Stony Brook University

Algorithms and Interfaces for Automated Non-Visual Skimming

A Dissertation Presented

by

Faisal Ahmed

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Computer Science

Stony Brook University

August 2012

Stony Brook University

The Graduate School

Faisal Ahmed

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree, hereby recommend

acceptance of this dissertation.

Dr. I.V. Ramkrishnan – Dissertation Advisor
Computer Science Department

Dr. C.R. Ramakrishnan - Chairperson of Defense
Computer Science Department

Dr. Yevgen Borodin – Dissertation co-Advisor
Computer Science Department

Dr. Jeffrey P. Bigham
Computer Science Department
University of Rochester

This dissertation is accepted by the Graduate School

Charles Taber
Interim Dean of the Graduate School

Abstract of the Dissertation

Algorithms and Interfaces for Automated Non-Visual Skimming

by

Faisal Ahmed

Doctor of Philosophy

in

Computer Science

Stony Brook University

2012

In our information-driven web-based society, we are all gradually falling victims to information overload [9]. However, while sighted people are finding ways to sift through information faster, computer users who are blind are experiencing an even greater information overload. These users access computers and Internet using screen-reader software, which reads the information on a computer screen sequentially using computer-generated speech through an audio interface and allows users to navigate using keyboard shortcuts or gestures. This interface does not give them an opportunity to know what content to skip and what to listen to. So, they either listen to all of the content or listen to the first part of each sentence/paragraph before they skip to the next one. In this dissertation, I investigate and develop methods for non-visual skimming that will empower visually impaired users to access digitized information significantly faster and, thus, reduce the cognitive load associated with non-visual browsing.

Visual skimming involves quickly looking through content while picking out words and phrases that are emphasized visually and/or carry the most meaning. My preliminary user study with 20 visually impaired participants who were blind has shown that users were able to skim and comprehend text much better if the summary included connected word phrases, as opposed to separate unrelated words, e.g., “hands down” instead of “hands”.

In this dissertation, I investigate the techniques employed by sighted users to skim web content, and design algorithmic methods to enable a computer-assisted skimming experience for screen-reader users. I design and evaluate algorithms for generating variable-length summaries to support skimming text at different levels of granularity and speed. I also design and evaluate a touch-based and shortcut driven screen-reader interfaces to support non-visual skimming. I have used participatory design approaches to devise all user interfaces. I conducted controlled and in-situ real-world experiments to evaluate the approach and interfaces. The Major contributions of this thesis include: 1) novel interfaces for non-visual skimming on regular computers and touch-screen devices; 2) novel computer algorithms that support skimming interfaces.

Dedication

..to Professor Parvin Akther and Mr. Masud Ahmed.. the greatest treasures of my life..

Table of Contents

List of Figures

Preface

Introductory sections: In Section 1 I give motivation of my dissertation, the problem statement and overview of the solution with a description of a realistic user scenario. In Section 2 I present a broad overview of the related work.

Research on Fast Navigation: Section 3 gives the detailed description of the user study I did and the hypotheses I proved as a part of my solution. In summary, I evaluated the utility of non-visual skimming and analyzed existing approaches to gain insight of how to design non-visual skimming algorithm.

Research on Algorithm of Non-Visual Skimming: In Section 4, I discuss my non-visual skimming algorithm (Section 4.2) and a shortcut based skimming interface, I designed. I also provide detailed description of both the experimental and user evaluation (Section 4.3, 4.4) of this algorithm and interface.

Research on Variable size Skimming using Touch Interface: In Section 5, first I talk about the enhancement of the non-visual skimming algorithm to support variable size/speed skimming (Section 5.1). Then I introduce the touch interface as a potential interface for variable speed skimming (Section 5.2). Afterwards for both topics, I give a detail design of the user evaluation done in realistic scenarios to prove the effectiveness of Non-Visual skimming with a touch interface supported by variable speed skimming.

Concluding Sections: Finally in Section 6, I summarize the contributions of this dissertation. I propose future directions of research and convey a general note on web accessibility research. A full list of cited works is given in Section 7.

Acknowledgements

I express my deepest gratitude to my Advisor Dr. I.V. Ramakrishan and co-Advisor Dr. Yevgen Borodin, to arise my enthusiasm towards accessibility research. Throughout all these years these two persons kept me charmed by their extraordinary talent and hard work. I wish them the very best in the future.

A heartfelt thanks goes to Glenn Dausch, our accessibility expert consultant, without whom I would find it hard to design usable accessibility tools.

My colleagues Muhammad Asiful Islam, Valentyn Melnyk, Yury Puzis, Andrii Soviak, Denys Katerenchuk were always been very helpful and used to criticize my work to make it more perfect.

I want to thank Terri Hedgpeth Ed.D., the Director of the Disability Resource Center at the Arizona State University for helping us organize and conduct the user study with the individuals who are blind.

Special thanks to all masters Students of Applied Logic Lab from 2010 to 2012 to helped me run numerous experiments. Special thanks go to Jyotirmoy Sundi, Ruchita Sarwagi and Nisha Punjabi who contributed in building accessibility tools.

Curriculum Vitae (2012)

Research Interests

Web Accessibility, HCI, Data/Web mining, NLP, User Interface, Applied Machine Learning

Education

PhD in Computer Science — **Expected : Aug, 2012**

Department of Computer Science

State University of New York at Stony Brook — CGPA **3.76**

MS in Computer Science — **2011**

Department of Computer Science — CGPA **3.76**

State University of New York at Stony Brook

Bachelor of Science in Computer Science and Engineering — **2006**

Department of Computer Science and Engineering, — CGPA **3.87**

Bangladesh University of Engineering and Technology (BUET)

Selected Work Experiences

01/2008 - Present: Research Assistant

- Conducting research on Accessible Information Technology for people with visual disabilities. Recent work is on Non-Visual Skimming and Touch Interface for improving accessibility of web content. Regarding NLP I am working on extracting clinical relationships from doctor-patient conversation/narratives. I am a key member of the Hearsay (www.cs.sunysb.edu/~hearsay/) R&D team. *Technologies Used:* C++, Java, JavaScript, JNI, MSTTS.

01/2011 - Present: Research Assistant

- Research assistant working with a company: Charmtech Labs LLC. Working on web mining. I designed algorithms for article/next page link/heading extraction on web pages. Our product Capti is available at Apple appstore. *Technologies Used:* Java, JavaScript, XML.

06/2011 – 08/2011: Technology Analyst, Morgan Stanley Smith Barney

- I have worked in MSSB TapsCo where I developed a reporting system for monitoring inbound & outbound file transmission and clustering transactions in mainframe,. *Technology Used:* Perl, JCL.

06/2010 – 08/2010: Technology Analyst, Morgan Stanley

- I have designed and developed a Batch Milestone Management application to manage MS's numerous business deliverables efficiently. I also worked on a trade volume console for visualizing incoming live traffic. My work includes predicting next day trade volume, automating the task of volume data extraction from datacenter, report creation and sending. *Technology Used:* Perl, ASP.NET, Sybase.

06/2009 – 08/2009: Research Intern, Microsoft Research

- Researched on building a generic accessible medical data/user interface on traditional Gaming Consoles (i.e., Xbox) at Microsoft Medical Media Lab. I designed and developed a prototype for patient to let them access and view their status using Xbox which is has been deployed at Washington Hospital Center. *Technology Used:* .NET.

01/2007 – 12/2007: Member R&D, Commlink Infotech (www.commlinkinfotech.com).

- Projects worked on: Web based ERP (Enterprise Resource Planning) system. I completed the Payroll Management System module. *Technologies Used:* .NET, PHP, HTML, MySQL, AJAX, JavaScript, Apache.

08/2006 – 12/2007: Member R&D, Streamstech Inc. (www.streamstech.com)

- Worked as a designer and developer of plug-ins for Open Source GIS (Global Information Systems) software. *Technologies Used:* C, C++, .NET framework

03/2006 – 04/2006: Departmental Website Development

- Worked as a team leader for developing the website for Computer Science & Engineering Department, BUET (www.buet.ac.bd/cse). *Technologies Used:* PHP, HTML, AZAX, JavaScript, MySQL, Apache, Flash.

05/2005 – 05/2006 : Hospital Management System

- Development of Hospital management system for City Hospital, Dhaka, Bangladesh. *Technology Used:* Visual Basic.

Teaching Experience

01/2008 – 05/2009: Teaching Assistant, Department of Computer Science, SUNY StonyBrook

- Introduction to Computer Science, Discrete Mathematics.

04/2009: Guest lectures, Department of Computer Science, SUNY Stony Brook

- Recitation class on Discrete Mathematics

07/2001- 07/2005: L**ecturer, Sunrise: Engineering University Admission Coaching Center, Bangladesh**

- Taught Mathematics, Physics, Chemistry

Awards & Honors

- **04/2012: Best Paper Award,** International Cross-Disciplinary Conference on Web Accessibility, W4A 2012.
- **05/2009: Best TA Award**, Computer Science Department, SUNYSB.
- **01/2008: Presidential fellowship** from Stony Brook University.
- **09/2006: 1st prize in Website Development Competition** arranged by the Department of CSE, BUET(www.buet.ac.bd/cse/common/credits.php).
- **05/2003: University Merit Scholarship** (awarded to the top 5 students of the department) for two semesters at BUET.

Programming Skills

- Java, C/C++, C#, Python, Perl, Assembly, SQL, MCML

- XML, XQuery, Oracle, MySQL, PostGreSQL, Sybase
- HTML, PHP, ASP.net, JSP, Java Script, Struts, Hibernate, LAMP
- Eclipse, MS Visual Studio .NET, MapWindow

Courses Taken

- **Graduate:** Compiler Design, Analysis of Algorithms, Theory of Database Systems, Logic in Computer Science, Machine Learning, Artificial Intelligence, Fundamental of Networks, Discrete Mathematics, Information retrieval
- **Undergraduate:** Operating system, Computer Architecture, Database, Digital Logic Design, Theory of Computation, Software Engineering, Numerical Methods, Computer Graphics.

Selected Projects

- **Hearsay:** I have worked as a member of development team of Hearsay. Hearsay is a non-visual web browser which provides context directed browsing experience to the visually impaired people. It provides various input and output interface to facilitate numerous user needs. *Technologies Used:* Java, C++, JavaScript, HTML
- **Radiology Search Engine:** A project in collaboration with Stony Brook University hospital. I led a team of four to make a radiology search engine which is built upon Lucene indexing System. The search engine includes account management, bookmark management, incremental indexing etc. The project is being under installation. *Technologies Used:* JSP, JavaScript, MySQL, HTML
- **Skimming:** I have analyzed various summarization techniques and designed an algorithm of skimming specially for blind individuals. I have evaluated the effectiveness of the approach experimentally and by doing user study. *Technologies Used:* Java, JavaScript

- **Extraction of Medical Information:** I have worked on extracting clinical relationships from doctor-patient conversation/narratives. I have used SNOMED to identify the medical terms on the doctor patient conversation as attribute and devised an algorithm to find the attribute value, if it has any. This task makes the data processing significantly faster. *Technologies Used:* Java, MySQL
- **Voice Activated Home Appliance System** - led a team of five to design and implement an appliance system for manipulating domestic electronic equipments by voice. *Technologies Used:* C++, MS Voice SDK
- **Classifying articles from 20 online Newsgroups –** I built a classifier using naïve Bayes classifier. *Technology Used:* Java.
- **Compression at bit level by increasing redundancy in static binary file --** Research project to find out maximum compression limit of static files. *Technology Used:* Java.
- **Compiler Design and Implementation** – I built a compiler from scratch, for an object oriented language. *Technologies Used:* flex, bison, C
- **Optical Character Identification** – Worked in a team to build a model for indentifying Bengali Characters, using neural network. *Technology Used:* C++

Patent

U.S. Patent Pending (13458961): Borodin Y., A. Dimitriyadi, Y. Puzis, **F. Ahmed**, V. Melnyk "Combining Web Browser and Audio Player Functionality to Facilitate Organization and Consumption of Web Documents". Charmtech Labs LLC, 2012

Publications

- **Ahmed, F.**, Y. Borodin, A. Soviak, M.A. Islam, I.V. Ramakrishnan, and T. Hedgpeth, *Accessible Skimming: Faster Screen Reading of Web Pages*, *to be appeared in UIST*. 2012, ACM: Cambridge, Massachusetts, USA.
- **BEST PAPER AWARD: Faisal Ahmed**, Yevgen Borodin, Yury Puzis and I.V. Ramakrishnan. *"Why Read if You Can Skim: Towards Enabling Faster Screen Reading".* To appear in: International Cross-Disciplinary Conference on Web Accessibility, W4A 2012.
- **Faisal Ahmed**, Muhammad Islam, Yevgen Borodin and I.V. Ramakrishnan. *"Assistive Web Browsing with Touch Interfaces".* In Proceedings of the 12th International ACM SIGACCESS Conference on Computers and Accessibility, (ASSETS'10).
- Muhammad Asiful Islam, **Faisal Ahmed**, Yevgen Borodin, I.V. Ramakrishnan, "*Thematic Organization of Web Content for Distraction-Free Text-to-Speech Narration*". In Proceedings of CIKM 2011.
- Islam, M.A., **F. Ahmed**, Y. Borodin, and I.V. Ramakrishnan, *Thematic Organization of Web Content for Distraction-Free Text-to-Speech Narration*, in *ASSETS*. 2012.
- Y. Borodin, **F. Ahmed**, M.A. Islam, S. Feng, Y. Puzis, V. Melnyk, G. Dausch, I.V. Ramakrishnan, *"Hearsay: A New Generation Context-Driven Multi-Modal Assistive Web Browser"*. In Proceedings of the 19th International World Wide Web Conference (WWW'10).

- Muhammad Asiful Islam, **Faisal Ahmed**, Yevgen Borodin, Jalal Mahmud, I.V. Ramakrishnan, *"Improving Accessibility of Transaction-centric Web Objects"*. In Proceedings of the 10th SIAM International Conference on Data Mining, (SDM'10).

- Yury Puzis, Yevgen Borodin, **Faisal Ahmed** and I.V. Ramakrishnan. *"An Intuitive Accessible Web Automation User Interface".* To appear in: International Cross-Disciplinary Conference on Web Accessibility, W4A 2012.

- I.V. Ramakrishnan, J. Mahmud, Y. Borodin, M. A. Islam, **F. Ahmed**, "*Bridging the Web Accessibility Divide"*. In Proceedings of the 4th Int'l Workshop on Automated Specification and Verification of Web Systems (WWV'08), Elsevier ENTCS, Volume 235, Pages 107-124, April 2009.

- Zhiyuan Zhang, **Faisal Ahmed**, Arunesh Mittal, IV Ramakrishnan, Rong Zhao, Asa Viccellio and Klaus Mueller, "*AnamneVis: A Framework for the Visualization of Patient History and Medical Diagnostics Chains*". In Proceedings of the Workshop on Visual Analytics in Healthcare (VAHC 2011).
- Borodin, Y., A. Sovyak, A. Dimitriyadi, Y. Puzis, V. Melnyk, **F. Ahmed**, G. Dausch, I.V. Ramakrishnan "*Universal and Ubiquitous Web Access with Capti*" To appear in: International Cross-Disciplinary Conference on Web Accessibility, W4A 2012.
- I.V. Ramakrishnan, J. J. Tithi, A. Bagate, V. Khot, **F. Ahmed**, D. Harrington, MD, R. Talati, MD "Organizing RadLex Lexicon for Efficient Retrieval of Radiology Documents" In Proceedings of IHI 2012.

1. INTRODUCTION

1.1. Problem Statement

The Web has permeated many aspects of our lives and has become essential for obtaining and exchanging information, paying bills, shopping, banking, making travel arrangements, applying for college or employment, connecting with others, participating in civic activities, etc. The utility of the Web is even greater for people with vision impairments who once required human assistance with most of these simple activities, and who can now do them independently on the Web.

To provide some background on the size of the target population who can benefit from the Web, according to the World Health Organization, there are 285 million people with vision impairments worldwide – 39 million blind and 246 million with low vision [83]. According to the American Foundation for the Blind, there are over 25 million Americans with severe vision loss who have trouble seeing even when wearing glasses; the same source reports that 196,000 of these Americans use the Internet [2].

Since the inception of the Web, researchers and practitioners have been working on screen-reading technology to make the Web more accessible to blind people [18, 25, 69]. Screen readers (e.g., JAWS [43], Window-Eyes [84], VoiceOver [80], SuperNova [77], NVDA [61]) are assistive technology tools that narrate the content of the screen using text-to-speech. Screen readers *typically* allow users to navigate within web pages back and forth if web content is properly marked up by HTML tags for headings, paragraphs, links, buttons, etc. The introduction of touch-screen devices such as iPhones, has enabled blind people to listen to the content that they touch, e.g., as the user is sliding a finger over the surface of the touch screen, VoiceOver screen reader [80] starts reading the content of the HTML element touched by the finger. Although screen readers enabled blind people to access web content, a large gap remains between the ways blind and sighted people interact with that content.

When sighted people look at a web page, they can get a quick overview of the page and find what they need. The structure and formatting of web pages further help sighted people find the information relevant to their goals. In content rich web pages, this still leaves lot of text to read through, imposing a *heavy cognitive load*. But, again, sighted people can selectively glance over the content, read slower or faster, skip content, pick out snippets of information such as names, dates, noun and verb phrases, etc. This allows sighted people to get the gist of the content and find the information they are looking for much faster, reducing the cognitive load. The process of quickly glancing over the content is called skimming.

Skimming [19, 51] is a speed-reading technique that allows sighted readers to get a general idea of the content and sift through information very quickly, without reading the entire content and, most importantly, without sacrificing basic understanding of the content. Skimming is also very useful while searching for specific information, especially, when keyword searching is not possible, for example, because the user does not know the exact keywords to search for. Unfortunately, due to the way screen-reading software works, the effectiveness of "skimming" available to people who are blind does not come anywhere close.

Because of the low bandwidth of the serial audio interface and because one has to hear the information before deciding if it is important, people who are blind read through significantly more content and spend considerably more time identifying the information they need [12]. In our preliminary user study with 20 screen-reader users, we have confirmed that increasing the speech rate and using shortcuts are the *only* strategies screen-reader users employ to skim through web pages. However, we found that newly blind people and older adults are much worse at understanding faster speech rate (similar results are reported in [76]). Having found the article in a web page, screen-reader users only use shortcuts to scan through the beginning of every line, sentence, or paragraph. Unfortunately, these functions are not equivalent to the ability to skim through information, and these strategies are no match to visual web browsing [31].

