.

Furthermore, there have been several advancements in sketch recognition frameworks, which was another motivation for us to focus on much more complex user interfaces supporting natural interaction. The first milestone in the recognition of sketched symbols without definite rules was the IDM feature extraction method [Ouyang and Davis, 2009]. This method opened the way to more accurate sketched symbol classifiers. Next, [Tumen et al., 2010] investigated the use of this feature extraction method with different machine learning heuristics, obtaining far more accurate classifiers. Moreover, [Yesilbek et al., 2015] conducted several experiments to find the intersection of the hyper-parameter ranges yielding the best classification

accuracies on different sketched symbol datasets when using the SVM classifier with different feature extraction methods.

Chapter 3

DATA COLLECTION STUDY

We designed a protocol and implemented software tools to collect speech and sketching simultaneously from people having different levels of expertise in soccer. By having people with ranging expertise in soccer, we established an ecologically validity between the tools and the protocol. The tools and the protocol are also highly customizable, meaning that they can be modified for use in different motion-based video retrieval use cases. Next, we constructed a visual vocabulary to ease sketching, and prepared some video clips to present during data collection sessions. Finally, the data collected from the participants was annotated and prepared for use in several machine learning models for multimodal interpretation.

3.1 Retrieval Use Case

We aimed to design a multimodal interpretation framework on simultaneously given sketching and speech, and thus to build a system that performs motion-based video retrieval with the given input. For this reason, the use case was initially planned to frequently involve motion scenarios regardless of the context. However, the open-ended nature of sketching and speech makes it difficult to handle different contexts at the same video retrieval framework. For example, sketched symbols and words uttered for describing a soccer drill differ largely from the ones used for chess matches. To build a retrieval system operating on both chess and soccer matches, we must put in twice as much effort as we would do for either of them. Within this context, we decided to narrow down on the use case and choose a single case involving motion scenarios. Considering its popularity, we finally selected *retrieval of soccer match videos* as the use case of our video retrieval system.

3.2 Software Tools

We developed 3 applications, one to collect speech and sketching simultaneously from the participants, another to provide a video search front-end for the participant, and a third to annotate the symbols drawn during a session.

3.2.1 Data Collection

We developed two applications running on Android tablets, one for the experimenter [Altıok, 2017b] and another for the participant [Altıok, 2017c]. As seen in Figure 3.1, both applications have a canvas with a background image that can be customized based on the use case scenario. These applications use a shared canvas, that is, whatever gets drawn in one tablet is replicated in another tablet and is also written to a text file on both sides for later use. To replicate the drawings, these applications communicate with each other over a Wi-Fi network. If a user wants to highlight a part of her drawing, then she can get her stylus pen closer to the screen. The wizard (experimenter) sitting across the opposite side can see where the user is highlighting. Moreover, these applications also record their users' videos through built-in tablet cameras while the applications are in use. Note that there are 3 buttons on the bottom-left corner. The leftmost button clears the entire canvas. The rightmost one activates the eraser mode. Users can erase any part of her drawing through eraser. By clicking the button in the middle, she can activate the drawing mode.

There are two outputs of each software:

- A text file (alternatively called a *sketch stream file*) including a list of the following actions recorded with a time stamp:
  - Starting a stroke
  - Ending a stroke
  - Adding a point to sketch
  - Hovering over a point

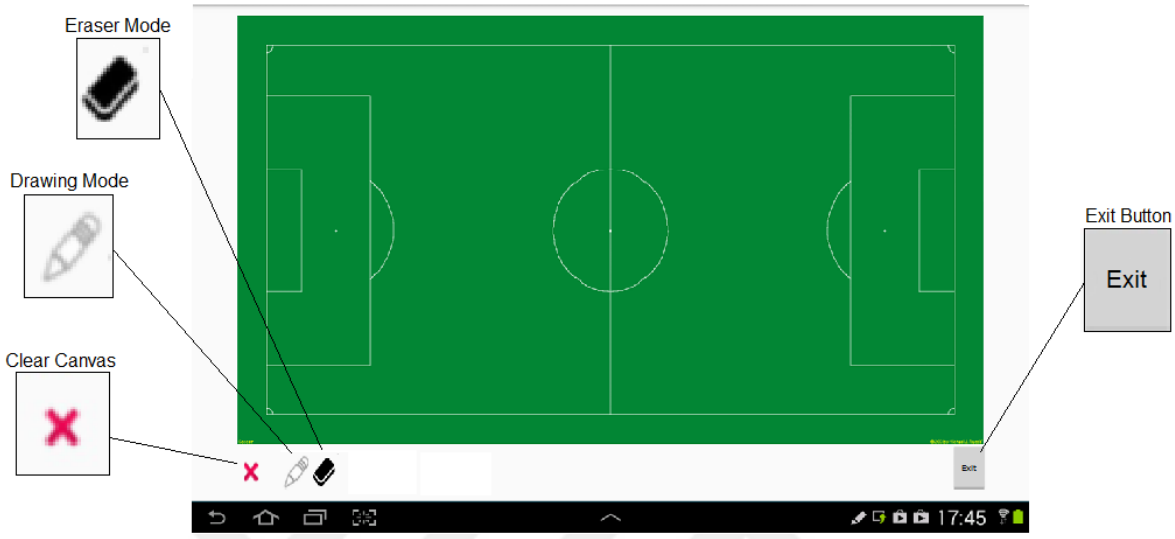


Figure 3.1: A screen-shot from data collection application, with soccer field background image chosen for soccer match retrieval (with the explanations of UI components)

  - Clearing entire canvas

- A video containing the entire session

For the transcription of each session video collected on the participant's side, we marked start and end points between which participants talked about a video clip to search. Afterwards, we connected speaker output to a virtual microphone through a virtual hardware driver [Muzychenko, 2015] to redirect the participants' voice to speech recognizer like a real-time utterance. Next, we transcribed the session video of the participant using IrisTK [Skantze and Al Moubayed, 2012] test recognizer application with Google Cloud Speech Recognition API support [Google, 2017]. Once we obtained the transcripts, we corrected mistranscribed words by hand.

3.2.2 Wizard Application for Video Search

Another application serves as the video search front-end for the participant [Altıok, 2017a]. This application uses video clips prepared for retrieval tasks.

The application consists of two windows, one for the experimenter and another for the participant.

- **Participant's Window:** This window is located at the participant's screen. The window contains a video display where a video clip selected on the experimenter's window is played 3 times.

- **Experimenter's Window:** This window supports two main functionalities: playing the video clip to search and navigating through videos. The window is presented in Figure 3.2.

  - **Playing a video clip to search:** For each motion event, when the experimenter finds the correct video clip, the participant and the experimenter move on to the next event by clicking the topmost button on the left panel. Once the button is clicked, a video clip of the next event is randomly picked and played 3 times on the participant's screen. If all events have been covered, then the application displays a message indicating that the session is over.

  - **Navigating through video clips:** For each motion event, once the participant draws and verbally describes the scenario of a video clip, the experimenter looks at the list of video clips on his side, and picks one for preview by clicking the green buttons. When the wizard is happy with his selection, he sends the video to play on the participant's screen by clicking the "send video" button at the bottom.

3.3 Visual Vocabulary

Due to the open-ended nature of sketching, participants needed help in establishing a visual vocabulary for describing objects and their motions. To this end we supplied them a visual vocabulary. We reviewed the notations used in *Soccer Systems and*

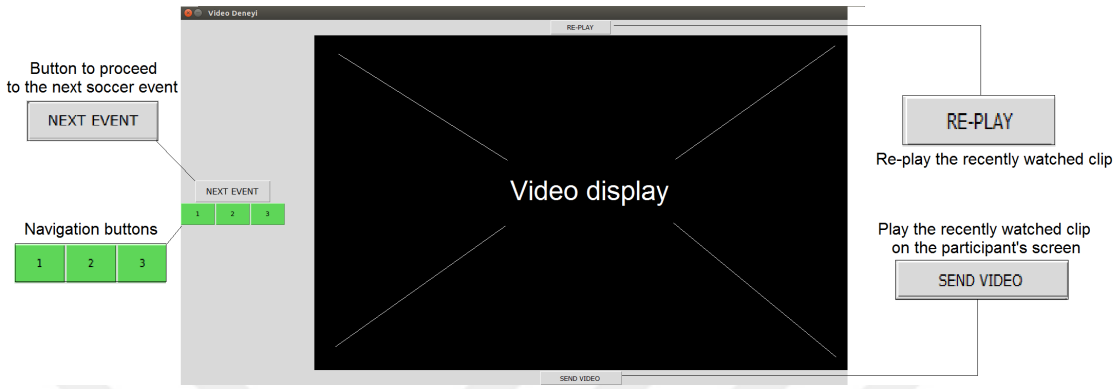


Figure 3.2: A screen-shot from wizard application for video search (experimenter's side) with the explanations of UI components

Strategies [Bangsbo and Peitersen, 2000] (see Figure 3.3) and Goal.com's live scoring system[1] (see Figure 3.4). We wanted participants to memorize as few types of symbols as possible, therefore we combined these notations and removed the zig-zag symbol for dribbling. There appears to be no need for indicating dribbling explicitly. A player motion arrow drawn from a player right after sketching a ball motion arrow coming to the player would imply dribbling. One can also say a player is dribbling while drawing a player motion arrow to indicate dribbling. The resulting visual vocabulary is presented in Figure 3.5.

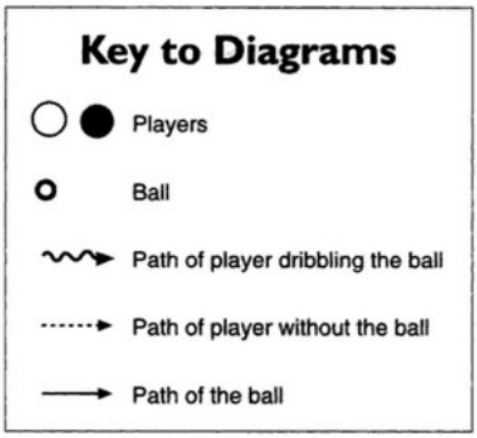


Figure 3.3: Drawing notation used in Soccer Systems and Strategies

Figure 3.4: Drawing notation used in Goal.com's live scoring system

3.4 Video Clips

Our goal is to retrieve clips describing motions in soccer videos. This requires a database of videos augmented with motion data of players and the ball. In the soccer match retrieval case, there are two alternative ways of finding such video clips:

- **Using motion sensors in matches:** Making arrangements with soccer clubs, mounting motion sensors on players and the ball and taking videos of these matches together with their motion data

- **Using a soccer simulation game:** Finding an open-source soccer game and modifying its source code to record coordinates of the ball and players together with frames at every moment.

Ideally, the first option is preferable. However, this choice requires a massive infrastructure for motion sensors mounted on players and several cameras located around a soccer field. Due to the infeasibility in infrastructure, the second option is more doable to prepare videos augmented with motion. Although compiling videos from a soccer game is far from reality, these videos are similar to the ones in real life in terms of events that might happen in a match such as passes, shots and kicks. Presenting a soccer clip from a soccer game would not affect participants' behavior of using speech and sketching. Hence, to prepare soccer video clips that will be presented to participants during sessions, we modified an open-source soccer simulation game named Eat the Whistle [Greco, 2008] in a way described above (see Figure 3.6 for a screen-shot). Afterwards, we recorded several soccer matches played between AI-controlled teams. For each match recorded, we compiled the subsequent frames into a

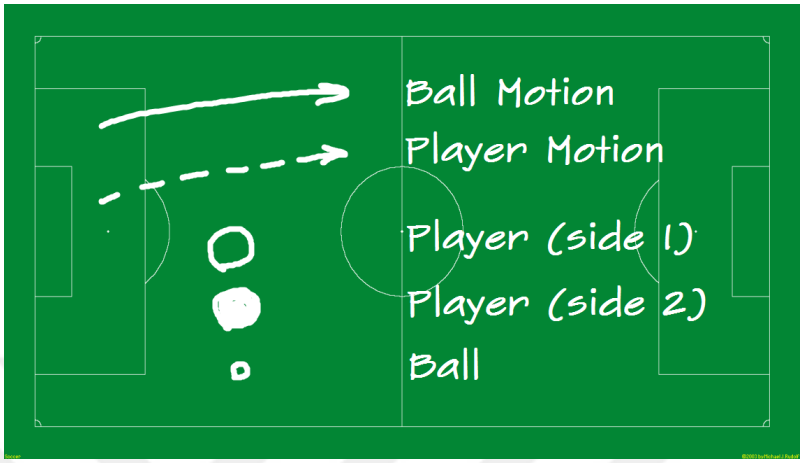


Figure 3.5: Drawing notation used in the data collection sessions

video. Finally, for each of the following events listed in Table 3.1, we prepared video clips from match videos.

Figure 3.6: A screen-shot from Eat the Whistle

3.4.1 Events for a Session

Searching for scenes of all events above might extend a session. After a while participant might get exhausted and not provide proper data. Thus we decided to divide

Corner kick	Shot hitting the post
Direct free kick	Shot saved by a goalkeeper
Indirect free kick	Throw in
Header	Shot going wide
Bicycle	Foul
Goal	Poke tackle
Pass	Sliding tackle

Table 3.1: Motion events that might happen in a soccer match

the events among participants. For each participant, we chose either the first or the last 7 events such that the successive participants do not describe the same events.

3.5 Method

This section presents the physical set-up and the protocol followed.

3.5.1 Physical Set-Up

We aim to investigate the incorporation of speech and sketching in video retrieval. To this end, we initially planned a typical Wizard-of-Oz set-up where participants are asked to draw and talk about a scenario on a computer without being told that the computer is controlled by a human. However, as we have established in our pilot runs, after a while, participants hesitate talking while staring at a computer screen that does not produce any feedback indicating that their speech is being attended to. To motivate participants to talk, we allowed limited feedback provided by the experimenter. The experimenter does not engage into a conversation, however gives feedback while listening to the participant by nodding his head or saying "OK, I'm listening to you" as needed.

As illustrated in Figure 3.7, each session takes place on a table with two tablets,

a computer with a screen plugged in. On the participant's side, there is a screen to display videos to the participant, and a tablet to describe motion in the video clip through sketching. On the experimenter's side, there is a computer including a software to pick up a video clip randomly and display it on the participant's screen, and another tablet to see the participant's drawing.

