Otto-von-Guericke-Universitat Magdeburg

Faculty of Computer Science

Master's Thesis

Development of a Reading Model for Gaze Data to classify different Types of Human Reading Behavior

Author:

Volkan Demir

May 31, 2018

Advisors:

Prof. Dr.-Ing. Andreas Nürnberger
Data and Knowledge Engineering

Dr.-Ing. Tatiana Gossen
Dipl.-Inform. Michael Kotzyba

Demir, Volkan:
Development of a Reading Model for Gaze Data to classify different Types of Human Reading Behavior
Master's Thesis, Otto-von-Guericke-Universitat Magdeburg, 2018.

Abstract

Reading is a complex skill that involves interaction of many different stages of information processesing [Rei00]. For instance, where the eyes looking at? How long do we look at? In order to find answers to those questions many reading models are developed. Nowadays, many eye tracking based research have been conducted to explore reading. Those readings can include long and short texts, text with pictures and text with graphics where they can be found in newspapers, on webpages, books and so on. Assuming that readers are reading whenever they look at text is not a good solution to detect and analyze the reading. For instance, such an assumption cannot distinguish between different types of reading, such as scanning from skimming. Thus, in this thesis a reading model are developed to classify different types of human reading behavior in eye movements recorded by an eye tracker. The classification uses the fixations, saccades and unclassified seqeunces with their features in the gaze data. The implemented method is Hidden Markov Model (HMM) as certain eye movement patterns are charecteristic for reading and HMM is used in sequence classification.

Keywords. Eye Movements, Types of Reading Behavior, Sequence Classification, Fitting Distributions, Hidden Markov Model

Acknowledgements

This thesis marks the end of my academic journey in Otto-von-Guericke University Magdeburg for graduating with a Master degree. I would like to thank all the academic stuff in the university, and all the people who were with me the last two years. I do not want to miss all that I have learned by working on this thesis.

First and foremost, I thank my advisors Tatiana Gossen, Michael Kotzyba and Johannes Schwerdt. I am grateful for their big support, patience, useful comments, and for their excellent guidance that helped me not to get lost on this path.

I would like to thank Prof. Dr. Ing. Andreas Nuremberg for giving me the opportunity to write my Master's thesis at his chair.

Finally, I thank my family and friends for their moral support and patience during the preparation of this thesis.

Contents

List of Figures

List of Tables

List of Code Listings

1. Introduction

Reading is a complex skill that involves the interaction of many different stages of information processing [Rei00]. For instance, where the eyes looking at? and how much time do we spend looking at it? In order to find answers to this type of questions researchers developed many reading models to explain how the eye movements are controlled reading. For example, E-Z Reader model, SWIFT and EMMA [Rei00]. However, those models explain reading if reading is happening, they are not able to detect reading. Thus, we require a new model that can identify reading. Furthermore, classify different types of reading behavior.

To date, many eye tracking based studies are referenced to detect reading process. Such reading could be on web pages, books, newspapers and so on. In the most cases, those materials could include not only text but also pictures. For instance, in Figure 1.1 one can see a webpage that includes text and picture with the location of fixation points of the eye movements. Besides, a reader may be looking for a piece of information on the text so he is not interested in full text but he might be looking for instance for a number. Thus, depending on the task, eye movements may show different type of processes; scanning the page, reading the passage, viewing the picture and many. Those processes are needed to be distinguished when it comes to statical analysis of experimental data.

A commonly used method to classify eye movements is a Region of Interest Analysis (ROI). This method basically assumes that if the reader looks at the text then s/he is reading. Such simple assumption may fail as eye movements in reading have different processes. For instance, when looking at a screen one may not be interested in text on it; one might probably think about something else while his gaze points may stay on text ROIs, although he is not reading. In these situations much more complicated technique is needed. Thus, one of the machine learning algorithms has been decided to be used: Hidden Markov Model (HMM) as certain eye movement patterns are characteristic for reading. By so, HMM is going to learn and recognize the pattern of reading process. This approach only uses the tracked eye movements extracted from the eye tracking software; Tobii Studio. The approach does not take

ROIs or any other method. Besides the standard HMM, we are also going to use the Naive Bayes HMM as a baseline for comparison of the methods.

We tested our approach on a large amount of experimental data which have been recorded before, using different study cases for [KGSN17b]. The states of *scanning*, *skimming*, *reading*, *media view* and *unknown* has been annotated in a fraction of that data to obtain a training and validation set.

1.1 Problem Description and Motivation

In this thesis, the problem of classifying reading behavior has been addressed. Determining user interest when interacting with a screen is not an easy task. Hence, it is important to know what types of reading behavior the user shows.

Given the problem, our motivation is the classification of reading behavior that can be used to support individual users in their web searches and in any other area that requires reading. In order to support individual users, their reading behavior with tracking eye movements is needed to be defined and to be classified which moment the user is doing. If what the user is doing currently can be detected, better individual support can be provided to the user. Results of reading behavior classification may be helpful to detect web search activities; such as Fact-Finding Search or Exploratory Search. For instance, if the user is doing Exploratory Search, then more search engine result pages may be given to him. However, this scope refers to future works of this thesis.

Regarding the problem description and motivation, the research questions of the thesis are;

RQ1. How accurate the developed reading model correctly detects reading behavior when compared with the labeled data?

RQ2. How good the model is able to distinguish between the types of reading behavior?

RQ3. How effects of using the developed model transition compared to the baseline?

Figure 1.1: An example of a webpage with fixation points. Adapted from [usa18]

1.2 Goal and Contributions

The goal of this thesis is to develop a reading model that is able to classify reading behavior of eye movement processes. By doing that, the following contributions are provided:

- A reading model is provided that uses gaze data to detect reading.
- The model is able to distinguish between different types of human reading behavior such as *scanning*, *skimming*, *reading*, *mediaview* and *unknown*.
- For a given problem, how to estimate the parameters of the gaze data distributions.
- In order to illustrate the performance of the model, an evaluation if the developed reading model correctly detects reading behavior regarding the labeled data are presented and compared to the baseline.

1.3 Structure of the Thesis

This thesis is organized as follows. In Chapter 2 existing scientific work related to the thesis topic is presented. Chapter 3 introduces the background concepts of *Eye Movements* and *Hidden Markov Model*. Chapter 4 introduces the experimental data, and it presents the development of a reading model with HMM. Chapter 5 presents and discusses the evaluation concept, how well the reading model to classify reading behavior. Chapter 6 first evaluates and discusses the approach and then summarize the thesis and provide possible further research directions.

2. Related Work

In this chapter, the related work to thesis is presented. Gaze data can be used to detect what kind of task the user is doing ([IB04], [KGSN17a]) or whether a person is reading or not ([CM01], [Gus10], [KH07], [BDvE08], [CGL+11], [BHB12]). As this thesis focuses on classifying reading behaviors, more details from reading detection works by the year of publication will be given.

In 2001, Campbell and Maglio [CM01] presented a reading detection algorithm. Their reading detection system relies on three mechanism: (a) eye movements in both x and y positions are averaged over 100 ms intervals. (b) evidence of reading is accumulated until it crosses a predefined threshold value. If the eye moves to the right (to reading direction), then reading evidence was increased. Vice versa, if the eye moves to the left (to regression direction), then reading evidence was decreased. (c) mode switching: if the evidence reaches a threshold, then reading is detected and the mode is switched to reading from scanning. To accumulate evidence of reading they track some specific eye movements. For instance, if the eye moves a short distance right then the event is categorized as "*read forward*" and awarded with +10 points. But, if the eye moves a short distance left then the event is categorized as "*regression saccade*" and awarded with -10 points. When the systems reaches +30 points threshold, it switches into reading mode. In their approach there are 13 different events with some points, so as an input to the algorithm they used fixation points and the difference vectors between them. According to [Gus10], this approach was tested on a Tobii system by Olsson 2007 and result was that it only worked for long reading sequences but had problems with short reading sequences. We consider this work is a traditional reading detection based on character lengths and reflects the state of the art of character-based classification methods.

In 2004, [IB04] provided an alternative and complementary approach to identifying user tasks based on analyzing patterns of eye movements. For that, they collected eye movements of the users who applied different computer-based tasks and they have shown that each task has unique eye movements. To classify those tasks, they simply calculated the percentage time spent on AOIs and the fixations within each AOI.

In 2007, Kollmorgen and Holmqvist [KH07] presented a method for reading detection using Hidden Markov Models with six states where three of them was considered as reading and the rest of three was considered as non-reading. The classes of each state were fixation and saccade with the duration and length features respectively and blink with no feature. The classification of the states was done by the Viterbi algorithm as reading or non-reading. The differences between our approach and theirs are that they only classify two classes where we do for five classes, and they do not model the fraction of the observation given the state. Besides, the emission distributions that they used in their HMM was discrete probability distributions while we applied continuous probability distributions. In this thesis, duration is treated as continuous variable that can take any value between two timestamps as well as the absolute saccadic direction where the angle can take any value between two angles. However, they also used a neural network to detect reading and they got almost similar results when compared with HMM.

In 2008, Buscher [BDvE08] presented an algorithm to detect whether a person is reading or skimming. As [CM01] did in 2001 their algorithm is also character-based reading detection. Their algorithm follows three steps. Those are; 1. Algorithm detects the fixations 2. Algorithm classifies the transitions from one fixation to the next one as features. 3. Scores related with the features are accumulated. Like in [CM01] if the accumulated score reaches the predefined threshold value (+30 for reading and +20 for skimming) then reading or skimming is detected. They also pointed out the idea of the algorithm is related to the [CM01]. However, their algorithm has some modifications. First of all, they concerned about the detection of fixations. Besides, their accumulation strategy is different. For instance, when the eye move, they calculate the distance and direction in letter space. If the eye move to the right and the movement is between 0 and 11 letter space which is called "*read forward*", then they award +10 points for reading and +5 for skimming. If the eye move to the left and the movement is between 0 and -6 letter space which is called "*short regression*", then they award -8 points for both reading and skimming.

In 2010, Gustavsson [Gus10] presented an evaluation of HMM and Neural Network for classification of reading in eye movements in his thesis. In his HMM, the hidden states are the read saccades, regressions and four directions of long saccades (forward, return, up and down sweeps). Besides, he set the state transition probabilities as uniform distribution, so in every state has equal probability to any state. For the emission distribution part of the HMM, he uses standard 2-dimensional Gaussian distributions to represent observations which are saccades in terms of x and y positions in pixels. After building HMM, he trains the model with the Baum-Welch algorithm. The differences between our approach and his are that our HMM parameters; observations, hidden states and the representation of the distributions are different. He uses just the features of the saccade. Furthermore, as the all state transitions are occurring in our work; we do not set the transition probabilities as uniform.

In 2011, Cole [CGL+11] constructed a line-oriented reading model based on EZ reader model to distinguish single fixations from the sequence of fixations. If there was a sequence of fixations near enough to one another, taking account of regressions, this was labeled as a "*reading*" sequence. The inputs to their algorithm are fixation

Table 2.1: A list of Related Works

Publication Year	Study Name	Type of Method(s)
2001	A Robust Algorithm for Reading Detection [CM01]	Reading detection based on character lengths
2004	Using Eye Gaze Patterns to Identify User Tasks [IB04]	Percentage time spent on AOIs and the fixations within each AOI
2007	Automatically Detecting Reading in Eye Tracking Data [KH07]	HMM & Neural Network
2008	Eye Movements As Implicit Relevance Feedback [BDvE08]	Character-based reading detection
2010	Real Time Classification of Reading in Gaze Data [Gus10]	Neural Network
2011	Task and user effects on reading patterns in information search [CGL+11]	Line-oriented reading model based on EZ reader model
2012	A Robust Realtime Reading-skimming Classifier [BHB12]	RBF-SVM
2017	Inferring User's Search Activity Using Interaction Logs and Gaze Data [KGSN17a]	HMM

locations and their duration. The output is a classification of the sequences as *reading* or *scanning*. From our understanding, their algorithms work in that way; 1. They detect all the fixations 2. They exclude the fixations that cannot exceed 113ms. 3. If the fixation location is within 35 pixels to 120 pixels parafoveal region; label the fixation sequence as "*reading*". Else, the fixation is not on the right side of parafoveal region and fixation is not a regression then label the fixation sequence as "*scanning*". The model is line-oriented, so when the sentence wraps to a new line the model breaks that into two reading sequences. That's a limitation of the model depending on the application.

In 2012, Biedert [BHB12] presented a classification method to distinguish *reading* from *skimming*. In order to build the classifier, they use two attributes as the main features which are the angularity of the saccade and forward speed. Angularity denotes how bent or vertical the saccades, and forward speed denotes how much progress in the reading direction of the text. They hire *RBF-SVM* (A machine learning method for classification) for training where they also conduct a grid search. Besides, they implement a reading detection algorithm similar to [BDvE08] for the baseline to their method.

In 2017, Schwerdt [KGSN17a] proposed a methodology and a model to identify web search activities using the sources logging files and eye tracking data. They collected the sources applying a user study where users had to perform three search tasks. In this thesis, we also use the same eye tracking data. In order to classify the tasks, they hire the Markov chain with the features of fixation durations, fixation occurrences and with logging file features such as clicking and scrolling.

A summary of the studies can be seen in tabular form in Table 2.1. The first column shows the publication year of the work, the second column shows the name of the work and the last column shows the type of method that they used in their works.

3. Background

Before explaining eye movements and reading behaviors, we need to understand what reading is. In literature, reading is defined as:

Definition 1.1 Complex cognitive process of decoding symbols in order to construct or derive meaning [Wik18c].

Cognitive process is the mental action of getting knowledge and understand through experience, though, and the senses. So, with the interaction between printed text and the reader's eyes, reading is shaped by readers prior knowledge, experiences, thoughts and so on. In reading, the most important and basic step for the eyes is to see, identify and recognize the printed word. If the eye can identify and recognize the word, then the word can be understood by the reader. To understand how word recognition is done by the eyes, the eye movements are explained in the next section.

3.1 Eye Movements

In this section, we will describe the basic types of eye movements that are relevant for the thesis as background information. The following topics will be discussed: *1. Fixation and Saccades*, *2. Regression* and *3. Refixation*.

Fixation and Saccade

During a reading, the eyes remain stationary for brief periods of time typically 200-300 ms, and this behavior is called *Fixation* (Table 3.1). Visual information is extracted from the printed page only during Fixations.

Between *Fixations*, the eyes make short and rapid movements that are called *Saccade*. *Saccades* usually take 20 to 50 ms (depending on the length of the movement), which is equal to between 1 and 3 degrees in reading, to complete its cognitive process. During Saccades, no visual information is extracted. In Figure 3.1, one can see an example of fixation and saccade movements on the text.