All of our 20 subjects wished there was a way to read and skim through information faster. Those who acquired blindness later in life said they were really missing the ability to skim, but they found the screen-reader support of skimming to be inadequate. The JAWS screen reader [43] has a naïve implementation of "skimming" [72] that allows users to read the 1st line or the 1st sentence of each paragraph, which is, essentially, equivalent to using shortcuts. It also lets users specify regular expressions and then listen to the content that matches those expressions.

However, twenty experienced JAWS users we interviewed either did not know about the JAWS skimming feature or considered it "*inconvenient"* and/or *"useless."* The users who were familiar with VoiceOver's touch interface said it was useful in that it gave them access to the two-dimensional layout of the web page, but it did not enable skimming any better.

1.2. Proposed Solution

Despite the fact that screen readers have enabled blind people to access the Web, screen-readers do little to save users' time. A lot of research has been done to help find main content in web pages [5, 11, 13, 14, 40, 55] and avoid having to listen through irrelevant content such as links and ads. However, very little research has been done to help blind screen-reader users process information faster. Because more and more information is becoming available on the Web, there is a pressing need for an assistive technology that can facilitate effective information processing for screen-reader users.

The goal of this research is to enable blind people to read web content faster with the help of non-visual skimming. Just as visual skimming helps sighted people, non-visual skimming will enable blind people to go through content two, three, four times faster than they presently can with the state-of-the-art assistive technologies. While rapid eye movements (saccades) help sighted people pick out bits and pieces of information, non-visual skimming will be enabled through intelligent algorithms that will identify the most salient information for blind people. Just as sighted people, screen-reader users will be able to speed up and slow down the process of skimming depending on the desired information density.

I envision that non-visual skimming can be made available both in regular screen-reader and touch-screen interfaces; this will enable people who are blind to control the process of skimming either by pressing keyboard shortcuts or by dragging a finger on the surface of a touchpad, tablet, or a smart-phone.

1.3. Use Scenario

Mary is a successful college student who maintains good grades despite the fact that she is legally blind. She has just enough sight to see people around her, but she cannot use computers without screen-reading software. Just as any student these days, Mary is overloaded with information; she constantly needs to look up reading material online, use the Blackboard [1] website to access homework assignments, use the online message board, keep up with the news, etc. The secret to her success, as well as having enough time to do her homework and still finding time to sleep is… non-visual skimming!

Truth be told, Mary could access computers rather well with screen readers even before skimming. Since she started using screen readers, she got used to listening to synthesized speech at three times the normal speech rate, and she got very proficient with different shortcuts allowing her to jump through headings, links, etc. Yet, she often had to stay up late to finish her homework due to the sheer volume of information she had to go through… until non-visual skimming became ubiquitous in screen readers.

Non-visual skimming empowers Mary to read through and search for information much faster on her laptop, her tablet, or her smart phone. She does her homework on her laptop, where she uses keyboard shortcuts control the speed of skimming. Between classes, she quickly revises reading materials on her tablet by dragging the finger on the screen at different speed of skimming. Even in her commute, Mary can skims web pages on her smart-phone with gestures – the faster she swipes the faster she can read.

Skimming helps her brush through different sections of web pages and skim through main content. Whenever Mary needs to scan through text, she starts skimming, and listens to snippets of information that help Mary get the gist of the content. While skimming, she can control the density of information and switch between skimming and regular reading. Whenever Mary needs to revise what she read before, she skims at 4 times the speed. Mary also finds skimming very useful when she knows what information she is looking for, but she cannot just do a keyword search for it. In such cases, she saves herself a lot of time by scanning the text at 10 times the speed. But the most important thing that skimming does for Mary is help her be more productive and compete with her sighted peers on equal footing.

2. RELATED WORK

2.1. Overview of Skimming

People typically read for enjoyment, to search for information, to complete a task, or to explore / review text [51]. Whatever the specific purpose of reading may be, in our information-driven society, people are faced with large volumes of information that they need to process. *Cognitive overload* [48, 78] is a fundamental problem that arises while processing a large volume of information in web pages; this is especially true for people with vision impairments who have to listen through most of web content in a sequential manner.

To reduce the cognitive load, sighted people employ a number of different speed-reading techniques (a.k.a. skimming or scanning) [50]. Skimming helps get the gist of the content or find specific information that could not be searched for otherwise. One of the simpler techniques is to look through the illustrations, titles, subtitles, and other visually prominent content. However, this only gives an idea of what the content is about. To gain a deeper understanding of the content one can read the first and/or the last sentence in each paragraph. However, to gain understanding of the content, one has to scan the entire content; more advanced sighted readers can scan through the content catching separate phrases instead of individual words. So without reading every word, advanced readers can get most of the information slowing down or accelerating depending on their information needs. This can often be accomplished by diagonal reading, when the reader moves the eyes diagonally over the content picking out the most important phrases [74].

Unfortunately, assistive technology does not provide people who are blind the ability to efficiently skim through information in a manner that is comparable to that of sighted people.

To simulate the skimming process, one could try to use eye tracking devices to figure out what words and phrases are chosen by the human eye. However, different people develop

different ad hoc approaches to skimming, and there is no guarantee that the words that "catch the eye" are the ones that carry the most meaning. An alternative approach is to look specifically for the words and phrases that help preserve the meaning of the text. This could be accomplished by using automated summarization.

Automated summarization approaches use computer algorithms [67] to condense textual content while preserving its gist. Visual skimming is akin to the extractive summarization, because when people scan the text they take it as it is. Our interviews with 20 screen-reader users also suggested that the extractive summarization approach that preserves the original content is most suitable for enabling non-visual skimming. I next review the existing extractive summarization approaches and explain why they are not suitable for non-visual skimming of web pages.

2.2. Summarization

Summarization is a process of condensing textual content into a shorter version, to help the audience get the gist in a short period of time. Summarization is supposed to highlight important points of the original long content (e.g., article, paper, book, etc.) to give the user the gist of the content, but summarization frequently does not preserve the mood or style of the original text. Automated summarization techniques aim to summarize text with the help of computer algorithms [67].

Unfortunately, summarization alone does not work the same way as skimming. A summary often does not preserve all of the salient content, because, otherwise, the summary will be too long. Using a summary, a blind person can get the gist of the content, but cannot control how much information appears in the summary and has no easy way to get more information. So, after reading a summary, the screen-reader user would often have to read the entire content again to look for missing information. In my research, I am trying to recreate the same experience for the blind people, which is enjoyed by the sighted people while skimming.

Several attempts have been made to use summarization in assistive technology tools. Harper et al. introduced Summate [36] – a summarization tool for blind individuals. Summate is a FireFox-based tool that summarizes web pages and presents the summary in an alert box. Another accessibility tool, AcceSS [63] attempted to simplify and summarize the web contents for web users. AcceSS did simplification by removing the clutter and retaining the important sections to give the user a preview of the page. Researchers at Oxford Brookes University have built a Web navigation tool called BrookesTalk [86] that uses information retrieval techniques to summarize a Web page for quick orientation. While these tools gave screen-reader users the ability to preview content, they have not given the blind users the ability to skim and switch between the summary and the original content easily.

Broadly, there are two main types of summarization techniques: summarization by extraction and summarization by abstraction. In short, extraction deals with picking out the most salient sentences or phrases from the original text [6, 29, 31], and abstraction is compressing sentences [8, 22], which involves paraphrasing the existing content, for example by automatically filling

out a template [53]. Most of the existing summarization techniques [23, 27, 32, 53, 85, 88] use one or both of these two techniques.

Because visual skimming involves "extracting" salient content with one's eyes, the extractive summarization appears to be the most suitable technique to support computerized skimming. Evaluation and comparison of the existing approaches to extractive summarization was not in the scope of this research, instead, I focused on confirming that non-visual skimming was useful for screen-reader users. Because I did not want shortcomings of the existing summarization technique to affect the results, I used human-generated summaries to identify the qualities of the ideal automated skimming approach.

2.3. Extractive Summarization Approaches

There are two main approaches that is dominating extractive summarization. Linguistic analysis and the use of term statistics was main focus while summarizing scientific documents. The other approach is the use of artificial intelligence, which utilizes knowledge-based methods for compressing information. The field has seen a lot of progress in the last decade, although there are a number of unsolved issues in extractive methods. In addition the natural language processing, the use of machine learning techniques where summarization models are trained from text corpus which contains huge number of documents and their summaries, greatly accelerated the advancement of extractive summarization. Now there are real-world systems which are able to summarize meetings [81], news broadcasts [59], the medical documents [26], and create biographies [73]. There are domain specific summarization systems as well. A sentence extraction based automatic summarization technique [38] targeted to make abstracts from presentations, proposes to a method using sentence location. They used latent semantic analysis to find the relevant and important sentences in a document which gives good results, but the processing time of the system is slow because of the time complexity of SVD decomposition. Companies like InXight, IBM, SRA, etc. made summarization products commercially available.

2.3.1. Summarization by Sentence Extraction

The majority of the existing extractive summarization approaches are doing sentence extraction [8, 20, 31, 36]. Such approaches typically analyze the entire text and examine the relationship between sentences to find the ones that have the most meaning or are very different from the rest of the text. JAWS screen reader [43] has a "skimming" feature [44] allowing users to read the 1st sentence or a line of each paragraph of text.

While sentence extraction can help get a general overview of text, this approach cannot enable skimming, in which the reader scans through all or most of the sentences picking out the most salient words and phrases. Sentence extraction leaves out most of the original content, which makes it unusable for scanning for information that cannot be found without reading the text, e.g., when keyword search is not possible due to unknown keywords. Sentence extraction

assumes coherent text and will not work in web pages, where there is unrelated content, e.g., in front pages of news sites. Therefore, keyword extraction approaches appear to be the only ones suitable for non-visual skimming of web pages.

I next examine the most relevant extraction approaches to determine if they can support a non-visual skimming interface.

2.3.2. Summarization by Keyword Extraction

In [86], the summary is constructed by extracting word trigrams that frequently occur in the entire document. This approach does not pick words from each sentence and needs long documents.

In [52] the author presents a keyword extraction approach based on the analysis of a whole document . The approach builds a tree where nodes represent words and edges represent order relationships. Higher weights are given to the words that have higher number of incoming edges, number of outgoing edges, frequency and tf-idf (term frequency multiplied by inverse document frequency) score. Finally, individual keywords that have the highest weights are extracted into the summary.

In [39], the authors build separate graphs for each sentence and weigh each vertex based on the word and structural dependencies within a sentence, as well as word frequency in the entire document. The summary is generated by selection of top ranked words. The compression of the summary depends on the predefined percent value.

An interesting relation-driven summarization algorithm is proposed in [42]. Authors describe three main stages: 1) concept activation, during which they reject the unimportant parts of the document and divide content words into two groups “triggers” and “filters”, 2) segmentation, during which they pick the words related to the words defined as “triggers” and decide which of those words should be skipped, and 3) identification of linguistic relations among the words, which helps to extract coherent phrases. Finally, the algorithm removes even more words, and

what remains is the summary. While the description of the algorithm is sketchy, I can see that the algorithm runs over a full document and does not guarantee coverage of each sentence.

While these all are viable summarization approaches for many applications, none of these is suitable for generating summaries that can simulate skimming. The major limitation of the reviewed approaches is that they all heavily rely on word frequency, which is a meaningful feature only when summarizing long documents. Unfortunately, the content of web pages is predominantly short. For instance, front pages of news sites typically have a number of short snippets each containing a headline followed by a few sentences from the linked articles; the articles themselves often contain only one long or a few short paragraph.

In addition, most of the reviewed approaches do not offer any guarantees that keywords will be extracted from every sentence. Some of them do not preserve the original order of the text or extract separate words instead of word phrases. All of these were the important criteria that I identified in my preliminary studies with people who are blind. Our subjects were able to understand the gist of the content much better when they were presented word combinations as opposed to separate words, and they did not want information to be skipped or presented out of order.

To overcome the problems listed above, a different summarization approach was needed to support the non-visual skimming interface. The desired approach should be able to summarize one sentence at a time without the dependence on the entire document, extract meaningful words and word combinations from every sentence, and preserve the original order of the extracted text.

In this research, I propose a skimming algorithm that satisfies these criteria.

2.4. Skimming in Accessibility

During our study, a number of blind subjects said they wished they could skim content. Unlike sighted users, blind people have to use screen readers to access computer content. Screen readers have come a long way from simple screen reading to intelligent content analysis that can segment web pages, categorize web objects by type, and offer the layered interface to navigate them [11, 18, 80]. However, screen readers still do not offer adequate support of skimming. So far, other than JAWS, no screen reader claims to support skimming. Also, Apple's VoiceOver [80] has several features that are relevant to skimming.

JAWS screen reader [43] has a feature which lets user create their own rules (regular expressions) for skimming. The rules basically tell JAWS to read certain pattern in article. Obviously the same rule cannot be always applied to all kinds of texts because same information can be presented in different patterns. In our interviews, however, screen-reader users said that JAWS skimming "was not useful," that JAWS skimming "was not enough to look into the content of an article," and that it was similar to the paragraph navigation feature which was used for a different purpose and with a different name.

VoiceOver (VO) screen reader [80] has a user friendly accessible interface for interacting with the screen content. VO has a number of features that enable users to read faster, however, none of them can be called skimming. VO allows users to drag a finger over the screen and listen to the content under the finger. In [5] Ahmed et al. mentioned granularity issues in VO's "dragging interface" and pointed out that there was no way to ensure that content was read sequentially. While this feature could, in theory, be used for skimming, currently VO simply reads the content under the finger without trying to determine how important that content is. VoiceOver, just like Hearsay[11], provides a layered view on web content allowing users to access elements of a particular type (e.g., headings, links, images, etc.). While this feature could give an overview of the content, it does not resemble real skimming.

SpeechSkimmer is a tool for audio skimming [7] which presents techniques for segmenting audio recordings and built a prototype for skimming speech. They exploit the properties of spontaneous speech and extract the salient parts of audio speech. The user interface was design decisions were made based upon observations and interviews [45] done by the members of

Speech Research Group. This work also reviews familiar systems that provide browsing or speech summarization features. They used time compression and speech detection techniques in SpeechSkimmer with a review on “pauses” and time compressed speech.

Based on our interviews with screen-reader users, none of the assistive technologies have adequate support of skimming. As a result, screen-reader users try to use regular screen-reader navigation shortcuts to simulate skimming, as was confirmed during our user study. Unfortunately, as I show in our experiments, such ad hoc skimming does not prove to be particularly effective.

2.5. The Need for Accessible Skimming

Our review of literature, as well as personal experience have shown that sighted people employ numerous speed-reading techniques; sighted readers can scan text with their eyes and pick out salient sentences and phrases that help the readers get the gist of the content. Assistive technology does not provide adequate support of skimming that would allow blind people to skim with efficiency comparable to that of sighted people. Summarization by itself does not satisfy the need for skimming; for screen-reader users, reading a summary is more like reading a separate, albeit shorter, narrative. I am not aware of any interfaces that maintain the connection between a summary and the original text – the feature that would allow the screen-reader user to switch seamlessly between them. Based on our interviews with screen-reader users, extractive summarization techniques that preserve the original content appear to be the best match for enabling the skimming interface. The design of an appropriate summarization approach is the subject of section 4 that is informed by the experiments described in sections 4.3and 4.4.

3. FASTER NAVIGATION WITH SKIMMING

3.1. Design and Evaluation of Accessible Skimming

I have conducted two user studies in order to verify the utility of skimming and evaluate my skimming interface. In the first study, 12 sighted subjects were asked to create *gold-standard* summaries of several articles, i.e. manually summarize the articles. In the second study, 20 blind subjects were asked to use the skimming interface to do two realistic tasks (listening and comprehension and searching for information) with and without the skimming interface.

3.1.1. Generating Gold-Standard Summaries

The quality of automated summarization is gradually approaching that of humans [67, 87]. However, to enable and evaluate the proposed skimming interface without committing to a particular automated summarization technique, we employed human-based summarization. By using sighted subjects to manually produce gold-standard summaries of articles, I was able to guarantee that the results of our experiments would not be affected by the shortcomings of a particular summarization method.

3.1.1.1. Participants

To generate the gold summaries, we recruited 12 sighted participants. All of them were college-educated and were fluent in English.

3.1.1.2. Experimental setup

We collected 6 news articles on different topics with 5-6 paragraphs in each of the articles on average. Each of the 12 sighted participants was asked to summarize 2 articles, thus, generating 4 gold-standard summaries for each article.

Since the purpose of skimming was to pick out the most salient information without rephrasing the original content, we stipulated that the subjects use extractive summarization. Since the point of skimming was to save time to the end users by presenting less information, we set the upper bound on the length of the summary to 1/3 of the original content. However, we did not want to dictate how exactly to do summarization, so each participant was provided only with the following "loose" guidelines:

- Each sentence is to be summarized separately;
- The summary may only include words and phrases found in the summarized sentence;
- Words in the summary should be in the same order they appear in the original text;
- The length of the summary should be no more than one third of the length of the original article;
- Summaries should be as informative as possible.

3.1.1.3. Results and Analysis

Sighted human subjects generated 4 gold-standard summaries for each of the 6 articles. Because of the loose guidelines, subjects generated summaries of variable length choosing different words. Specifically, some summaries were composed of mostly nouns, others also had prepositions, and yet others preserved some verbs, adjectives, and adverbs. **Figure 1** shows that in the original text nouns (31%), adjectives (9%), adverbs (4%), verbs (15%) preposition (13%) were the most common parts of speech.

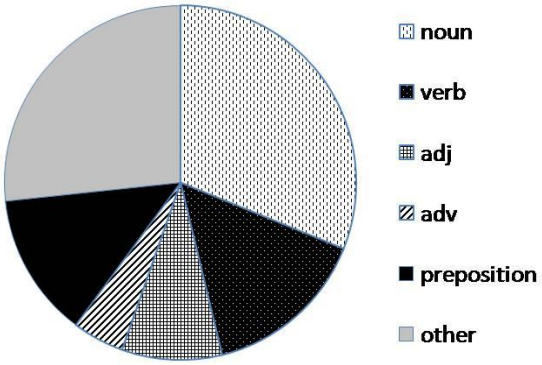


Figure 1. Parts-of-speech distribution in original text

After the gold-standard summarization, the ratios changed: nouns (54%), adjectives (11%), adverbs (11%), verbs (12%), and prepositions (7%), and these were still the most commonly chosen parts of speech (**Figure 2**).