3.5.2 Protocol

In each session, the participant and the experimenter sit across a table. Initially, the experimenter provides a set of sample sketched symbols describing the visual vocabulary for constructing drawings. To practice the use of the visual vocabulary, the experimenter opens a video clip, and requests the participant to describe the movement of objects in the clip. Next, they move on to the search tasks. In the search tasks, there are different motion events to describe through speech and sketching. For each motion event, the experimenter picks up a video clip of the event randomly, and plays the clip on the participant's screen 3 times. Then, the participant draws the motion of the objects in the clip on her tablet while talking at the same time. The experimenter then looks at the videos of the event on his screen, finds a video that is similar to the scenario drawn and uttered by the participant, and plays the video on the participant's screen 3 times. Finally, the experimenter asks whether the video is the one presented at the beginning or not. If the participant says yes, they proceed to the next motion event. Otherwise, the participant tries to produce more accurate description of the scene to be retrieved by improving the drawing and speech, and the retrieval process repeats.

3.6 Output and Annotation

Out of 18 participants, 7 participants were soccer fans, 6 were soccer players, 3 were soccer coaches and 2 were soccer referees. Altogether, the sessions took 661 minutes and 24 seconds. Average duration of a session was 36 minutes and 44 seconds. On all the sketched symbols and utterances collected from these sessions, the following

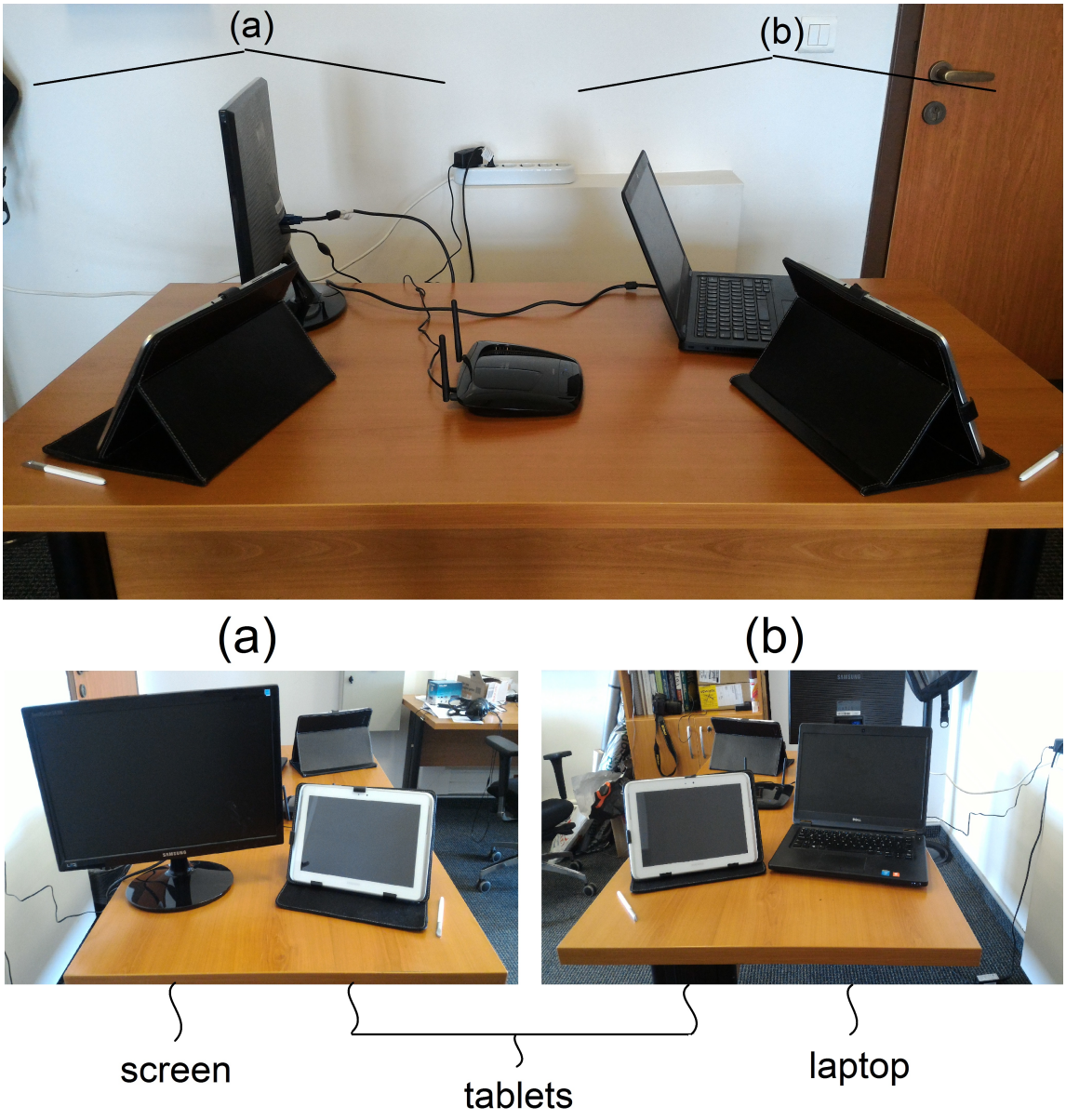


Figure 3.7: Physical set up; (a) for the participant, (b) for the experimenter

annotation methods were applied.

3.6.1 Sketched Symbol Annotation

We implemented an application to annotate individual sketched symbols [Altıok, 2017d]. This software basically takes a sketch stream file generated by a collector application as input. As shown in Figure 3.8, the right pane includes the stream. A user can select a part of the stream, and see the drawing generated by the selection on the left panel. Then she can save the drawing with a name, in the following formats:

- **Points table:** A list of *(x,y, time, strokenum)* where *(x,y)* refers to the coordinates of each sketch point, *time* refers to the time of generation of this point, and *strokenum* refers to the stroke ID the point belongs to. In each sketched symbol, stroke ID starts from 0 and is incremented by one when a new stroke starts.

- **Image:** An image of the drawing.

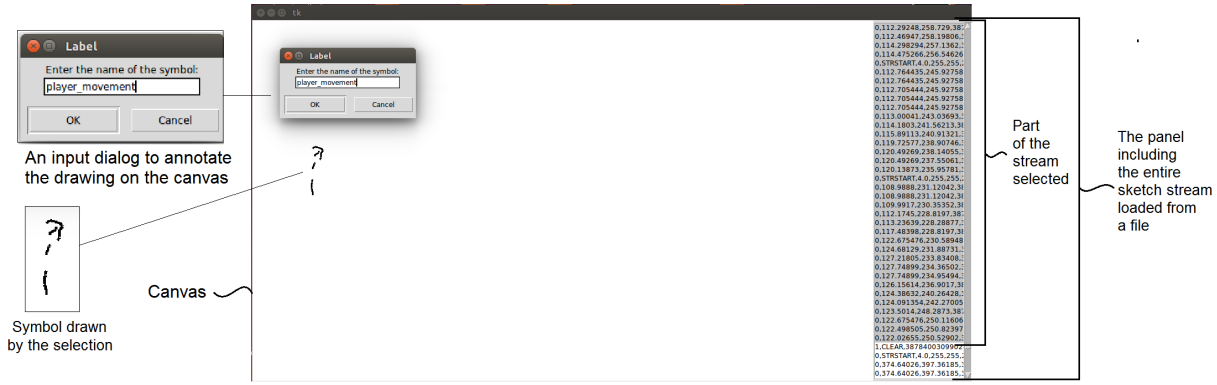


Figure 3.8: A screen-shot from sketched symbol annotation application with the explanations of UI components

Using this software, annotation of the sketched symbols was done in two ways:

Ball	88
Ball Motion	313
Player Motion	250
Player (side 1)	480
Player (side 2)	396
Total	**1527**

Table 3.2: Distribution of the sketched symbols among symbol types

West	283
East	181
Total	**464**

Table 3.3: Distribution of labeled motion arrow symbols among the directions

- **Annotation by symbol type:** We labeled symbols with one of the following classes: ball, ball motion arrow, player motion arrow, player from side 1 and player from side 2. The distribution of the symbols among such classes is presented in Table 3.2 together with some samples (see Figure 3.9). Note that the ball symbol was not frequently used, thus ball symbols were excluded from the training set of our sketched symbol classifier. Location of the ball can be clearly inferred from the starting point of a ball motion arrow.

- **Annotation by direction:** We also labeled the motion arrow symbols with their directions (heading to either the west or the east). Note that some motion symbols were excluded from the training set since they did not have an arrow head, which is an important indicator of direction. The distribution of the labeled motion symbols among the classes in this annotation method is presented in Table 3.3.

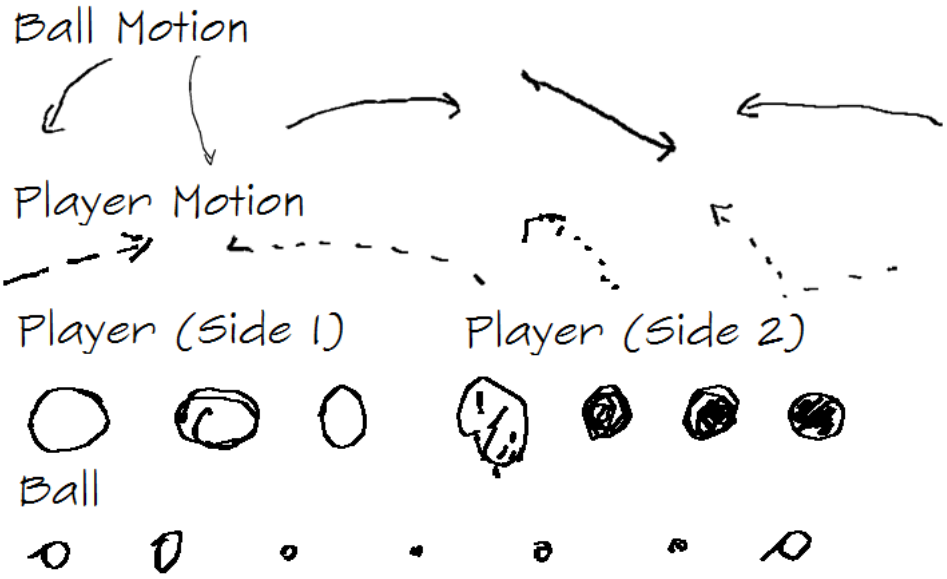


Figure 3.9: Sample symbols collected in the sessions

3.6.2 Utterance Annotation

We also collected 9296 words in the sessions. A summary of the words uttered is presented in Table 3.4. Some sample utterances are also shown in Table 3.5[2].

For each of the 183 phrases describing soccer drills, we generated n-grams with n ranging from 1 to 6. Next, each n-gram was labeled with one of the following motion events:

- **Bicycle**
- **Corner kick**
- **Dribbling**
- **Free kick:** All kicks far from or close to the goalpost are included in this category.
- **Header**

Words uttered	9296
Minimum words uttered in a session	233
Maximum words uttered in a session	925
Average number of words uttered in a session	516.4

Table 3.4: Summary of collected utterances in the sessions

Segment	**Event being described**
...the ball is coming to the post...	Shot hitting the post
...he is throwing the ball...	Throw in
...he is heading to the ball...	Header
...he is passing the ball in this way...	Pass
...this is an indirect free kick...	Indirect free kick

Table 3.5: Some sample utterances

Bicycle	198
Corner kick	406
Dribbling	2590
Free kick	687
Header	218
Pass	3694
Shot	4081
Tackle	2262
Throw-in	312
None	39663
Total	**54111**

Table 3.6: Summary of the n-grams generated from utterances

- **Pass**

- **Shot:** All of the shots heading to the goalpost, going wide, cleared by the goalkeeper and resulting in a goal are included in this category.

- **Tackle:** All of the sliding tackles, poke tackles and fouls are included in this category.

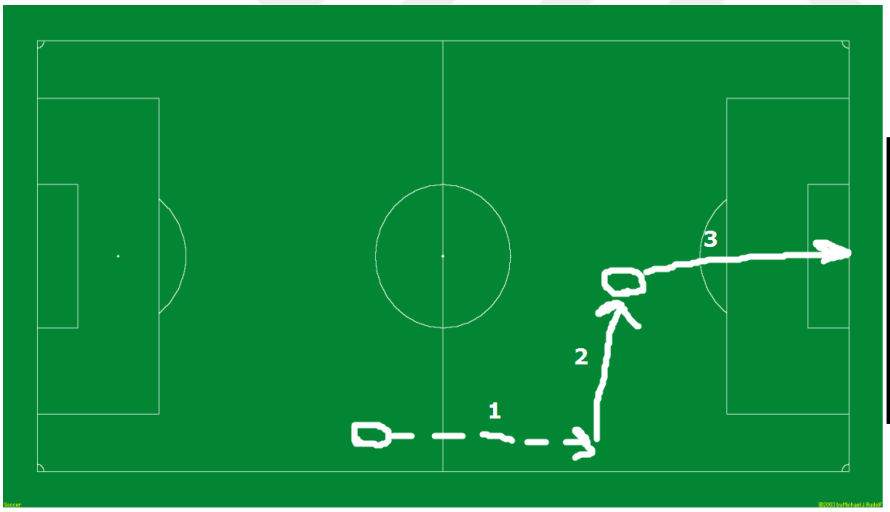
- **Throw-in**

- **None:** The n-grams implying more than an event or none of the events above are labeled with *None*.

Using this annotation method, we aimed to build a text-classification model that is capable of finding motion events described in the sub-segments of a long text segment. The distribution of n-grams among the motion events is also presented in Table 3.6.

3.6.3 Sketched Symbol - Utterance Mapping

To measure the performance of our multimodal interpretation framework, we also labeled all motion arrows drawn in the data collection sessions with its corresponding motion event described by a participant. At this point, some motion arrows were not labeled because of describing none of the motion events. For instance, a ball motion arrow which indicates a ball coming to a player but whose source of arrival is not certain could not be associated with any motion event. Through the strategy described above, we have gathered up 397 motion arrow - motion event mappings from 148 scene descriptions. A sample mapping is illustrated in Figure 3.10.



"this player came here and then passed the ball to his friend finally he scored"

Motion arrow - event mapping

Motion arrow #	Ball / Player motion?	Motion event
1	Player	Dribbling
2	Ball	Pass
3	Ball	Shot

Figure 3.10: A sample mapping between motion arrows and motion events

Notes to Chapter 3

1 A sample is available on-line at http://www.goal.com/tr/match/galatasaray-vs be relax

2 Note that the utterances were collected in Turkish.

Chapter 4

MULTIMODAL INTERPRETATION

Using the recent advancements on sketched symbol classification and natural language understanding, we have developed a multimodal interpretation framework that is able to understand the sequence of motion events from simultaneously given sketching and speech inputs. To come up with the most accurate framework, we have run several experiments on different sketched symbol and text classification models. Next, we have come up with some different co-interpretation algorithms having different combinations of sketched symbol and text classifiers. These co-interpretation models have exhibited notable performance on the data collected from participants. This implies that our multimodal interpretation algorithm is convenient with the data collected through our software tools and protocol for building bimodal and natural interfaces for expressing motion.