Task	Fixation Duration (ms)	Saccade Length (Degrees)
Silent Reading	225	2
Oral Reading	275	1.5
Visual Search	275	3
Typing	400	1

Table 3.1: Approximate Fixation Duration and Saccade Length in Reading. Adapted from [Ray78]

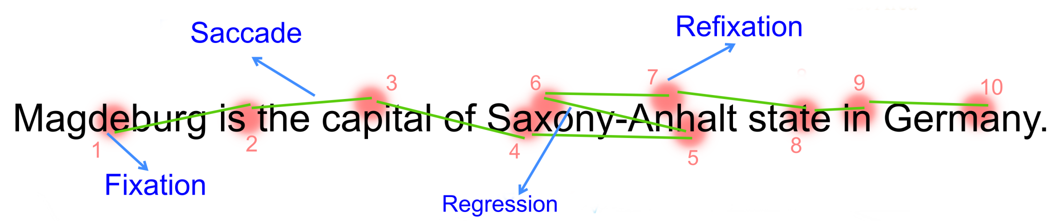


Figure 3.1: Example for *Fixations* and *Saccades*. The red dots represent fixation points and green lines between fixations represent saccades.

Regression

Reader does not always move their eyes in the direction of the text. About between 10% and 15% of all saccades in reading are *regressions*. That is, saccades move backward in the text. For instance, in Figure 3.1 while we are on the word; *"Anhalt"*, we saccade to backward and fixation on the word *"Saxony"* again.

Refixation

Refixation is the fixation on a word more than once in succession, in particular if it is a long word. So, for example in Figure 3.1, after we backward to word *"Saxony"*, the eyes saccade the next word and fixate *"Anhalt"* once again. In Figure 3.2, one can see that when the word length increase, the probability of refixation also increase. Because many letter words are hard to fixate at once, and user fixates the word again. On the other hand, as the word length decreases, the probability of skipping increases. For instance, on Figure 3.1 the word "of" between the "capital" and "Saxony" is skipped.

3.2 Types of Human Reading Behaviour

In this section, the types of human reading behavior that are used in the reading model are presented. The data has been labeled using the following descriptions of the reading behaviors.

Scanning

Scanning is a form of rapidly reading a text. Generally, it focuses on gaining a particular piece of information of a text rather than the understanding of the text as its whole [RM15]. The eye movement is horizontal oriented as the eyes sweeping over

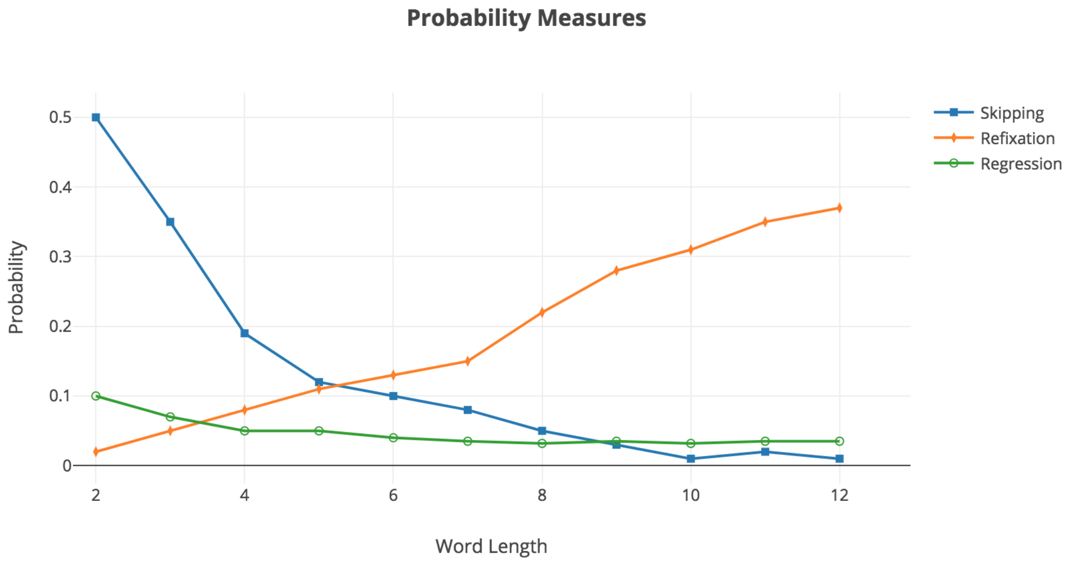


Figure 3.2: Benchmark Data for Fixation Probabilities. Adapted from [Nut08]

the text with the aim of finding a piece of information such as keywords and phrases (names, numbers or dates). According to White [WWMP15], the eye movement during *Scanning* is characterized by shorter overall reading time, fewer fixations, longer saccades and higher skipping rates. Clark [CRH+14] state that longer scan paths can be observed with respect to duration and length. In general, *Scanning* requires full attention and can be compared to a "mental spotlight". *Scanning* and *Skimming* are not often distinguished and rather merge into the same category called *Scanning*.

Skimming

Skimming is a form of rapidly reading a text. According to Geoffrey [RM15] the focus of *Skimming* is understanding the general meaning and obtain a brief summary of the content for a given text. According to Clark [CRH+14] it is characterized by less and shorter fixations, less saccadic regressions, long saccades and rather vertical eye movements. In *Skimming*, reading speed is much faster when compared with normal reading (600-700 vs. 280 words per minute). Besides, participants capture the main content but lack clear details [Rei00]. Duggan [DP11] describes the skimming strategy as a result of a reading experience after a threshold of information content is reached. If the information content is low, it is more possible that readers switch to *Skimming* after *Reading*. *Scanning* and *Skimming* are not often distinguished and rather merge into the same category called *Scanning*.

Reading

Reading is a clear defined behavior with organized eye movements according to Rayner [Ray08]. It is characterized by moving from line to line and fixating almost every word in each line to ensure complete understanding of the text. Function words (e.g., a and the) are skipped very often even in normal reading though and are only fixated 35% of the times they are encountered. Also, 2-3 letter words are only fixated in 25% of the cases.

Table 3.2: Summary of Types of Reading Behaviour

Reading Behaviour	Form	Focus	Eye Movement	Character
Scanning	rapidly reading a text	gaining a particular piece of information	horizontal oriented as a sweeping the eyes over the text	shorter reading time, fewer fixations, longer saccades and higher skipping rates.
Skimming	rapidly reading a text	understanding the general meaning of a text and obtain a brief summary	vertical eye movements	less and shorter fixations
Reading	normal reading	complete understanding a text	from line to line	fixating almost every word in each line
MediaView	almost no reading	visual media	restricted eye movements in a specific area	longer fixations and less saccades [RMR08]
Unknwon	no reading	no focus	outside the screen or undefined eye movement	overly long saccades if exist

MediaView

MediaView is a state when the user is not looking at the text but visual media such as images, tables, videosgifs, graphs or other media.

Unknown

Unknown is a state for events that could not be defined in the above behaviors. This state can be though as a container for several sub-states like views on the keyboard, adjusting glasses, etc.

Definitions are summarized in Table 3.2. The second column defines the reading form, the third column defines the main focus of the behaviors, the fourth column explains where the eyes move, and the last column gives the characterization of the behaviors. Besides, the visual projection of those reading behaviours can be seen in Chapter A.

3.3 Hidden Markov Model

The Hidden Markov Model (HMM) is one of the machine learning methods. In order to make a definition of HMM, first the *Markov chain* is introduced. Both Markov chain and HMM are the extension of the finite automata [JM00]. A Markov chain is a special format of weighted automaton where the weights are probabilities and the inputs determine automaton states. Markov chain is only used for clear sequences. For instance weather events; such case we only have the events such as hot, cold and warm. Given those events with a time sequence, Markov chain tries to find the type of weather for another unknown day. Thus, we can conclude that a Markov chain is useful when we want to compute probabilities for events that we can observe in the real world. However, sometimes the events that we are interested in, may not be directly observable. HMM separates the observations from the states; the observations are visible, however the state sequences that that we are trying to find them are hidden. For instance, given example above, HMM calculates the probability of having cold weather by observing another types of activity such as wearing a jacket at cold weather for each day. In this case, probability of having cold weather is a hidden state (we can not see the weather) and wearing a jacket at cold weather is an observation (this is an event we can observe and this observation is used to decide what is the weather like).

Hidden Markov Model can be defined as follows:

Definition 2.1 Hidden Markov Model (HMM) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (i.e. hidden) states [Wik18b].

In this thesis, we used a discrete first order Hidden Markov Model as the observations (Fixations, Saccades and Unclassified) of our model are defined. A set of five elements (Table 3.3) can be used to describe an HMM. In the next section, we will define the components of HMM.

Components of Hidden Markov Model.

An HMM is created by its transition probability distributions, its emission probability distributions and its initial probability distributions. Given Table 3.3, HMM Notation can be defined as λ = (A, B, π)

Formula	Definition
$S = s_1,s_2,...s_K$	a set of K states
A = $a_{11},a_{12}...a_{n1}...a_{nn}$	A transition probability matrix A, each a_{ij} representing the probability of moving from state i to state j
Y = $y_1,y_2...y_T$	A sequence of observations
B = $b_i(o_T)$	An emission probabilities, present the probability of an observation y_T being generated from a state i
$\pi_i = \pi_1,\pi_2...\pi_K$	An initial probability distribution over states. π_i is the probability that the Markov chain will start in state i

Table 3.3: Components of Hidden Markov Model. Adapted from [JM00]

In our case, the observations are fixation, saccade and unclassified sequences out of Tobii's filter. We assume that these observations can be described by a Markov model with hidden states as we have set of observations, and furthermore states that we are trying to distinguish between Scanning, Skimming, Reading, MediaView and Unknown states. All possible transitions between these five states can occur.

Problems of Hidden Markov Model

The three fundamental problems for HMMs are:

1. Suppose we have an observed sequence, O, and a Hidden Markov Model, λ, what is the probability that the sequence was generated by the model, P(O|λ)?

2. Suppose we have an observed sequence, O, and a Hidden Markov Model, λ, what is the most likely path, Q, through the hidden states, or $argmax_Q$P(Q|O,λ).

3. Suppose we have a set of sequences, O, what are the parameters of a Hidden Markov Model, λ, that maximize the probability P(O|λ)?

There are solutions to those problems: For problem 1, the solution is the forward algorithm. For problem 2, the most used algorithm is Viterbi algorithm. For problem 3, there is no manageable algorithm for finding the global maximum but algorithms to iteratively find local maximums exists, for example Baum-Welch algorithm which is an Expectation Maximization algorithm [Rab90].

In this thesis, classification of a sequence was done by using the Viterbi algorithm.

Viterbi Algorithm

The Viterbi algorithm is a dynamic programming algorithm that computes the most likely sequence of hidden states. Inputs for the algorithm are:

- Observation space O = $(o_1, o_2, ...o_N)$
- State space S = $(s_1, s_2, ..., s_K)$
- Initial probabilities pi = $(\pi_1,\pi_2...\pi_K)$
- A sequence of observations Y = $(y_1, y_2..., y_T)$
- Transition matrix A
- Emission matrix B

Output: The most likely hidden state sequence X = $(x_1, x_2..., x_N)$

In Listing 3.1, an example of a pseudo-code listing is given.

```
function VITERBI(O,S,π,Y,A,B) : X
     for each state i ∈ {1,2,...,K} do
         T_1[i,1] ← π_i · B_{iy_1}
         T_2[i,1] ← 0
     end for
     for each observation i = 2,3,...,T do
         for each state  j ∈ {1,2,..,K} do
             T_1[j,i] ← max_k(T_1[k,i−1] · A_{kj} · B_{jy_i})
             T_2[j,i] ← argmax_k(T_1[k,i−1] · A_{kj} · B_{jy_i})
         end for
     end for
     z_T ← argmax_k(T_1[k,T])
     x_T ← S_{Z_T}
     for i ← T,T−1,...,2 do
         z_{i−1} ← T_2[z_i,i]
         x_{i−1} ← s_{z_{i−1}}
     end for
     return X
 end function
```

Listing 3.1: Viterbi Algorithm

3.4 Experimental Data

The data used in this thesis was acquired from a user study of *DKE group* of *Otto von Guericke University*. In this user study users had to perform three search task: two *Exploratory* search tasks ($Expl_1$ and $Expl_2$) and at most twelve *Fact-Finding* search tasks (*Fact*). Subjects had at most 20 min to complete each exploratory and lookup tasks. Besides, Latin Square study had been performed to differ the order of the tasks.

- **Design 1:** *Fact*, $Expl_1$, $Expl_2$
- **Design 2:** $Expl_1$, $Expl_2$, *Fact*
- **Design 3:** $Expl_2$, *Fact*, $Expl_1$

Using Tobii studio, each first part of the designs has been annotated (e.g. Expl_1 for Design 2) as *1_Scanning*, *2_Skimming*, *3_Reading*, *4_MediaView* and *5_Unknown* by two different experts. Even if there is a guideline for annotation (Section 3.2), as human beings understanding is different from each other, we have observed some differences on annotations. In Table 3.4 matching rates of the annotations are given for all test subjects. First column defines the name of the data, second column shows the matching rates of the annotations and third column gives the number of the rows. Thus, the annotated data with the same events has been taken. For instance, the main reason of the differences on annotations is while the first expert pushes the button at 10th row and finishes at 30th row for the annotation of one of the reading behaviour, the other expert can push while he is at the 12th row and finish at 35th row. As a result, only the rows between 12th and 30th rows are considered as matched. Hence, for the test subject 21, the 63.93% of the data which has 8781 rows has been taken. Here, matching defines the intersection of matched annotations.

DataSet	Match (%)	# Rows
Proband-21-D1	63.93%	8781
Proband-25-D2	71.76%	8586
Proband-27-D1	64.02%	7283
Proband-28-D2	54.83%	6244
Proband-30-D1	56.08%	14614
Proband-31-D2	28.64%	3809
Proband-32-D3	59.00%	13993
Proband-33-D1	51.67%	9701
Proband-34-D2	55.98%	9503
Proband-35-D3	49.84%	9122
Proband-36-D1	51.75%	9723
Proband-37-D2	51.21%	5920
Proband-38-D3	35.49%	11141
Proband-39-D1	51.70%	10491
Proband-40-D2	55.50%	11500
Proband-41-D3	68.66%	8026

Table 3.4: Matching rates of the annotations.

Data Definition

Gaze points of users during search sessions can be thought as a sequence of events. In Table 3.5, an example of gaze data is given: the first column, StudioEvent, is the annotations of the five different possible states. The second, the third and the fourth columns show the index of the gaze event types respectively. Those indexes are unique for the type of its gaze. The fifth column shows the type of the gaze event. In the sixth column, there are a durations of the gaze events in ms. Lastly, the seventh column defines the absolute saccadic direction in degree.

- **StudioEvent:** Manual logging event includes the annotations of the eye movements. Those can be 1_Scanning, 2_Skimming, 3_Reading, 4_MediaView or 5_Unknown. The Gaze Plot visualizations for those eye movements can be seen in Figure 3.5, Figure 3.6 and in Figure 3.7. More plots can be found in the appendix.
- **GazeEventType:** The type of eye movement classified by Tobii Studio's fixation filter. According to this filter, an eye movement can be a Fixation, Saccade or Unclassified.
- **GazeEventDuration:** Duration of an eye movement event. For instance, a reading fixation is generally taking 220ms.
- **AbsoluteSaccadicDirection:** The angle between two fixation points where the previous fixation location is the origin of the coordinate system. This angle is calculated based on the fixation locations (Fixation Point 1 and Fixation Point 2). In Figure 3.3, there is an example of how absolute saccadic direction value is calculated and in Figure 3.4 illustrates the eyes movements during reading.