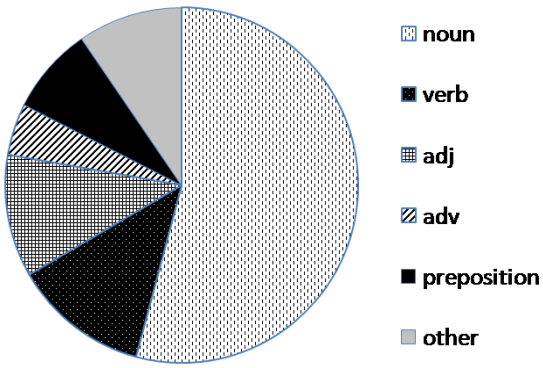


Figure 2. Parts-of-speech distribution in gold-standard summaries

This indicates that these parts of speech carried the most meaning in the original text. The "other" category included interjections, pronouns, determiners, predeterminers, etc. In almost all summaries, the punctuation was preserved. We believe this is because punctuation partitions sentences and clauses and helps preserve some of the original semantics.

The analysis of the parts of speech in the gold-standard summaries served as the key to designing the experiments for blind subjects. The results of this experiment prompted us to choose 3 types of summarization to be used in our subsequent experiments. To obtain the three types of summaries, we first merged the 4 gold-standard summaries of each article into a single summary using the following approach: if at least two subjects chose a certain word – we included it in the final summary. We preserved the original word order and punctuation. We, then, created two more types of summaries: one with only nouns and the other with nouns and prepositions. In **Table 1**, we provide samples of the 3 summaries (passages A, B, and C) for a paragraph (D) taken from a New York Times article [62]. Note the increasing level of detail from one summary to the other.

A: Gold summary with nouns only:
Twitter, 10 person startup San Francisco, Obvious. Mixture networking microblogging. idea, people omnipresence. Use Iran election.
B: Gold Summary with nouns and prepositions only:
Twitter, 10 person startup San Francisco, Obvious. Mixture of networking microblogging. on idea, people omnipresence. Use in Iran election .
C: Combined gold summary:
Twitter, 10 person startup San Francisco, called Obvious. Mixture of social networking microblogging. based on idea, people enjoy virtual omnipresence. Use in Iran disputed election.
D: Original paragraph:
Twitter, which was created by a 10 person startup in San Francisco was called Obvious. It is a heady mixture of messaging, social networking, 'microblogging' and something called 'presence.' It's shorthand for the idea that people should enjoy an 'always on' virtual omnipresence. Twitter's rapid growth made it the object of intense interest. The object of fair amount of ridicule, as it was derided as high tech trivia or the latest in time-wasting devices. But its use in Iran in the wake of the disputed presidential election of June 2009 brought it new respect. It was used to organize protests and disseminate information in the face of a news media crackdown.

Table 1. Gold-Standard Summaries.

3.1.2. Skimming in Comprehension and Searching Scenarios

To evaluate accessible skimming, we conducted an extensive user study with 20 blind subjects who used our skimming in listening-and-comprehension, as well as in search scenarios; these two common scenarios, where skimming could be most useful, were selected in informal discussions with blind screen-reader users. We then conducted a user study that demonstrated that accessible skimming improved user efficiency in both scenarios. Furthermore, this study helped obtain both quantitative and qualitative data that further validate the usability and effectiveness of the accessible skimming interface. In this section, we describe the experimental set up and discuss the results of the user study.

3.1.2.1. Participants

Disability Resource Center of Arizona State University helped us to recruit 20 blind subjects who participated in the user study. Gender representation was approximately equal. The ages of participants were evenly distributed and varied from early twenties to late sixties. Four of the subjects considered themselves expert computer users, ten of them were very comfortable and six were comfortable with computers. All subjects were well-versed in the use of screen readers, with JAWS as their primary screen reader. Computer usage was more than 20 hours per week for ten subjects, 11-20 hours per week for seven subjects, 6-10 hours per week for one subject, and 1-5 hours a week for the remaining two subjects.

3.1.2.2. Experimental Setup: Listening-and-Comprehension

When experienced sighted readers want to get a gist of the text, they can quickly scan through the text picking out salient information. To test how well our skimming interface can facilitate speed-reading, we designed an experiment testing how much information our subjects could retain while skimming, compared to regular screen reading as a baseline.

In a within-subjects experiment that had a 4 by 4 cell design, we had each subject listen to 4 different articles using 3 types of summaries A, B, and C, and the full text D (Table 1). We varied the order of the tasks for counterbalancing and to avoid bias.

We considered including a condition that would test the 1st-sentence-of-each-paragraph summary (similar to JAWS skimming), but, after trying it in a pilot study, we concluded that such a summary contained too little information and could not be used effectively in either the listening-and-comprehension or the searching scenarios.

Before the experiment, the subjects were given a sample listening-and-comprehension task to practice with. They were not allowed to use navigation shortcuts in this task. Each task took approximately 10 minutes.

Every task was followed by a set of 10 questions that tested the retention of information. The first question was open-ended and the rest were multiple-choice. The questions we formulated

covered the major points of each article without regard for which content actually appeared in the corresponding gold summary and covered the parts of speech that appeared in the gold summaries: 1 question was on the article topic (i.e. "What is the article about?" – "Twitter"), 4 questions on the nouns (e.g., "What was the name of the tweeter start up" – "Obvious"), 3 questions on the verbs (e.g., "What was twitter used for in Iran?" – "organize protests"), 1 on the numeric values (e.g., "How many people organized Twitter?" – "10"), and 1 on the adjectives/adverbs (e.g., "What kind of interest did Twitter generate?" – "Intense"). We approximately followed the distribution of the parts of speech in the original articles and were consistent in this break-down across all articles.

3.1.2.3. Experimental Setup: Ad-Hoc Searching Scenario

When experienced sighted or blind readers want to find some information in a lengthy text, but cannot search for the exact keywords, they quickly skim the text until they find a relevant passage and then start reading normally. It is also notable that many screen-reader users do not even use the search feature. To assess whether accessible skimming can help in this scenario, we designed an experiment that tested whether the subjects could find the answer to a specific question quickly using accessible skimming vs. their own ad-hoc skimming techniques as the baseline.

In a within-subjects experiment that had a 2 by 2 cell design, we had each subject skim through 2 different articles using the combined gold summary C and the full text D. Throughout the rest of the thesis, we will use the term *skimming* to mean skimming using gold summary and we will use *reading* to mean reading of text using regular shortcuts. The conditions were counterbalanced across subjects to avoid bias. Before each task, the subjects were asked a question, the answer to which they had to find in the article. The questions were formulated used different wording than the actual article, so that the subjects could not search for the exact keywords, e.g., "What is the name of the country in which the results of the voting caused riots?" – "Iran." The subjects were instructed not to try keyword searching.

The answer to the question was contained in the 4th paragraph of each article to increase the time required to find the answer. Having the answer in the beginning of the articles would have caused the users to find the answer equally fast with and without skimming, defeating the purpose of the experiment. Further, we only used one type of summary in this experiment to avoid the learning effect that would have been observed had our subjects figured out that the answer was toward the end of the article. Varying the location of the answer would have increased the variation of the completion times.

In this task, we measured the time it took subjects to reach the sentence that contained the answer to the question, as well as the time needed to understand and give an answer to the question. We also measured the number of keystrokes the subjects pressed to find the answer. Finally, we observed and noted the ad-hoc skimming strategies employed by the subjects while reading the full text. The subjects were given a sample task before the experiment and 5 minutes to practice with the skimming interface. They were allowed to use any navigation shortcuts which they were used to. Each task took approximately 10 minutes.

3.1.2.4. Questionnaires

Following each task in either of the scenarios, the participants were asked to rate the perceived difficulty of the task on a 5-point Likert scale (1=Very Easy to 5=Very Hard) (**Figure 4**). At the end of the experiments, we read statements about the skimming experience to the subjects and asked them to rate the statements on a 5-point Likert scale (1=Strongly Disagree to 5=Strongly Agree) – **Table 2** summarizes the ratings.

3.1.3. Interface for Accessible Skimming

To evaluate the accessible skimming interface, we used Hearsay [11] screen-reading platform. We used IVONA TTS [41] voice *"Eric"* with speech rate of 180 words per minute. The screen-reader interface enabled subjects to use all the typical navigation shortcuts used by the mainstream screen-readers such as JAWS [43], e.g., paragraph / sentence / word / character navigation, pause / resume, etc. The articles were plain text containing no links or images.

To enable skimming in the searching scenario, we added a new shortcut that allowed users to switch seamlessly between the summary and the original article preserving the current reading position. This feature helped emulate the behavior of experienced sighted readers who usually can scan quickly and then, at any point, slow down to read the text regularly. When switching from the regular screen-reading to the skimming mode, the reading position was reset to the closest preceding word that was in both the original text and the summary. In the searching-with-skimming condition, the subjects could use all navigational shortcuts. The search shortcut was disabled.

3.1.4. Hypotheses and Results

To test the two general hypotheses proposed in the introduction, we formulated a series of specific hypotheses that we accepted or rejected based on the statistical significance testing. In this section we provide the details and the results on all tested hypotheses. Also we present the results of the analysis of the post-completion questionnaire.

We considered statistical significance level of alpha = 0.01, while conducting the t-test tests on the data. The results may have been significant even with a smaller rejection region, but we did not want to increase the probability of the Type II error. The following are the specific hypotheses tested in the experiments and the results.

3.1.4.1. Hypotheses and Results for Reading and Comprehension

H1: The questions on general comprehension of the articles can be answered equally well for summary A, B, C, and D.

Results: Subjects were able to answer correctly the question "What is this article about?" with accuracy of 80% (St. Dev.=40%) for summary A, 90% (St.Dev.=30%) for summary B, 100% (St. Dev.=0) for summary C, 100% (St. Dev.=0) for summary D.

Discussion: One of our important goals was to check if skimming adversely affected the general understanding of a text. This hypothesis clearly shows after listening to summary D, the

subjects had no problem answering the general comprehension question. This was not the case with summaries A and B that contained no verbs, adjectives, or adverbs.

H2: The use of verbs, adverbs, and adjectives (VAAs) in skimming, in addition to nouns, enables greater comprehension compared to skimming without VAAs.

Results: Subjects were able to give correct answers to 56.5% (St. Dev. = 13.8%) and 62% (St. Dev. 16.7%) of questions after reading summary A and B respectively without VAAs. In contrast, subjects were able to answer correctly 77.5% (St. Dev.=11.6%) and 80.5% (St. Dev.=14.3%) of questions (St. Dev. = 12%) after listening to summary C and D respectively with VAAs. Using the one-tailed paired t-test, we found the differences in the question answering accuracy to be statistically significant ($t=6.352$, $df=19$, $p<0.0001$), accepting H2.

Discussion: The parts-of-speech distributions in the original text and the gold-standard summary presented in **Figure 1** and **Figure 2**) show the essential parts of speech that need to be preserved in the summary. This hypothesis, although unsurprising, suggests that VAAs can be very important in a summary.

H3: Comprehension of information about action increases if the summary includes Verbs, Adjectives, and Adverbs (VAAs).

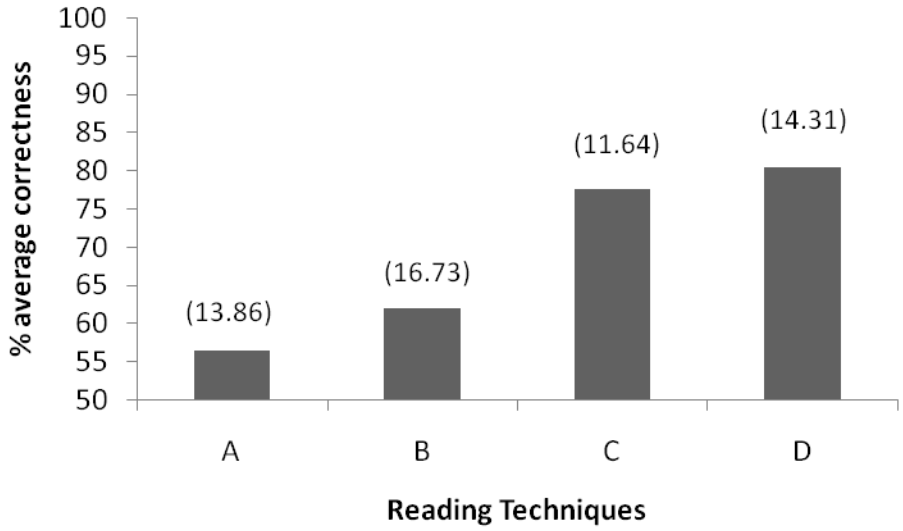


Figure 3. Average correctness (St. Dev.) of question answering with different summaries and full text D

Results: Our questions about actions / verbs were designed to check subjects' comprehension level of the actions described in the articles, they did not test the ability to guess / remember the exact verb. Subjects were able to answer questions about actions correctly in 50% (St. Dev. = 35%) and 45% (St. Dev.=27%) of cases for summaries A and B respectively. For summaries C and D, subjects were able to answer questions correctly in 78.3% (St. Dev. = 24.8%) and 76.6% (St. Dev.=26.7%) cases on average respectively. We compared accuracy of A and B vs. C and D and the one-tailed paired t-test showed that a statistically significant difference. This was an obvious result, since summaries A and B did not contain VAAs and we still asked questions about VAAs. However, we have also compared the accuracy of answers for questions about only nouns, which were present in all summaries. Subjects answered questions on nouns correctly: 63.7% (St. Dev.=26.2) for summary A, 72.5% (St. Dev.=17.9%) for summary B, 81.2% (St. Dev.=15.9%) for summary C, and 83.7% (St. Dev.=18.6%) for summary D. The differences between A and B, B and C, C and D were not found to be significant, but the difference between A and C was: ($t=-2.570$, $df=19$, $p=0.009$). This result suggests that punctuation and VAAs together have improved the answers of noun questions. Thus we can only partially accept H3.

Discussion: It is notable, that the difference between A and B, as well between C and D, was not statistically significant, which means that subjects' ability to answer questions about actions was comparable for the gold-standard summary and the original text.

H4: After reading each of the summaries A, B, C, and D subjects can answer increasingly more questions correctly.

Results: Subjects were able to give answers with the average accuracy of 56.5% (St. Dev. = 13%), 62% (St. Dev. = 16%), 77.5% (St. Dev. = 11%), and 80.5% (St. Dev. = 14%) after listening to summaries A, B, C, and D respectively (**Figure 3**), even though the actual order of reading these summaries was randomized.

A one-way ANOVA was used to test accuracy differences in answering four types of summaries A, B, C, and D. The accuracy with which subjects gave answers differed significantly across the four types of summaries, $F(3, 76) = 13.44$, $p < 0.0001$. The post-hoc test for linear

trend shows that there is a significant trend, that answering accuracies increases as you read summary type A, B, C, and D respectively, Slope 0.04375, $p < 0.0001$.

Discussion: Overall, we observed an increase in the average correctness, but we could not accept H4 for all pairs. Adding prepositions to the nouns (summary B) did not result in significantly better accuracy than using only nouns (summary A). While the presence of prepositions gave the subjects a better idea about the relationship between nouns, it was not enough to help them answer questions more accurately. It is notable, that the biggest difference in comprehension was with the addition of verbs, adjectives, and adverbs. Even more interesting is that *there was no significant difference in question answering accuracy between gold-standard summary and the original text. This implies that summary C carried almost as much information as the full text.*

H5: Subjects perceive the questions to be increasingly easier for summaries A, B, C, and D respectively.

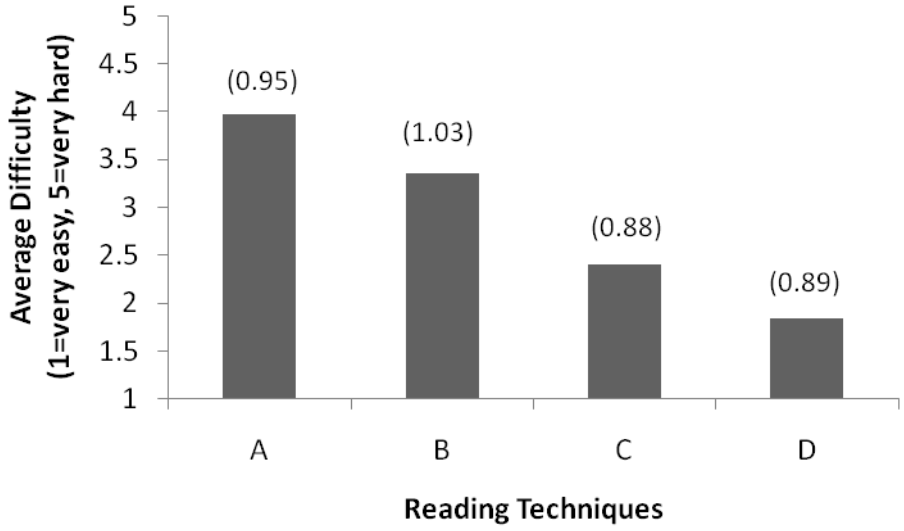


Figure 4. Average perceived difficulty (St. Dev.) of question answering with summaries and full text D

Results: Using the Likert scale (1=very easy to 5=very hard) the subjects rated the difficulty of the listening-and-comprehension task on average as 3.98 (St. Dev. = 0.95) for summary A, 3.35 (St. Dev. =1.03) for summary B, 2.4 (St. Dev. =0.88) for summary C, 1.8 (St. Dev. =0.89) for the full text, which shows a clear decrease in the difficulty level (**Figure 4**).

A one-way ANOVA was used to test differences of the difficulties perceived in answering four types of summaries A, B, C, and D. The difficulty perceived by the subjects while answering questions, differed significantly across four types of summaries, $F(3, 76) = 21.05$, $p < 0.0001$. The post-hoc test for linear trend shows that there is a significant trend, that answering difficulty decreases as you read summary type A, B, C, and D respectively, Slope -0.3738, $p < 0.0001$.

Discussion: The results show that the subjects thought the questions for gold-standard summary were almost as easy as those for the original text. It is notable that, although subjects thought the questions about the full text were a little easier than those for the gold-standard summary C, we did not find the difference to be statistically significant.

3.1.4.2. Hypotheses and Results for Searching Experiment

H6: Finding information using skimming is faster than in ad hoc searching.

Results: While searching for information, subjects took on average 76.35 seconds (St. Dev. = 24.89) to reach the answer using skimming and 148.45 seconds (St. Dev. = 45.16) using ad-hoc searching (**Figure 5**). There was a significant difference in the speed of skimming ($t=8.011$, $df=19$, $p<0.0001$).The time it took the subjects to actually answer the questions was on average 103.7 seconds (St. Dev. 37.30 seconds) and 169.4 seconds (St. Dev. = 49.54 seconds) for skimming and reading respectively. The one-tailed paired t-test ($t=6.448$, $df=19$, $p<0.0001$) found this difference to be statistically significant; thus, we accept H6.