4.1 Sketched Symbol Classification

Although the state-of-the-art sketched symbol classifiers leverage neural networks [Seddati et al., 2016], such heuristics do not seem to give the best results on our symbol dataset since the amount does not seem sufficient (see Figure 3.2) for training. For small datasets, the best results have been obtained through support vector machines with the RBF kernel trained on the IDM feature space [Ouyang and Davis, 2009, Yesilbek et al., 2015, Tumen et al., 2010]. Thus, we evaluated only the SVM-RBF classification combined with the IDM feature extraction pipeline. The assessment was done using grid search with 5-fold cross validation on the (C,γ) SVM-RBF hyperparameter pairs such that $-1 \leq \log_2 C \leq 10$ and $-10 \leq \log_2 \gamma \leq 1$. The distribution of accuracy rates among the symbol classes where the overall accuracy rate gets the

maximum is presented in Figure 4.1. As seen in the figure, the accuracy rates for every class is either close to or above 90%. This means that the SVM classification method with the IDM feature extraction mechanism is plentiful for figuring out individual sketched symbols expressing motion.

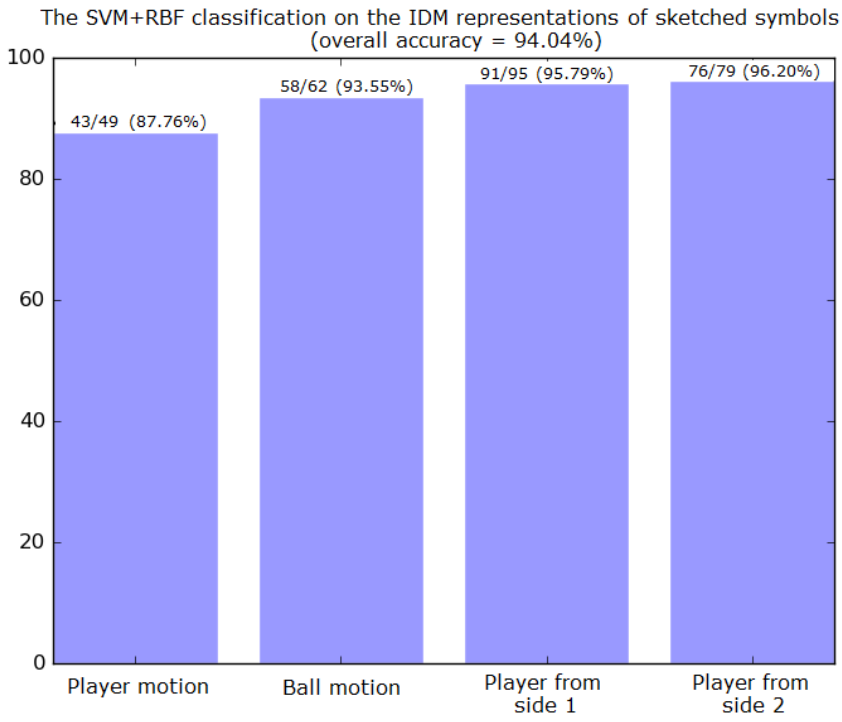


Figure 4.1: Classification accuracies of the SVM-RBF + IDM classifier / feature extractor combination

4.2 Natural Language Understanding

A speech input is basically a long text where a sequence of motion events is described. To find out the individual motion events mentioned in such texts, a text classifier must be trained in a way that it classifies sub parts as accurately as possible. To this end, we created a data set of n-grams from the transcripts collected from the participants. Because of the irregular nature of the spoken language, these n-grams

are not grammatically correct and thus called **short texts**. Then, we built text classification models by leveraging the recent short text classification frameworks. In our work, we tested the some variants of the neural network-based model proposed by [dos Santos and Gatti, 2014] and [Yu et al., 2013]'s SVM and logistic regression-based model (all implemented in a library called **LibShortText**) listed below. Note that despite not being solely designed for short texts, FastText [Joulin et al., 2016] was also included in the experiments because of its considerably short runtime at both classification and training.

- **Unigram + SVM:** Support vector machine classification on the unigram feature domain [Yu et al., 2013].
- **Bigram + SVM:** Support vector machine classification on the bigram feature domain [Yu et al., 2013].
- **Unigram + LR:** Logistic regression classification on the unigram feature domain [Yu et al., 2013].
- **Bigram + LR:** Logistic regression classification on the bigram feature domain [Yu et al., 2013].
- **CHARSCNN:** A long-short-term-memory (LSTM) based deep neural network constructing mathematical representations of texts by learning word embeddings at both word and character levels [dos Santos and Gatti, 2014].
- **CHARSCNN (word embeddings only):** A variant of the CHARSCNN model such that the mathematical representations of texts are constructed by learning only word-based word embeddings.
- **CHARSCNN (character embeddings only):** A variant of the CHARSCNN model such that the mathematical representations of texts are constructed by learning only character-based word embeddings.

- **Unigram + FastText:** A FastText classifier trained on the unigram features [Joulin et al., 2016].

- **Bigram + FastText:** A FastText classifier trained on the bigram features [Joulin et al., 2016].

To assess the classification performance of all these variants, the n-grams are randomly divided into 5 chunks, using 4 for training and 1 for testing. Hyper-parameter search for the SVM and LR-based classifiers was done internally by the LibShortText library [Yu et al., 2013], whilst the default parameters were used in the CHARSCNN and FastText models. Figures 4.2, 4.3, and 4.4 show the F1, precision and recall rates of all frameworks respectively.

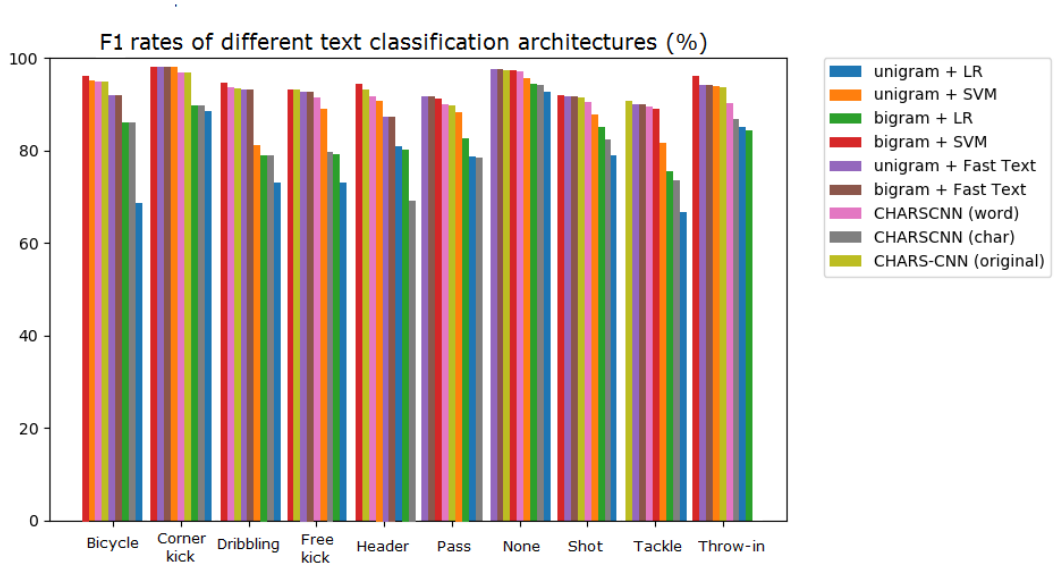


Figure 4.2: F1 rates of different text classification architectures

As seen in the figures, all the rates are above 65%, meaning that a sufficient infrastructure on understanding motion events from short texts seems to have been established. Additionally, although support vector machines had a remarkable performance on the data set, using these traditional models will not be a proper design

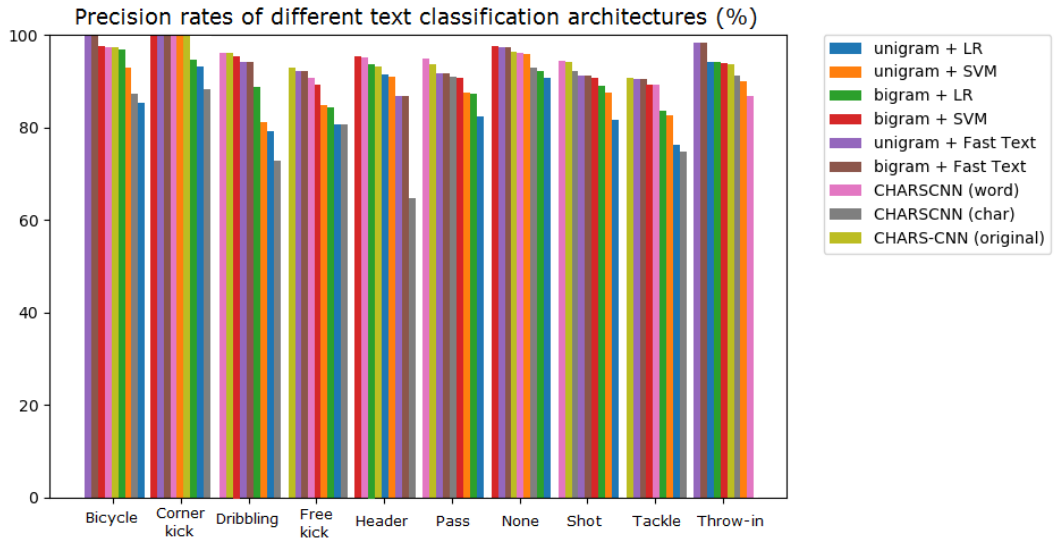


Figure 4.3: Precision rates of different text classification architectures

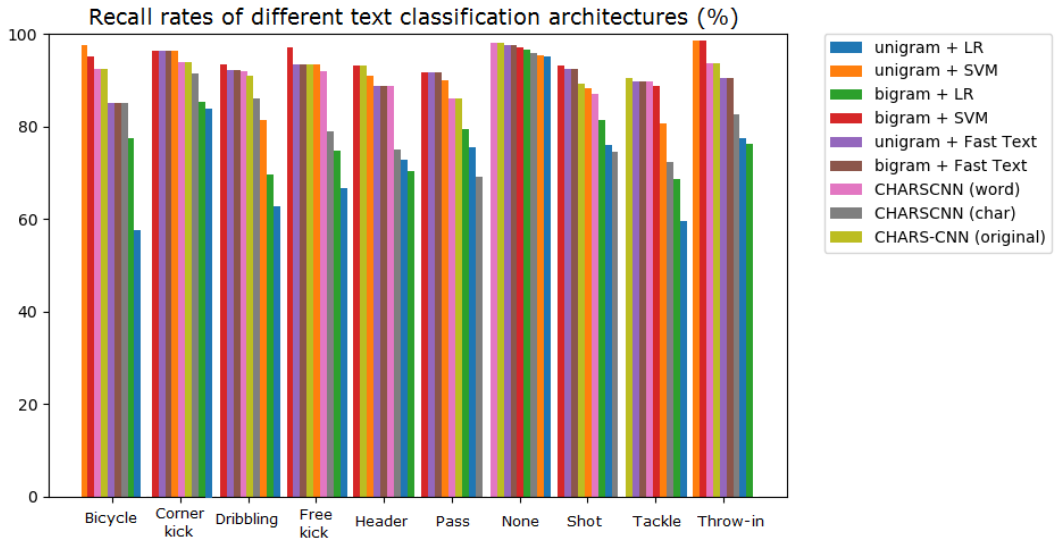


Figure 4.4: Recall rates of different text classification architectures

decision for building a multimodal video retrieval system in real life. This is because such models benefit from word frequencies to compute features, leading to misclas-

sification of the sentences with words not included in the training set. Considering all these reasons, we decided to use the FastText model on bigram features that had reached the highest F1 rates. All the F1 rates are above 95%, and it was expected to find out the motion events in a transcript most accurately during the user evaluation studies of the video retrieval system.

4.3 Co-Interpretation of Sketching and Speech

Using the sketched symbol and short text classifiers trained, we have designed a heuristic algorithm that is able to match the sketched motion symbols to the sequence of motion events found in a text. This algorithm gives the ability to convert multimodal input into plain text that will be used later to find the soccer match segments having the motions described. A sample matching is illustrated in Figure 4.5.

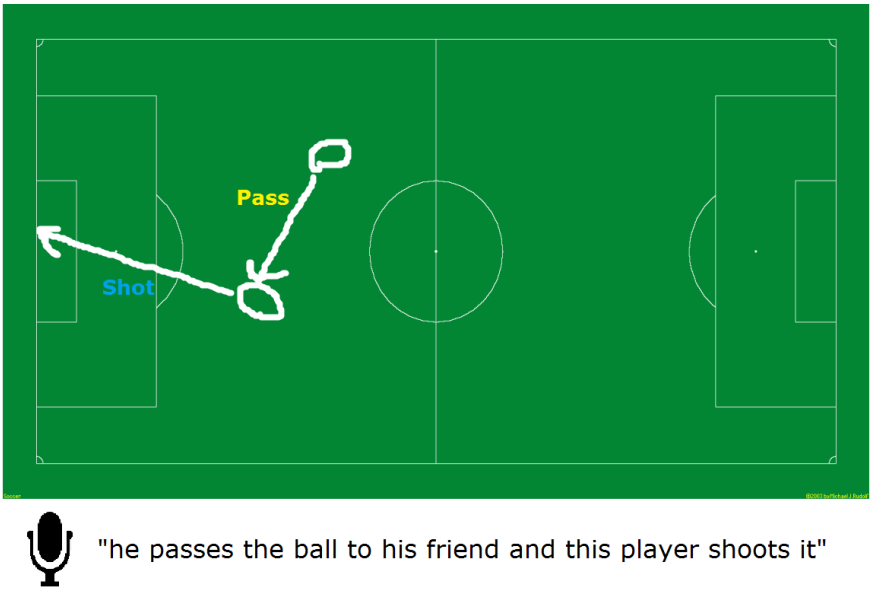


Figure 4.5: An illustration of sketched symbol - motion event matching

The multimodal interpretation algorithm consists of 2 steps. In the first phase, the transcribed speech input is split into short texts using the text classifier. Since the text classifier was trained on the n-grams with n ranging from 1 to 6, lengths of

the resulting segments must be between 1 and 6 words (both inclusive). The optimal segmentation must have the maximum segmentation log-probability, which is the sum of the maximum classification log-probabilities of all individual short text segments. This problem was modeled to be a dynamic programming problem as follows:

$f(.)$: The function returning the maximum log-probability of a short text segment

$T[i]$: The maximum sum of classification log-probabilities of short text segments when segmentation is performed on the first i word

W : The array of words in a transcribed text

$T[i] = max_{j=1,2,...min(6,i)}\{T[i-j] + f(W[(i-j+1):i])\}$

$T[n]$: The value to maximize

The input of the second step is the array of motion event classes with the highest probability. Initially, the *None* entries of the array, which do not express any motion event, are excluded. Next, the remaining entries are distributed among the sequence of motion arrow symbols drawn through another dynamic programming problem formulated as follows:

E : The array of motion events (excluding *None*s)

Ball motion events = [Header, corner kick, pass, bicycle, free kick, shot, throw-in]

Player motion events = [Tackle, dribbling]

$b(.)$: number of ball motion events in an array

$p(.)$: number of player motion events in an array

$$r(arrow, E) = \begin{cases} -b(E) & \text{if } p(E) = 0 \text{ and the arrow indicates a player motion} \\ 1 - b(E) & \text{if } p(E) > 0 \text{ and the arrow indicates a player motion} \\ -p(E) & \text{if } b(E) = 0 \text{ and the arrow indicates a ball motion} \\ 1 - p(E) & \text{if } b(E) > 0 \text{ and the arrow indicates a ball motion} \end{cases} \tag{4.1}$$

K : The array of sketched motion arrow symbol classes (sorted by drawing order, from the first to last)

$$T[i,j] = \begin{cases} max_{p=0,1,\ldots,i-1}\{T[i-p, j-1] + r(K[j], E[(i-p+1):i])\} & \text{if } j > 1 \\ r(K[j], E[1:i]) & \text{if } j = 1 \end{cases} \quad (4.2)$$

n : number of items in the E array

m : number of items in the K array

$T[n, m]$: The value to maximize

To explain further, this dynamic programming model tries to distribute the motion events detected in a transcribed text among the motion arrow symbols drawn by a user through preserving the temporal order of both events and drawings. There are 2 kinds of award functions, one for matching a ball motion arrow, and another for matching a player motion arrow. Each function yields 1 point if there is at least a motion event relevant to a motion arrow symbol. However, if there is at least a motion event irrelevant to an arrow symbol, it cuts down by 1 point for each irrelevant motion event. This leads to a homogeneous distribution of all motion event entries by preserving the relevance between motion arrows and events. Note that depending on the speech and sketch inputs, there might be some motion arrows with no motion events assigned.