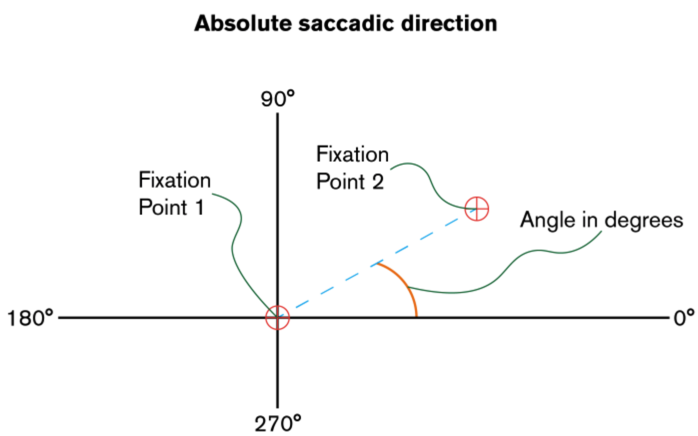


Figure 3.3: How absolute saccadic direction angle is calculated [Pro18].

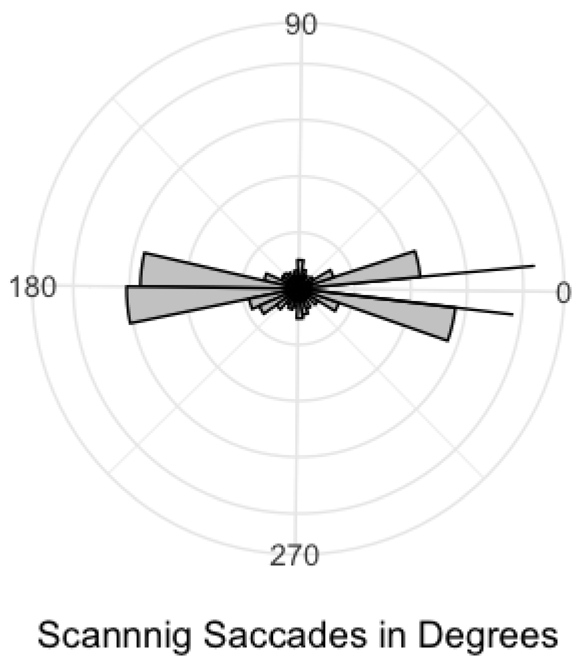


Figure 3.4: Visualizing absolute saccadic direction in 360°histogram. The angle densify in horizontal as the eyes sweeping over the text during Scanning.

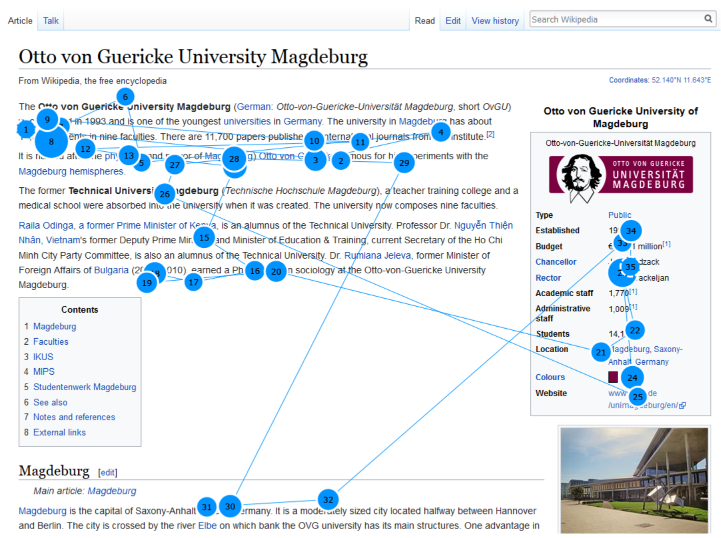


Figure 3.5: The Gaze Plot visualization exported using Tobii studio shows the sequence and position of fixations (blue dots) and saccades (lines between dots) for the eye movements of 1_Scanning and 2_Skimming.

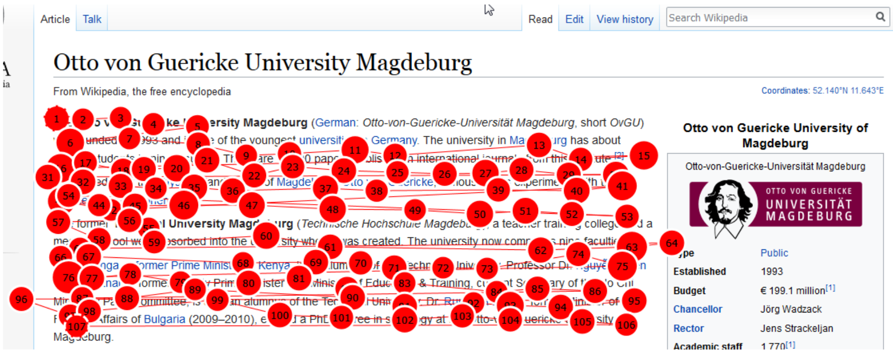


Figure 3.6: The Gaze Plot visualization exported using Tobii studio shows the sequence and position of fixations (red dots) and saccades (lines between dots) for the eye movements of 3_Reading.

Figure 3.7: The Gaze Plot visualization exported using Tobii studio shows the sequence and position of fixations (red dots) and saccades (lines between dots) for the eye movements of 4_MediaView.

Table 3.5: Example Gaze Data, extracted from Tobii Studio

StudioEvent	**FixationIndex**	**SaccadeIndex**	**UnclassifiedIndex**	**GazeEventType**	**GazeEventDuration**	**AbsoluteSaccadicDirection**
1_Scanning	6	-	-	Fixation	1117	7,94
1_Scanning	-	6	-	Saccade	33	-
1_Scanning	7	-	-	Fixation	367	87,97
1_Scanning	-	7	-	Saccade	83	-
1_Scanning	-	-	1	Unclassified	50	-
3_Reading	-	8	-	Saccade	50	-
3_Reading	8	-	-	Fixation	67	54,52
3_Reading	-	9	-	Saccade	67	-
3_Reading	9	-	-	Fixation	150	10,54
3_Reading	-	10	-	Saccade	117	-
3_Reading	10	-	-	Fixation	67	24,89
3_Reading	-	11	-	Saccade	34	-
3_Reading	11	-	-	Fixation	117	0,48
3_Reading	-	12	-	Saccade	67	-
3_Reading	12	-	-	Fixation	133	16,39
3_Reading	-	13	-	Saccade	17	-
3_Reading	13	-	-	Fixation	650	258,11
3_Reading	-	14	-	Saccade	17	-
3_Reading	14	-	-	Fixation	683	258,69
3_Reading	-	15	-	Saccade	133	-
1_Scanning	-	-	2	Unclassified	17	-
3_Reading	-	16	-	Saccade	17	-
⋮	⋮	⋮	⋮	⋮	⋮	⋮

4. Implementation

In this chapter, the Hidden Markov Model that has been introduced in Chapter 3 is implemented to classify the types of reading behavior.

4.1 Processing Data

After tracking eye movements by using Tobii Pro X2-60 eye tracker, the data become available as a list of gaze points. In Tobii Studio, refresh rates are measured in Hz and indicates the number of times per second the monitor refreshes the pixels on its display [Pro18]. Tobii studio's refresh rate is set to 60 Hz by default, that means pixels will be refreshed for each in 16.6 ms, and this causes the Tobii studio's fixation filter writes the same data in succession. For instance, in Table 4.1 a fixation that has 185 ms duration time has written 11 times ($185/16.6 = 11.1$) in a row. Besides, when the expert pushes the button for the annotation, Tobii inserts a new row with the annotation value, but that creates a problem. The problem is having different row length of the same data because of the new added rows for each pushes by Tobii Studio. Because of the nature of gaze points (i.e. a fixation cannot be followed by another fixation) a raw data needed to be cleaned to be used in HMM. To achieve that, the following steps are applied on data:

1. The rows are deleted before the "ScreenRecStarted" and after the "ScreenRecStopped" events, because the annotations are created between these two events.

2. The first 20 min of the data is taken (the first part of the search tasks where the annotations are created for).

3. Unique key values are given for unclassified gaze event. There are already given unique key values for Fixations and Saccades, but Tobii does not give one for unclassified.

4. Because of the annotation is just written on one row (the newly added row by Tobii) but needs to address all the rows until the next annotation is seen, empty columns for StudioEvent are filled by the present annotation value.

5. Only the observations with the unique key values are extracted. (As mentioned above, not all 11 rows but only the one of them is extracted.)

6. Two annotated data of the same subject are compared, and only the data has been annotated with the same events is taken.

7. The data has been separated as Train (2/3) and Test (1/3) Set to be used by HMM.

In Table 3.5, the cleaned data is illustrated.

The data are now available, consisting of fixations, saccades and unclassified. One such sequence can be written as:

$Y = (F_1, S_1, F_2, S_2, U_1 ...)$

where F_T stands for fixations consisting of its duration in ms (d_i), S_T stands for saccades consisting of its absolute saccadic direction in angle (x_i, y_i), and U_T stands for unclassified gaze type consisting of its duration in ms (d_i). From the data in Table 3.5, we can get a sequence as:

$Y = (F_{1117ms}, S_{7.94^\circ}, F_{367ms}, S_{87.97^\circ}, U_{50ms} ...)$

4.2 Building Hidden Markov Model

Hidden Markov Models (HMM) are well suited for sequence analysis tasks. Unlike the Markov Chain, HMM considers not only the state transition probabilities but also the features linked with each state. Those states consist of measurable observations. In this thesis, fixation durations, absolute saccadic directions, and unclassified durations are taken as observations.

By using the given model parameters (A, B, π), states and observations, an HMM can be computed via following joint distribution formula (Equation 4.1).

Table 4.1: Example Raw Gaze Data, written with 60Hz refresh rate by Tobii Studio

FixationIndex	SaccadeIndex	GazeEventType	GazeEventDuration
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
7	-	Fixation	185
-	7	Saccade	51
-	7	Saccade	51
-	7	Saccade	51
⋮	⋮	⋮	⋮

$$P(Z) = P(S_1) \cdot P(Y_1|S_1) \cdot \prod_{l=2}^{L} P(S_1|S_{t-1}) \cdot P(Y_l|S_l) \tag{4.1}$$

Where S indicates state space S $\in$ {*Scanning*, *Skimming*, *Reading*, *MediaView*, *Unknown*}, Y indicates observations Y $\in$ {*Fixation*, *Saccade*, *Unclassified*}, and Z is the dataset. P(S_1).P($Y_1|S_1$) is the initial probabilities π, P($S_1|S_{t-1}$) is the transition probabilities A, and P($Y_l|S_l$) is the emission probabilities B.

Architecture

Figure 4.1 shows the architecture of HMM. Oval shapes at the top represent the hidden states and at the bottom represent observations. S_t is the hidden state at time t, $S_t \in \{S_1, S_2, S_3...\}$. Y_t is the observation at time t, $Y_t \in \{Y_1, Y_2, Y_3...\}$. The arrows in the diagram show conditional dependencies. From the Figure 4.1, it is seen that the conditional probability distribution of the S_t is only depends on S_{t-1} and also Y_t only depends on S_t. By having all these information, in order to build an HMM the following features need to be defined:

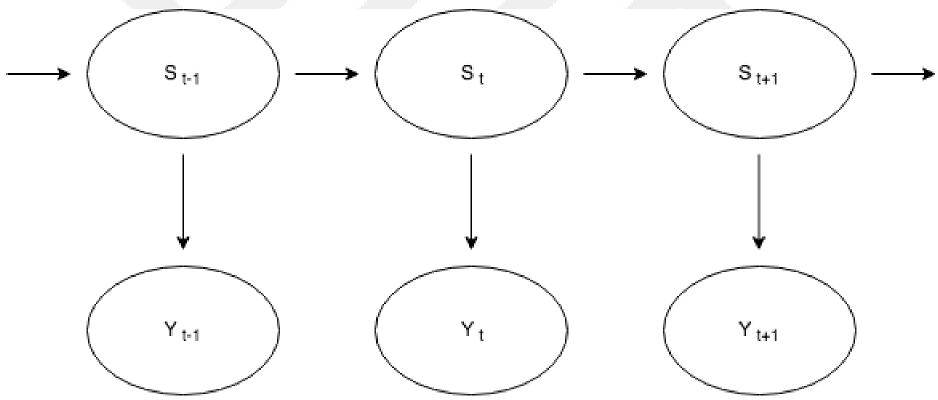


Figure 4.1: The general architecture of HMM.

- Number of states
- Probability distribution of the initial state
- Transition probability distribution
- Type of emission distributions
- Parameters for the emission distributions

Number of states

The state space that we use for the detection of the different types of reading behavior are consist of five states which are *Scanning*, *Skimming*, *Reading*, *MediaView* and *Unknown*.

Table 4.2: Initial probabilities of the gaze data. For example, the state *Skimming* has 0.1875 initial probability.

Scanning	***Skimming***	***Reading***	***MediaView***	***Unknown***
0.25	0.1875	0.09375	0	0.46875

Table 4.3: Transition probabilities of the gaze data. The state changes that goes itself again (diagonal) has higher probabilities.

	Scanning	**Skimming**	**Reading**	**MediaView**	**Unknown**
Scanning	0.978165939	0.0043668122	0.006113537	0.0004366812	0.0109170306
Skimming	0.005144995	0.9873713751	0.003274088	0.0004677268	0.0037418148
Reading	0.001833092	0.0004823927	0.996623251	0.0003859141	0.0006753497
MediaView	0.002624672	0.0026246719	0.010498688	0.9816272966	0.0026246719
Unknown	0.010728402	0.0062111801	0.005646527	0.0005646527	0.9768492377

The probability distribution of the initial state

Initial state distribution is an array that stores the probability of the start point at state i in an observation. $\pi = \pi_i$ where

$\pi = \mathrm{P}[q_1 = S_i\ ],\ 1 \leq i \leq \mathrm{N}$

One can see an example array for initial probabilities on Table 4.2. In our data, most of the state sequences are starting with the *Unknown* state, where *MediaView* has no initial.

Transition probability distribution

$\mathrm{A} = a_{ij}$ stores the probability of state j following the state i. $a_{ij} = \mathrm{P}[q_{t+1} = S_j \mid q_t = S_i]$, $1 \leq i\ ,\ j \leq \mathrm{N}$ where any state can reach any other state in a single step. If there is no transition between two states, there can be the situations where $a_{ij} = 0$. But, we know from our data that all transitions among all states are occurring. The transition from one state to another model the changes from one of the types of reading behavior to the other one. These transitions generally have a low probability (e.g. from Scanning state to Skimming state) when compared with the state changes that goes itself again (e.g. from Reading state to Reading state). Table 4.3 shows all transition probabilities between the states and Figure 4.2 presents the graphical representation of the states with their transition probabilities.