Discussion: Subjects were able to navigate to the answer location 1.9 times faster using skimming compared to using ad-hoc searching. However, to actually answer the questions, subjects were 1.6 times faster while using skimming compared to ad hoc searching. This can be a substantial time saving to screen-reader users.

H7: There is a difference between the time to reach the answer and the time to actually answer the question.

Results: While using skimming, subjects spent on average 76.35 seconds (St. Dev. = 24.89) to reach the answer and 103.7 seconds (St. Dev. = 37.3) to actually answer to the questions. This difference is significant verified by a one-tailed paired t-test ($t=-6.092$, $df=19$, $p<0.0001$). While using ad-hoc searching, subjects spent on average 148.45 seconds (St. Dev. = 45.16) to reach the answer and 169.4 seconds (St. Dev. = 49.5) to actually answer the questions. One-tailed paired t-test showed this difference is significant ($t=-6.092$, $df=19$, $p<0.0005$). Thus, we accept H5.

Discussion: The result show that it takes subjects time to realize that they have found the answer when they are either skimming or using ad-hoc searching. We expect the difference to be dropping for longer text. But there is still a substantial gain in the speed. We have also computed the differences between the time to answer and time to reach the answer, and then compared these differences between skimming and ad-hoc searching – *it was not significant.*

We found that skimming was almost twice as fast (1.9) as ad hoc searching for reaching the answer. But because, while skimming, it took subjects longer to realize that they have found the answer, skimming was only 1.6 times faster compared to ad-hoc searching. We believe the reason may have been the lack of practice with this new kind of interface. Some of the user comments (in the Testimonials Section) also mentions that it requires practice.

H8: Skimming was easier than ad hoc searching.

Results: We found that, on average, people rated the difficulty of the tasks with skimming as 1.25 (St. Dev. = 0.55) and the difficulty of tasks with reading as 2.3 (St. Dev. = 0.98). Using the one-tailed paired t-test, we found skimming appeared to be significantly easier than reading ($t=-4.702$, $df=19$, $p<0.0001$), rejecting the null hypothesis.

Discussion: This was a surprising finding, especially considering that the subjects have had very little experience with skimming. Recall, that in the reading-and-comprehension scenario, the subjects rated questions for the full text as easier, even though that difference was not found to be statistically significant.

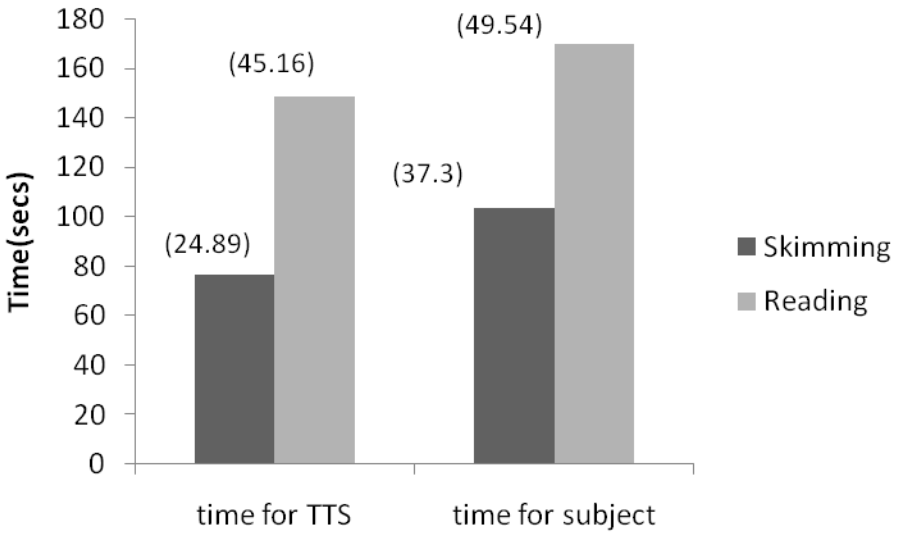


Figure 5. Average time to reach and understand the answer to a question in skimming vs. normal reading with (St. Dev.)

H9: The number of keystrokes pressed while skimming seemed to be always greater than the number of keystrokes used in normal mode. But the one-tailed paired t-test showed there was no difference between these two variables (t=1.475, df=19, p=0.922).

We also found no relationship between: **H10**) the completion time and the number of hours the subjects used computers per week, **H11**) the completion time and subjects' comfort level with computers, **H12**) the amount of time spent at the computer weekly and the perceived difficulty of the tasks (for either searching or listening and comprehension), and **H13**) the comfort level with computers and the perceived difficulty of the tasks (for searching or listening and comprehension).

3.1.5. Post Completion Questionnaire – Analysis

The post completion questions and answers about the skimming experience are presented in **Table 2**. The analysis showed that the standard deviation for all questions was within one point, demonstrating high consistency.

From the statements made by subjects, we can conclude that skimming has a lot of potential to improve reading experience for screen-reader users. Although the subjects thought that their comprehension was somewhat impeded in the skimming mode, almost all subjects agreed that skimming made the reading faster and they wanted to use skimming in the future. We did not see

significant difference in the question-answering performance between the gold-standard summary and the regular text. Two of the subjects mentioned that the tasks may have been easier if they could get more time to practice before the experiment.

3.2. Ad-Hoc Skimming / Searching Techniques

In our experiments we observed the subjects using their own ad-hoc techniques for skimming using regular navigational shortcuts. Some of the subjects used paragraph navigation to quickly move across paragraphs. A lot of subjects were reading sentences half way. Some subjects did not use any shortcuts and just listened to the entire text.

In the follow up interview, we asked open-ended questions to gain deeper understanding into ad hoc skimming techniques. One of the expert users who had 15 years of experience in using JAWS said: "I usually read the first one or two words of a sentence and then go to the next one. Obviously first 1-2 words cannot always give me the whole picture, but it works to some extent." This is an interesting comment that can lead to a simple, yet effective skimming approach. Seven of the subjects mentioned that they often speed up the speech rate for faster reading (in our experiment, the speech rate was controlled and was fixed at 180 words per minute). Most of the subjects would have tried to use JAWS search feature to look for certain information if they knew what exactly they were looking for. Two of the subjects, who frequently read research papers, used JAWS skimming. However, they said JAWS skimming is not comparable to our skimming.

3.3. Qualities of Summarization for Skimming

Literature review, examination of visual speed-reading technique, and the experiments with the gold-standard summaries helped us identify the main characteristics of the automated summarization technique that is best suited for supporting non-visual skimming.

Extractive summarization [17, 32, 47, 53, 54, 88] appears to be the most appropriate method for supporting non-visual skimming. Evaluation and comparison of different existing approaches

to extractive summarization was not in the scope of this research, instead, we focused on confirming that non-visual skimming was useful for screen-reader users.

Nouns carry the most information and should be the majority of the summary. Prepositions help establish relationship between the nouns. And verbs, adjectives, and adverbs often add additional meaning.

The ideal summarization technique should use some syntactic and/or semantic analysis of text for extracting salient parts of the original content. Human generated summaries can serve as a benchmark for evaluating the selected automated summarization methods and the statistical analysis of them can provide clues to the design of automated summarization.

General Statements	Avg. (St. Dev.)
I wish I could look through articles faster, than I can with a screen reader	3.80 (1.15)
I often experience difficulties looking for desired information within an article	3.40 (0.99)
Skimming made reading through articles faster	4.20 (1.00)
I understood articles equally well when skimming and when reading the full article	2.70 (1.34)
I want to use skimming in the future	4.30 (1.03)

Table 2. Average 5-Point Likert scale values (St. Dev.)
(Scale 1=Strongly Disagree to 5=Strongly Agree)

3.4. Testimonial

To complete the research, we are offering these verbatim quotes of our subjects commenting on the skimming:

"If I don't know what I am looking for, this is definitely very handy. I can decide whether to read further or not."

"Search doesn't always work well because there may be multiple results, while skimming is more contextual."

"It's definitely a valid operational concept, which will be very useful."

"I think skimming would be really helpful for long articles, books. In the skimming mode, it was like I was going over my class notes!"

"Usual way of reading takes a lot longer than skimming."

"It's easier, more practical, to find the most important things. I usually have difficulties in reading longer class assignments."

"I usually read the article in its full to get the idea, but with skimming I did not have to read it entirely. It's definitely faster."

"Skimming makes work faster if you need a synopsis."

"It gives key information, gets rid of unnecessary information. Sometimes I increase speech rate to quickly read the article, which is no longer required if we use this method."

"I usually speed up the speech rate to read faster and use paragraph navigation. With skimming it made it easier by giving important words. But it needs more time to practice."

Although the majority of feedback was elated, not all subjects were happy with the skimming approach:

"Skimming breaks up information… Introduces disorganization... There were no reference points… I don't know where I am..."

"Skimming is faster, but important info is sometimes missing."

Nevertheless, the subjects who made those comments did well in the listening-and-comprehension experiment and were able to complete the searching task faster with skimming than in the regular reading mode. This suggests that accessible skimming, just like visual skimming, may require some practice and getting used to.

4. ALGORITHM: NON-VISUAL SKIMMING

4.1. Overview of Our Approach

At high level our algorithm works as follows.

Firstly every sentence is parsed to extract grammatical relations amongst its words. Secondly a lexical tree based on these relations is constructed, where each node of the tree represents a word in the sentence. Thirdly for every word in this tree, its grammatical (i.e., POS tags) as well as structural features (related to indegree/outdegree, etc.) are extracted. These features are fed to a trained classifier to determine whether or not to include the word in the summary. Finally a subtree consisting of all these words is constructed. This subtree represents the skimming summary that users interact with via an interface.

Observe that three key elements of our algorithm are a natural language parser, classifier and a skimming interface. Below we describe these aspects for understanding the Skimming algorithm as well as its experimental evaluation. For extracting the relations we used the Stanford Natural Language Parser [49].

4.1.1. Stanford Parser

In our preliminary user study with 20 participants who were blind, we have found that screen-reader users were able to skim and comprehend text much better if the summary included connected word phrases, as opposed to separate unrelated word, e.g., “hands down” instead of “hands”. To capture the relationships between words in a sentence, we used Stanford parser [49], which is a probabilistic natural language parser that has tools for analyzing the grammatical and syntactical structure of sentences.

The relations identified by the parser are very simple and, in the words of the developers, "quite accessible to non-linguists and effective in relation extraction applications". These typed dependencies are represented as relations between pairs of words (i.e., "John likes cake" the object of "John" is "cake"). Apart from the grammatical relations Stanford Parser also identifies the POS (Parts of Speech) of the words.

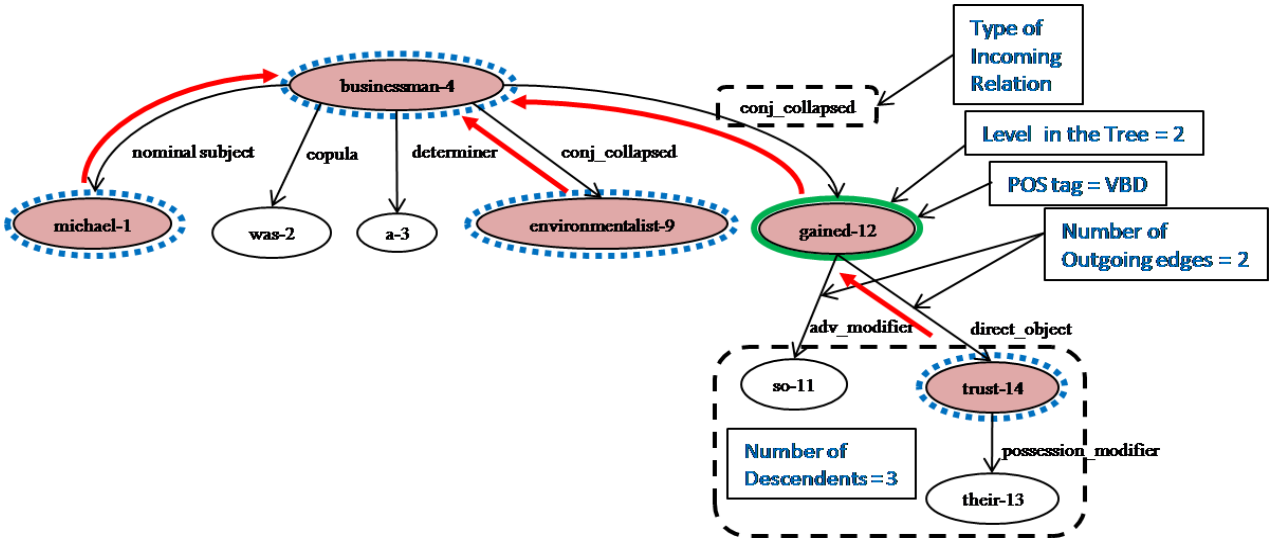


Figure 6. Sentence tree and features for "*Michael was a businessman, as well as an environmentalist, and so gained their trust.*"

Stanford parser identifies a total of 48 different binary grammatical relations [56]. A grammatical relationship holds between a governor word and a dependent word. For example, the sentence *"Chewing gum can improve memory, say UK psychologists."* is processed into the following relations:

Relation Name (Governor-Position → Dependent-Position)

Adjectival modifier (gum-2 → chewing-1);

Nominal subject (improve-4 → gum-2);

Auxiliary (improve-4 → can-3)

Nominal subject (say-6 → memory-5)

Clausal complement (improve-4 → say-6)

Noun compound modifier (psychologists-8 → uk-7)

Direct object (say-6 → psychologists-8)

Using the grammatical relations generated by Stanford Parser a directed graph can be constructed, where the nodes are the words and the edges are the relations from the governor to

the dependent (**Figure 6**). In the words of the developers, "the dependency graph is a tree (a singly rooted directed acyclic graph with no re-entrances)." [50]. Throughout the thesis, we will treat the dependency graph as a tree and will refer to it as a *sentence tree*.

4.1.2. Dataset Generation for Training the Classifier

The quality of automated summarization is gradually approaching that of humans [41, 88]. However, in my previous work [3] I explain why the existing automated summarization approaches are not suitable for skimming web pages. Part of the reason is that they have been trained on and fine-tuned for specific type of summaries, which are not suitable for skimming. Since we could not find an existing dataset of human-authored summaries that would be appropriate for skimming, we created our own.

To generate the gold-standard skimming summaries, we recruited 24 sighted participants. All of them were college-educated and were fluent in English. We collected 24 news articles on different topics with 5 paragraphs in each of the articles on average. The total size of the raw text included 674 sentences (17K words).

Each participant was asked to summarize 3 different articles, thus, generating 3 gold-standard summaries for each article. We then used a majority-vote approach to combine each of the 3 summaries into one, i.e., we chose only the words that were selected by at least 2 people, resulting in 43% compression ratio.

Since the purpose of skimming was to pick out the most salient information without rephrasing the original content, we stipulated that the subjects use extractive summarization. Because the point of skimming was to save time to the end users by presenting less information, but still have enough information to understand the content, we requested that the summary length be between 1/3 and 1/2 of the original content. However, we did not want to dictate how exactly to do summarization, so each participant was provided only with the following "loose" guidelines:

- Each sentence is to be summarized separately;

- The summary may only include words and phrases found in the summarized sentence;
- Word order and punctuation has to be preserved;
- Summaries should be as informative as possible.

The summaries are available at http://www.cs.sunysb.edu/~faiahmed/doc/data.zip.

4.1.3. Interface

Summarization by itself does not satisfy the need for skimming; for screen-reader users, reading a summary is more like reading a separate, albeit shorter, narrative. Therefore, we designed an interface that allowed screen-reader users to switch seamlessly between reading the skimming summary and the original web page preserving the current reading position. This feature helps emulate the behavior of experienced sighted readers who usually can scan quickly and then, at any point, slow down to read the text regularly. When switching from regular screen-reading to skimming, the reading position is set to the closest preceding word that was in both the original text and the summary.

The skimming interface was integrated into our HearSay [11] screen-reading platform, which is a Java application that enables a screen reading interface for the Firefox web browser. HearSay supports all the typical navigation shortcuts available in the mainstream screen-readers such as JAWS [43], e.g., paragraph / sentence / word / character navigation, pause / resume, etc.

The skimming interface can be invoked on any web page content regardless of its length as long as the content is organized in sentences (e.g., snippets of news articles along with headlines), but lists of links that can be found in menus, on the other hand, should be read in their entirety.

4.2. Skimming Algorithm

In this section, we describe the underpinnings of the proposed Skimming algorithm; specifically, we describe feature extraction, training classifiers and the skimming algorithm.

4.2.1. Feature Selection

The results obtained from the Stanford parser enabled us to identify a number of features that could be used for training a classifier. The features we chose were the Number of Outgoing Edges, Level in the Tree, Number of descendents, Incoming relation type, and POS tag (**Figure 6**). Each of the numerical features has been normalized with a scaling factor. The feature, Number of outgoing edges was normalized by the number of nodes in the tree. The feature, Level in the Tree was scaled by the highest level of a node in the tree. Finally, the feature, Number of descendents was normalized by the highest number of descendents a node can have in the tree.

4.2.2. Training the Classifier

We used an open source machine learning library, called Weka [35], to train classifiers using various machine learning models. In order to generate training data points we need to generate feature vectors with their label/class. First, we used Stanford Parser [49] (Section 4.1.1) to generate feature vectors for each word in each sentence in our dataset (Section 4.1.2). Then, to label/classify a feature vector, we check if the corresponding word appears in the gold summary; if it appears, we label the feature vector as "YES" class otherwise as "NO" class. "YES" class means this word is selected to be used in summary and "NO" class means otherwise. These datasets are used to train different classifiers (i.e., Linear SVM, Multilayer Perceptron, Random Forest) through the Weka machine learning library with 10-fold cross-validation; using 591 sentences (15300 words) for training and 83 sentences (1700 words) for testing.

The classifier-generated summary is formed by selecting words classified as "YES" and rearrange them by preserving the order in original sentence. To measure the accuracy of the classifier, we represented both the gold-standard summary and the classifier-generated summary as arrays of words. Because many sentences had repeating words, it was necessary to align the

words for more accurate measurements. Therefore, we ran the LCS (Longest Common Subsequence) algorithm to align the two arrays and compare the summaries.

We report the results in the first group in Table 3, under the caption "Output of the Classifiers alone". We found Linear SVM performs the best, therefore we decided to use it to classify words (*SkimSentence:Line 12*) in our proposed Skimming algorithm *SkimSentence* (section 4.2.3). In section 4.3 we discuss this result in details and also show our algorithm *SkimSentence* (section 4.2.3) improves the result.