This 2-step matching algorithm has been inspired by [Hse et al., 2004]'s work on segmenting sketched symbols. His work basically splits sketched symbols into primitive tokens such as arcs and lines. The segmentation process is based on a dynamic programming model where segments are fitted to either an arc or a line.

We evaluated our matching algorithm on the data set of sketched symbol - motion event mappings. Since the algorithm can assign multiple motion events to a single motion arrow symbol, *a motion event - arrow match* is said to be correct if *the correct motion event is among the ones matched to the arrow.* The matching algorithm was evaluated through different text classification models (CHARSCNN and FastText with bigram features). The evaluation results are presented in tables 4.1, 4.2 and 4.3.

As seen in the tables, the FastText-based segmentation is more accurate than the CHARSCNN-based segmentation. This is because FastText is more accurate than

Criteria	**Value obtained using CHARSCNN**	**Value obtained using FastText**
Average accuracy rate in a search task (number of correctly matched motion arrows in a search task / number of all motion arrows in a search task)	68.88% ± 33.61%	75.38% ± 30.16%
Overall accuracy rate (number of all correctly matched motion arrows drawn in all search tasks / number of all motion arrows drawn in all search tasks)	63.48% (252/397)	69.02% (274/397)

Table 4.1: Accuracy rates of the multimodal interpretation algorithm

CHARSCNN on short text classification. Moreover, there are high accuracy rates on the motion events with many examples such as pass, tackle and free kick. However, this is not the case for the dribbling class since the variation of the words within this class is more than the other classes. It is thought that having more keywords but less samples in a motion event might lessen the accuracy rate. Furthermore, as seen in Table 4.3, our multimodal interpretation framework overperforms the random matcher. Since there are 8 distinct ways of matching to a ball motion (7 ball motion events + matching none of the events), a random matcher is 12.5% accurate. In the case of player motions, there are 3 different ways of matching (2 player motion events + matching none of the events) and thus the accuracy is 33.3%. Within this context,

Motion event	**Accuracy (using CHARSCNN)**	**Accuracy (using FastText)**
Header	40% (2/5)	60% (3/5)
Corner kick	87.5% (7/8)	87.5% (7/8)
Pass	78.57% (77/98)	84.69% (83/98)
Bicycle	40% (2/5)	20% (1/5)
Free kick	90.91% (10/11)	90.91% (10/11)
Shot	73.44% (47/64)	96.88% (62/64)
Throw in	60% (6/10)	70% (7/10)
Tackle	73.68% (28/38)	86.84% (33/38)
Dribbling	46.2% (73/158)	43.04% (68/158)

Table 4.2: Distribution of accuracy rates of the multimodal interpretation algorithm among the motion events

our 2-step multimodal interpretation heuristic seems applicable for use in our final multimodal video retrieval system. For its higher accuracy, FastText has been chosen to be the text classification model in the video retrieval system.

4.4 User Interaction Mechanism

Based on the sketched symbol classifier, text classifier and the multimodal interpretation model, the user interaction mechanism of the video retrieval system is planned as shown in Figure 4.6. As seen in the figure, the video retrieval system has been designed to take speech and sketching simultaneously. Users are allowed to draw through a stylus pen. To distinguish individual symbols, users must stop drawing for a short while once a symbol is complete. If a user stops drawing for more than a second, coordinates of the points inside the recent symbol are sent to the sketched symbol classifier. Depending on the classification result, the recent symbol is beautified. On the speech side, the speech input is transcribed through Google's speech recognition

Motion arrow type	**Accuracy (random matcher)**	**Accuracy (using CHARSCNN)**	**Accuracy (using FastText)**
Ball motion	12.5% (1/8)	75.12% (151/201)	86.06% (173/201)
Player motion	33.3% (1/3)	51.5% (101/196)	51.5% (101/196)

Table 4.3: Distribution of accuracy rates of the multimodal interpretation algorithm among the motion arrow types

API [Google, 2017]. To overcome mistranscriptions due to network breakages, users are also allowed to type whatever they said directly into the system through keyboard. Finally, once a user finishes describing a scene, the transcribed text and the sketched symbol classification results are evaluated altogether by the multimodal interpretation algorithm.

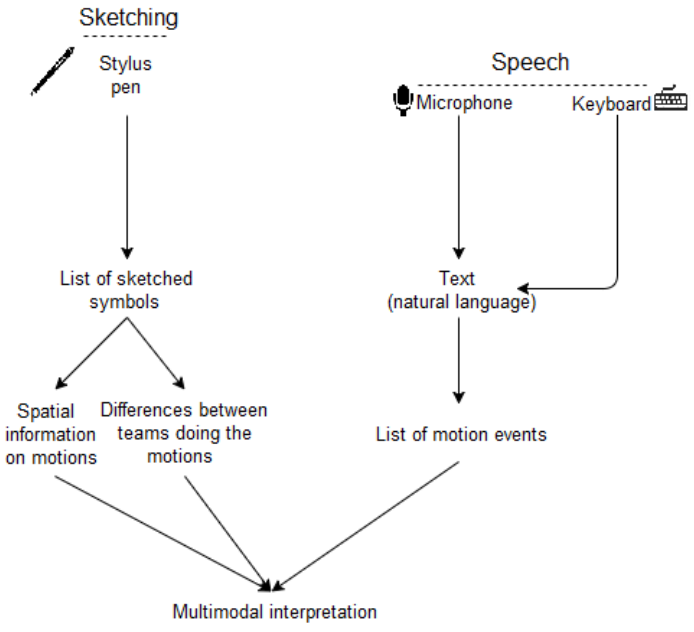


Figure 4.6: The user interaction mechanism of the video retrieval system

4.5 Search Query Generation

The multimodal input must be converted into a plain text search query to perform retrieval on the database back end. There are 3 steps of generating back end search queries. As the first step, the end points of the motion arrows drawn in the canvas are detected. The end points of a motion arrow are assumed to be the furthest points in the arrow [Hammond and Davis, 2006b]. These end points are then marked as starting and ending points depending on the orientation of arrows. If an arrow is heading to the west, the endpoint with the smaller x coordinate is marked as the starting point and the other is marked as the end point. The situation gets reversed if it is heading to the east. To find out the orientation, we also trained a binary SVM classifier on the IDM feature domain. The 5-fold cross validation results of the model are depicted in Figure 4.7[1]. As seen in the figure, the classifier seems accurate to figure out the orientation of an arrow if its head is drawn clearly.

As the second step, each motion arrow is associated with a player symbol. Our system does not care about the team names but their differences. The player doing a motion is assumed to be the closest one to the arrow within 40 pixels radius. The teams are distinguished by being filled of player symbols. Moreover, if a motion arrow is associated with a player, its starting point is updated to the center of the player associated[2]. Furthermore, if there is another arrow within 40 pixels radius of the ending point of an arrow, the teams doing these motions are assumed to be the same.

As the last step, the multimodal interpretation algorithm is executed. For each arrow, the end points, the team information and its motion events are packed into a plain text query. Figure 4.8 illustrates the process of generating a plain text search query from multimodal input.

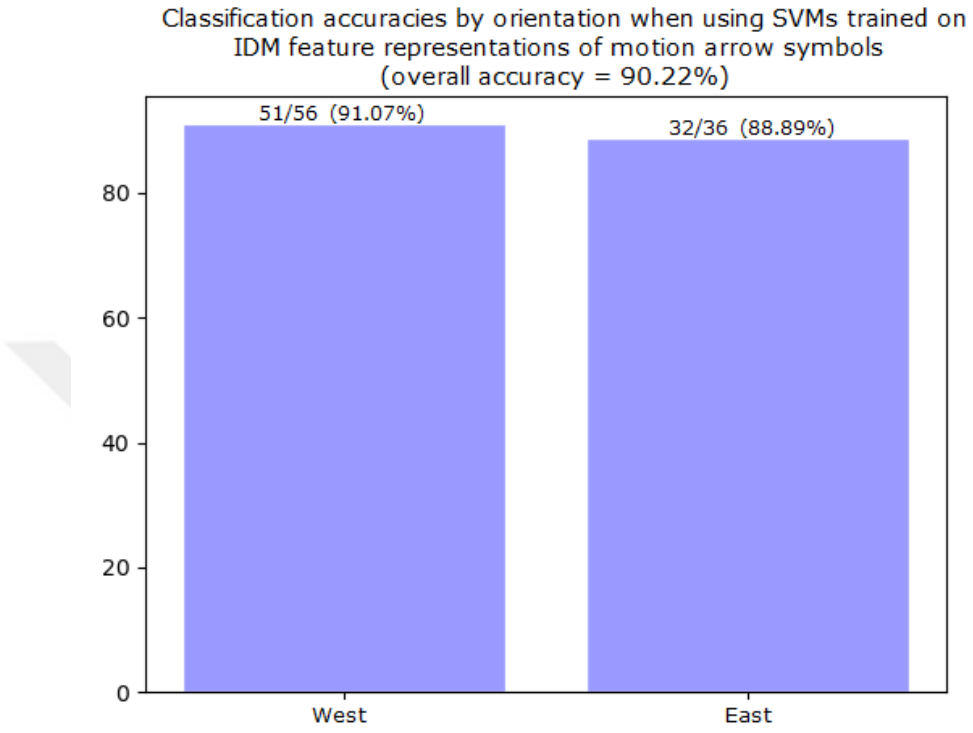


Figure 4.7: Classification accuracies of the SVM-RBF + IDM classifier / feature extractor combination for finding out the orientation of a motion arrow symbol

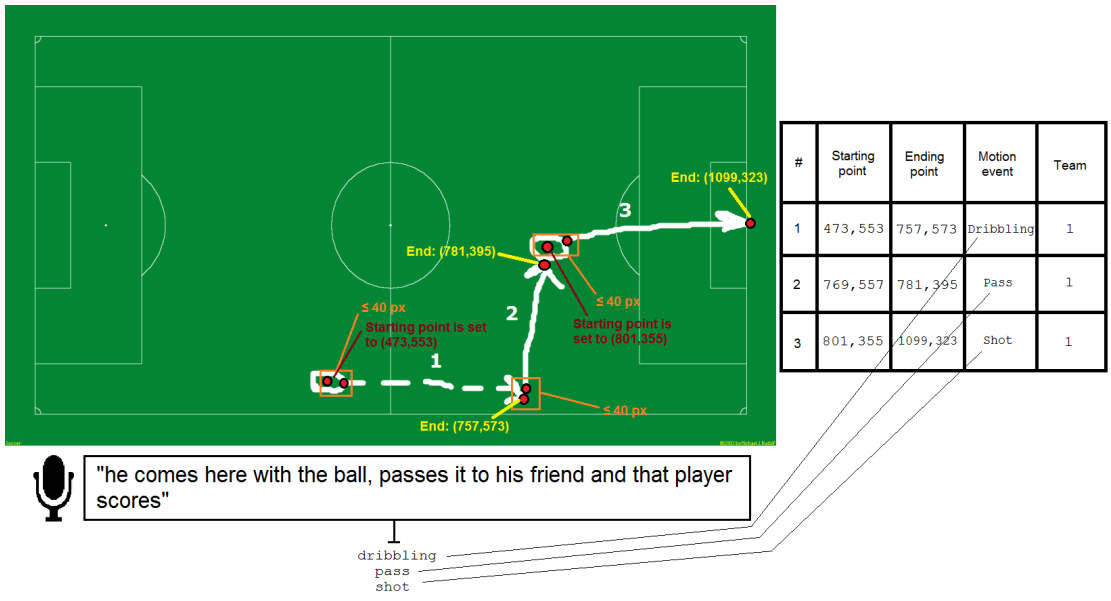


Figure 4.8: An illustration of the database query generation

Notes to Chapter 4

1 Note that the hyperparameter ranges used in the grid search are as the same as the ones used for the symbol classification.

2 The center of a player symbol is the average of all points (center of mass) included in the symbol.

Chapter 5

DATABASE SYSTEM

To enable motion based retrieval using plain text queries generated from multi-modal inputs, a database system has been designed and some retrieval algorithms have been implemented. The following subsections give a brief overview on the database system design and the retrieval algorithms that comply with the plain text search query format.

5.1 Database Design

Our database system treats every single motion event (header, corner kick, pass, bicycle, free kick, throw-in, tackle, shot and dribbling) as a primitive motion scenario. All motion scenarios described by a user are considered to be an aggregation of the primitive motion scenarios. Following this approach, we designed the database to store and index a set of primitive soccer match cutouts each of which is represented by a feature vector $(id, fs, fe, sx, sy, ex, ey, t, e)$. All features are explained below.