Type of emission distributions

The emission probability distributions of the states are the probability distributions over all fixation durations, absolute saccadic directions and unclassified durations. Probability distributions can be classified as continuous probability distributions or discrete probability distributions, depending on whether they are defined with continuous or discrete variables. If a variable can take any value between two specified values, it is called as continuous, otherwise, discrete [Dis].

Durations which are represented in ms and absolute saccadic directions which are represented in angle are the continuous variables, because the time can take any

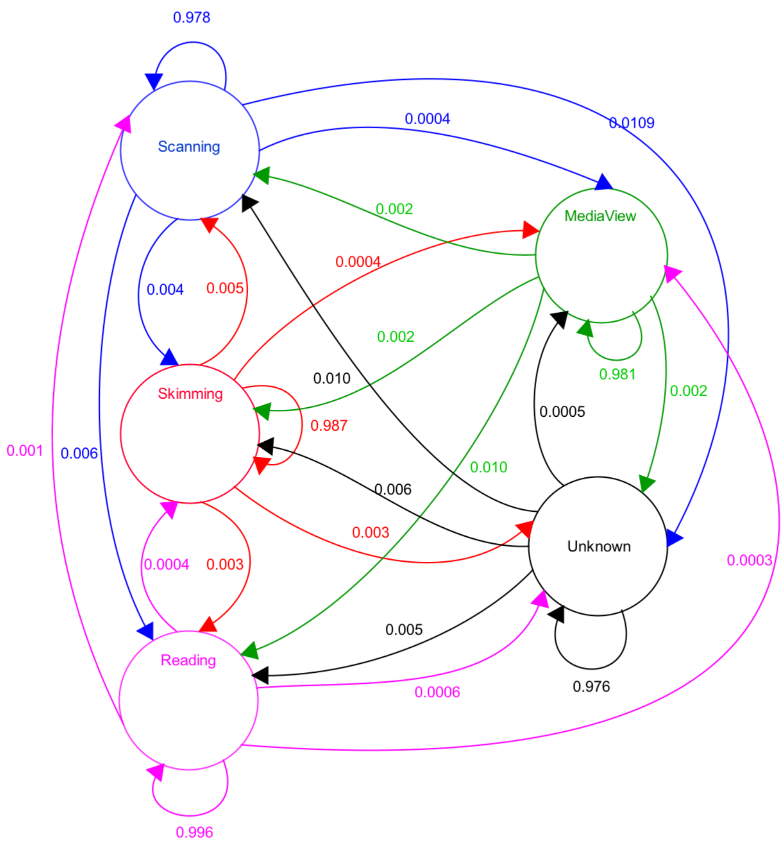


Figure 4.2: Graphical representation of the Five-State transion probabilities. States are represented in oval shapes and their transitions from one state to the another are represented by the arrows.

value as well as angle. For instance, an angle could be 52.6° or 52.3° either, that proves it can take any value between 52° and 53°.

A discrete probability distributions are different from continuous probability distributions. For instance, continuous probability distribution cannot be described in tabular form. Instead of that, in order to describe continuous probability distribution, an equation or formula is used. This equation is called a *probability density function* which is known as *pdf* [Dis]. Before modeling an HMM, we need to define *pdf*s of emission distributions. To do that, the data has been fitted to distributions.

Fitting Distributions

Fitting distributions is a task to find a mathematical function that represents a variable. In this thesis, we have tested whether our observations $y_1, y_2, y_3, ...y_T$ are being a sample of an unknown population with a *pdf*. There are commonly two steps to identify fitting distributions. Those are:

1. Exploratory data analysis with graphics

2. Goodness of fit test

1. Exploratory data analysis with graphics

Exploratory data analysis is one of the most used method. Getting statistics (skewness, kurtosis, etc.) and using graphical techniques (histogram, density, etc.) can give feeling of *pdf* to be used [Fit]. For instance, in Figure 4.3 we have plotted a histogram with its *density*. Next step is the drawing a quantile-quantile (Q-Q) plot (Figure 4.4 b), it basically compares fitted data (data has been fitted to Gamma distribution as the histogram of the data looks like Gamma distribution.) and empirical distributions in terms of the dimensional values of the variable. It is a graphical technique for determining if a data set come from a fitted population. In Q-Q plot, on the y-axis there are empirical quantiles, on the x-axis there are the ones got by the theoretical model. Besides, if the empirical data come from the population with the chosen distribution, the points should fall approximately along with reference line [Fit]. As one can see on Figure 4.4 b, points fall aproximately along with reference line which means choosen distribution is true. The *density* plot and the *Cumulative Distribution Function cdf* plot (Figure 4.4 c) can be considered as the basic of the goodness of fit plots. *Cdf* plot displays the theoretical *cdf* of the fitted distributions and the empirical *cdf* based on data. While the *pdf* graph mainly shows the shape of data, the *cdf* graph is useful to actually determine how well the distributions fit to data. The Q-Q plot emphasizes the lack of fit at the distribution tails while the P-P plot (Figure 4.4 d) emphasizes the lack of fit at the distribution center [Fit].

In addition to empirical plots, descriptive statistics may help to find distribution type, especially the skewness and kurtosis [Fit]. In order to find the distributions to be fitted the data, some values for common distributions are displayed in Figure 4.5. For some distributions (normal, uniform, logistic, exponential) represented by a single point, there is only one value for the skewness and kurtosis. For others, there is lines (gamma and lognormal) or larger areas (beta). Yet, the skewness-kurtosis plot should be regarded as indicative only [Fit].

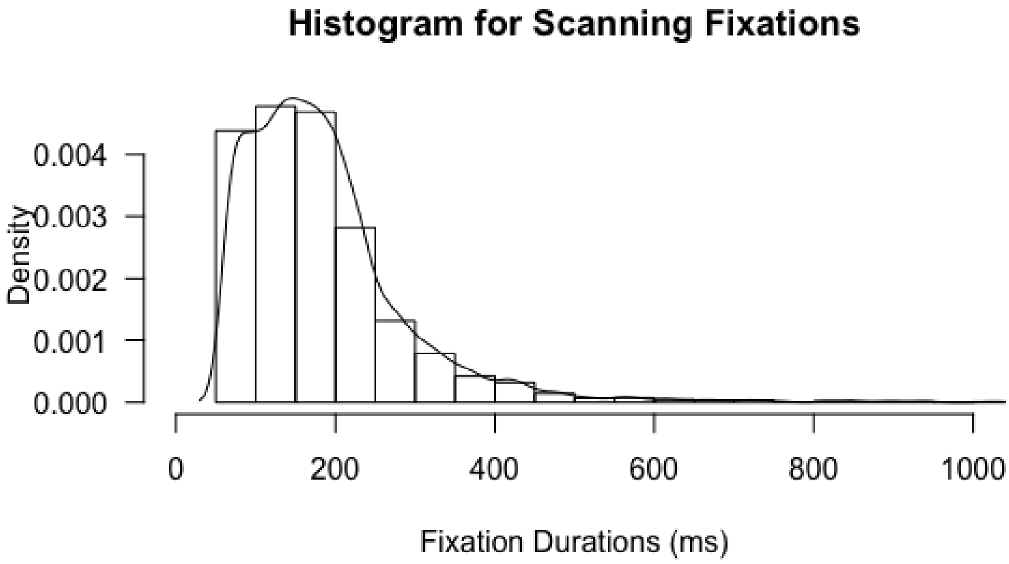


Figure 4.3: Histogram for Scanning Fixation Durations.

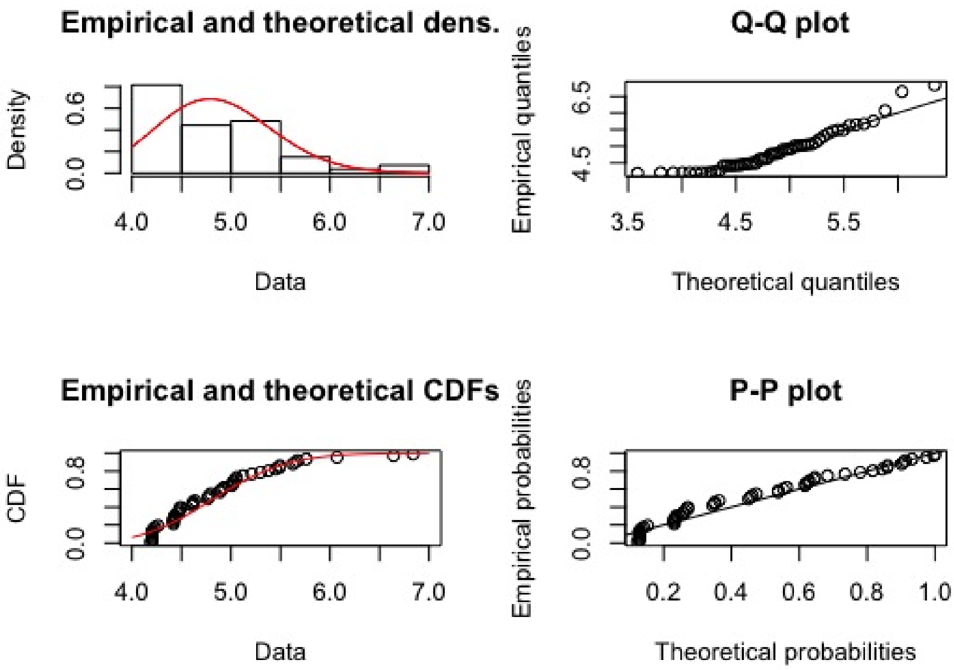


Figure 4.4: Four Goodness-of-fit plots for Gamma distribution fitted to continuous data.

2. Goodness of fit test

Histograms and other graphical techniques can help to find distribution type, but graphics could be quite subjective, so there is need to do some statistical testing to see if data is actually from the chosen population. These tests are called goodness-of-fit test. Some of these are chi-square, Kolmogorov-Smirnov, Anderson-Darling, and Shapiro-Wilk. All of these tests are for statistical null hypothesis. The following hypothesis is applied to goodness-of-fit test:

H_0 = The data is suitable for a specified distribution.

H_1 = The data is NOT suitable for a specified distribution.

Statistical significance should be defined as a threshold for the null hypothesis. This value depends on the application but it is usually considered as 0.05. If the result of goodness-of-fit test value (p-value) is greater than the significance level, we can conclude that observed frequencies are not significantly different from the expected frequencies given in the null hypothesis. Thus, we applied Kolmogorov-Smirnov test on the observations. As you can see the application of Kolmogorov-Smirnov test on *Scanning Fixations* durations in Figure 4.6, the p-value is greater than significance value which means we accept the null hypothesis. Hereby, we can conclude that *Scanning Fixations* durations are significantly similar to gamma frequencies that we specified. Not only the *Scanning Fixations* but also other types of fixations also show the same distribution type. In Figure 4.7 histograms are plotted with their *gamma pdf*s for all the types of reading behavior. We can also provide a proof from literature; according to Simola [SSK08], a gamma distribution has often been used for modeling fixation durations, because its negatively skewed distribution resembles the data.

Parameters for the emission distributions

When the distribution type is identified, we can estimate the parameters of the chosen distribution. In previous section, we have found that fixation durations show gamma distribution. The gamma distribution is a two-parameter family of continuous probability distributions [Wik18a]. There are three different forms of gamma distribution parameters, which are:

1. Shape parameter k and scale parameter θ

2. Shape parameter k and rate parameter β

3. Shape parameter k and mean parameter μ

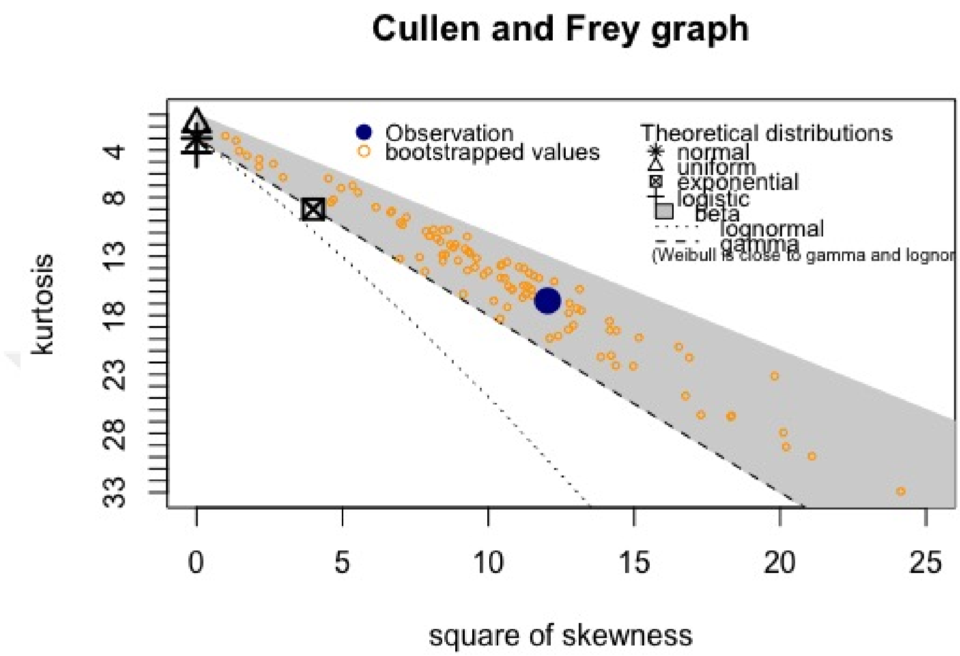


Figure 4.5: Skewness-kurtosis plot for a continuous variable; Scanning Fixation Durations.

```
> ks.test(scanningfixations,"pgamma",fit.gamma$estimate[1],1/fit.gamma$estimate[2])

        One-sample Kolmogorov-Smirnov test

data:  scanningfixations
D = 0.080265, p-value = 0.1891
alternative hypothesis: two-sided
```

Figure 4.6: Kolmogorov-Smirnov Test.