4.2.3. Skimming Algorithm

The classifier helped us pick up separate words in a sentence, however, our preliminary user studies demonstrated that people tend to understand content better in the presence of phrases. Therefore, we designed the *SkimSentence* Algorithm to extract grammatically related important words of the sentence, while keeping comprehension level almost the same as our Gold-Standard summaries. The algorithm uses a trained classifier (Section 4.3.1) and another algorithm *MinConnectedTree* to skim the given sentence. The pseudocode for both algorithms is presented in this section.

Algorithm: *SkimSentence*

R: Set of Typed Dependency Relations {(W_{g1}, W_{d1}), (W_{g2}, W_{d2}), … (W_{gn}, W_{dn})} between words in a sentence, where Governor W_{gi} depends on Dependent W_{di} where $1 \leq i \leq n$.

T: Sentence Tree: (Parents: Child) ≡ (Governor : Dependent)

Input: S: sentence to be skimmed

Output: K: skimmed sentence

```
1.  SN ← {} //initialize set of summary nodes
2.  K ← [] //initialize list of words
3.  R ← TypedDependencies(S) //stanford parser
4.  T ← ConstructTree(R) //from typed dependencies
5.  For each node N in T
6.     Do O ← #ofOutgoingEdges(N)
7.        L ← LevelInTree(N)
8.        C ← #ofDescendants(N)
9.        I ← IncomingRelationType(N)
10.       P ← POSTags(N)
11.       F ← <O,L,C, I, P> //feature vector
12.       Do If InSkimming(F)=True // SVM-classify
13.          Then SN.add(N)
14. MT ← MinConnectedTree(SN)
15. K ← OrderAndListWords(MT)
16. Return K
```

Algorithm: *MinConnectedTree*

TN: Temporary node

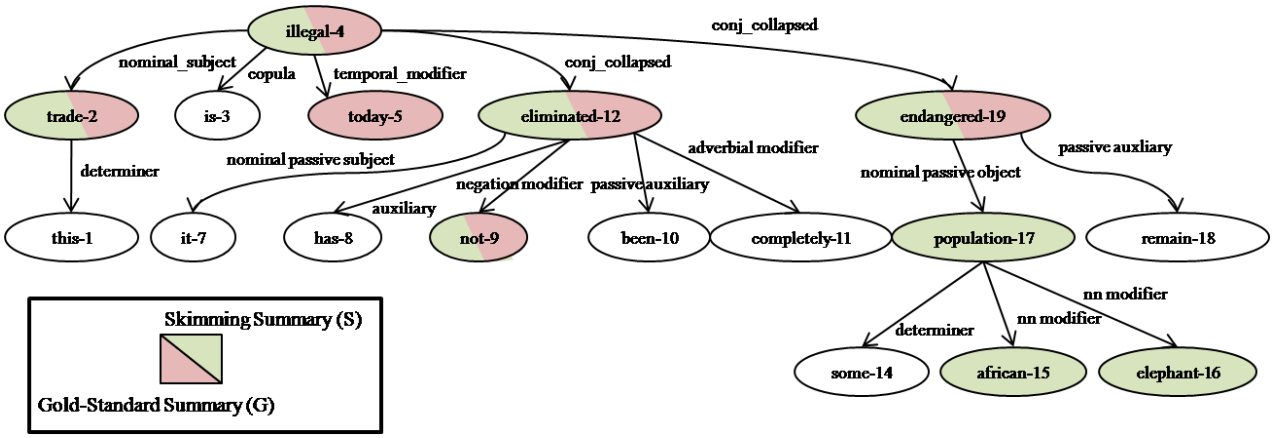
Input: S**:** Set of nodes of a sentence tree

Output: MS: Set of minimum connected nodes

```
1.  MS ← {} //initialize set of nodes
2.  For each node N in S
3.     Do TN ← N
4.        While TN ≠ NULL and TN.visited = False
5.           Do TN.visited ← true
6.              MS.add(TN)
7.              TN ← TN.Parent
8.  Return MS
```

SkimSentence first uses the Stanford Parser to find all the typed dependencies / grammatical relations (Section 4.1.1) in the given sentence (*SkimSentence:Line 3*). Then it uses the list of

grammatical relations to construct the sentence tree (Section 4.1.1) (*SkimSentence:Line 4*). An example sentence "*Michael was a businessman, as well as an environmentalist, and so gained their trust.*" **Figure 6** illustrates the sentence tree corresponding to this sentence.



Skimming Summary (S): "*trade illegal, not eliminated, african elephant population endangered.*"
Gold-Standard Summary (G): "*trade illegal today not eliminated endangered*"
(F): *"This trade is illegal today, but it has not been completely eliminated and some African elephant population remain endangered."*

Figure 7. Sentence tree illustration of the typical Skimming algorithm output compared to the human/gold summary

Once the tree is constructed, it creates a set of summary nodes which are chosen by our trained classifier (*SkimSentence:Line 5-13*). In order to choose the summary nodes, it first analyzes each of the nodes in the tree and identifies their features (*SkimSentence:Line 5-10*). #ofOutgoingEdges calculates the number of outgoing edges (normalized by total number of nodes) of a node (*SkimSentence:Line 6*). The method LevelInTree finds the level of a node in the Tree (normalized by the highest level of a node in the tree), where root is at level 1, children of the root is at level 2 and so on (*SkimSentence:Line 7*). #ofDescendents returns the total number of descendents (normalized by the highest number of descendents of a node in the tree) a node has (*SkimSentence:Line 8*). The type of the relation between a node and its parent is determined by the method IncomingRelationType (*SkimSentence:Lines 9*). POSTags method finds the parts of speech of the word corresponding to a node (*SkimSentence:Line 10*). Going back to our example in **Figure 6** let's focus on the node "gained-12". In the figure we can easily identify its features, number of outgoing edges is 2/5, level in the tree is 2/4, number of descendants is 3/9, type of incoming relation is conj_collapsed, POS tag is VBD.

With all the node features the algorithm constructs a feature vector F (*SkimSentence:Line 11*). In our example, the feature vector for the node "gained-12" is *<2/5, 2/4,3/9, conj_collapsed, VBD>*. Then the feature vector is fed to the trained SVM model (Section 4.3.1) in order to classify it to either "YES" or "NO". The InSkimming method takes a feature vector and returns True/False depending on whether the node is classified to "YES"/"NO" respectively, using the trained linear SVM classifier (section 4.2.2) (*SkimSentence:Line 12*). If InSkimming returns True (which means it can be taken as part of skimmed sentence), the node is added to a set of summary nodes (*SkimSentence:Line 13*). In this example, the nodes "michael-1", "businessman-4", "environmentalist-9", and "trust-9" are chosen as summary nodes. In the figure these nodes are circled with blue dots (**Figure 6**).

The set of summary nodes constructed this way, does not guarantee if they have any grammatical relations between them. Moreover, adding more nodes which are grammatically related to these nodes will increase the comprehension level of the skimmed sentence. So to make sure that a minimum relation exists among these summary nodes, a second algorithm *MinConnectedTree* is used, to construct a subtree that connects all the summary nodes with minimum number of edges (*SkimSentence:Line 14*)

The algorithm *MinConnectedTree* constructs a tree with minimum number of edges necessary to cover a given set of tree nodes. The algorithm runs two nested loops. The outer loop iterates over the given set of nodes (*MinConnectedTree:Line 2*). For each node it does a traversal towards root of the tree (*MinConnectedTree:Line 4-7*). In the inner loop, while traversing towards the root, it checks if the node has already been visited or not. If the node has not been visited, it is marked as visited and is added to a set of output nodes (*MinConnectedTree:Line 6*). After the nested loops are executed, a set of nodes are returned, which forms a tree with minimum number of edges (*MinConnectedTree:Line 8*). In **Figure 6**, from each of the summary nodes (circled with blue dots), the algorithm traversed towards the root and picked up any non summary nodes on the way. The traversal paths have been marked with bold red line in the figure. The only non-summary node that has been picked up (apart from the summary nodes) by *MinConnectedTree*, is "gained-12". All the nodes that have been selected by *MinConnectedTree* are shaded in the figure.

In the *SkimSentence* algorithm, once the tree with minimum number of edges is constructed (*SkimSentence:Line 14*), a skimmed sentence is made by ordering the nodes according to the position of the corresponding words in the original sentence, using the method OrderAndListWords (*SkimSentence:Line 15*)

Next we present the outcomes of our algorithm on a sample sentences using the sentence trees. For better understanding we color the nodes. The words that appear in the Gold-standard summary are colored with red and the words chosen by our *SkimSentence* algorithm are colored with green. If a word is chosen by our algorithm as well as human/gold skimming, half of the node is colored with red and other half is colored with green. In **Figure 7**, we present an example of a sentence tree and the gold-standard and skimming summaries.

4.3. Experiments on the Dataset

4.3.1. Experimental setup

In order to evaluate our skimming algorithm, we conducted experiments on our dataset. In this section we do a comparative study between the "summaries generated by Classifiers alone" and the "summaries generated by *SkimSentence*".

	Random Forest	Multi-Layer Perceptron	Linear SVM
Precision	58.92%	59.93%	59.27%
Recall	54.01%	66.75%	68.63%
F-measure	0.56	0.63	0.64
Compression	33.62%	41.16%	43.20%
After applying *SkimSentence*			
Precision	57.36%	58.34%	63.40%
Recall	63.77%	75.78%	76.57%
F-measure	0.60	0.65	0.69
Compression	40.95%	48.34%	46.81%

Table 3. Output of three different classifiers

We report the results using the standard metrics of precision, recall, and F-measure, as well as the compression ratio. We define *precision* as the ratio of the number of the words appearing in both arrays to the size of our skimmed summary. We define *recall* as ratio of the number of the words appearing in both arrays to the size of the gold-standard summary. *F-measure* is the harmonic mean of recall and precision. Finally, we define the *compression ratio* as the ratio of the number of words in a skimmed summary to the number of words in the original sentence.

4.3.2. Results and Discussion

Although we tried all of the models available in Weka, in **Table 3** we report only on the top three classifiers: Random forest, Multi-layer perceptron, and linear SVM-based model (implemented in the lib-svm [16] software). Of the three classifiers, we chose linear SVM for further experiments as it had the best classification results on our dataset. The comparative performance of different classifiers on our dataset was not in the scope of this research, especially, because classifier performance often varies depending on the size of the training dataset [65, 82].

The SVM classifier yielded the best combination of the precision: 59.2%, recall: 68.6%, F-measure: 0.64 and, the compression ratio: 43.2%. By running the Skimming algorithm on the classifier results, we improved the precision to 63.4%, recall to 76.5%, and F-measure to 0.69, at the cost of the compression of 46.8%. The 4% improvement of precision and 8% improvement of recall, at the 4% difference in compression justify the use of the Skimming algorithm on top of the classification results.

While these measures give the overall idea of the algorithm's accuracy, **it should be noted that, for the purposes of skimming, we favored recall over precision**, because the correctly recalled words typically carry the most meaning. At the same time, we balanced recall with precision not to introduce too much "noise"; **it is also notable that the "noise" words that were "erroneously" chosen by the algorithm could carry just as much meaning as the words in the gold-standard summary**, because while shortening a sentence, a human may choose just one of any two equally meaningful words. This observation is supported by the user study that showed no statistically significant difference between in the performance of subjects over the gold-standard and skimmed summaries (Section 4.4.6). Finally, in our experiments, we also balanced the compression ratio, so that the skimmed summary was no longer than the gold-standard one.

The summaries are available at http://www.cs.sunysb.edu/~faiahmed/doc/data.zip.

4.4. User Study

4.4.1. Participants

The Disability Resource Center at the Arizona State University helped us recruit 23 participants who are blind to take part in the user study. Gender representation was 7 male and 16 female. The ages of the participants were evenly distributed and varied from early twenties to late sixties. Three of the subjects considered themselves expert computer users, fourteen of them were very comfortable, five were comfortable and one was mildly comfortable with computers. All subjects were well-versed in the use of screen readers, with JAWS as their primary screen reader. Internet usage was more than 20 hours per week for twelve subjects, 11-20 hours per week for eight subjects, 6-10 hours per week for two subject, and 1-5 hours a week for only one subject.

4.4.2. Experimental Setup

To evaluate the accessible skimming interface, we built the proposed Skimming interface into our HearSay [11] screen-reading platform, which is a Java application that enables a screen reading interface for the Firefox web browser. HearSay supports all the typical navigation shortcuts available in the mainstream screen-readers such as JAWS [43], e.g., paragraph / sentence / word / character navigation, pause / resume, etc. The articles chosen for the study were typical web news articles, however, they did not contain any links or images to reduce the variation among the articles. We used RealSpeak [70] Samantha voice with the speech rate of 140 words per minute.

To evaluate the interface and the underlying algorithm, the subjects were asked to perform three tasks in each of the listening-and-comprehension and searching scenarios. In each of the three tasks the subjects were either using the Gold-standard summary, Skimming summary, or the original text of the articles.

4.4.3. Listening and Comprehension scenario

In a within-subjects experiment that had a 3 by 3 cell design, we had each subject listen to 3 different articles using 3 types of skimming: Gold-standard summary G, Skimming summary S,

S: Skimming by SkimSentnece
Elephants cows live family herds young, adult males bulls tend roam. Having baby elephant commitment. Elephants have longer pregnancy mammal 22 months. Cows give birth to one calf two to four years. Birth, elephants weigh 200 pounds stand 3 feet tall. African elephants, unlike asian relatives, are not domesticated. Range africa rain forests central west africa. Elephants found desert. Herd mali elephants migrates route desert search water.
G: Gold/Human Skimming
Female cows live family herds with young males bulls roam own. Having baby serious commitment. Longer pregnancy 22 months. Birth to calf every two four years. At birth weigh 200 pounds 3 feet tall. African elephants not easily domesticated. Range sub-saharan africa and central west. Northernmost elephants found in malis sahel desert. Herd of mali elephants migrates in circular route search water.
F: Original paragraph:
Female elephants cows live in family herds with their young, but adult males bulls tend to roam on their own. Having a baby elephant is a serious commitment. Elephants have a longer pregnancy than any other mammal almost 22 months. Cows usually give birth to one calf every two to four years. At birth, elephants already weigh some 200 pounds (91 kilograms) and stand about 3 feet (1 meter) tall. African elephants, unlike their Asian relatives, are not easily domesticated. They range throughout sub-Saharan Africa and the rain forests of central and West Africa. The continent's northernmost elephants are found in Malis Sahel desert. The small, nomadic herd of Mali elephants migrates in a circular route through the desert in search of water

Table 4. Skimming and Gold-standard summaries

and the full text F (Table 4). We varied the order of the tasks for counterbalancing and to avoid bias. Before the experiment, the subjects were given a sample listening-and-comprehension task to practice with. They were not allowed to use navigation shortcuts in this task. Each task took approximately 10 minutes, which included the question answering section as well.

A set of 12 questions was asked to test the overall comprehension and retention of information. The first question was open-ended and the rest were multiple-choice. The questions we formulated to cover the major points of each article without regard for which content actually appeared in the corresponding gold summary. The questions covered several parts of speech that carried information: 1 question was on the article topic (i.e. "What is the article about?" – "African Elephants"), 4 questions on the nouns (e.g., "In search of what do the Mali elephants migrate?" – "Water"), 3 questions on the verbs (e.g., "According to the text, African Elephants cannot be easily" – "Domesticated"), 2 on the numeric values (e.g., "How much does an elephant weigh at birth?" – "200 pounds"), and 2 on the adjectives/adverbs (e.g., "African elephants are described as" – "Hungry"). We approximately followed the distribution of the parts of speech in the original articles and were consistent in this break-down across all articles. **All questions and summaries are available for download at**

http://www.cs.sunysb.edu/~faiahmed/doc/data.zip.

4.4.4. Searching Scenario

In a within-subjects experiment that had a 3 by 3 cell design, we had each subject skim through 3 different articles using the Gold-standard summary G, Skimming summary S, and the full text F. The conditions were counterbalanced across subjects to avoid bias. Before each task, the subjects were asked a question, the answer to which they had to find in the article. The subjects were instructed not to try keyword searching because some would and some would not depending on their experience with screen readers. However, to make the scenario realistic, the questions were formulated using the wording different from that of the actual article, so that one could not easily search for the exact keywords, e.g., if the original sentence was "*Cows usually give birth to one calf every two to four years.*" The question was: "How frequently do female elephants have calves?" **All questions and summaries are available at http://www.cs.sunysb.edu/~faiahmed/doc/data.zip.**

The answer to the question was contained in the 4th and the last paragraph of each article to increase the time required to find the answer. Having the answer in the beginning of the articles would have caused the users to find the answer equally fast with and without skimming,

defeating the purpose of the experiment. Varying the location of the answer would have increased the variation of the completion times.

In this task, we measured the time it took subjects to give an answer to the question. We also measured the number of keystrokes the subjects pressed to find the answer. The subjects were given a sample task before the experiment and 5 minutes to practice with the skimming interface. They were allowed to use any navigation shortcuts which they were used to. Each task took approximately 10 minutes.

4.4.5. Questionnaires

Following each task in either of the scenarios, the participants were asked to rate the perceived difficulty of the task on a 5-point Likert scale (1=Very Easy to 5=Very Hard, **Figure 8b**, **Figure 10b**). At the end of the experiments, we read statements about the skimming experience to the subjects and asked them to rate the statements on a 5-point Likert scale (1=Strongly Disagree to 5=Strongly Agree) –

Table 2 summarizes the ratings. Finally, we also asked general questions about subjects' computer use, age, comfort level, onset of blindness, etc. We used these questions to test for performance differences among groups of subjects.

4.4.6. Hypotheses and Results

We formulated a series of specific hypotheses that we accepted or rejected based on the statistical significance testing. In this section, we provide the details on the hypothesis and discuss the results. Finally, we present the results of the analysis of the post-completion questionnaire. *We used a t-test for statistical significance testing with the significance level of alpha = 0.01.* The following are the specific hypotheses tested in the experiments and the results.

H1: After listening to summaries S and G, subjects can answer reading-and-comprehension questions equally well.

Results: Subjects were able to answer all of the reading-and-comprehension questions correctly with accuracy of 71.01% (St. Dev.=11.19%) for summary S and 71.37% (St.Dev.=9.67%) for summary G (**Figure 8a**). Using the one-tailed paired t-test, we found that

there is no statistically significant difference between the question answering accuracy for summary S and G (t=-0.134, df=22, p=0.894), accepting H1.

We also found no statistically significant differences for specific types of questions between summaries S and G (**Figure 9**).