- id: A unique number created for every soccer match. This prevents compiling primitive motion scenarios from different matches into the same video clip.
- fs: The frame number where the cutout starts
- fe: The frame number where the cutout ends
- sx: The x coordinate of the point where the primitive motion starts
- sy: The y coordinate of the point where the primitive motion starts

- ex: The x coordinate of the point where the primitive motion ends
- ey: The y coordinate of the point where the primitive motion ends
- t: The team of the player involved in the primitive motion scenario (gets either 1 or 2)
- e: The event name of the primitive motion scenario (either of header, corner kick, pass, bicycle, free kick, throw-in, tackle, shot and dribbling)

We recorded 5 soccer matches between the AI-controlled teams from the Eat the Whistle game [Greco, 2008]. For each match, we extracted primitive motion events and stored their feature representations in ADAMpro, which is a scalable database utility for big multimedia collections [Giangreco and Schuldt, 2016]. To extract features of each primitive motion cutout, a state machine has been modeled and implemented. All state machines are explained below. Illustrations of the complex ones are also provided.

- **Header:** All the subsequent frames from one where a player is heading the ball, to the latest one where the ball is free and has not been picked up by someone else are considered to be about a header event (see Figure 5.1).
- **Corner kick:** All the subsequent frames from one where a corner kick is set, to the latest one where the ball is free and has not been picked up by someone else are considered to be about a corner kick event.
- **Pass:** If the team having the ball does not change but the player does, all the subsequent frames from the one where the ball gets free to the one where the ball is picked up again are considered to be about a pass event (see Figure 5.2).
- **Bicycle:** All the subsequent frames from one where a player is doing a bicycle kick, to the latest one where the ball is free and has not been picked up by someone else are considered to be about a bicycle event.

- **Free kick:** All the subsequent frames from one where a free kick is set, to the latest one where the ball is free and has not been picked up by someone else are considered to be about a free kick event.

- **Shot:** If the ball is picked up by the opponent's goalkeeper or it goes wide, all the subsequent frames from the one where the ball gets free to the one where it goes wide or caught by the goalkeeper are considered to be about a shot event.

- **Throw-in:** All the subsequent frames from one where a throw-in is set, to the latest one where the ball is free and has not been picked up by someone else are considered to be about a throw-in event.

- **Tackle:** All the subsequent frames from one where a player tries to steal the ball to the latest one where the ball is free and has not been picked up by someone else are considered to be about a tackle event.

- **Dribbling:** All the subsequent frames from one where the ball is picked up by a player to the one where he loses it are considered to be about a dribbling event (see Figure 5.3).

Moreover, regardless of belonging to a primitive motion event, all match frames are saved into a disk in order to prevent disjointedness among the video clips of primitive motions. If two successive primitive events include some other frames in between, these frames are also included in the video clip that will be presented in the search results.

5.2 Retrieval Framework

The ADAMpro utility supports two modes of retrieval:

- **Boolean retrieval:** Retrieval based on exact matching between a query and a document

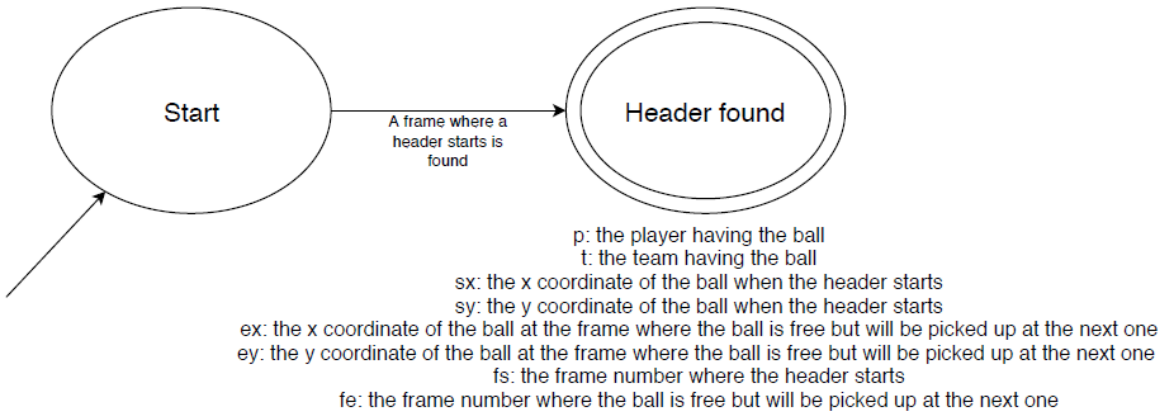


Figure 5.1: An illustration of the state machine finding out the headers in the ball motion record of a soccer match

- **Vector space retrieval:** Retrieval based on the similarity between a query and a document. The less the distance between the feature vectors of two items is, the more similar they are.

Our retrieval framework exploits these two methods to bring video clips having the motions described. Firstly, for each motion entry in a plain text query, all primitive video clips having one of the motion events listed in the entry are queried through boolean retrieval (see Algorithm 1). Next, on the video clips found in the previous step, a vector space retrieval is performed by using the position information as the query vector. Finally, starting from the first entry, for each cutout found in a motion entry, the closest cutout of the next motion entry happening within the first 250 frames from the first frame is sought[1]. If there is such a cutout, this cutout is marked as the cutout of the next motion entry, and the seeking process repeats between the next two motion entries. Otherwise the seeking process is terminated. If the chain of seeking processes happen till the last motion entry, then the frames between the first frame of the first entry's cutout and the last frame of the last entry's cutout are compiled into a video clip to be presented as a search result. Note that the results are presented through sorting them by their similarity scores in descending order (see

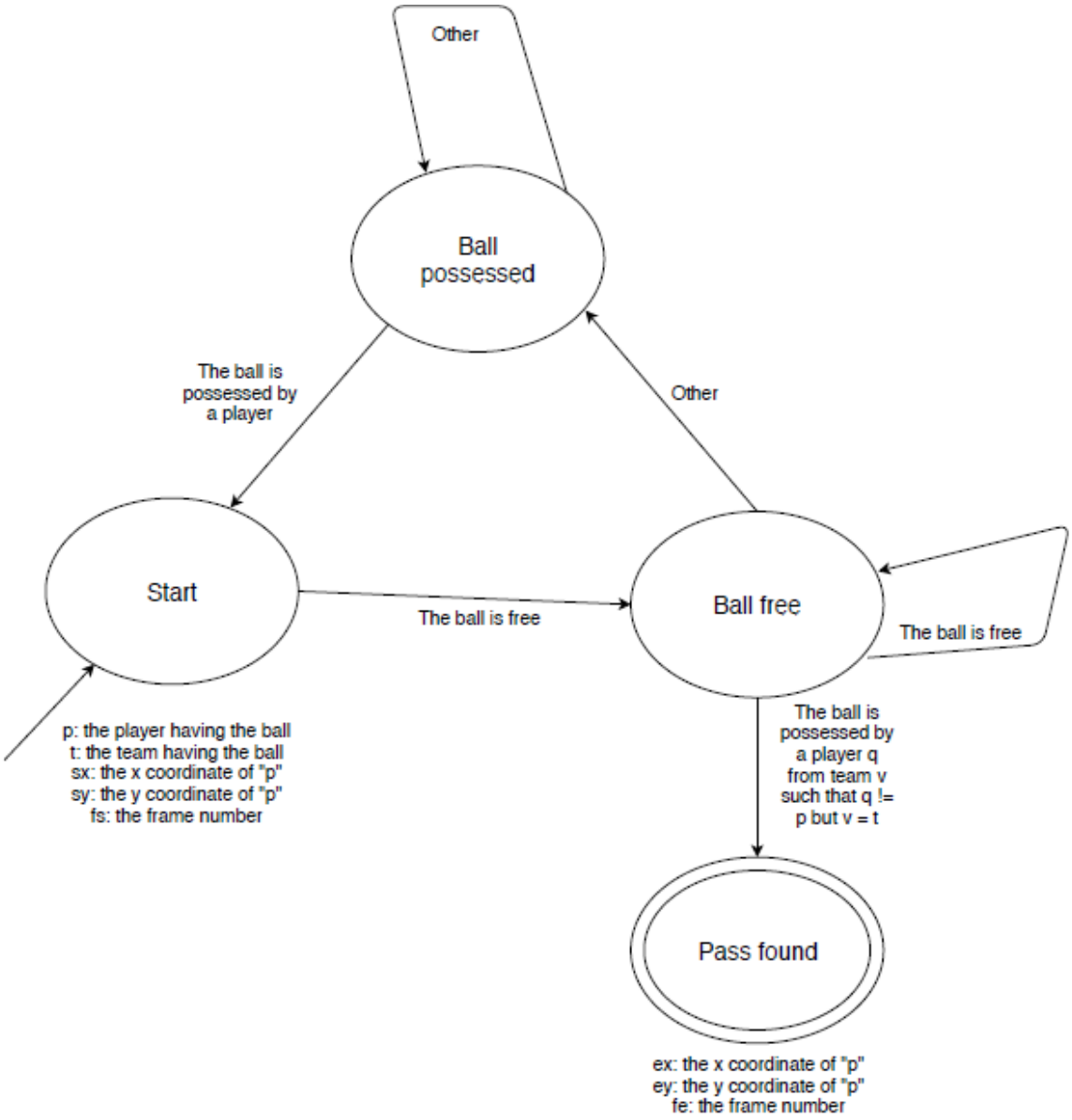


Figure 5.2: An illustration of the state machine finding out the passes in the motion record of a soccer match

Algorithm 2). The similarity score of a video clip is defined as the sum of the squared Euclidean distances of the cutouts included[2].

Algorithm 1: An algorithm that finds primitive video cutouts for each motion entry in a search query

Data: K (an array of motion event entries in a search query)

Result: results (an array of search results for each motion entry in the search query)

$results \leftarrow []$

$max\_number\_of\_results \leftarrow 10$

for *each entry in K* **do**

$boolean\_query \leftarrow e == entry.motion\_events[0]$

$possible\_event\_count \leftarrow entry.motion\_events.length$

for *each n in* $[1, 2, ..., possible\_event\_count - 1]$ **do**

$boolean\_query \leftarrow boolean\_query || e == entry.motion\_events[n]$

end

$query\_vector \leftarrow < query.sx, query.sy, query.ex, query.ey >$

results.append(database.vector_space_retrieval($query\_vector$, $max\_number\_of\_results$, database.boolean_retrieval($boolean\_query$)))

end

Algorithm 2: An algorithm yielding search results as video clips

```
Data: results (an array of search results for each motion entry in the search query)
Result: S (an array of video clips with their similarity scores)
max_frame_count ← 250
motion_query_count ← results.length
first_motion_entry_results ← results[0]
first_motion_entry_event_count ← first_motion_entry_results.length
S ← []
for each n in [0, 1, ..., first_motion_entry_event_count − 1] do
    first_event_clip ← first_motion_entry_results[n]
    first_motion_event_frame_range ← first_event_clip.frame_range
    frame_range ← first_motion_event_frame_range
    similarity_score ← first_event_clip.similarity_score
    for each m in [1, 2, ..., motion_query_count − 1] do
        next_results ← results[m]
        next_results_count ← next_results.count
        min_frame_diff ← max_frame_count
        closest_clip ← None
        last_frame_range ← None
        for each p in [0,1,...,next_results_count-1] do
            frame_diff ← next_results[p].frame_range.start
            if frame_diff ≤ min_frame_diff and next_results[p].id = first_event_clip.id then
                min_frame_count ← frame_diff
                closest_clip ← next_results[p]
                last_frame_range ← next_results[p].frame_range
                similarity_score_for_closest_clip ← closest_clip.similarity_score
            end
        end
        if closest_clip = None then
            break
        end
        else
            frame_range ← frame_range ∪ last_frame_range
        end
        if n = first_motion_event_entry_count − 1 then
            S.add((frame_range, similarity_score))
        end
    end
    // Sort S by the first column (their similarity scores) in descending order
    S.sort(column_number = 1, 'descending')
end
```

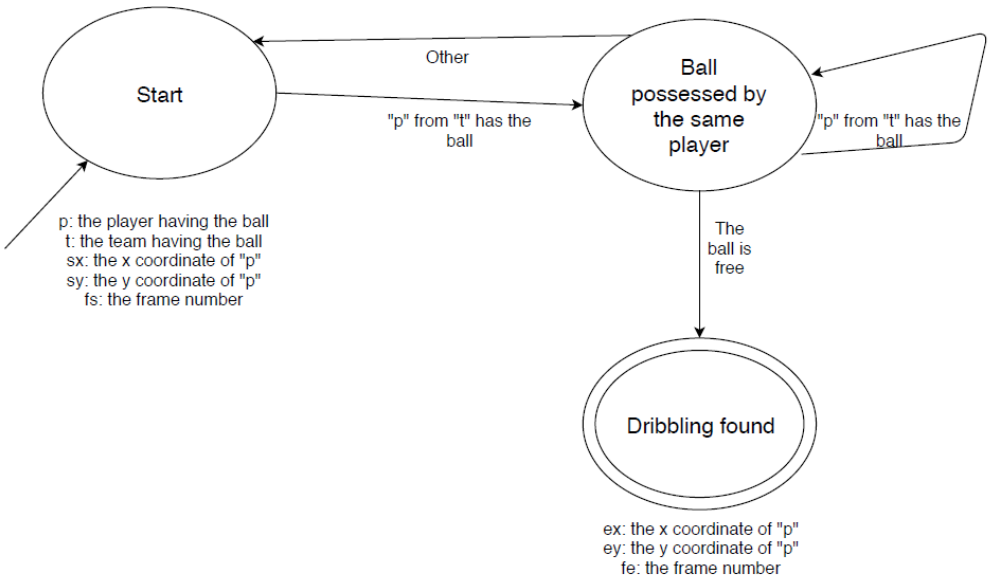


Figure 5.3: An illustration of the state machine finding out the dribbling events in the motion record of a soccer match

Notes to Chapter 5

1 Note that the candidate cutouts are from the same soccer match.

2 The squared Euclidean distances of the cutouts are computed by ADAMpro.

Chapter 6

USER EVALUATION STUDIES

A video retrieval system has been built by integrating the multimodal input evaluation framework into the database back end[1]. More details on the system's UI components, the protocol followed in the user evaluation studies with some insights on the results are provided in the following subsections.

6.1 User Interface Design

The user interface components of the video retrieval system are shown in Figure 6.1. As seen in the figure, the UI consists of two main columns. The left column includes a text input field where a speech input is transcribed and displayed. Users can change its content through keyboard in case of having a network breakage on Google's speech recognition system [Google, 2017], or having a mistranscribed text. At the bottom of the text field, there is a drawing canvas with a soccer field background image. The buttons just below the canvas can clear the canvas, remove the recently drawn symbol, starts the multimodal interpretation and kicks a search. The bottommost section on the left panel is a video display. The right panel includes search results as thumbnails. When a user clicks on a thumbnail, the corresponding video clip is played at the video display. Note that whenever the compilation of a video clip is done, the clip is sent to the right panel without waiting for the other clips to be compiled. Thus users can watch the video clips that have been already compiled while the search is in progress, enabling a better user experience than presenting all the clips in one go. To re-play the recently watched video, users can click on the blue button below the display. During the user studies, whenever they think that they have found the correct video, they can click on the green button to evaluate the correctness of the

recently watched video.