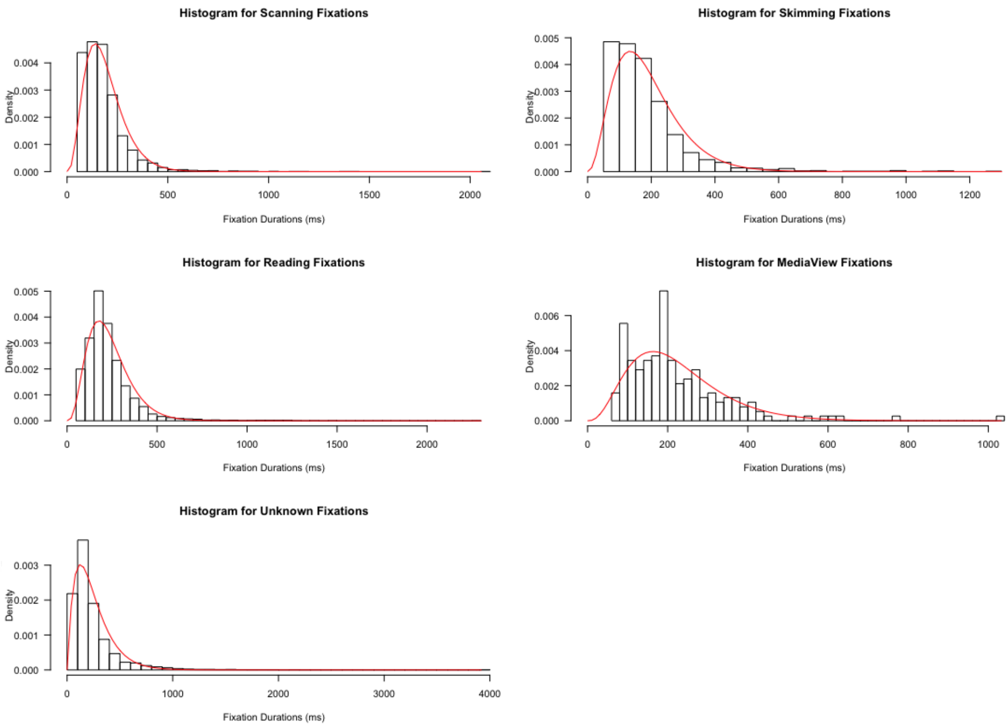


Figure 4.7: Histograms for the types of reading behavior with their gamma pdfs.

Those all parameters are real numbers. Rate parameter is the inverse of scale parameter $\beta = 1/\theta$ and mean is $\mu = k/\beta$. In this thesis, we used the first form, shape and scale as the parameters of gamma distribution. In order to model HMM, we need to find probability density function of the emission distribution. An observation O which is gamma distributed with shape k and scale θ is denoted by:

$$O \sim \Gamma(k, \theta) \tag{4.2}$$

and the corresponding probability density function is:

$$f(x; k, \theta) = \frac{x^{k-1}e^{-x/\theta}}{\theta^k \Gamma(k)} \tag{4.3}$$

By having the Equation 4.3, the gamma distribution parameters with *Maximum Likelihood Estimate (MLE)* using the *Newton-Raphson method* can be estimated. This method is a technique for solving equations numerically. Simply solving the equation Equation 4.4 where the first derivative Equation 4.5 and the second derivative Equation 4.6 and stop the approximation when convergence value is less than 0.0000001, the MLE of shape and scale can be found. Equation 4.4 refers to line 21, Equation 4.5 refers to line 17 and Equation 4.6 refers to line 19 in Listing 4.1 pseudocode.

for k 1,2,...

$$shape^k = shape^{k-1} - \frac{l'(shape^{k-1})}{l''(shape^{k-1})} \tag{4.4}$$

where

$$l'(a) = nlog(\frac{a}{\bar{X}}) - n\frac{\Gamma'(a)}{\Gamma(a)} + \sum_{i=1}^{n} logX_i \tag{4.5}$$

$$l''(a) = \frac{n}{a} - n(\frac{\Gamma'(a)}{\Gamma(a)})' \tag{4.6}$$

```
function gamma_MLE(x) : shape, scale
    n <- length(x) # length of the observations
    mean_x <- mean(x) # mean of the observations

    #initiate the shape, scale, and convergence values
    shape_prev <- n*(mean_x^2)/sum((x-mean_x)^2)
    scale_prev <- sum((x-mean_x)^2)
    converg <- 1000

    # initiate two vectors to store alpha and beta in each step
    shape_est<-shape_prev
    scale_est<-mean_x/shape_prev

    # Newton-Raphson
    while(converg>0.0000001){
        #first derivative of shape_k-1
        der1<-n*log(shape_prev/mean_x)-n*digamma(shape_prev)+sum(log(x))
        #second derivative of shape_k-1
        der2<-n/shape_prev-n*trigamma(shape_prev)
        #calculate next shape
        shape_next<-shape_prev-der1/der2
        # get the convergence value
        converg<-abs(shape_next-shape_prev)
        # store estimators in each step
        shape_est<-c(shape_est, shape_next)
        scale_est<-c(scale_est, mean_x/shape_next)
        # go to next alpha
        shape_prev<-alpha_next
  }
  shape<-shape_next
  scale<-mean_x/shape_next

  return(shape,scale)
}
end function
```

Listing 4.1: Estimating gamma distribution parameters using MLE

The types and parameters of the other emissions (Saccade: Absolute saccadic direction and Unclassified: Duration gaze events) have been found with the same techniques that have been applied for the Fixations. In Figure 4.9, it can be seen that absolute saccadic directions show the multimodal Gaussian distribution. In order to find modes and parameter values of the distribution, R package [SFMR16] has been used. In Figure 4.8, as it is seen there are eight modes for the distribution that shows multimodal Gaussian distribution is going to have eight mean values and eight standart deviation values. The goodness of fit test also has been applied to Unclassified durations and its distribution has been found as exponential distribution where the plots are given in Figure 4.10.

```
----------------------------------------------------
Density estimation via Gaussian finite mixture modeling
----------------------------------------------------

Mclust V (univariate, unequal variance) model with 8 components:

 log.likelihood    n df       BIC      ICL
      -38302.41 7333 23 -76809.52 -79406.2

Clustering table:
   1    2    3    4    5    6    7    8
 902  810  779 1192 1038  838  816  958
```

Figure 4.8: Density estimation for the Absolute saccadic direction values.

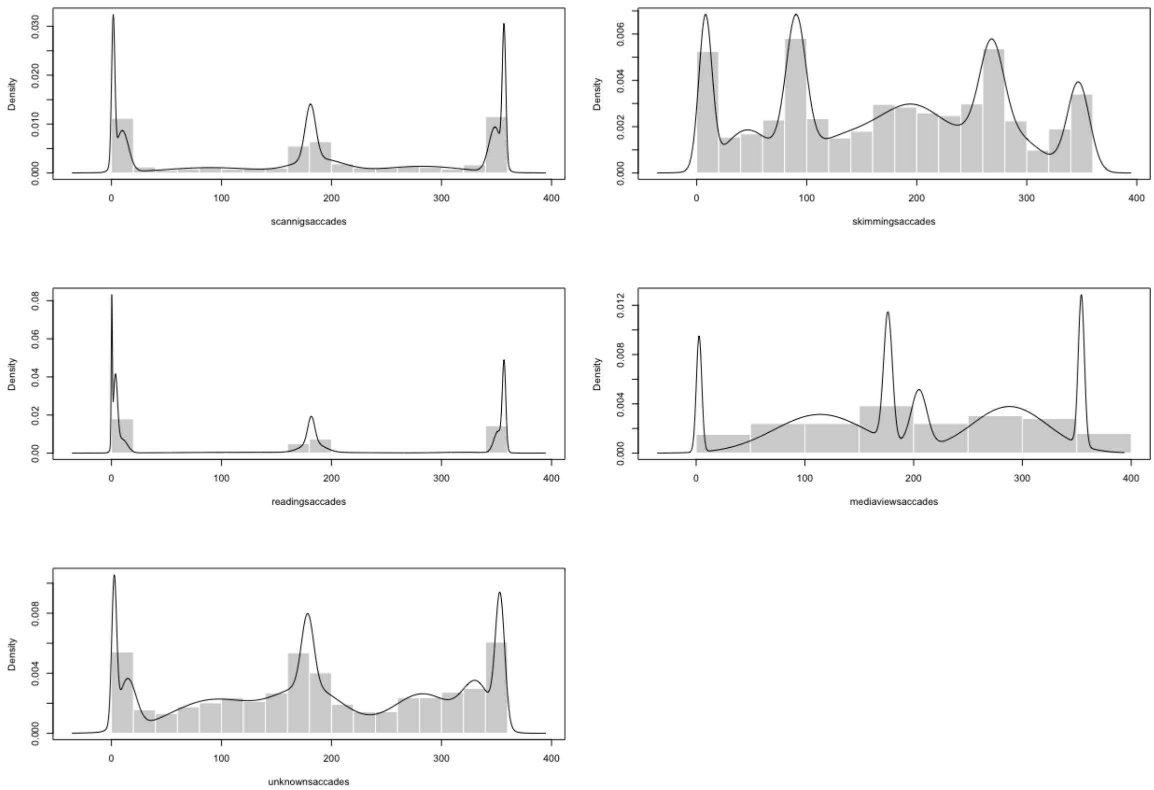


Figure 4.9: Histogram for Absolute Saccadic Directions.

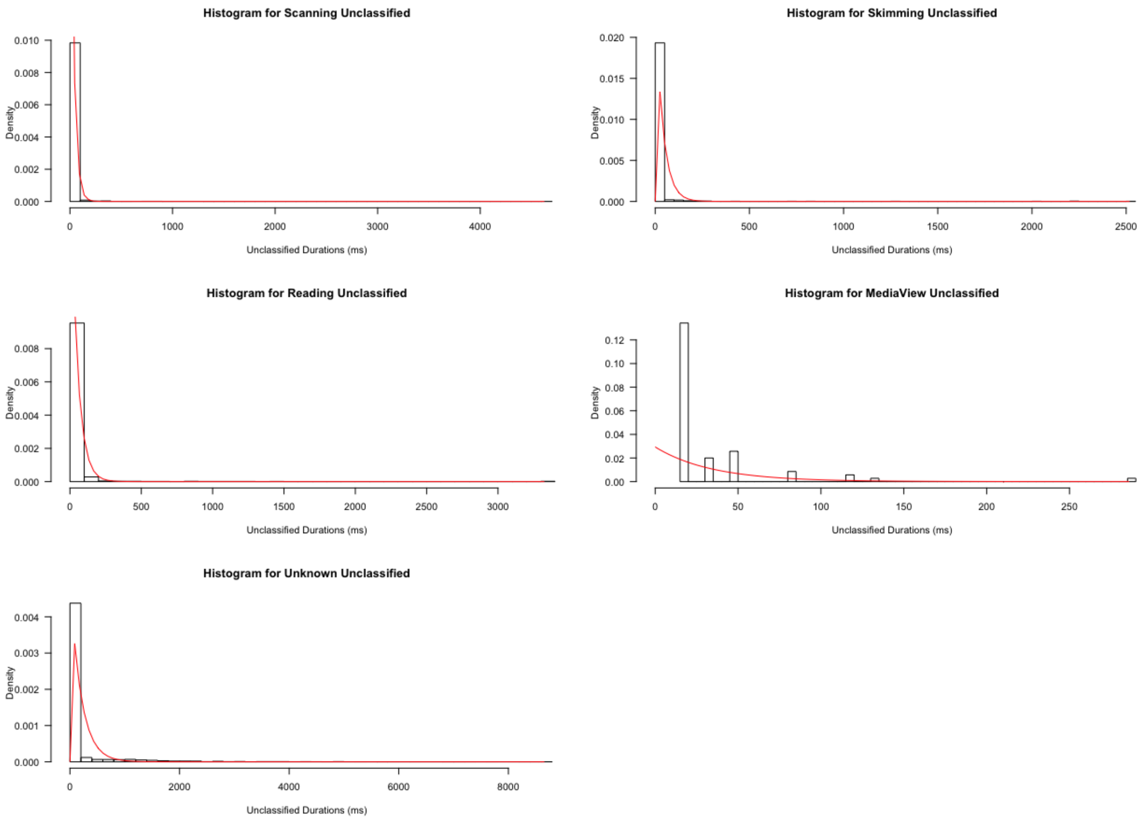


Figure 4.10: Histogram for Unclassified Durations.

4.3 Learning HMM

When building an HMM there are two parts which are very important: first, the design of the structure (e.g. what states are there and how they are connected). Second, parameter values of the HMM; the initialization, transition, and emission probabilities. In this section, we will discuss how the parameters are estimated, in other words, we will be showing how the HMM is learned.

We have a set of training sequences; $x^1,\ldots,x^n$ and the joint probability of all the sequences are the product of the probabilities of the individual sequences. Thus, we can calculate the log probability of the sequences:

$\log(x^1,\ldots,x^n|\theta) = \log \mathrm{P}(x^1,\ldots,x^n|\theta) = \sum_{j=1}^{n} \log \mathrm{P}(x^j|\theta)$

where θ represents the entire parameters of the model (i.e. transition, emission). This is equal to the log likelihood of the model.

Estimation of the parameters can be grouped into two categories. Those are;

1. Estimation when the state sequence is known (Supervised);
   - MLE
2. Estimation when the state sequence is unknown (Unsupervised);
   - Baum-Welch

– Viterbi training

In this thesis, as our sequences are fully annotated, we are going to train our model with the first method; MLE. When the paths are unknown for the training sequences *Baum-Welch* or *Viterbi training* algorithms can be used. When all the sequences are known, we can count the number of times where each transition is used by the training sequences and for the emission we have already estimated maximum likelihood in Section 4.2. Let it for transition A_{ij} from state i to j, then the maximum likelihood estimation is given as:

$$a_{ij} = \frac{A_{ij}}{\sum_{j'} A_{ij'}} \tag{4.7}$$

By the MLE method, we look for parameter $\theta(a_{ij})$ which maximize the probability of the training set. If there are insufficient data, maximum likelihood estimation is vulnerable to overfitting [DREKJM98]. For instance, if there is a state i that never occurs in the training sequences, then the estimation equations are undefined for that state because of the result of zero divided by zero is indeterminate [HMMa]. In order to avoid from such problems Equation 4.7 can be smoothed by adding some pseudo counts before using it.

- A_{ij} = number of transitions i to j in training data + γ_{ij}

With this way each transition probability can be guaranteed to be greater than zero and this also guarantees that transition occurring in the training data always has a higher probability than one which does not occur.

4.4 Getting most likely sequences: Viterbi Decoding

Although it is not possible to identify what state the model in by looking at the corresponding observation, we are interested in sequence of states. In order to find what the observation sequence "means" by considering the states is called *decoding* in the field of speech recognition [DREKJM98]. There are many ways for decoding, in this thesis we will use the most common one, called the *Viterbi algorithm*. In Chapter 3, we gave the background information for the algorithm.

The most likely sequence can be found recursively. For instance, for each possible hidden state sequence, we run the *Viterbi algorithm* and compute the likelihood of the observation sequence given the hidden state sequence.