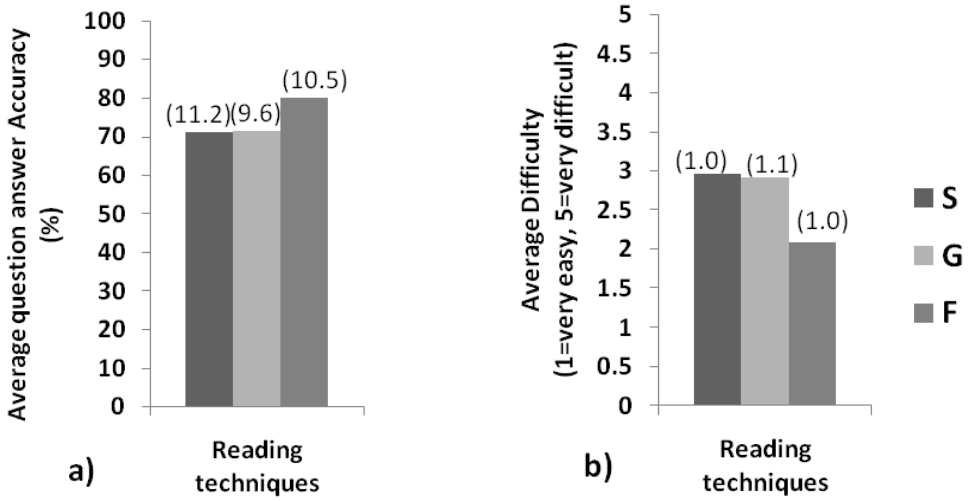


Figure 8. Average accuracy and perceived difficulty (St. Dev.) of question answering with S, G and F

Discussion: One of the most important findings of these experiments was that the subjects were able to answer accurately about 70% of the questions about the text after listening to either skimming summary S or the gold-standard summary G.

H2: After listening to summary S and full text F, subjects can answer reading-and-comprehension equally well.

Results: Subjects were able to answer 71.01% (St. Dev.=11.19%) of the reading-and-comprehension questions correctly for summary S and 80.07% (St. Dev.=10.58%) for full text F (**Figure 8a**). A one-way ANOVA was used to test accuracy differences in answering three types of summaries S, G, and F. The accuracy with which subjects gave answers differed significantly across the three types of summaries, F (2, 66) = 5.48, p = 0.0063. The post-hoc test for linear trend also shows that there is a trend, that answering accuracies increases as you read summary type S, G and F respectively, (Slope 0.04529, p = 0.0047), thus rejecting H2.

Discussion: This result was as expected, because, after all, full text was easier to understand and contained all the answers to the questions. However, it is notable, that while the length of text shortened by more than 50%, the performance in question answering lowered by less than 10%. We believe that the subjects could not answer 100% of questions after reading the full text due to information overload.

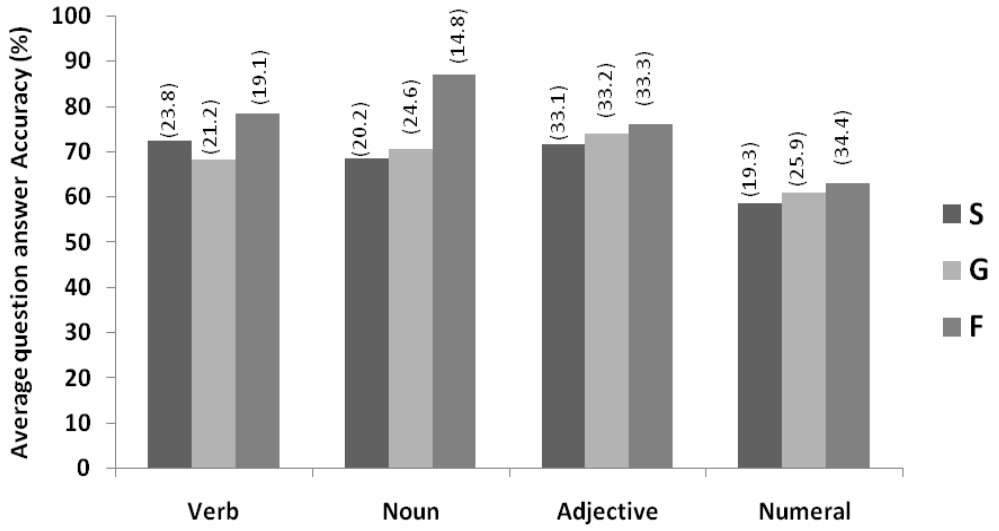


Figure 9. Accuracy of different question types

H3: Subjects perceive question-answering tasks to be of the same difficulty after listening to a summary G/S and full text F.

Results: After the completion of each listening-and-comprehension task, we asked the subject to rate their difficulty on a Likert scale (very easy=1 and very difficult=5). Subjects rated the difficulty level of the task with summary S as 2.95 on average (St. Dev.=1.02), rated summary G as 2.92 on average (St. Dev.=1.16), and rated the full text F as 2.08 on average (St. Dev.=1.06). A one-way ANOVA was used to test the difficulty in answering three types of summaries S, G, and F. The difficulty with which subjects gave answers did not differ significantly across the three types of summaries, $F(2, 66) = 2.501$, $p = 0.0898$. , thus we accept H3. (**Figure 8b**)

Discussion: This result was unexpected because reading full text is supposed to be easier than reading a skimmed summary. However, reading the full text may be putting a higher cognitive load on the subject resulting in information overload, and making it harder to

remember information and answer questions. The fact that subjects rated both G and S summaries at the same level of difficulty suggests that our Skimming summary is very close to human-generated summaries.

H4: Subjects could find information faster while skimming with G/S, than when using screen-reader navigation shortcuts.

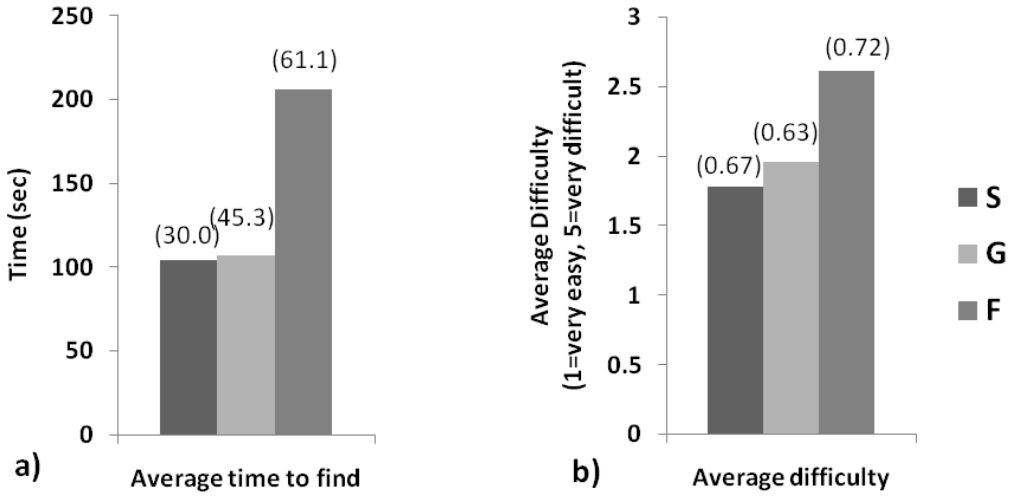


Figure 10. Average time to find and average difficulty in the searching task using skimming vs. regular shortcuts

Results: While using skimming while searching for the answer to a question, the subjects took on average 104.13 seconds (St. Dev. 30.00) with S and 107.08 seconds (St. Dev. = 45.33) with G. Using regular screen-reader navigation shortcuts, it took subjects 205.78 seconds (St. Dev. = 61.17) to answer the questions (**Figure 10a**). A one-way ANOVA was used to test the differences among times subjects took while searching for the answer to a question, with S, G and using regular screen reader shortcuts. The time they took differed significantly across three techniques, $F(2, 66) = 35.02$, $p < 0.0001$. The post-hoc test using Tukey's multiple comparison test shows the difference in time for S and F was significantly different (mean diff = -100.60, q=10.41). Same goes for G and F as well (mean diff = -97.35, q=10.08); thus, we accept H4.

Discussion: Subjects were able to find the answer almost twice as fast using skimming, compared to using regular screen-reader navigation shortcuts. This result demonstrates that skimming can save screen-reader users a substantial amount of time, depending on the size of the text. In our case, since each article had 5 paragraphs, the subjects saved up to 30 seconds per

paragraph while searching. The fact that there was no difference between S and G again speaks in favor of our algorithm.

H5: Subjects perceived ad-hoc searching to be more difficult compared to using skimming S or G.

Result: In the search task subjects perceived a difficulty level of 2.60 (St. Dev. = .72) on average while using screen-reading shortcuts without skimming. With skimming S, subjects perceived a difficulty of 1.78 (St. Dev. = 0.67) and with skimming G 1.95 (St. Dev. = 0.63) on average. Conducting one-way ANOVA test shows significant differences in the perceived difficulty level while finding answers with skimming S, G and using regular screen-reader shortcuts, $F(2, 66) = 9.487$, $p = 0.0002$. The post-hoc test for linear trend also shows: the perceived difficulty increases as you use summary type S, G and screen-reader shortcuts respectively, (Slope 0.4130, $p = 0.0001$ Another post-hoc test, Tukey's multiple comparison test shows that using screen-reader shortcuts is significantly more difficult (mean diff = -0.83, $q = 5.84$) than skimming S. Similarly using screen-reader shortcuts were more difficult than using skimming G (mean diff = -0.65, $q = 4.61$); so we accept H5.

Discussion: As the search task using screen-reader shortcuts was involved with listening to full sentences instead of summaries, subjects had to put effort and time to listen in full and find the answer to the question. So using screen-reader shortcuts turns out to be difficult.

H6: The number of keystrokes pressed was the same between searching with skimming and searching with navigation shortcuts.

Result: The average number of keystrokes pressed was 36.60 (St. Dev = 16.21) for Skimming summary S, 35.65 (St. Dev. = 15.85) for Gold Summary G, and 37.08 (St. Dev. = 26.61) for Full text F. A one way ANOVA test showed no statistically significant differences between the number of keystrokes while searching with S, G and using navigation shortcuts $F(2, 66) = 0.0301$, $p = 0.9703$. Thus, we accept H6.

Discussion: These findings demonstrate that skimming does not require more effort in terms of pressing shortcuts, compared to the regular navigation. At the same time, skimming increases the navigation speed twofold (H4).

H7: Subjects were not able to distinguish between human- and computer-generated summaries.

Results: After each listening-and-comprehension, subjects were asked to identify whether the text was written by a human. On average in 66.66% of cases (St. Dev.=48.3%) cases they mistook Skimming Summary S for the Gold-standard summary G. And, on average, in 76.2% of cases (St. Dev. 43.6%) cases they were not able to distinguish between Skimming Summary S and Gold Summary G. The one-tailed paired t-test shows that there is no significant difference between the choices subject made while detecting S and G as human written or not ($t=1.369$, $df=20$, $p=0.186$). This means most of the time they were not able to distinguish between Gold-standard summary and Skimming summary; confirming H7.

Discussion: This is another result that supports the claim that the proposed Skimming algorithm could generate summaries that were almost as good as those written by humans.

We observe no difference in performance on these tasks between groups of subjects:

H8: who lost their eyesight before the age of 10, 15, or 20 and those who lost sight later.

H9: belonging to different age categories.

H10: with different comfort level and different amount of time spent with computers.

4.4.7. Post Completion Questionnaire Analysis

The post completion questions and answers about the skimming experience are presented in **Table 5**. The analysis showed that the standard deviation for all questions was within one point, demonstrating high consistency.

From the statements made by subjects, we can conclude that skimming has a lot of potential to improve reading experience for screen-reader users. Although the subjects thought that their comprehension was somewhat impeded in the skimming mode, almost all subjects agreed that skimming made the reading faster and they wanted to use skimming in the future. We did not see

significant difference in the question-answering performance between the summaries and the regular text.

General Statements	Avg. (St. Dev.)
I wish I could look through articles faster, than I can with a screen reader	4.30 (0.82)
I often experience difficulties looking for desired information within an article	3.60 (1.11)
Skimming made reading through articles faster	4.22 (0.85)
I understood articles equally well when skimming and when reading the full article	2.82 (1.07)
I want to use skimming in the future	4.26 (1.00)
I would like to see skimming implemented in a commercial screen reader	4.60 (1.15)

Table 5. Average 5-Point Likert scale values (St. Dev.)
(Scale 1=Strongly Disagree to 5=Strongly Agree)

One of the subjects made a comment that non-visual skimming is "…very useful to glimpse over articles. May be very useful for test takers and also for sighted people (while driving and reading news/articles)". This reveals a possible application of the skimming that would allow not only people who are blind, but also sighted users to take advantage of audio skimming. Skimming can be useful to the sighted users who want to relax their eye and listen to web content, but do it faster.

5. VARIABLE SPEED SKIMMING ON TOUCH INTERFACE

5.1. Variable Size Summaries

5.1.1. Background

Summarization algorithms for non-visual skimming will enable blind users to process text content in web pages faster; however, the advantage of visual skimming is in the ability of the reader to change the speed of skimming freely, depending on the relevance of the passage. For non-visual skimming to be as useful, it should also let the user read at different speeds, too. If I leave the speech rate unchanged, variations in speed can only be achieved by changing the length of the summary. Several works have been done on generating summaries/abstracts of different size. In this section we will mention the selected ones.

In the work [34], the author attempted to bring the idea of variable size audio scanning service for blind, which is called telegraphic text reduction. They have introduced 8 different reduction levels of texts applied on text documents. This is basically a finite state machine, where they first separate tokens and identify parts of speech. Then the POS annotated text is passed through verb group markers and then a filter which identifies the head verbs of the verb groups (i.e., Head of Passive verb group). Next it is passed through a noun group annotator which identifies the principal and head nouns (i.e., FreeNoun, PrepNoun). A couple of other filters are used to identify the syntactic dependencies (i.e., Subject-Verb, Verb-Object). The reduction level they present here is based on different parts of speech, verb groups, syntactic relations. One strict rule they mentioned that the negation should always propagate with verb. Few example rules are: 1) pick only the proper names and no subclasses 2) only subjects, object nouns, no sub-clauses, 3) only subjects, head verbs, object nouns, no sub-clauses etc. If we closely look at the rules, we can see it is increasingly adding different parts of sentences to create different reduction levels.

The reduction technique they used has several shortcomings. First, the authors have identified features and predefined rules based on those features. There was no basis or proof

upon which the features have been selected. In addition there is no guarantee on how it is going to perform in different domain, because the important information may differ with various domains, where general rule cannot be applied. The ideal approach to create generic rule to support all kinds of domain, would be a training a model and use it to identify salient parts of a sentence, which I am doing (Section 4). The second shortcoming is, it does not ensure summarizing every sentence and it can skip a sentence entirely if it finds it does not contain any useful information according to the rule. These types of compressed sentence cannot enable skimming interface, as the screen reader users wants to keep a mapping between the reduced sentence and the original sentence [3]. The skimming interface will better simulate the visual skimming if we can let user switch between skimming and normal reading mode seamlessly (without changing the reading position). Therefore if a sentence is entirely skipped while preparing the reduced version of the sentence, then the system will have nothing to read to the user if he/she switches from normal reading mode to skimming for that particular sentence. Finally although they gave an example scenario of the possible application of their approach, they have never evaluated their technique/system with screen reader users in realistic environment, due to which we cannot justify the effectiveness of their technique.

A tool called Summate [36] provides variable length summaries which is based on sentence locations in a document. This particular tool parses the DOM and attempts pick the first sentence of each paragraph. At most 4 sentences are used: the first, the first sentence of the last paragraph, the sentence laying in the upper (75%) quartile and lower (25%) quartile. They consider sentences inside quotations as it is. The output of the summarizer can vary in sizes, depending on the number of sentences used in the summary. The authors also denote the quality of summaries by the terms High, Medium and Low. For example summary using four sentences contains more content and more expressive compared to those using one sentence. There for the summaries using four sentences are denoted as high quality and summaries using one sentence are denoted as low quality.

There is a firefox plugin [28] called Great Summary [79] for summarizing web pages specially articles. Once the tool is installed the tool appears in the firefox menu and can be applied on article web pages. After the tool is invoked it generates a couple of sentences as summary of the article, and reports if summary cannot be created. This plugin also provides an

option where user can select the number of sentences to be used to create summary; available options are 1 to 100 sentences. This tool enables variable length summaries by offering this option. Great Summary presents summary in a separate tabs in a new web page. This tool is basically about reviewing the summaries. But user often has to go back and read the original content to make sure if there is any missing information or to read further details. Therefore this tools is not appropriate to support non-visual skimming. Pluribo: Instant Summaries [64] is another plugin for firefox that targets the website Amazon.com. It basically generates summary of the customer reviews on a particular product. It helps user to learn the key lessons on the pros and cons of the product without going through all the past comments made by other people. Here you can also control the size of the summary. This is definitely a useful tool, but cannot be applied in general purpose summarization. One of the most popular firefox plugins SummaryFox [37] is a tool for Mac OS X, which provides selective summarization. User can highlight the text to summarize and select "Summarize" from the right-click menu, the summary service will come up with the summarized version of the selected text. This comes with extra option of tuning the summary size using a slider: from 1% to 100%. SummaryFox is same as other plugins in terms of presenting the summary and the original text but not having a mapping between them.

A statistical keyword extraction technique [57] presents an algorithm which is applicable to single document without using a corpus. This method is based upon extracting frequent words and the words that co-occurs with frequent words. They basically observe whether the distribution of co-occurrences of a term *x* with frequent terms is biased to a particular subset of frequent terms; if it is then the term *x* is probably a keyword. They used statistical measure to calculate the degree of bias. The main goal of this paper is to find a keyword extraction method that does not use corpus. In the first step they identify co-occurring terms. Once they identify co-occurring terms, they are clustered based on the similarity of distributions of co-occurrences among them. Subsequently they used pairwise clustering which basically combines the terms that co-occur more frequently, to yield relevant terms in the same cluster (i.e., doctor, nurse, hospital). Afterwards they calculate a special statistical measure of each term. Finally the algorithm outputs the given number of terms having the largest statistical measure they calculated. So user can specify the number of words they want and the algorithm outputs it. The main drawback of this algorithm is, they use the whole document to extract keyword where they calculate the word frequencies in the document, which is not useful for skimming. We explained

in Section 2 that a lot of web documents usually have small snippets of information, where the frequency of words will be a distinguished feature to identify a keyword. Besides this particular technique is likely to miss a sentence if they don't find any of the words in it does not co-occur with frequent terms with a bias. This is also against the requirement of the skimming interface, which requires each sentence to be summarized.