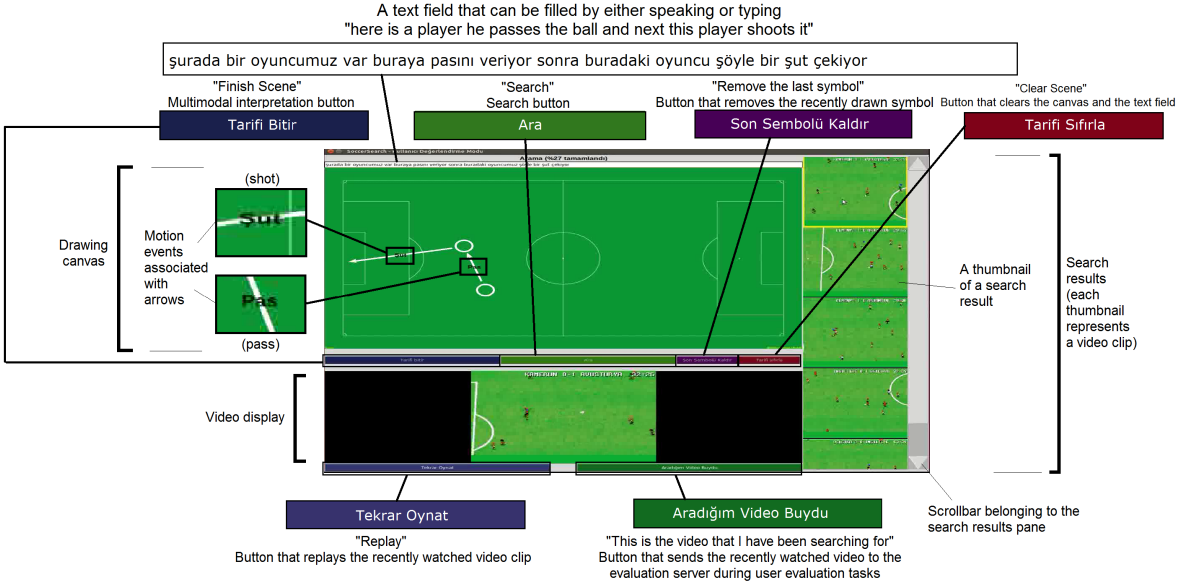


Figure 6.1: The UI components of the video retrieval system

Moreover, to enhance the user experience while drawing, a symbol is beautified when a user stops drawing it. As illustrated in Figure 6.2, if she stops drawing a symbol for more than a second, the symbol is sent to the sketched symbol classifier and beautified depending on the classification result. In this way users would think that the system is interacting with them. If the symbol is classified as a player, the system draws a circle with 40 pixels of radius. Otherwise, the system detects the endpoints of the symbol, and draws an arrow (either dashed or solid).

To illustrate the joint interpretation of sketching and speech on the drawing canvas, each motion event is written around the center of mass of the arrow the event is assigned to (see Figure 6.3). If a user thinks that a motion event is not assigned to an arrow correctly, then she can click on the motion events listed on the arrow, and add the desired event(s) through marking them on a pop-up window.

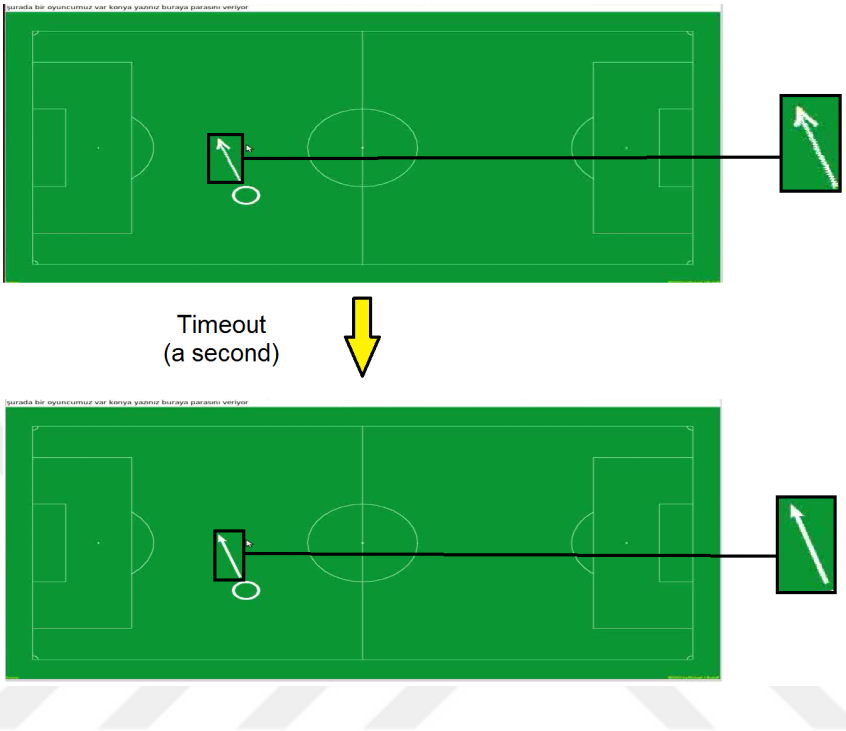


Figure 6.2: A sample symbol beautification in the video retrieval system

6.2 Method

The user evaluation studies take place around a table shown in Figure 6.4. Initially, the participant sits in front of the computer located at the right half of the table. Next, the experimenter gives a brief overview of the study through a presentation on the computer. After the presentation, the experimenter plays two short video clips, and asks the participant to draw the motions of the ball and players in these clips on a paper having a soccer field background image using the symbols given in the visual vocabulary (see Figure 6.5). Then the experimenter reviews the symbols drawn. If there is a misdrawn symbol, he asks the participant to re-draw it.

Once the training on sketched symbols is complete, the experimenter limits access to the visual vocabulary, and the search tasks begin. The participants sits on the left side of the table. There are 5 search tasks, a warm-up task and four actual tasks. The participant's interactions with the search engine is not included during the warm-up

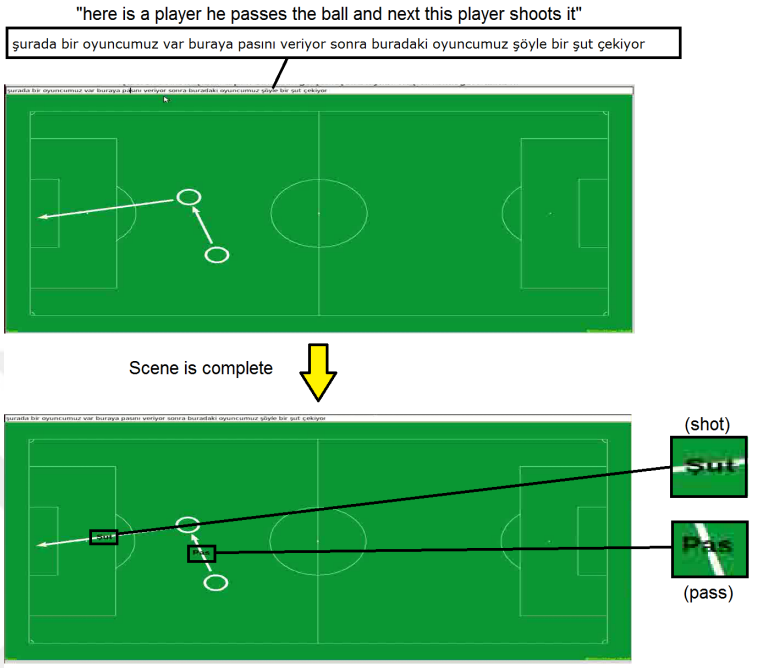


Figure 6.3: A sample illustration of multimodal interpretation on the drawing canvas

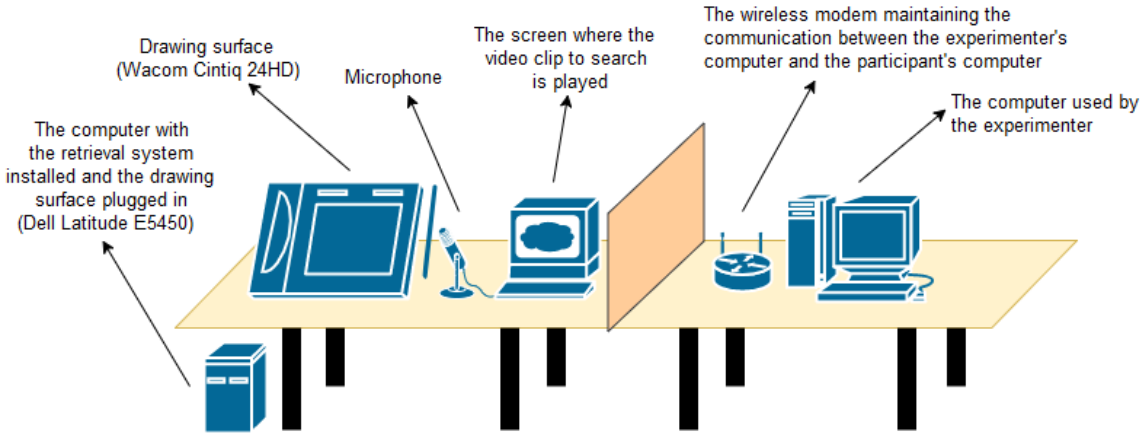


Figure 6.4: The physical set-up of the user evaluation studies

task. For each search task, a cutout from one of the 5 soccer matches indexed is randomly selected and played on the screen[2]. Next, the participant tries to find this video clip through our video retrieval system in 8 minutes. Within this time period,

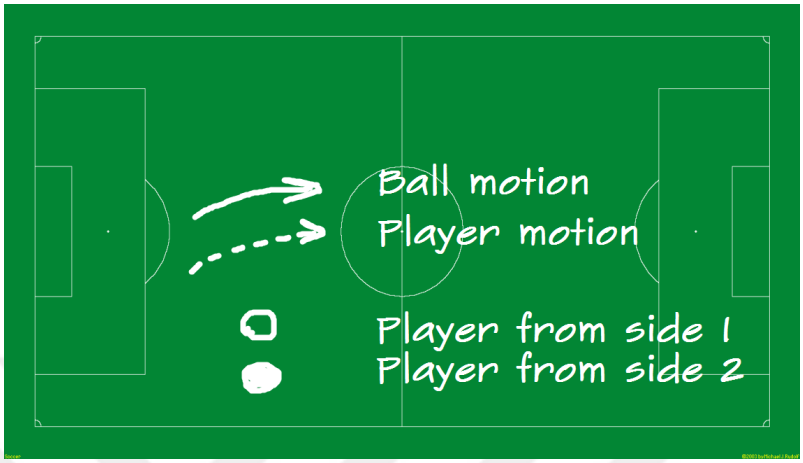


Figure 6.5: The visual vocabulary used during the user evaluation studies

the participant is allowed to do as many video searches as she would like. Whenever she thinks that she has found the correct video, she clicks on the green button below the video display of the retrieval system[3]. Then, the computer located at the right half of the table evaluates if the video clip sent by the participant is correct. If the video clip is correct[4], then the search task successfully ends. Otherwise, the search task continues until the time is over. If the time is over and the correct video clip could not be found, the search task fails.

Moreover, in order to observe how participants established their search strategies based on time and number of attempts, we reviewed the scoring functions used in the assessment of video retrieval systems. The metric developed by [Schoeffmann et al., 2014] uses two parameters: the time passed till finding the correct video (t) and the number of attempts made until finding the correct video (p). The scoring function is as follows (s denotes the points gained from a search task, and t_{max} denotes the time allowed in a search task):

$$s(t,p) = \frac{100 - 50\frac{t}{t_{max}}}{p}$$

This formula enables doing a trade-off between time and number of attempts. The change with respect to time is linear whilst the change on the score is inversely proportional to the square of number of attempts. This motivates the participants

to do as many attempts as possible till finding the correct clip. The participants favoring time cannot be distinguished from the ones favoring number of attempts. To overcome this problem, we created a new scoring function having a parametrized trade-off space between time and number of attempts. The function is derived by the functions yielding the point loss per second or attempt, namely T and P.

$T(t; w, \alpha, t_{avg}) = -100w(\frac{t}{t_{avg}})^{\alpha}$ (t: the time passed till finding the correct video clip, t_{avg}: the arithmetic mean time passed till finding the correct video or having a failure)

$P(p; w, \beta, p_{avg}) = -100(1-w)(\frac{p}{p_{avg}})^{\beta}$ (p: the number of attempts till finding the correct video clip, p_{avg}: the arithmetic mean number of attempts till finding the correct video clip)

The -100 coefficient in both functions implies that a participant can lose up to 100 points. The w coefficient is a real number in the $[0, 1]$ range. This coefficient determines the linear contribution of time and number of attempts to total loss. The last factor in both functions is an exponentiation of the fraction of a time passed or a number of attempts made ($\alpha, \beta \in [0, 1]$ and $\alpha, \beta \in \Re$). Having $\alpha = 0$ in T means that every unit of time passed has the same point loss. If $\alpha > 0$, the point loss increases with respect to time. However, as α approaches to 1, the point loss for a particular time or attempt count decreases.

By taking the indefinite integrals of the loss functions as follows, the scoring function used in our evaluation studies has been derived. Note that the constants coming from the integrals are set to 0.

$$S(t, p; w, \alpha, \beta, t_{avg}, p_{avg})$$
$$= 100 + \int T(t; w, \alpha, t_{max}) d(\tfrac{t}{t_{avg}}) + \int P(p; w, \beta, p_{avg}) d(\tfrac{p}{p_{avg}})$$
$$= 100 - 100w(\tfrac{t}{t_{avg}})^{\alpha+1}\tfrac{1}{\alpha+1} - 100(1-w)(\tfrac{p}{p_{avg}})^{\beta+1}\tfrac{1}{\beta+1}$$

The points gained from a search task is the product of video clip similarity[5] and the value obtained from the scoring function. If the search task fails, then the points gained from the task is zero. Taking the summation of the points gained from actual search tasks, the total points collected in an evaluation study is computed.