Figure 4.11 shows an example of the Viterbi trellis for computing most likely sequence for the observation sequence at time *t1*; *Fixation* and time *t2*; *Saccade* and so on. The detailed algorithm can be found in Chapter 3. The idea is to process the observation sequence left to right. Each cell, *vt(j)* represents the probability that the HMM is in state j after seeing the first t observations and passing through

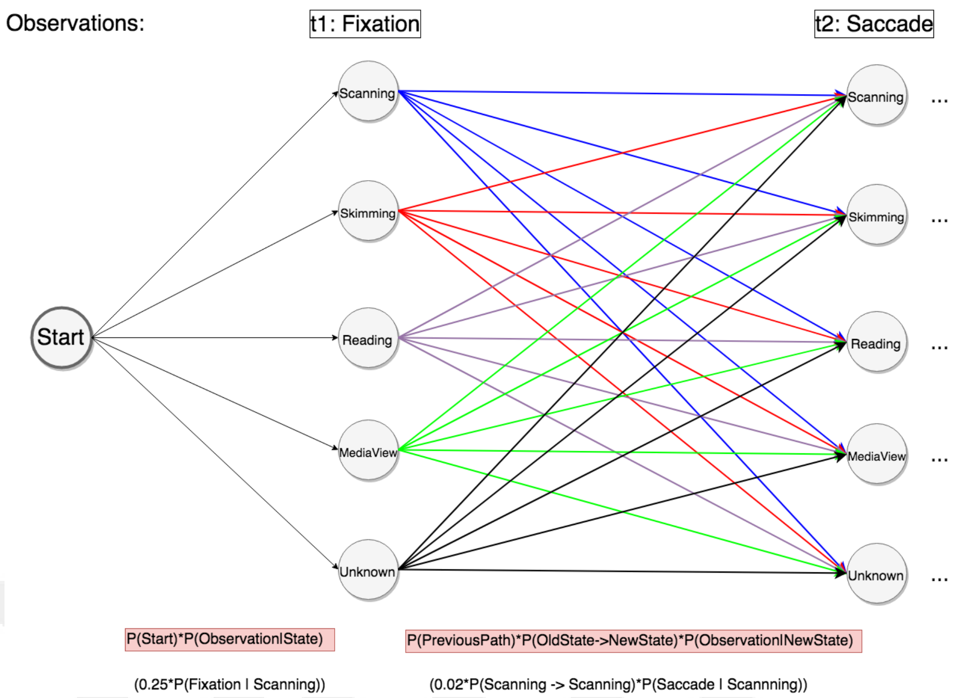


Figure 4.11: The Viterbi trellis for computing most likely sequence through the hidden states. Hidden states are in circles, observations in rectangle.

the most probable state sequence $q_0, q_1, q_2 ...., q_{t-1}$, given the automaton θ [HMMb]. For instance, for the first cell *vt(Scanning)*, by seeing the observation as *Fixation*, we choose the maximum value according to Equation 4.8. The three probabilities ($vt-1$, a_{ij} and $b_j(o_t)$) are multiplied for extending the previous path to compute the Viterbi probability.

$$vt_j = maxv_{t-1}(i)a_{ij}b_j(o_t) \tag{4.8}$$

- $vt-1$ = the previous Viterbi path probability
- a_{ij} = the transition probability from the Old State to the New State
- $b_j(o_t)$ = the state observation likelihood of the observation o_t given the current state j.

This methodology is implemented in given R code: Listing 4.2. In the first 18 code lines, the HMM is calculated according to the given observations (see the first column of Table 4.4); for instance, if the observation is *Fixation* then *pdf* of gamma is used. Hence, "dens.emission" of the models (see lines 6, 11 and 16 on Listing 4.2) calculates the density values of states. When the p matrix (see Table 4.4) with five columns (number of states) and N rows (number of rows in Test data) is completed, the Viterbi as seen in the code lines between 19-26 can be applied; for every sequence Viterbi is calculated with using initial probabilities, transition probabilities and with p matrix that stores *pdf* values of distributions. In line 19, length(x$N) represents the number of sequence in Test data. In lines between 20 and 25, there is an internal code that is imported from C code that does all calculations. NN is the cumulative sum of the sequences, for instance, let's say that the first sequence is consist of 94 rows so tmp will be calculated for p[(NN[i]+1):NN[i+1],] that equals to p[1:94]. After that, maximum values of states in tmp is chosen to assign states: state[1:94] = tmp$q+1 in code line 25.

Listing 4.2: Computing Viterbi probability according to observations

```
1  p = matrix(nrow=sum(x$N), ncol=5)
2  for (i in 1:sum(x$N)) {
3   if (obs[i]=="Fixation") {
4    p[i,] <- sapply(1:5,
5    fn <- function(state)
6    modelgamma$dens.emission(x$x[i],state,modelgamma))
7   }
8   else if (obs[i]=="Saccade") {
9    p[i,] <- sapply(1:5,
10   fn <- function(state)
11   modelnorm$dens.emission(x$x[i],state,modelnorm))
12  }
13  else{
14   p[i,] <- sapply(1:5,
15   fn <- function(state)
16   modelexp$dens.emission(x$x[i],state,modelexp))
17  }
18 }
19 for(i in 1:length(x$N)) {
20   tmp <- .C("viterbi_hmm",a=logtrans,pi=logpi,
21   p=as.double(log(t(p[(NN[i]+1):NN[i+1],]))),
22   N=as.integer(x$N[i]),NN=as.integer(nseq),K=as.integer(5),
23   q=as.integer(rep(-1,x$N[i])),loglik=as.double(c(0)),PACKAGE='mhsmm')
24   loglik=loglik+tmp$loglik
25   state[(NN[i]+1):NN[i+1]] = tmp$q+1
26 }
27 ans <- list(s=state,x=x$x,N=x$N,loglik=loglik)
```

After prediction is completed by *Viterbi algorithm*, we can evaluate the accuracy of the model by comparing the actual annotations in test data with Viterbi prediction. In the next sextion, we will evaluate our reading detection model.

Table 4.4: The head of the p matrix. Each row represents the *pdf* value of the given observation for the five states. For instance, the first observation is Saccade so the first line values are calculated with the type of Saccade distribution which is multimodal Gaussian.

States / **Observations**	**Scanning**	**Skimming**	**Reading**	**MediaView**	**Unknown**
[1] **Saccade**	1.458293e-49	2.364287e-54	4.791658e-46	1.839593e-99	1.450633e-76
[2] **Fixation**	2.743948e-03	3.554251e-03	2.645266e-03	3.356914e-03	3.390768e-03
[3] **Saccade**	2.677557e-83	6.506183e-97	3.359612e-77	1.390294e-164	2.836231e-128
[4] **Fixation**	2.469367e-03	3.209230e-03	2.358244e-03	4.082267e-03	3.527636e-03
[5] **Saccade**	1.045819e-85	5.430100e-100	2.007842e-79	3.327712e-169	5.894765e-132
[6] **Fixation**	1.184882e-03	1.075008e-03	1.155198e-03	1.511598e-03	1.402613e-03
⋮	⋮	⋮	⋮	⋮	⋮

5. Evaluation

In this chapter, the evaluation of our reading detection approach and our baseline against it are presented. In order to compare standard HMM with another method, we used the naive Bayes as a baseline. Because of the time limitation, instead of implementing more complicated model as a baseline, the less complicated model is chosen and the comparison between these two models is made.

In naive Bayes, we simply remove the dependencies between the states. So, there is no successor independent of predecessor. In Figure 5.1 we see the general architecture of the naive Bayes. So, the formulation of the model;

$$P(Z) = \prod_{l=1}^{L} P(S_l) \cdot P(Y_l|S_l) \tag{5.1}$$

When compared with Equation 4.1, we only remove the Markovian chain because this dependency decouples here independence. However, the emission part of the formula stays the same. In other words, we downgrade the standard HMM.

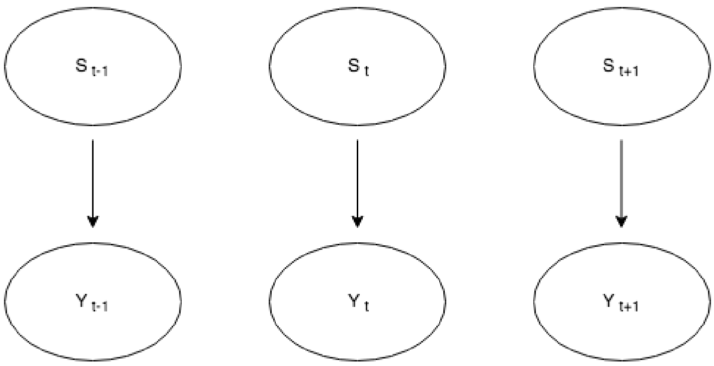


Figure 5.1: Architecture of naive Bayes. Downgraded from Standard HMM.

The experiments were carried out on double core of an Intel Core i5 MacBook Pro with 2.6 GHz of the clock and 8 GB of RAM memory on OS X High Sierra operating system. Furthermore, the algorithms have been implemented in R programming and compiled with RStudio.

In order to measure the performance of an HMM, the evaluation metrics such as Accuracy, Precision, Recall and F1 Score are used. Besides, a brief explanation of the “confusion matrix” are provided. In this thesis, we have used DKE dataset from [KGSN17a] to classify the types of reading behavior among *scanning*, *skimming*, *reading*, *mediaview* and *unknown*.

Once the model is built, the most important question arises how good is the model? Thus, evaluating the model takes an important part to see how good the classification is.

A confusion matrix is a table that is generally used to measure the performance of a classification model on a test data for which the real values are known [Per]. There are four parameters to be extracted from a confusion matrix, those are:

True positive TP These are the correctly predicted positive values which mean that the value of the actual class is yes and the value of the predicted class is also yes. E.g. if the actual class value indicates the reader is doing *scanning*, predicted class tells you the same thing [Per].

On multiclass classification, TP values are the diagonal column cells of the confusion matrix.

True negative TN These are the correctly predicted negative values which mean that the value of the actual class is no and the value of the predicted class is also no. E.g. if the actual class says the reader is not doing *scanning*, the predicted class tells you the same thing [Per].

On multiclass classification, TN values are the correctly rejected prediction for a certain class. For instance, for a *scanning*, it is all the values except the real class row of the *scanning* and prediction column of the *scanning* on the confusion matrix.

False Positives FP When the actual class is no and the predicted class is yes [Per]. E.g. if the actual class says reader is not doing *scanning* but the predicted class tells you that the reader is doing *scanning*.

On multiclass classification, FP values are incorrectly identified as predictions for a certain class. For instance, for a *scanning*, it is prediction column of the *scanning* except for the first cell on the confusion matrix.

False Negatives FN When actual class is yes but predicted class in no [Per]. E.g. if the actual class value indicates that the reader is doing *scanning* and predicted class tells you that the reader is doing *skimming*.

On multiclass classification FN, values are for a certain class. For instance, for a *scanning*, it is the real row of the *scanning* except for the first cell on the confusion matrix.

In Figure 5.2, the graphical representation of how these parameters are calculated on confusion matrix is illustrated.

Once these four parameters are counted, the following metrics can be calculated:

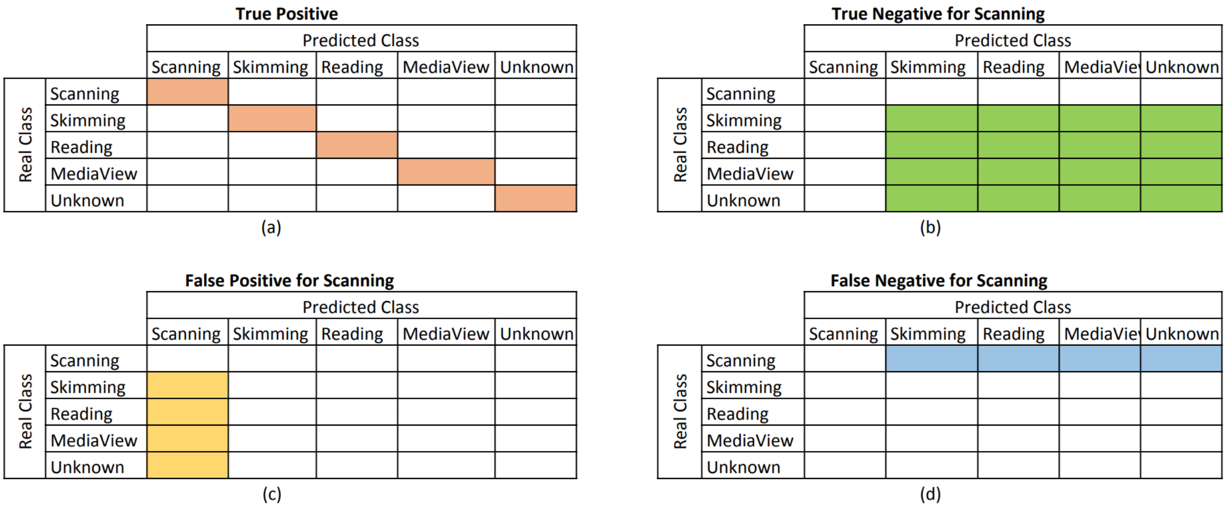


Figure 5.2: How the parameters of the confusion matrix are calculated. For example, the sum of each colored cells gives the parameter values of the *Scanning*.

Accuracy

Accuracy is the performance measure and it is simply the ratio of correctly classified states to the total number of classifications [Per]. One may assume that the model is the best when it has high accuracy but this is the case when the datasets are symmetric where the values of false positives and false negatives are almost the same [Per]. Hence, not only the accuracy but also the other evaluation metrics (Precision, Recall and F1 score) should be considered to evaluate the performance of the model. Accuracy can be defined in a formula as follows.

$$\text{Precision} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}}$$

Precision

This measure represents the percentage of the positive classifications that were correct and it is defined as follows.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

For all five states, each precision values are calculated and sum of them are divided into the number of states to get an average precision. On Table 5.3 and Table 5.4 one can see the precision values of each state of the models.

Recall

This recall measure (also called sensitivity) represents how many percents of the actual positive cases that were classified correctly and it is defined as follows.

Table 5.1: Confusion Matrix for standard HMM

		Predicted Class					
		Scanning	Skimming	Reading	MediaView	Unknown	*Row Sum*
Real Class	Scanning	3474	110	13	19	10	3626
	Skimming	620	2063	158	1	15	2857
	Reading	35	160	3983	0	132	4310
	MediaView	372	0	0	424	4	800
	Unknown	14	41	270	0	8369	8694
	Column Sum	4515	2374	4424	444	8530	

Table 5.2: Confusion Matrix for NB-HMM

		Predicted Class					
		Scanning	Skimming	Reading	MediaView	Unknown	*Row Sum*
Real Class	Scanning	1601	51	1154	9	811	3626
	Skimming	424	1408	316	0	709	2857
	Reading	574	277	2262	230	967	4310
	MediaView	62	98	147	411	82	800
	Unknown	272	174	1005	84	7159	8694
	Column Sum	2933	2008	4884	734	9728	

$$\text{Recall} = \frac{\text{True Positives}}{\text{True positives} + \text{False negatives}}$$

For all five states, each recall values are calculated and sum of them are divided into the number of states to get an average recall. On Table 5.3 and Table 5.4 one can see the recall values of each state of the models.

F1 Score

The F1 score is defined as:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

which is the harmonic mean of the precision and recall measures. This measure is often used to get a single score for a classifier to simplify comparison.

Test results

Given metrics above, to measure the performance we need to create a confusion matrix. In Table 5.1 and Table 5.2, we show the confusion matrices of standard HMM and naive Bayes respectively. The confusion matrix basically shows how many times a real class member (e.g *Scanning*) predicted as other class members.