In this objective, I experiment with different text analysis tools and design novel summarization algorithms that could generate variable-size summaries for powering variable speed non-visual skimming. Just as mentioned in the introductory sections, I will focus on summarizing continuous textual content only. Once the properties are identified, I will find/design summarization algorithms that support different compression ratios. The confidence score of a classifier, which is at the base of the proposed summarization algorithm, may be turn out to be very useful for "continuous" control of the summary size, i.e. it may enable us to obtain a summary of just about any size.

5.1.2. Generation of Variable-Size Summaries

5.1.2.1. Basis

In my work [4] (Section 4.3), I have experimented with several classifiers to generate extractive summaries. I have observed that different classifiers return summaries of different sizes. In addition, some of the classifiers (in our experiment MultiLayer Perceptron and SVM) had a confidence score that expressed the estimated probability that the classifier categorized the results correctly. Therefore, if such a classifier is trained to categorize each word separately, then, by varying the confidence-score threshold, it may be possible to vary the *compression ratio* of the text and produce summaries of different size.

I will first formulate the requirements for the structure of variable size summaries to support both the line-by-line and diagonal skimming. I hypothesize that regardless of the summary size, whenever possible, variable-length summaries should have a good coverage of the target text, e.g., they should include M words per sentence/paragraphs or M words for every K words of text. Good coverage, as long as the words are not repeated, can give the user both a good understanding of what the text is about and enable the user to search for information effectively.

One of the possible requirement can be that larger summaries should include the content of smaller summaries, i.e., S_n is a full text of length *n*, and S_1 is this text's one-word summary (which contains at least one word), S_2 is a two-word summary (which contains at least two words), etc., then summaries should have the following property: $S_1 \subseteq S_2 \subseteq \ ... \ \subseteq S_n$. This property will enforce consistency across summaries; making it easy for the user to recognize a passage if the user skims through it more than once at different speeds. In contrast, if $S_i \nsubseteq S_{i+1}$, then passage recognition may be inhibited, especially if the two summaries share little content. These properties will be verified both in focus groups and in controlled experiments with blind users. The experiments may yield more properties that summaries will need to follow, for instance, that shorter summaries should have mostly nouns.

In section 4, I have designed an algorithm *SkimSentence* to choose words from a sentence to create a constant size summary that can support skimming interface. We follow a machine learning based method to train a model using the features we extract from words in the sentence. Later we use the model to classify the words whether they are "in Summary" or "not in Summary". The top 3 classifiers we used for constant size summarization are: Random forest, Multi-layer perceptron, and linear SVM-based model. The performance comparison of different classifier is beyond the scope of this dissertation. With the classifier and with SkimSentence algorithm, SVM-based model outperformed all other classifier in terms of F-measure and compression. One of the features that we did not use training the classifier is whether the word-node is the root of the tree. The root of the tree is grammatically connected/related to all other words directly or indirectly. In a sentence tree the child node (dependent node) is dependent on its parent node (governor node). Hence all the word-nodes are directly/indirectly dependent on the root node.

The standard term that is used in statistics, to measure the reliability of an estimate is confidence score/level/coefficient. It is a specific kind of interval estimate for a population parameter. Confidence score is calculated from the observation which is different from sample to sample. Sample to sample calculation is different because if the experiment is repeated it includes the parameter of interest most of the times. Confidence score determines the probability of the parameter to be contained in the observed interval.

According to [24] "A confidence interval with a particular confidence level is intended to give the assurance that, if the statistical model is correct, then taken over all the data that might have been obtained, the procedure for constructing the interval would deliver a confidence interval that included the true value of the parameter the proportion of the time set by the confidence level." There is a difference between point estimates and interval estimates. Usually a point estimate is a single value which represents the estimate of a parameter of interest (i.e., mean of some data). On the other hand, interval estimate provides a range within which the parameter is likely to lie. Normally the confidence interval is presented in a table with estimates of the parameters to show the reliability of the estimates.

The probability value associated with a confidence interval is called confidence score (1-α). Normally it is written as a fraction. An example of the confidence score is can be (1-0.09) = 0.91, i.e. a 91% confidence score. A practical example: suppose a soccer tournament organizer predicted that, for the current season, team X will win 50% of the matches they play. The organizer may assign a 95% confidence score to the interval 50% plus/minus 5%. Therefore the organizer thinks it is very likely that team X will get points from 45% to 55% of the total matches they play. With the same information the organizer may assign a 90% confidence score to the interval from 47% to 53%. So it's little bit less likely than before that X will get points from 47% to 53% of the total matches they play. The length of the confidence interval is primarily influenced by the sample size used in the estimation.

5.1.2.2. Detailed Procedure and Algorithm

In our experiment, Linear SVM-based classifier and Multilayer perceptron provides a confidence score (estimation of output class probabilities) for each classification, which is the empirical probability that the classification is correct. In [60] the authors presented a comparison of different post-processing methods for multi-class probabilities with standard SVM. They proved, SVM allows estimating posterior probabilities more accurately than classical multilayer perceptron.

We use as open source machine learning library Weka (section 4.3.1) to train a model using SVM classifer and use the trained model to classify words in a given sentence (whether they are in the summary or not) along with classification confidence score. Therefore instead of

outputting a class, it outputs confidence score/probability of classifying the word in “Yes” class (Yes means the word will be extracted for summary) and that of “No” class. In section 4.1.1 we described how we used Stanford parser to generate a sentence tree. From the tree we identified different features (section 4.2).

From the main characteristics we determined in section 3.3, we formulated we should do extractive summarization and each of the sentences should be summarized. And for variable level summary if we want to use a threshold that will control the size of the summary, the most reliable parameter is confidence score, because it can be used to rank the words according to their importance. In order to have a linear control over the summary size we have designed a simple technique to reassign the confidence scores in such a way which will be a linear function of the summary size and will not lose information on word ranking. There are few steps we do after we obtain the confidence score associated with each word. In the first step, we sort the words according to the confidence scores. Then we reassign scores to the words from a range of 0.0 to 1.0 in a equal spaced manner (i.e. the one with lowest confidence score will get the score of 0.0 and the one with highest confidence score will get the score of 1.0). Finally reorder the words to preserve the original position.

The steps are described with an example in **Table 6**:

Original Sentence:	*"Amy is a busy student"*
Words with confidence score (SVM)	*(Amy, 1.0) (is, 0.6) (a, 0.5) (busy, 0.8) (student, 0.7)*
Sort by Confidence Score	*Amy busy student is a*
Reassign 1.0-0.0 equally spaced way	*(Amy, 1.0) (busy, 0.75) (student, 0.5) (is, 0.25) (a, 0.0)*
Reorder by original position	*(Amy, 1.0) (is, 0.25) (a, 0.0) (busy, 0.75) (student, 0.5)*

Table 6. Basic steps of the algorithm with an example

After we refine the word-scores, we can now use a parameter to specify the size of the summary. Using this parameter we can control how big the summary would be in terms of percentage of original text. This parameter can take values from 0.0 to 1.0. For example if the parameter is set to 0.5, then all the words with score greater or equal to 0.5 will be extracted from the sentence to form the summary. This means we will not be using (1.0-0.5 = 0.5) around 50%

of the words, so the summary size will be around 50% of the original. If the parameter is set to 0.7, then we extract all the words with score equal or more than 0.7, which means we throw away around 70% of the words and summary will use (1.0-0.7 = 0.3) around 30% of the words from original text. We used the term “around” while mentioning the percentage, because we have to consider the word as a whole while choosing them for summary rather considering part of it. The words are not of equal sizes. Therefore we cannot use the word “exactly” when we mention the summary size in percentage of original text size.

In addition to these: two additional rules have been included: First, the root of the sentence tree will always have the confidence score of 1.0, which means it will always be included in the summary. The reason is, to maintain the mapping between the original sentence and its summary for better comprehensibility for screen-reader users; we have to have at least one word in the summary. This is required to support the skimming interface, so that it can switch between sentence and its summary seamlessly without changing the reading position. Second rule is, we will assign the punctuations a confidence score of 1.0. In section 3.1.1.3 we found that punctuations were preserved by the human summarizer in most of the cases, because punctuation partitions sentences and helps contain original semantics. Therefore we keep the punctuations those appears in the middle of a sentence.

An example sentence and its variable length summaries are given below (**Table 7** and **Table 8**).

Original Sentence: *Afterwards, they often spray their skin with a protective coating of dust.*

After sorting and reassigning scores, generated by the SVM Classifier for each word in the sentence is presented in the **Table 7**

Words	Confidence Score (2 decimal digits)	Reassigned Score
Afterwards	0.26	0.5454545454545454
,(coma)	0.37	1.0
they	0.24	0.6363636363636364
often	1.0	0.18181818181818182
spray	0.24	1.0
their	0.24	0.2727272727272727
skin	0.64	0.9090909090909091
with	0.0	0.0
a	0.24	0.36363636363636365
protective	0.24	0.45454545454545453
coating	0.64	0.8181818181818182
of	0.0	0.09090909090909091
dust	0.64	0.7272727272727273
.(period)	1.0	1.0

Table 7. Words and corresponding reassigned scores for sentence

Next we present how we use the threshold to generate variable length summaries in the same sentence (**Table 8**)

Summary	Threshold
Afterwards, they often spray their skin with a protective coating of dust.	0.0
Afterwards, they spray their skin a protective coating dust.	0.2
Afterwards, they spray skin protective coating dust.	0.4
they spray skin coating dust.	0.6
spray skin coating.	0.8
spray.	1.0

Table 8. Summaries for different threshold for the sentence
"***Afterwards, they often spray their skin with a protective coating of dust.***"

Based on the above we have designed the algorithm *VariableSummary*. The algorithm takes a sentence and a confidence-score threshold to generate a summary of a certain importance and size. So for different thresholds it will generate summaries of different importance and different sizes.

Algorithm: *VariableSummary*

R: Set of Typed Dependency Relations {(W_{g1}, W_{d1}), (W_{g2}, W_{d2}), … (W_{gn}, W_{dn})} between words in a sentence, where Governor W_{gi} depends on Dependent W_{di} where $1 \leq i \leq n$.

S: Set of <confidence score, word node> tuples: {(CS_1, N_1), (CS_2, N_2),… (CS_n, N_n)}.

T: Sentence Tree: (Parents: Child) ≡ (Governor : Dependent)

Input: S: sentence to be skimmed; CT: Confidence Score Threshold

Output: M: summary as list of words

```
1.  S ← {} // set of <confidence score, word node> tuples
2.  M ← [] //initialize list of summary words
3.  R ← TypedDependencies(S) //stanford parser
4.  T ← ConstructTree(R) //from typed dependencies
5.  For each node N in T
6.     Do O ← #ofOutgoingEdges(N)
7.        L ← LevelInTree(N)
8.        C ← #ofDescendants(N)
9.        I ← IncomingRelationType(N)
10.       P ← POSTags(N)
11.       F ← <O, L, C, I, P> //feature vector
12.       confidence_score ← getConfidenceScore(F)
13.       S.add(confidence_score, N)
14.  M ← WordsAboveThreshold(S, CT)
15.  Return M
```

Algorithm: ***WordsAboveThreshold***

Input: SN: set of <confidence score, word node> tuples; CT: Confidence ScoreThreshold

Output: M: summary as list of words

```
1.  M ← [] //initialize list of summary words
2.  FN ← {} //initialize final set of nodes
3.  SortedSN, newSN ← {} //init list of <score, node> tuples
4.  SortedSN ← Sort SN by confidence score in ascending order
5.  NewSN ← distribute scores 0.0-1.0 equally, in SortedSN
6.  For each tuple <S, N> in NewSN
7.     Do If S >= CT
8.        Then FN.add(N)
9.  M ← OrderNodesGetWords(FN)//get words in original order
10. Return M
```

5.1.3. User Study : Variable Size Summarization

In this user study I compare several types of summaries of different sizes generated by our *VariableSummary* algorithm in terms of comprehension level as it presented to blind people. This experiment will help us establish the relation between summaries of compression level output by our summarizer. In addition it will compare the effectiveness of the summarization algorithm output to human generates summaries. In the experiments, we will measure the ability of the subjects to understand and retain information after listening to or reading different summaries. We will also measure subjective perception of quality of skimming summaries. Statistical significance tests (e.g., such as two-tailed paired t-test and analysis of variance) will be used to compare the effects of the interfaces on the dependent variables.

5.1.3.1. Experimental Setup

To evaluate our skimming algorithm we have used the same interface we used the user study described in section 4.4. The articles we chose for the study were normal news articles from

Original paragraph: S0 : 0% Compression of the Original Content
Female elephants cows live in family herds with their young, but adult males bulls tend to roam on their own. Having a baby elephant is a serious commitment. Elephants have a longer pregnancy than any other mammal almost 22 months. Cows usually give birth to one calf every two to four years. At birth, elephants already weigh some 200 pounds (91 kilograms) and stand about 3 feet (1 meter) tall. African elephants, unlike their Asian relatives, are not easily domesticated. They range throughout sub-Saharan Africa and the rain forests of central and West Africa. The continent's northernmost elephants are found in Malis Sahel desert. The small, nomadic herd of Mali elephants migrates in a circular route through the desert in search of water
S20 : 20% Compression of the Original Content
Female elephants cows live family herds their young, adult males bulls tend to roam their own. Having a baby elephant serious commitment. Elephants have longer pregnancy other mammal almost 22 months. Cows give birth to one calf two to four. birth, elephants weigh some 200 pounds 91 kilograms stand 3 feet 1 meter tall. African elephants, Asian relatives, are not easily domesticated. They range sub-Saharan Africa the rain forests central and West Africa.The continent's northernmost elephants found Mali's Sahel desert. The small, nomadic herd Mali elephants migrates a circular route the desert search of water.
S40 : 40% Compression of the Original Content
Female elephants cows live family herds young, adult males bulls tend roam. Having baby elephant serious commitment. Elephants have longer pregnancy mammal 22 months. give birth to one calf to four. birth, elephants weigh 200 pounds 91 kilograms 3 feet 1 meter. African elephants, Asian relatives , are not. range sub-Saharan Africa rain forests central West Africa. northernmost elephants found Malis Sahel desert . small, nomadic herd Mali elephants migrates circular route desert search water.
S60 : 60% Compression of the Original Content
elephants cows live family herds, adult males bulls. Having baby commitment. have longer pregnancy mammal 22. give to calf to four. , weigh 200 pounds kilograms 3 feet 1.Aafrican elephants, Asian, not. range Africa rain forests West Africa. elephants found Sahel desert. , herd Mali elephants migrates route desert search water.
S80 : 80% Compression of the Original Content
elephants live family, bulls. Having baby. have pregnancy 22. give to to. , weigh 200 3 feet. elephants, Asian, . range West Africa. found desert. , herd migrates desert search.

Table 9. Summaries of Different Compression level

different websites. To evaluate the skimming algorithm subjects were asked to perform five tasks in a scenario called "Listening and Comprehension". In each of these tasks subjects will use any of the five type of summaries: S0(0%), S20(20%), S40(40%), S60(60%), S80(80%); generated by our *VariableSummary* algorithm. In a within-subjects experiment that had a 5 by 5 cell design, we had subjects listen to 5 different articles using 5 different compression levels (**Table 9**). We varied the order of the tasks for counter balancing and to make it unbiased. In the beginning the subjects were a sample listening-and-comprehension task to get familiar with the speech and system. This particular task did not require any use of the shortcuts. Including the question answering section each task took 10 minutes on average. A set of 12 questions were asked. We followed the same question categories as we used in section 4.4.3.

After completing each task the participants were asked to rate the perceived difficulty of the task on a 5-point likert scale (1=Very Easy to 5=Very Hard). At the end of experiments we read statements about the skimming experience to the subjects and ask them to rate the statements on a 5-point likert scale (1=Strongly Disagree to 5=Strongly Agree) (**Table 10**).

5.1.3.2. Power Analysis

Power analysis is an important part of experimental evaluation. It allows us to find out the minimum sample size needed to detect an effect of a given size with a given statistical significance level. Similarly once can find the probability of detecting an effect of a given size with given statistical significance level and sample size. Power analysis can be done in either *a priori* or *post hoc* style. *A priori* style power analysis is done before the data is collected, and is typically used to approximate sufficient sample sizes to achieve adequate power. *Post-hoc* style analysis is done after data collection is done, which we don't consider for our user study. Basically as the power of the statistical test increases, the probability of a Type II error (making a false negative choice) occurring decreases.

Four measures have close relationship, sample size, effect size, significance level, power. Given three of these we can determine the fourth.

For this particular user study we will be dealing with 5 groups of data (five summaries of different compression levels) in order to discover the relationship among them. We will no ANOVA test (Analysis of Variance) to check if these five groups of data are statistically significantly different and if there is any linear/nonlinear trend among these groups.

I have used **R** (statistical programming language) [30] and a power analysis library [66] to do the power analysis. The formula used in power analysis for ANOVA is:

```
pwr.anova.test(k = , n = , f = , sig.level = , power = )
```

Where

```
k = number of groups

n = sample size

f = 0.1, 0.25, and 0.4 represents small, medium, and large effect
sizes respectively

sign.level = significance level of the test (i.e., 0.05)

power = Probability of rejecting a false null hypothesis (i.e., 0.7)
```

For comparing 5 different groups of data, a large effect f=0.4, significance level=0.05 and for power 0.8 we execute the following command in **R**

```
pwr.anova.test(k = 5, f = 0.4, sig.level = 0.05, power = 0.7)
```

The computed estimated sample size is n = 13.08497

So the minimum number of subjects required for the study is 14 ≈ 13.08497.

5.1.3.3. Participants

Disability Resource Center of Anonymous State University helped us recruit 15 participants who were visually impaired, for our user evaluation. 6 of the participants were male and 9 were female. The age distribution of the participants was evenly distributed from 25 to 62. All of them were well-versed in screen readers having JAWS as their primary screen reader. 3 of them

considered themselves expert user, 6 of them were very comfortable, 5 of them were comfortable, 1 of them were mildly comfortable with computer. Distribution of internet usage was: 4 subjects use more than 20 hours per week, 8 subjects use 11-20 hours per week, 1 subject uses 6-10 hours per week, 1 subject uses 1-5 hours per week and 1 subject never uses internet.

5.1.3.4. Hypotheses and Results

I have formulated specific hypotheses which I accept or reject based on the statistical significance test. We will present the hypotheses and the results from user evaluation and discussion on the results as well.

We considered statistical significance level of alpha = 0.05, while conducting the ANOVA(analysis of variance) tests on the data. As discussed in the power analysis (section 5.1.3.2) we achieve the power of 0.7 for the statistical test we conduct on the data. Because we use relatively high value of power for power analysis and use the minimum number of subjects required (Section 5.1.3.2), therefore the probability of the Type II error will be low in ANOVA we conduct here.