6.3 Results

We conducted the user evaluation studies on our video retrieval system by involving 15 participants. The following criteria were used in the assessment of the system:

- Sketched symbol classification performance
- Multimodal interpretation performance
- Search performance of the participants
- Search runtime
- The search strategies established by the participants

6.3.1 Sketched Symbol Classification Performance

To evaluate the classification performance of the sketched symbol classifier used in the video retrieval system, initially, the symbols that were drawn once and not removed from the drawing canvas later were detected by hand. These symbols, namely **permanent symbols**, are thought to be classified correctly. For each permanent symbol, we also inspected **the number of trials** to get to the correct classification. Having less number of trials for a symbol indicates that the symbol classifier works accurately on the symbols drawn by the participants. Within this context, we kept the statistics of the number of trials for each permanent symbol. As presented in Table 6.1, the most frequently drawn symbols are the player symbols, having the least number of trials. The symbol type with the greatest number of trials is the player motion arrow heading to the west. Analyzing the results altogether, the mean number of trials for all classes is below 1. Considering the sizes of the sketched symbol datasets collected and used for benchmarking [Eitz et al., 2012], we concluded that the support vector machine based classification with the IDM feature extraction method [Ouyang and Davis, 2009] has exhibited a remarkable performance on our

dataset of sketched symbols. We believe that this finding is useful since it can open the way to data efficient learning methods for sketched symbols.

Class or orientation	**Mean number of trials**	**Maximum number of trials**	**Number of permanent symbols**	**Number of permanent symbols with the number of trials greater than or equal to 1**
Player from side 1	0.15 ± 0.56	6	184	21
Player from side 2	0.12 ± 0.40	3	130	12
Ball motion/east	0.21 ± 0.70	6	101	21
Ball motion/west	0.28 ± 0.80	7	103	28
Player motion/east	0.27 ± 0.77	5	53	14
Player motion/west	0.46 ± 1.43	8	41	14

Table 6.1: Summary of the number of trials for permanent symbols

6.3.2 Multimodal Interpretation Performance

Out of the 250 motion arrow symbols drawn, the participants added motion events to 77 symbols (30.80%) through a pop-up window. This means that only 77 symbols were not matched with the correct motion event. In other words, the multimodal

interpretation system is $100\% - 30.80\% = 69.20\%$ accurate on the motion arrows drawn by the participants. Moreover, on average, the participants added motion events to only 30.42% of the motion arrow symbols drawn during a search task (with a standard deviation of 17.70%). Considering the fact that our dataset of n-grams is relatively smaller than the data sets used in benchmarking of the popular text classification models [dos Santos and Gatti, 2014], we concluded that our multimodal interpretation model has exhibited a notable performance on the speech and sketching data combined.

6.3.3 Search Performance of Participants

Out of the 60 video clips searched by the participants, 37 were found correctly (see Figure 6.6 for the number of clips found correctly by each participant). This means that our video retrieval system's accuracy is $37/60 = 61.67\%$. Taking the complexity and ambiguity of both speech and sketching into account, this accuracy rate is noteworthy. This retrieval system is the first of its kind in terms of the modalities used. Thus it constitutes a baseline for the video retrieval systems in the future.

Moreover, for each motion event, we analyzed the fraction of the video clips found correctly. As seen in Table 6.2, the highest accuracy was obtained in corner kicks, and the second in shots. For other events, the accuracy rates were ranging between 41% and 61%. These statistics imply that the sketching and speech produced by the participants to describe shots and corner kicks were consistent with each other. For other events, the sketching and speech used by the participants might not be consistent among the participants. Due to the lack of graphics quality of the Eat the Whistle game, the participants might have misinterpreted the motion events presented in video clips, resulting in wrong video clips returned by the retrieval system. For instance, some participants have interpreted a free kick as a pass, and the retrieval system has brought only passes rather than free kicks.

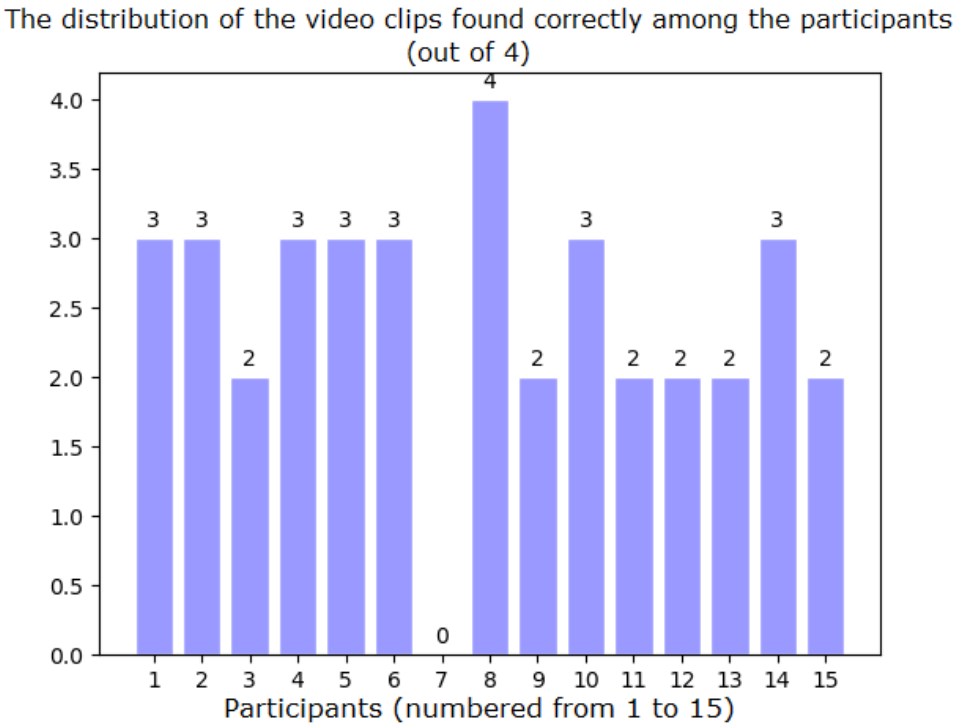


Figure 6.6: The distribution of the video clips found correctly among the participants

6.3.4 Search Runtime

An average completion time of a search was 83.5 seconds (with a standard deviation of 67.58 seconds). On average, the participants have found the correct video within the first 26.69 seconds of a search runtime (with a standard deviation of 21.87 seconds). Combining these results, the participants were able to find the correct video within the one-third ($26.69/83.5 \sim 1/3$) of a search runtime. Most of a search runtime is spent on the compilation of the video clips found. These imply that the retrieval system seems to list the desired results on the top, thus the retrieval algorithm serves to a good user experience. Note that this system is the first of its kind in terms of the modalities employed. Thus that runtime is considered to be a baseline for the multi-modal motion retrieval systems in the future.

Motion event	**Video clips presented**	**Video clips correctly found by a participant**	**Accuracy**
Header	13	8	61.54%
Corner kick	10	9	90.00%
Pass	12	5	41.67%
Bicycle	12	6	50.00%
Free kick	8	4	50.00%
Shot	13	11	84.62%
Throw-in	14	8	57.14%
Tackle	21	11	52.38%
Dribbling	34	20	58.82%

Table 6.2: Distribution of video clip retrieval accuracies among the motion events

6.3.5 Search Strategies

Using our new scoring metric on different combinations of w, α and β, we projected how the search established by the participants strategies based on time[67] and attempt count[8]. All the participants' scores are presented in figures 6.7, 6.8, 6.9, 6.10 and 6.11. As seen in the figures, the scores when $w = 1$ with different (α, β)[9] values are greater than the ones when $w = 0$. The participants' scores increase in parallel to the linear weight on the time loss. Most of the participants have favored time against attempt count, thus the loss in overall scores due to time was less than the ones due to attempt count.

Moreover, for different values of w[10], we took the average scores of each participant over different combinations of α and β. As shown in Figure 6.12, 5 participants have reached the maximum score when $w = 0$ whereas the others (10 participants) have reached the maximum when $w = 1$. This implies that majority of the participants have used time more efficiently than attempt count. Also, our retrieval system seems

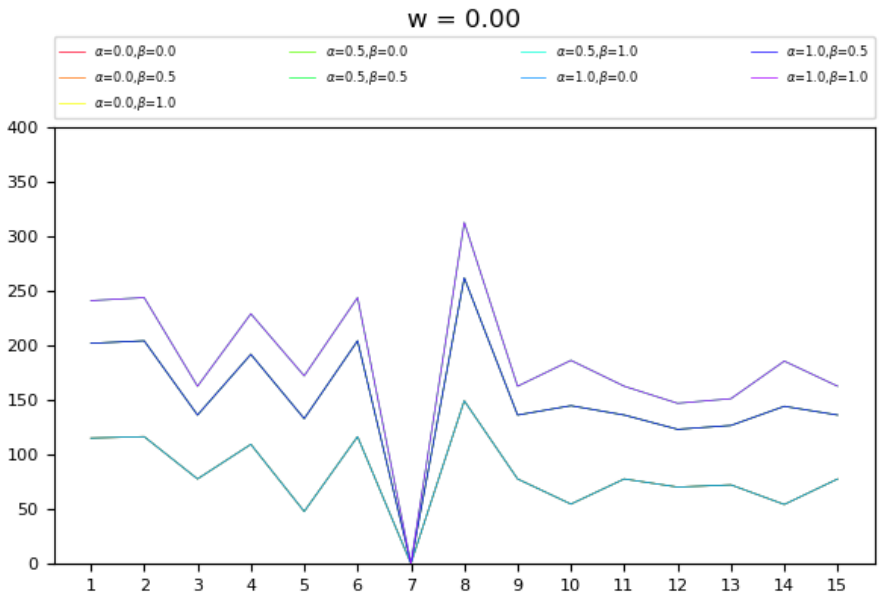


Figure 6.7: The participants' scores for different combinations of α and β when $w = 0$

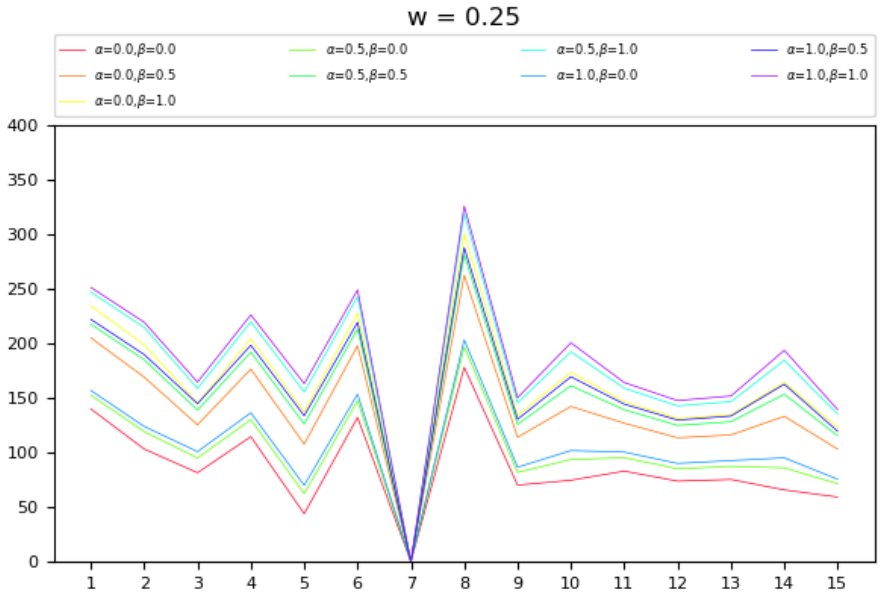


Figure 6.8: The participants' scores for different combinations of α and β when $w = 0.25$

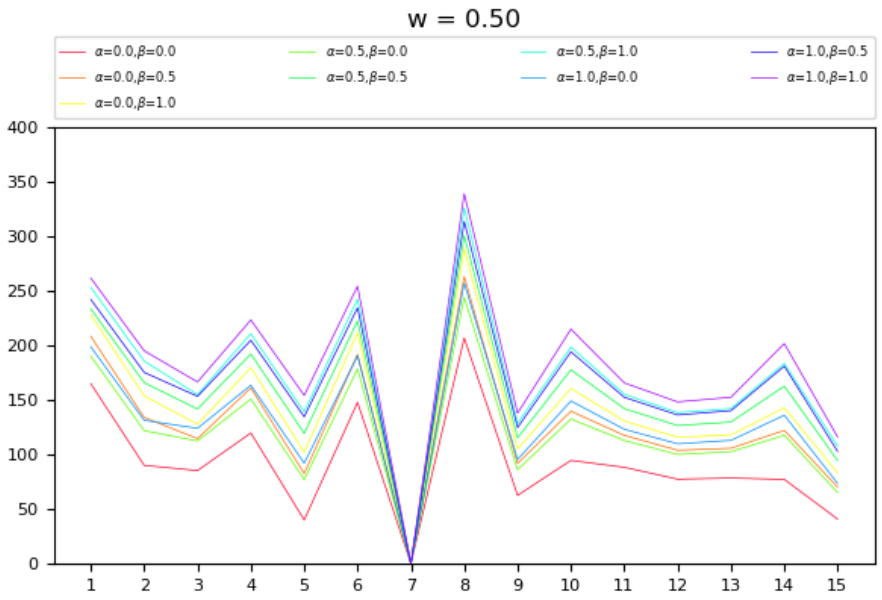


Figure 6.9: The participants' scores for different combinations of α and β when $w = 0.5$

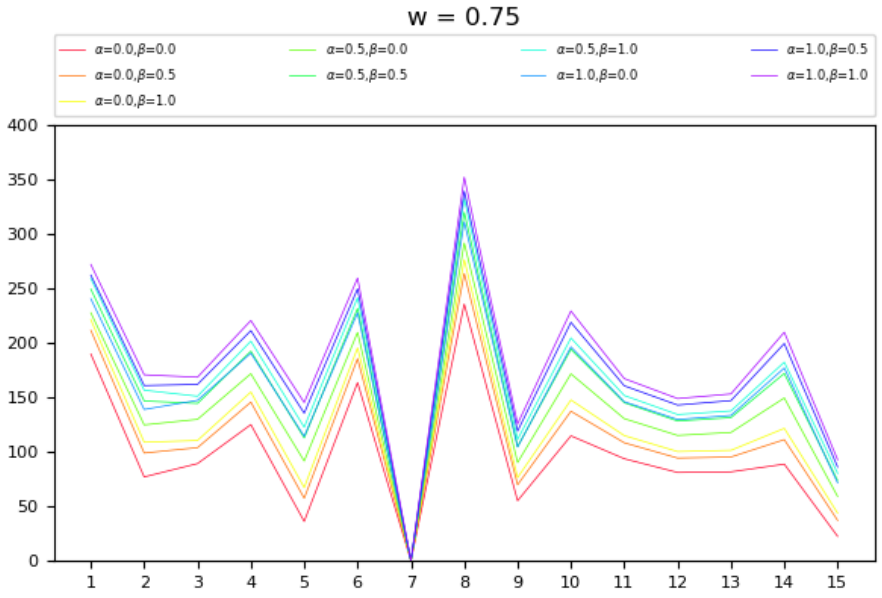


Figure 6.10: The participants' scores for different combinations of α and β when $w = 0.75$

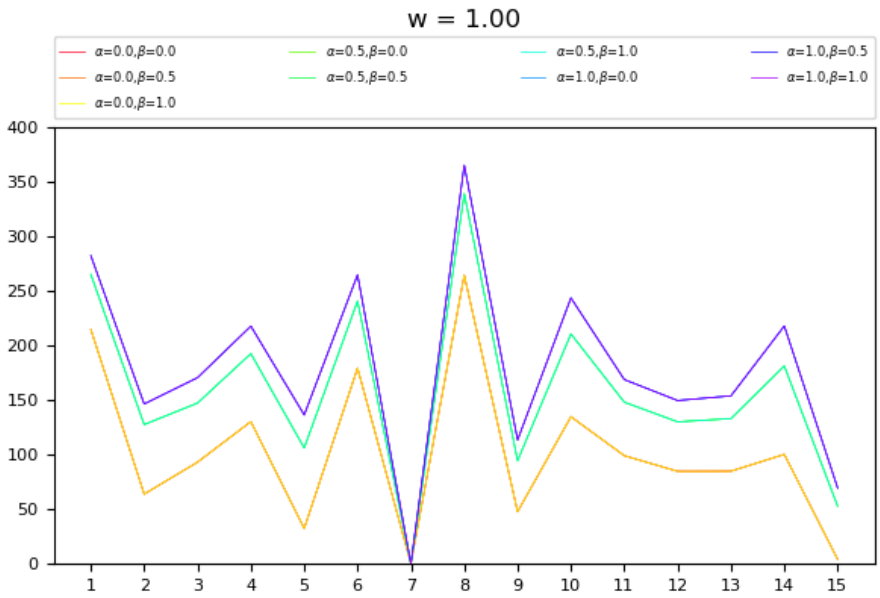


Figure 6.11: The participants' scores for different combinations of α and β when $w = 1$

to support both time-favoring and attempt-count-favoring search strategies without huge performance gaps.