By having the confusion matrices, precision and recall values are calculated for each state (Table 5.3 and Table 5.4) and then the average values with the F1 score are given in Table 5.5.

Moreover, the two models are compared on the graphical representation. In Figure 5.3 and Figure 5.4, horizontal bars with the different states whose colors are

Table 5.3: The precision and recall values of the states of standard HMM with their average values.

Standard HMM		
States	**Precision**	**Recall**
Scanning	0.76943522	0.95808053
Skimming	0.86899747	0.7220861
Reading	0.90031646	0.92412993
MediaView	0.95495495	0.53
Unknown	0.98112544	0.9626179
Average	0.89496591	0.81218289

Table 5.4: The precision and recall values of the states of naive Bayes with their average values.

Naive Bayes		
States	**Precision**	**Recall**
Scanning	0.54585748	0.44153337
Skimming	0.70119522	0.49282464
Reading	0.46314469	0.52482599
MediaView	0.5599455	0.51375
Unknown	0.73591694	0.82344145
Average	0.60121202	0.55927509

Table 5.5: Given the confusion matrices, the values of the evaluation metrics.

	Test Results			
	Accuracy	Precision	Recall	F-measure
Standard HMM	0.90269631	0.89496591	0.81218289	0.85156724
Naive Bayes	0.63296692	0.60121202	0.55927509	0.57949648

different for each state are presented. As standard HMM is state dependent (see Equation 4.1), state changes are not very frequent and it is easy to observe state matches with colors in Figure 5.3. On the other hand, as the naive Bayes is independent of predecessor states (see Equation 5.1) it is not easy to observe state matches.

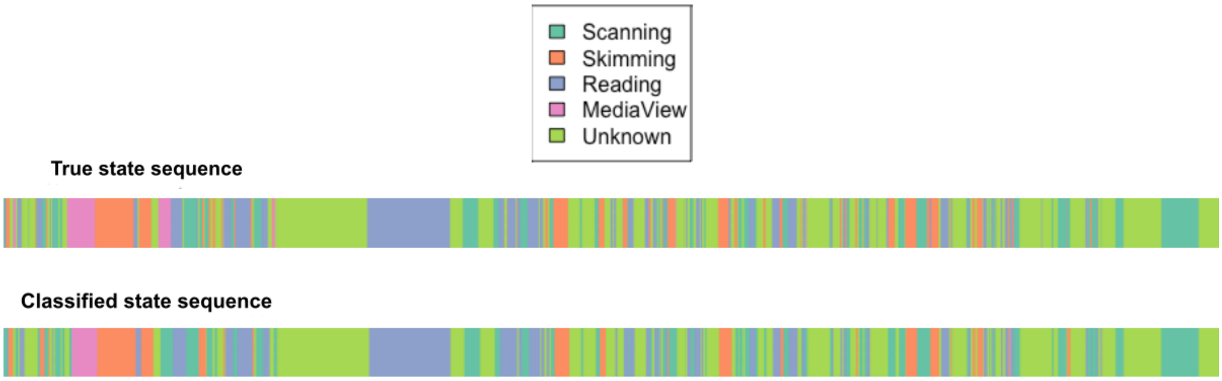


Figure 5.3: Comparison of test and classified set for Standard HMM. The horizontal bars show the different states whose colors are defined in the legend.

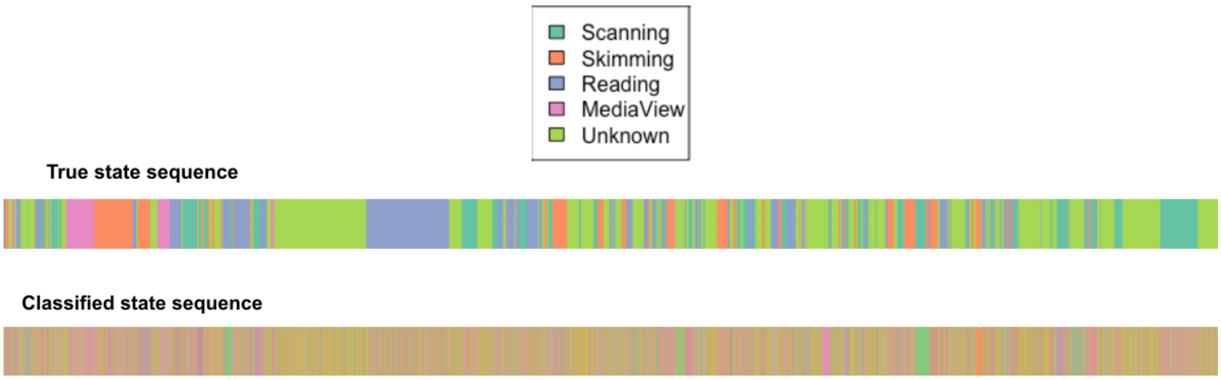


Figure 5.4: Comparison of test and classified set for naive Bayes. The horizontal bars show the different states whose colors are defined in the legend. As the classified state changes are very frequent, it is hard to observe state matches with color.

6. Conclusion and Future Work

Conclusion

In this thesis, we presented a reading model for gaze data to classify different types of human reading behavior. One of the main contributions of our work is to detect and distinguish *scanning*, *skimming*, *reading*, *mediaview* and *unknown* reading behavior from each other.

Gaze patterns have the information needed to recognize and distinguish the types of reading behavior. Thus, to have labeled data, those patterns have been labeled by two different experts. Common labels of the experts are accepted as the most accurate conclusion of the gaze patterns.

From an experimental point of view, our contribution lies in the comparison of the performance of classifiers in both standard HMM and naive Bayes. Considering the results, we answer the research questions of our thesis as follows:

RQ1. How accurate the developed reading model correctly detects reading behavior when compared with the labeled data?

In this thesis, *reading* is distinguished from *scanning* and *skimming*. However, for a simplicity *mediaview* and *unknown* can be grouped as "*non-reading*" and the rest of the types as "*reading*". However, even if the *reading* is considered together as *scanning*, *skimming* and *reading*, reading alone behavior shows better outcome where the evaulation metric values (precision and recall respectively) of the *reading* alone behavior is 0.900 and 0.924. Its evaulation metric values are better than the average values of *scanning*, *skimming* and *reading* all together (0.846 and 0.868 respectively).

RQ2. How good the model is able to distinguish between the types of reading behavior?

In confusion matrix Table 5.1, we can see that our developed model detects reading behavior with the 90% accuracy. The diagonal column cells of the confusion matrix give us the correctly detected types of reading behavior when compared with the labeled data.

RQ3. How effects of using the developed model transition compared to the baseline?

The model has been evaluated comparing with the naive Bayes and results (Table 5.5) show that while standard HMM shows 90% accuracy, our baseline the naives Bayes, on the other hand, shows 63% accuracy. Thus, we can say that standard HMM classifier performs better than the naive Bayes classifier. Precision, Recal, and F-Measure values of the standart HMM also outperform the naive Bayes . Consequently, using the transition probability of the states helps to classify the types of reading behavior much more accurate.

Furthermore, the another contribution of the thesis is that, the gaze data has been fitted to its distributions; Fixation durations of the gaze data show Gamma distribution, where Saccade length show multimodal Gaussian distribution and Unknown durations show Exponential distribution. Hence, the parameters of the distribution types have been found with *MLE*, and estimated the parmeters *shape* and *scale* on Gamma distributed Fixation durations data using the *Newton-Raphson method*. The parameters of the multimodal Gaussian; *mean* and *sigma*, and the parameter of the Exponential; *rate* also have been estimated with *MLE* after fitting their data to their distribution types.

Future Work

Many different experiments, tests, and implementations have been left for the future due to the time limitation of the thesis. Future work is related to the improvement of a reading detection model, new proposals to try different methods, or simply curiosity. The performance of the reading detection model can be improved in several ways.

One of the limitations of the standard HMM is that it is weak at capturing the long-range relations between the observed values [Bis06]. Thus, standard HMM can be improved by adding extra links to the Figure 4.1. One of the solutions is to extend the standard HMM to give the Autoregressive hidden Markov model, an example is shown in Figure 6.1.

Given a hidden state sequence of the model;
$Q = Q_{1:T}$ where $Q \in \{Scanning, Skimming, Reading, MediaView, Unknown\}$
and an observed feature sequence;
$C = C_{1:T}$ where $C \in \{\text{Fixation, Saccade, Unclassified}\}$
A joint probability can be defined as in Equation 6.1. It can be seen from the formula that when compared with standard HMM the only changes is in the emission part as it depends on both current state and all past output.

$$P(C, Q) = \prod_t P(Q_t|Q_{t-1}) \cdot P(C_t|C_{1:t-1}, Q_t) \tag{6.1}$$

In this thesis, we have used duration feature for Fixation and Unclassified observations, and absolute saccadic direction for Saccade observation. To further improve, the combination of some features such as saccadic amplitude, and relative saccadic

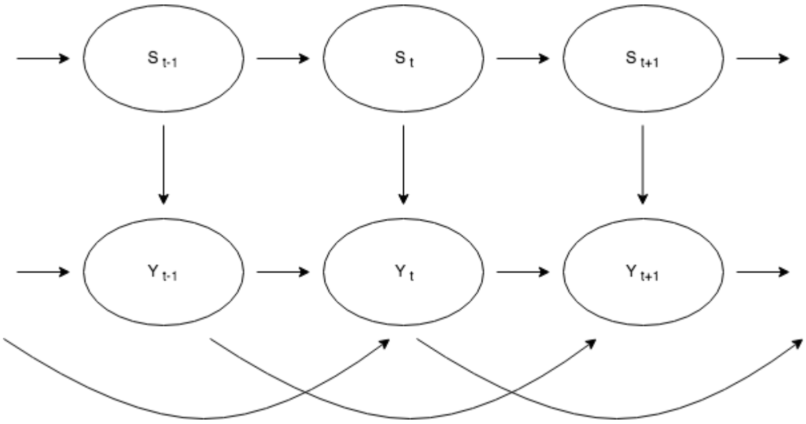


Figure 6.1: Architecture of Autoregressive HMM, in which the distribution of the observation Y_t not only depends on a subset of previous observations but also depends on the hidden state S_t. In this figure, Y_t depends on the two previous observations Y_{t-1} and Y_{t-2}. This number of dependency can be increased.

direction (definitions of the terms can be found in [Pro18]) for saccade can be used. Furthermore, character-based reading classifier (e.g. [BDvE08]) can be combined to our model.

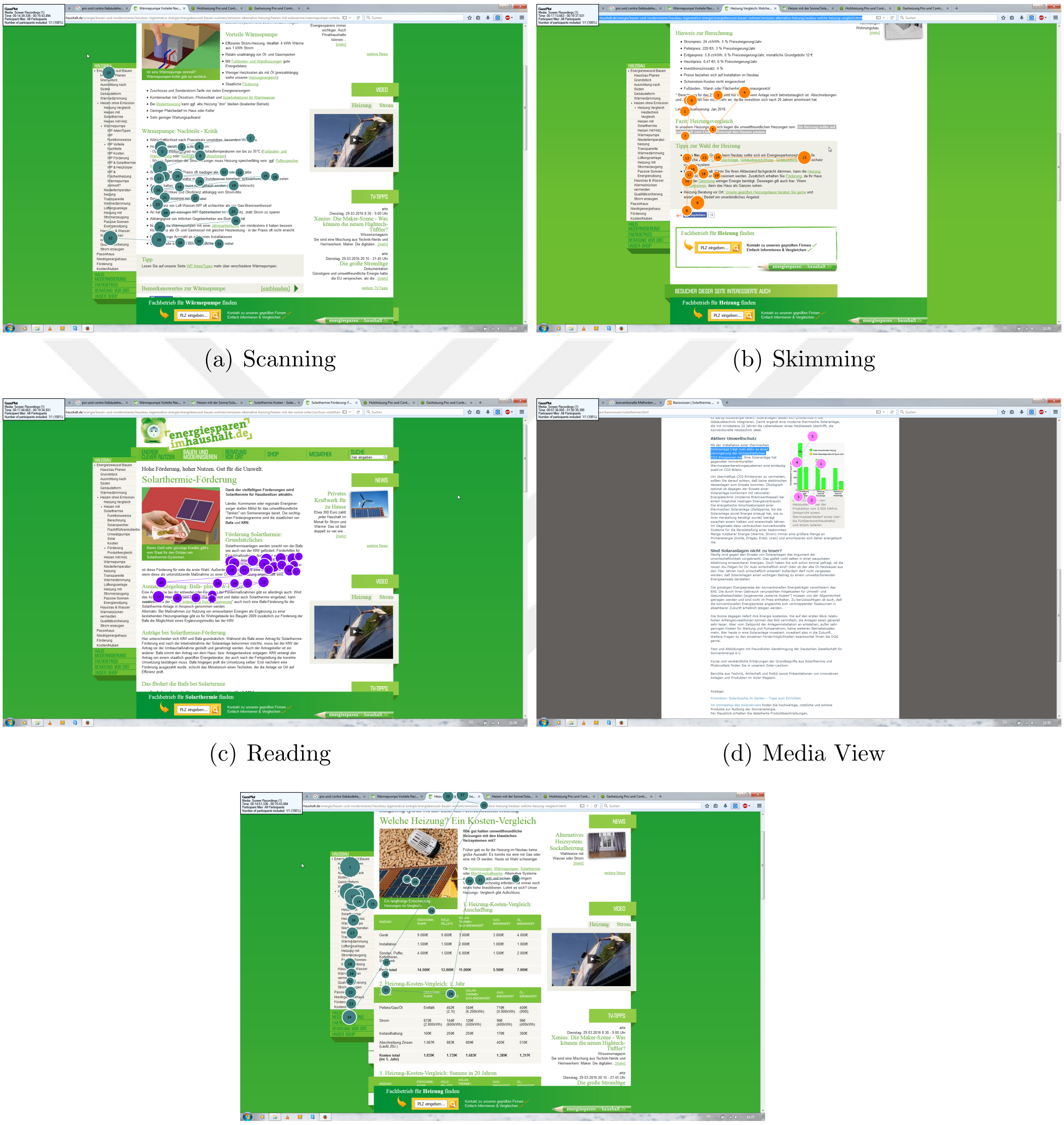
In Section 3.4, it has been pointed out that for the matched annotations of the two annotated data, only the intersection of the matched rows is taken. Here, instead of taking the intersection, a much better approach may be applied to not the loose information of the rows outside the intersection. For instance, comparing the timestamps of the annotations may be a solution. However, this is not tested in this thesis and has been left for the future work.

Obviously, this work can be enhanced to unsupervised learning, so that the model can classify types of reading behavior without the need of labels for training data.

Besides, the classification of the types of reading behavior knowledge might be used to support users during their web searches. For instance, it is more likely that a user who performs an exploratory search will read the text on the screen. As the model is able to detect reading, and classify the types of it, some keywords can be suggested to the user or related pages that allow examination from different perspectives during his web search activity. On the other hand, a user who performs a fact-finding search is more likely *scanning* or *skimming* the text on the screen as he tries to find a piece of information. Hence, some statements might be highlighted to support user.

It can be also the case that reading speed of different users, the amount of the regressions and their durations with respect to the font and size can be analyzed for further developments.

A. Appendix

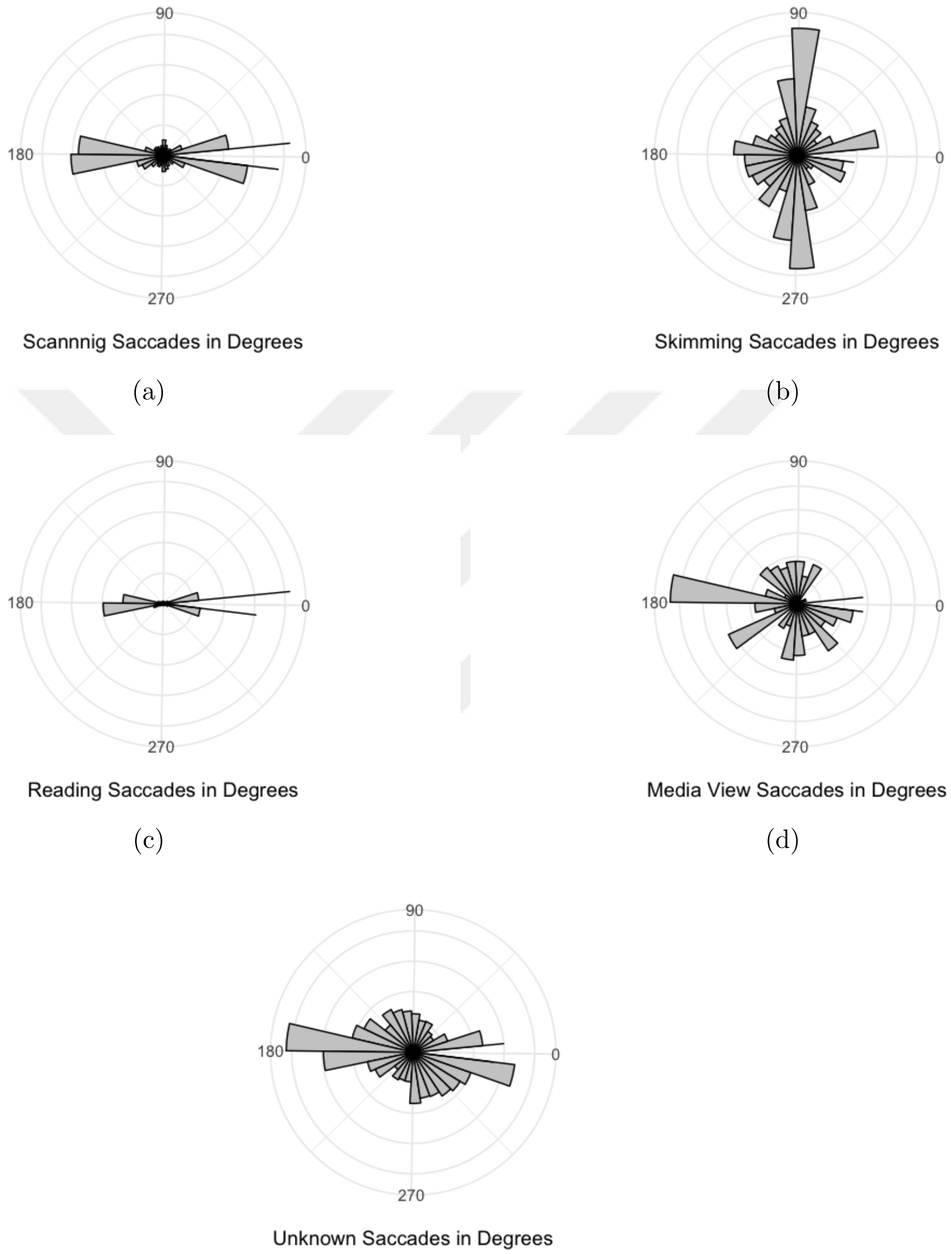


(a) Scanning (b) Skimming

(c) Reading (d) Media View

(e) Unknown

Figure A.1: The gaze points of the five different states extracted from one of the users data using Tobii Studio.



(a) (b) (c) (d) (e)

Figure A.2: 360 Degree Histogram of the five different states of the training data. The length of the states: Scanning:7333, Skimming:2582, Reading:5973, MediaView:250, Unknown:10224.

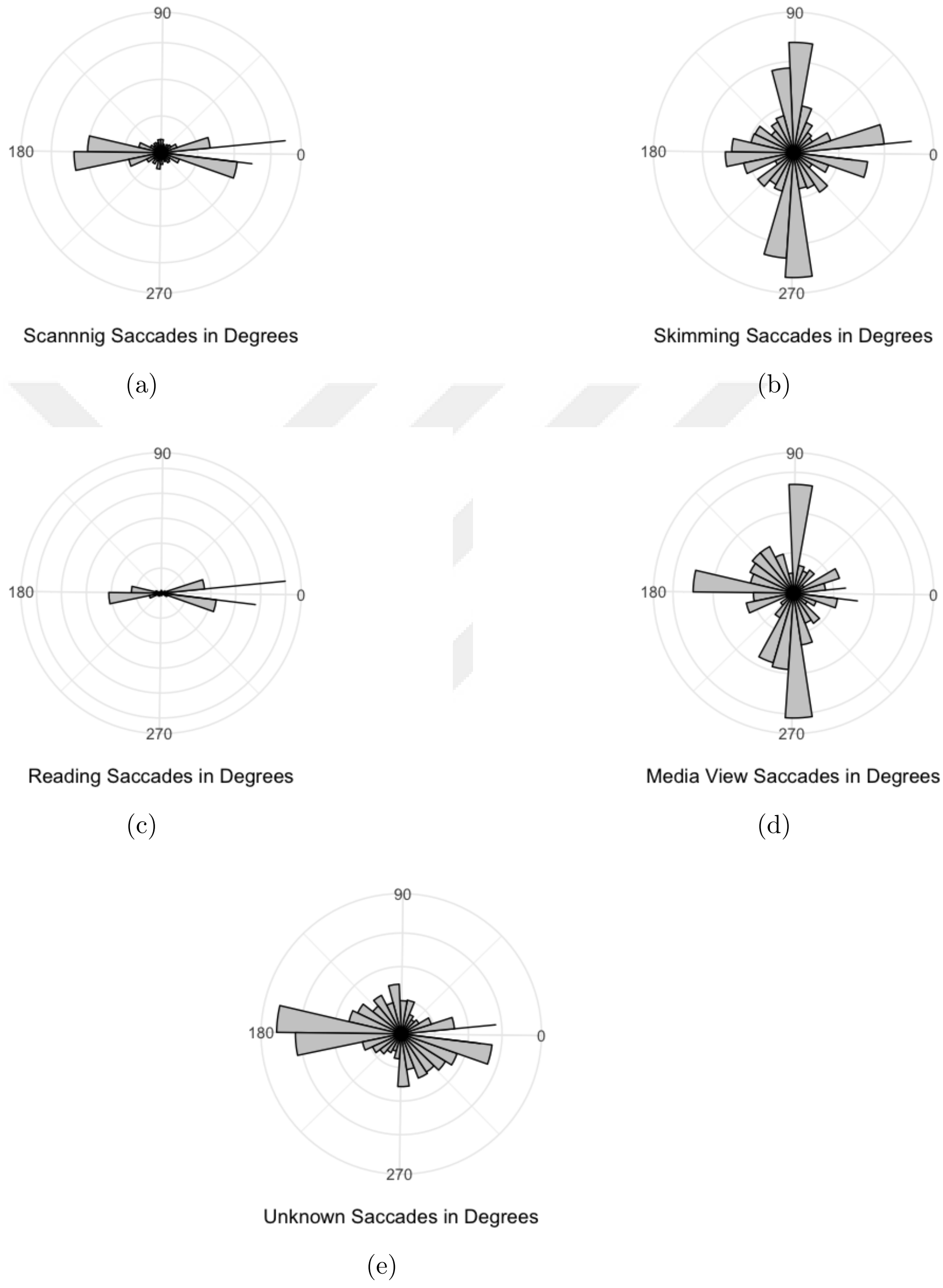


Figure A.3: 360 Degree Histogram of the five different states of the test data. The length of the states: Scanning:4452, Skimming:1785, Reading:1854, MediaView:342, Unknown:4248.

Bibliography

[BDvE08] Georg Buscher, Andreas Dengel, and Ludger van Elst. Eye movements as implicit relevance feedback. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, pages 2991–2996, New York, NY, USA, 2008. ACM. (cited on Page 5, 6, 7, and 47)

[BHB12] Ralf Biedert, Andreas Hees, Dengel, and Georg Buscher. A robust realtime reading-skimming classifier. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, pages 123–130, New York, NY, USA, 2012. ACM. (cited on Page 5 and 7)

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. (cited on Page 46)

[CGL+11] Michael J. Cole, Jacek Gwizdka, Chang Liu, Ralf Bierig, Nicholas J. Belkin, and Xiangmin Zhang. Task and user effects on reading patterns in information search. *Interacting with Computers*, 23(4):346 – 362, 2011. Cognitive Ergonomics for Situated Human-Automation Collaboration. (cited on Page 5, 6, and 7)

[CM01] Christopher S. Campbell and Paul P. Maglio. A robust algorithm for reading detection. In *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, PUI '01, pages 1–7, New York, NY, USA, 2001. ACM. (cited on Page 5, 6, and 7)

[CRH+14] Malcolm Clark, Ian Ruthven, Patrik O'Brian Holt, Dawei Song, and Stuart Watt. You have e-mail, what happens next? tracking the eyes for genre. *Information Processing & Management*, 50(1):175 – 198, 2014. (cited on Page 11)

[Dis] Probability distributions: Discrete vs. continuous. http://stattrek.com/probability-distributions/discrete-continuous.aspx. Accessed: 2018-03-10. (cited on Page 24 and 26)

[DP11] Geoffrey B. Duggan and Stephen J. Payne. Skim reading by satisficing: Evidence from eye tracking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1141–1150, New York, NY, USA, 2011. ACM. (cited on Page 11)

[DREKJM98] Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme J. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, volume 3. 01 1998. (cited on Page 34)

[Fit] Fitting distributions with r. https://cran.r-project.org/doc/contrib/Ricci-distributions-en.pdf. Accessed: 2018-02-15. (cited on Page 26)

[Gus10] Carl Johan Gustavsson. Real time classification of reading in gaze data. In *Master's Thesis*, 2010. (cited on Page 5, 6, and 7)

[HMMa] Hidden markov model training. http://digital.cs.usu.edu/~cyan/CS7960/HiddenMarkovModel_training.ppt. Accessed: 2018-02-16. (cited on Page 34)

[HMMb] Speech recognition architecture: Hmms and viterbi. http://www.cs.brandeis.edu/~cs136a/CS136a_Slides/CS136a_Lect8_HMMs_Viterbi.pdf. Accessed: 2018-03-09. (cited on Page 35)

[IB04] Shamsi T. Iqbal and Brian P. Bailey. Using eye gaze patterns to identify user tasks. 2004. (cited on Page 5 and 7)

[JM00] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2000. (cited on Page xiii, 12, and 13)

[KGSN17a] Michael Kotzyba, Tatiana Gossen, Johannes Schwerdt, and Andreas Nürnberger. Inferring user's search activity using interaction logs and gaze data, 2017. (cited on Page 5, 7, and 40)

[KGSN17b] Michael Kotzyba, Tatiana Gossen, Johannes Schwerdt, and Andreas Nürnberger. Exploration or fact-finding: Inferring user's search activity just in time. In *ACM SIGIR Conference on Human Information Interaction & Retrieval*, CHIIR '17, pages 87–96, New York, NY, USA, March 2017. ACM. (cited on Page 2)

[KH07] Sepp Kollmorgen and Kenneth Holmqvist. Automatically detecting reading in eye tracking data. 01 2007. (cited on Page 5, 6, and 7)

[Nut08] Antje Nuthmann. Cognitive modelling course: Computational modelling of eye-movement control in reading. 2008. (cited on Page xi and 11)

[Per] Accuracy, precision, recall and f1 score: Interpretation of performance measures. http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures. Accessed: 2018-03-09. (cited on Page 40 and 41)

[Pro18] Tobii Pro. Tobii Studio User's Manual Version 3.4.5. https://www.tobiipro.com/siteassets/tobii-pro/user-manuals/

tobii-pro-studio-user-manual.pdf, 2018. [Online; accessed 25-February-2018]. (cited on Page xi, 16, 21, and 47)

[Rab90] Lawrence R. Rabiner. Readings in speech recognition. chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pages 267–296. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. (cited on Page 14)

[Ray78] Keith Rayner. Eye movements in reading and information processing. *Psychological Bulletin*, pages 618–660, 1978. (cited on Page xiii and 10)

[Ray08] Rayner. Eye movements during reading, scene perception, visual search, and while looking at print advertisements. In *Visual marketing: From attention to action*, 2008. (cited on Page 11)

[Rei00] Rayner Reichle. Comparing the e-z reader model to other models of eye movement control in reading, 2000. (cited on Page v, 1, and 11)

[RM15] P. Rodeghero and C. McMillan. An empirical study on the patterns of eye movement during summarization tasks. In *2015 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–10, Oct 2015. (cited on Page 10 and 11)

[RMR08] Keith Rayner, Brett Miller, and Caren Rotello. Eye movements when looking at print advertisements: The goal of the viewer matters. 22:697–707, 07 2008. (cited on Page 12)

[SFMR16] Luca Scrucca, Michael Fop, Thomas Brendan Murphy, and Adrian E. Raftery. mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, 8(1):205–233, 2016. (cited on Page 31)

[SSK08] Jaana Simola, Jarkko Salojärvi, and Ilpo Kojo. Using hidden markov model to uncover processing states from eye movements in information search tasks. *Cognitive Systems Research*, 9(4):237 – 251, 2008. (cited on Page 28)

[usa18] usability.gov. Eye Tracking. https://www.usability.gov/how-to-and-tools/methods/eye-tracking.html, 2018. [Online; accessed 15-January-2018]. (cited on Page xi and 2)

[Wik18a] Wikipedia. Gamma distribution. https://en.wikipedia.org/wiki/Gamma_distribution, 2018. [Online; accessed 22-March-2018]. (cited on Page 28)

[Wik18b] Wikipedia. Hidden Markov Model). https://en.wikipedia.org/wiki/Hidden_Markov_model, 2018. [Online; accessed 15-January-2018]. (cited on Page 13)

[Wik18c] Wikipedia. Reading (process). https://en.wikipedia.org/wiki/Reading_(process), 2018. [Online; accessed 15-January-2018]. (cited on Page 9)

[WWMP15] S. J. White, K. L. Warrington, V. A. McGowan, and K. B. Paterson. In *Eye movements during reading and topic scanning: Effects of word frequency.*, 2015. (cited on Page 11)

With this, I declare that I have written this thesis on my own, distinguished citations, and used no other than the named sources and aids.

Magdeburg, den 31. May 2018