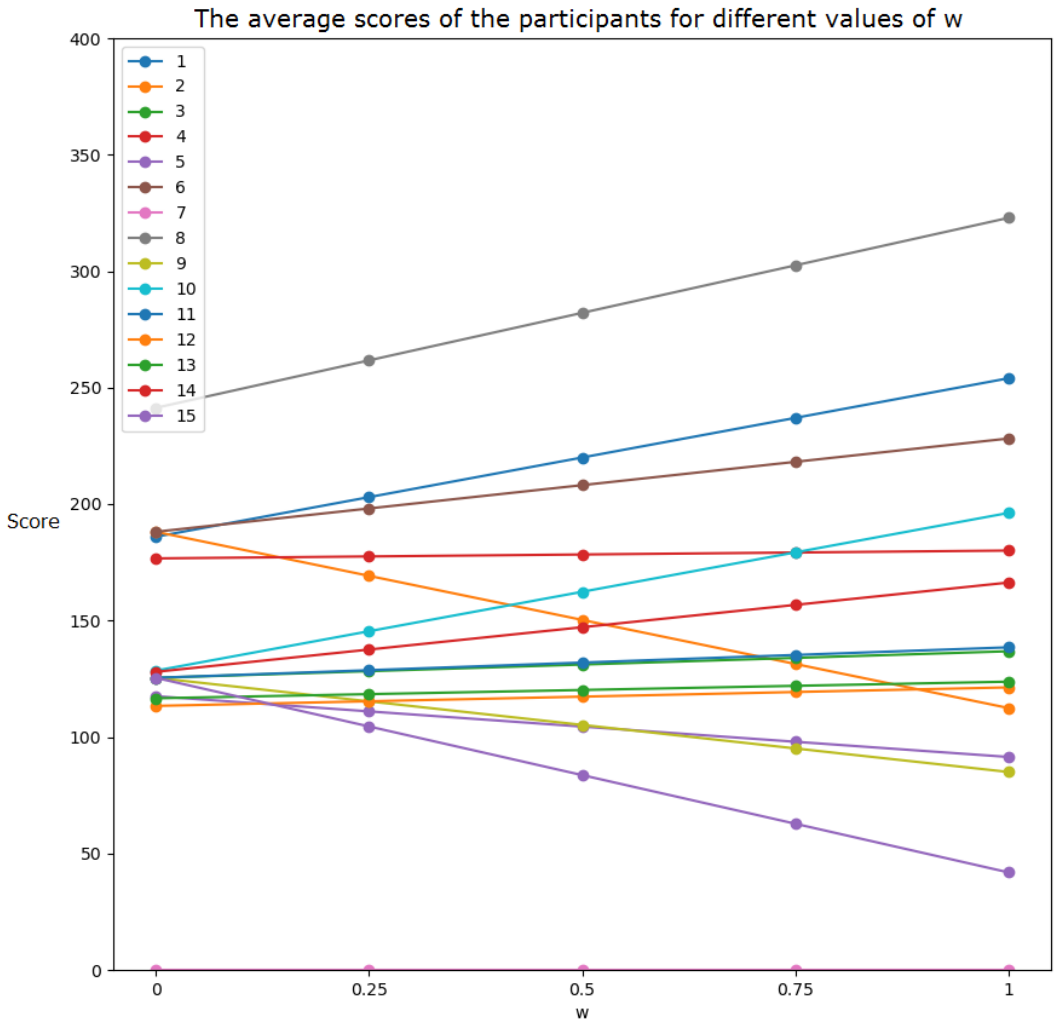


Figure 6.12: Average scores of the participants for different values of w (participants are enumerated from 1 to 15)

Notes to Chapter 6

1 Source code of the system is available online at http://github.com/ozymaxx/soccersearch

2 Among the 4 video clips played in the actual tasks, one clip includes a single primitive motion event, one clip includes two consecutive primitive motion events, one clip includes three consecutive primitive motion events and one includes four consecutive primitive motion events.

3 This action is called *an attempt* in the rest of the paper.

4 A video clip is said to be correct if the intersection of its frames with the frames of the actual clip is at least 80 percent of the frames of the actual one.

5 The number of frames at the intersection between the video clip found by the participant and the correct one / the number of frames in the correct video clip

6 We used second as the unit of time.

7 The average duration of finding a correct video clip was 285.32 seconds, hence t_{avg} was taken as 285.32.

8 The average number of attempts made in a search task was 1.63, hence p_{avg} was taken as 1.63.

9 α, $\beta \in \{0, 0.5, 1\}$

10 $w \in \{0, 0.25, 0.5, 0.75, 1\}$

Chapter 7

CONCLUSIONS

To have a prior knowledge on how humans articulate motion through speech and sketching, we designed a protocol and implemented a set of software tools for encouraging participants to draw and talk in a video retrieval task. The protocol and the software tools were assessed on the soccer match retrieval use case scenario. The ecologically validity in these tasks was formed by involving participants with ranging level of expertise in soccer. The software tools are customizable based on the retrieval use case, and the protocol is easy to follow. Afterwards, to build a natural interaction mechanism for depicting motion, we developed a multimodal interpretation model that is able to understand the sequence of motion events specified through speech and sketching. This heuristic model is based on two sub-models, a sketched symbol classifier and a text classifier. Considering the exiguity of the data we collected, the multimodal interpretation model and its sub-models are thought to exhibit a remarkable classification performance. All these results also show that the software and protocol used in data collection make a good couple with the multimodal interpretation model for building a speech-and-sketch-based user interfaces for characterizing motion. As the final step, we developed a multimodal video retrieval system by combining the natural interaction mechanism with a database back-end designed for big multimedia collections. The system was assessed through user evaluation studies. The study results show that the retrieval system works accurately on real-life multimodal inputs. However, this system is the first of its kind in terms of the modalities employed. This system has just constructed a baseline for the video retrieval systems in the future. There is a big room of improvement on the multimodal interpretation and the video retrieval algorithm. We believe that this system has opened the way for the

studies concerning multimodal interaction for effective and efficient video retrieval.

BIBLIOGRAPHY

[Adler and Davis, 2007a] Adler, A. and Davis, R. (2007a). Speech and sketching: An empirical study of multimodal interaction. In *Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling*, pages 83–90. ACM.

[Adler and Davis, 2007b] Adler, A. and Davis, R. (2007b). Speech and sketching for multimodal design. In *ACM SIGGRAPH 2007 courses*, page 14. ACM.

[Al Kabary and Schuldt, 2013] Al Kabary, I. and Schuldt, H. (2013). Towards sketch-based motion queries in sports videos. In *Multimedia (ISM), 2013 IEEE International Symposium on*, pages 309–314. IEEE.

[Altıok, 2017a] Altıok, O. C. (2017a). Ad-Hoc Video Search Application for Multi-Modal Data Collection.

[Altıok, 2017b] Altıok, O. C. (2017b). Multi-Modal Collector (Experimenter's Side).

[Altıok, 2017c] Altıok, O. C. (2017c). Multi-Modal Collector (Participant's Side).

[Altıok, 2017d] Altıok, O. C. (2017d). Sketch Annotator for Multi-Modal Collector.

[Bangsbo and Peitersen, 2000] Bangsbo, J. and Peitersen, B. (2000). *Soccer systems and strategies*. Human Kinetics.

[Cohen et al., 1997] Cohen, P. R., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. (1997). Quickset: Multimodal interaction for distributed applications. In *Proceedings of the fifth ACM international conference on Multimedia*, pages 31–40. ACM.

[Collomosse et al., 2009] Collomosse, J. P., McNeill, G., and Qian, Y. (2009). Storyboard sketches for content based video retrieval. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 245–252. IEEE.

[dos Santos and Gatti, 2014] dos Santos, C. and Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78.

[Eitz et al., 2012] Eitz, M., Richter, R., Boubekeur, T., Hildebrand, K., and Alexa, M. (2012). Sketch-based shape retrieval. *ACM Trans. Graph.*, 31(4):31–1.

[Giangreco and Schuldt, 2016] Giangreco, I. and Schuldt, H. (2016). Adampro: Database support for big multimedia retrieval. *Datenbank-Spektrum*, 16(1):17–26.

[Google, 2017] Google (2017). Cloud Speech Recognition API.

[Greco, 2008] Greco, G. (2008). Eat the Whistle.

[Hammond and Davis, 2006a] Hammond, T. and Davis, R. (2006a). Ladder: A language to describe drawing, display, and editing in sketch recognition. In *ACM SIGGRAPH 2006 Courses*, page 27. ACM.

[Hammond and Davis, 2006b] Hammond, T. and Davis, R. (2006b). Tahuti: A geometrical sketch recognition system for uml class diagrams. In *ACM SIGGRAPH 2006 Courses*, page 25. ACM.

[Hse et al., 2004] Hse, H., Shilman, M., and Newton, A. R. (2004). Robust sketched symbol fragmentation using templates. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 156–160. ACM.

[Joulin et al., 2016] Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

[Kazi et al., 2014] Kazi, R. H., Chevalier, F., Grossman, T., and Fitzmaurice, G. (2014). Kitty: sketching dynamic and interactive illustrations. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 395–405. ACM.

[Mauceri et al., 2015] Mauceri, C., Suma, E. A., Finkelstein, S., and Souvenir, R. (2015). Evaluating visual query methods for articulated motion video search. *International Journal of Human-Computer Studies*, 77:10–22.

[Muzychenko, 2015] Muzychenko, E. (2015). Virtual Audio Cable.

[Ouyang and Davis, 2009] Ouyang, T. Y. and Davis, R. (2009). A visual approach to sketched symbol recognition. In *IJCAI*, volume 9, pages 1463–1468.

[Oviatt et al., 2000] Oviatt, S., Cohen, P., Wu, L., Duncan, L., Suhm, B., Bers, J., Holzman, T., Winograd, T., Landay, J., Larson, J., et al. (2000). Designing the user interface for multimodal speech and pen-based gesture applications: state-of-the-art systems and future research directions. *Human-computer interaction*, 15(4):263–322.

[Papineni et al., 1997] Papineni, K. A., Roukos, S., and Ward, T. R. (1997). Feature-based language understanding. In *Fifth European Conference on Speech Communication and Technology*.

[Schoeffmann et al., 2014] Schoeffmann, K., Ahlström, D., Bailer, W., Cobârzan, C., Hopfgartner, F., McGuinness, K., Gurrin, C., Frisson, C., Le, D.-D., Del Fabro, M., et al. (2014). The video browser showdown: a live evaluation of interactive video search tools. *International Journal of Multimedia Information Retrieval*, 3(2):113–127.

[Seddati et al., 2016] Seddati, O., Dupont, S., and Mahmoudi, S. (2016). Deepsketch 2: Deep convolutional neural networks for partial sketch recognition. In *Content-*

Based Multimedia Indexing (CBMI), 2016 14th International Workshop on, pages 1–6. IEEE.

[Skantze and Al Moubayed, 2012] Skantze, G. and Al Moubayed, S. (2012). Iristk: a statechart-based toolkit for multi-party face-to-face interaction. In *Proceedings of the 14th ACM international conference on Multimodal interaction*, pages 69–76. ACM.

[Skubic et al., 2007] Skubic, M., Anderson, D., Blisard, S., Perzanowski, D., and Schultz, A. (2007). Using a hand-drawn sketch to control a team of robots. *Autonomous Robots*, 22(4):399–410.

[Thorne et al., 2004] Thorne, M., Burke, D., and van de Panne, M. (2004). Motion doodles: an interface for sketching character motion. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 424–431. ACM.

[Tumen et al., 2010] Tumen, R. S., Acer, M. E., and Sezgin, T. M. (2010). Feature extraction and classifier combination for image-based sketch recognition. In *Proceedings of the Seventh Sketch-Based Interfaces and Modeling Symposium*, pages 63–70. Eurographics Association.

[Vergo, 1998] Vergo, J. (1998). A statistical approach to multimodal natural language interaction. In *Proceedings of the AAAI98 Workshop on Representations for Multimodal Human-Computer Interaction*, pages 81–85.

[Yesilbek et al., 2015] Yesilbek, K. T., Sen, C., Cakmak, S., and Sezgin, T. M. (2015). Svm-based sketch recognition: which hyperparameter interval to try? In *Proceedings of the workshop on Sketch-Based Interfaces and Modeling*, pages 117–121. Eurographics Association.

[Yu et al., 2013] Yu, H., Ho, C., Juan, Y., and Lin, C. (2013). Libshorttext: A library for short-text classification and analysis. *Rapport interne, Department of Computer*

Science, National Taiwan University. Software available at http://www. csie. ntu. edu. tw/˜ cjlin/libshorttext.