H1: After reading each of the summaries S0, S20, S40, S60 and S80 subjects can answer decreasingly less questions correctly.

Results: Subjects were able to answer with average accuracy of 0.75% (St. Dev. = 0.09%), 0.67% (St. Dev. = 0.10%), 0.56% (St. Dev. = 0.11%), 0.44% (St. Dev. = 0.11%), 0.10% (St. Dev. = 0.10%) after listening to summaries S0, S20, S40, S60 and S80 respectively (**Figure 11**). We would like to mention that the actual order of reading these summaries was randomized to remove the bias.

A one-way ANOVA was used to test accuracy differences in answering four types of summaries A, B, C, and D. The accuracy with which subjects gave answers differed significantly across the four types of summaries, $F(4, 70) = 106.6$, $p < 0.0001$. The post-hoc test for linear trend shows that there is a significant trend, that answering accuracies decreases as you read summary type S0, S20, S40, S60 and S80 respectively, Slope -0.1536, $p < 0.0001$.

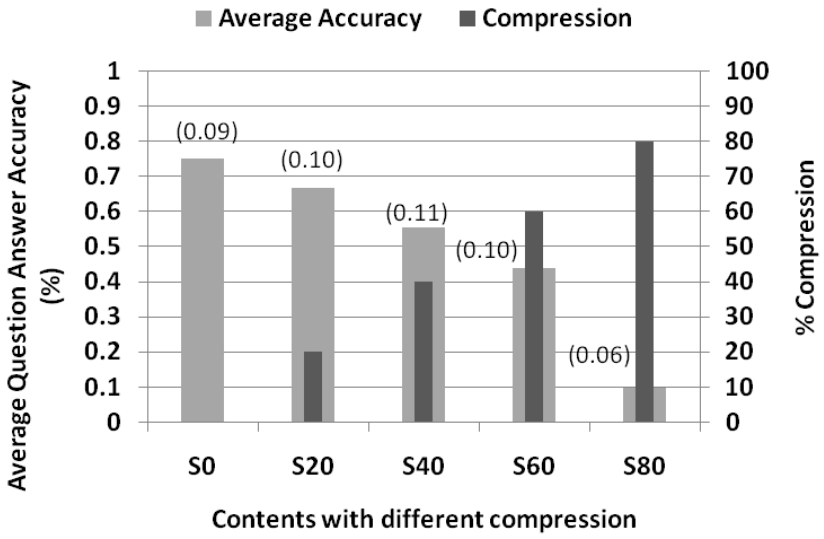


Figure 11. Average correctness (St. Dev.) of question answering with summaries S0, S20, S40, S60 and S80

Discussion: Overall, we observed a gradual increase in the correctness of the answers. This means the comprehension level of the summaries increases as the compression level decreases. We also found these summaries are significantly different in terms of user comprehension level, which ensures that no two consecutive summaries have almost similar content. So everything we present a different summary to the subjects, it has significantly more or less content than the previous ones. So these five different summaries can be used to support skimming interface to support variable speed skimming.

H2: Subjects perceive the questions to be increasingly harder for summaries S0, S20, S40, S60 and S80 respectively.

Results: We have used a Likert scale (1=very easy to 5=very hard) based on which subjects rated the difficulty of the listening-and-comprehension task on average: 2.10 (St. Dev. = 0.70) for summary S0, 2.70 (St. Dev. = 0.10) for summary S20, 3.60 (St. Dev. = 0.71) for summary S40, 4.23 (St. Dev. = 0.78) for summary S60 and 4.73 (St. Dev. = 0.46) for summary S80, which shows a clear increase in the difficulty level (**Figure 12**).

To test the differences of the difficulties perceived in answering five types of summaries S0, S20, S40, S60 and S80 we conducted one-way ANOVA test. The difficulty perceived by the subjects while answering questions using different summaries were significantly different from each other, $F(4, 70) = 31.78$, $p < 0.0001$. After conducting a post-hoc test for linear trend we found there is a significant trend, that answering difficulty increases as the user read summary type S0, S20, S40, S60 and S80 respectively, Slope 0.6867, $p<0.0001$.

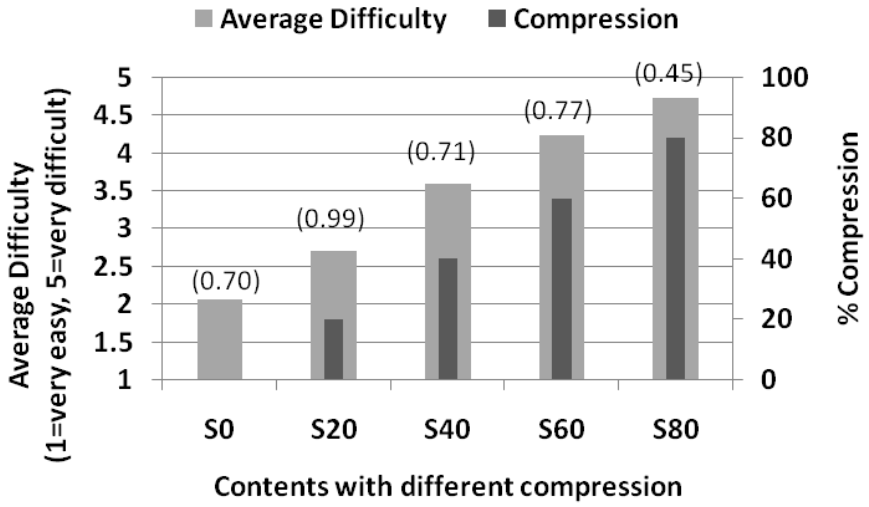


Figure 12. Average perceived difficulty (St. Dev.) of question answering with summaries S0, S20, S40, S60 and S80

Discussion: It is quite usual that the subjects found the task using summary with no compression S0 to be the easiest one, because this is the original article and it contains all the information. Most importantly the higher compression level of the content, less easy the task associated with it.

5.2. Touch based Skimming Using Variable Size Summary

5.2.1. Background

While the use of shortcut-driven screen reading remains prevalent among blind people, touch-based and, especially, hybrid (shortcut and touch) interfaces are gaining in popularity. An exemplary touch-based screen-reading interface can be found on Apple's iOS devices such as iPad and iPhone, where, by using gestures and finger dragging with VoiceOver (VO) [80] screen reader, blind people can access the user interface and browse the Web. An example of the hybrid interface can be found in Apple's Macbooks, where VO users can utilize both keyboard and touch to interact with content.

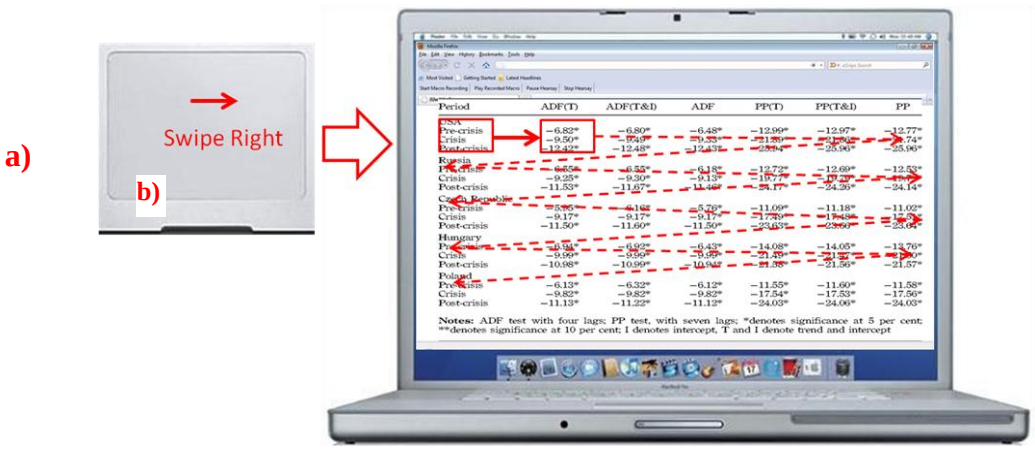


Figure 13. Finger Fatigue on VoiceOver Gesture Interface on MacBook

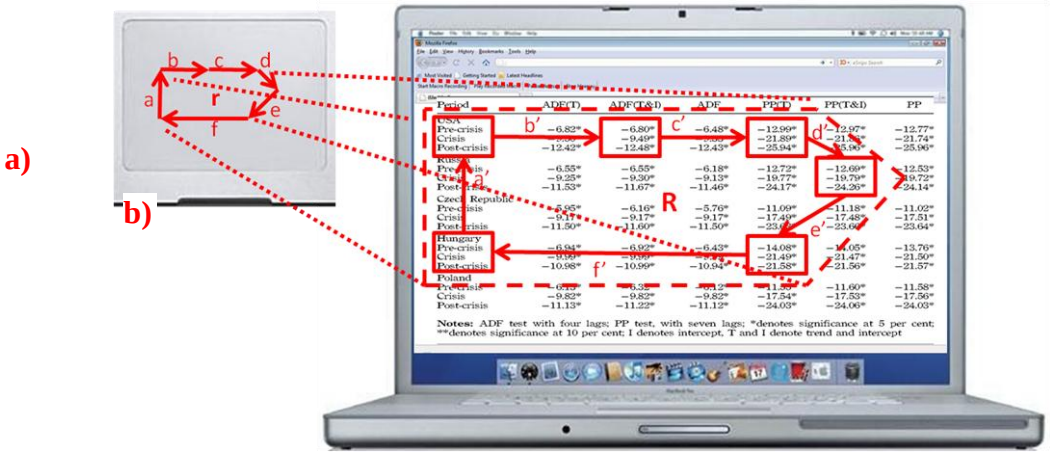


Figure 14. Fat Finger on VoiceOver Dragging Interface on MacBook

With these interfaces, users can listen to the beginning of each sentence/paragraph or listen to the content under the finger. Regrettably, this interface does not match the effectiveness of visual skimming. Furthermore, In my work [5], I have shown that "finger fatigue" (**Figure 13**) and "fat-finger" (**Figure 14**) are the two problems that make interaction with the VO's touch-based interface ineffective. Specifically, swiping through the content on the page requires too much interaction and causes "fatigue"; and dragging the finger over the touch-pad/screen makes it easy to skip over the content that is "thinner" than the finger.

Another work [68] discusses issues of small viewing form factor of touch based devices for web browsing and proposes a technique to automatically summarize and structure the web documents into readable and browsable format. They basically focus on the structure of the web documents and split the web documents into several smaller documents for contextual analysis. Once the contextual analysis is done, they get summarization of each document and which is used to label the documents. They use abstractive summarization approaches to, which means they reorder/rephrase the words to create the labels – this breaks one of the main requirements of skimming as per the requirements gathered from blind users and experiments [3]. Finally they present the webpage as a table of labels sorted by natural order. This natural order is most of the times misleading because sequential list cannot hold the context/semantics of geometric positions of the web segments. From the table of labels, the user is able to click a label of his/her choice to read in details. This basically breaks the flow of the web page and thus fails to preserve the spatial positions of the content which narrows the gap between visual and non-visual web browsing.

A unique application for summarization is provided by, handheld touch based devices such as personal digital tablets and smart-phones. These mobile environments have many opportunities to offer customized summaries. The main summarization techniques followed by these hand-held devices are document keyword extraction, and small summaries by sentence shortening. Word compaction [21] is also used for some environment. However, the ideal technique for sentence compression is based on syntactic and semantic analysis, which can also be applied [10, 34]. One of the papers [15] demonstrates that, the speed in web navigation and number of button presses or pen actions are greatly improved by applying techniques based on combination of keywords and single-sentence summaries.

A radial scroll tool [75] for small and large scale touch displays has been presented to enable faster document navigation. Instead of dragging an elevator like scroll bar or using several key

presses like page up/page down or up/down arrow key, user can make gestures on the document surface and move their finger in a circular fashion either clockwise or anticlockwise and get the document scrolled. They basically gave an idea on how to resolve "finger fatigue" for a particular case of long document scrolling. This work enables faster document browsing and can be added on top of touch-based skimming interface to scroll multipage documents. As we want to imitate the skimming experience of sighted reader, we want them to move their finger over the document and listen to all the content that comes under their finger.

In [58], a two-handed interface has been described which is a new feature to the touch based PDAs. They basically let user interact with touch based devices with thumb in combination with the pen or other finger. The device can detect the combined movement of two points on its screen without additional hardware and the thumb will act as a support to the task of pen/finger. This two-handed gesture interface can be handy if we can use it to change the settings for the skimming interface.

For mobile touch screens there is a work [46] investigating multi touch interaction techniques to make it accessible to blind people. Authors target the touch interfaces because it's still largely inaccessible to the blind users and there is not easy and efficient accessible alternative to this. Because of the blind people's inability to visually locate the objects on mobile screens, they face this accessibility issue. Here the authors introduce a "Slide Rule" which consists of very generic audio based gesture designs that allows access to mobile applications or blind people. They have categorized the applications into major subclasses and assign gestures for different features of that application. We consider these "Slide Rule"s while designing our touch-based skimming.

In this objective, I design a touch-base interface for non-visual skimming. Furthermore, I investigate whether the skimming interface can alleviate the finger fatigue and fat-finger problems. Just as mentioned in the introductory sections, I will focus on skimming continuous text that is not interrupted with other links.

5.2.2. Designing the Touch-based Skimming Interface

5.2.2.1. Strategy and Methodology

Although a lot of work has been done on touch interface to make things faster, but none of them focused on navigating web faster for people with visual impairment other than VO. As people with visual impairment prefer simpler gesture compared to complex one, we built our touch-based skimming interface as simple as possible.

In the previous section, we investigated whether touch-based skimming can be enabled by both dragging and gestures. We explored different designs for a touch-based skimming interface. We follow the participatory design principles and arrange focus-group sessions to engage accessibility consultants, and end users in the design process. Following the implementation, we evaluate and compare different interface designs in controlled and in-situ [71] user studies. We use the Android platform to enable skimming on touch screens and we use our HearSay platform to experiment with touch-based skimming on laptops with touch-pads.

We extended the typical screen-reading gesture interface, such as that implemented by VoiceOver (VO) [80], with gestures enabling the user to initiate and stop skimming. For example a down-pointing double-finger swipe may be used to initiate line-by-line continuous-mode skimming from the current reading position. The continuous skimming may help avoid the finger fatigue problem that may occur when users try to swipe through a lot of the content, trying to skim by reading the beginning of each sentence. Powered by customized summarization algorithms (Section 5 and 5.1.2), the skimming interface will narrate the most salient pieces of information, which can be easily missed when only reading the beginning of each sentence. The pinch-in and pinch-out gesture will control the desired speed of skimming, allowing the system to vary the compression ratio of skimming (Section 5). The pinch-in and pinch-out gesture is audio aided (sounds of different pitches), which will help user to know if he/she will read more or less. The user may return to the regular reading mode by choosing the lowest compression ratio by pinching-in until it reaches the lowest compression ratio.

I will extend the typical finger-dragging interface, such as that implemented by VoiceOver, to enable skimming by dragging. The resulting system will skim depending on the compression level chosen by the user by doing pinch-in and pinch-out. As soon as the user do a ”tap-and-

hold" and starts dragging the finger on the touchable surface; the system will then read in the desired skimming speed (currently chosen compression level). This interface may help alleviate the fat-finger problem, since the system will not be skipping any content as happens with VoiceOver. The effectiveness and usability of this kind of skimming in real world scenarios will be determined in user studies.

It is notable that, in VoiceOver, the geometric coordinates of gestures have no effect on what content is be read, while, in the dragging interface, both on touch screens and Macbooks, the finger position is translated to the corresponding content on the screen. Our preliminary work [5], suggested that a dragging interface that does not do coordinate translation may be more usable; for example, dragging the finger anywhere on the touch pad of a regular laptop does move the mouse pointer from its current position, rather than placing it on the screen in the position corresponding to the touch-pad coordinates. For these reasons in addition to co-ordinate translation we are providing swipe and two-finger-scroll based skimming, which will let the user control an invisible second cursor on the screen to navigate the sentences.

In this objective, I will also designed an interface for shortcut-based variable-speed skimming, because I expect that a number of screen-reader users will continue to utilize their old screen readers that do not support touch. The interface described in section 3.1.3, with a slight modification, can support variable-speed skimming. Similar to touch-based skimming, shortcut-based interface may allow only 3-4-5 speeds at most, before it becomes very difficult to operate.

In our preliminary pilot study, we did not find the backward skimming very useful, because it creates confusion in the information flow and users were performing well in the tasks. Therefore although we have the infrastructure to support skimming/variable-speed skimming backwards, we did not include it in the actual user study.

5.2.2.2. Interface

Based on the discussion we had in the previous few sections, we have designed our touch-based skimming interface. The platform we target here is the "Android" [33]. The main reasons behind this choice is it's open source. More and more devices are supporting android these days. From phone or tablet in the TV in camera in home everywhere people android is getting popular. Android is giving an open source platform for developing various kinds of applications and share them in an open marketplace for making the growing community even stronger by exploiting the power of product distribution. Android provides a very generic single application model which bridges/supports all the android supported devices. So once we develop an app for Android platform we can deploy it to a wide range of devices (i.e., tablets, smart-phones). The SDK itself comes with powerful APIs to manage the User Interface and backend hardware at the same time. Besides, the learning curve of the android development is not that high, so given our time constraint to develop a research prototype, we chose the android platform to develop the touch-based skimming interface and deploy our variable-speed skimming algorithm in it.

We used "Samsung Galaxy Tablet 10.1" as device to build the prototype. For TTS we used IVONA TTS [41] voice *"Amy"* with speech rate of 180 words per minute. The speech rate was determined from interviews and user study done with blind people in my previous work (section 3 and 4).

And exemplary interface presented by VoiceOver, is the basis of our skimming interface. In addition to what Voiceover provides to read an article content our goal is to add the skimming feature on top of that. I will maintain a cursor to keep track of the current sentence being read. The gestures we design here will use the cursor as a reference point and process the user commands accordingly.

A gesture we will implement here is "single finger swipe". Swipe with single finger will enable the user read the next or previous sentence. Swiping in the left direction on the touch interface will read the previous sentence of the current cursor/sentence of the article. Swiping in the right direction will read the next sentence of the current cursor/sentence of the article. At any point of time we can stop the system in the middle of reading a sentence, by making appropriate

gesture. For this gesture the user does not have to worry about the gesture position on the touch interface. He can make gesture anywhere on the surface he/she wants.

To enable continuous reading of the article, we incorporate a gesture "tow finger swipe down". Swiping with two fingers in downward direction makes the system read the all the sentences until the end of the article starting from the current sentence or cursor position. At any point of time you can stop reading by making the gesture for pause. Making the pause gesture once again will make it resume from the sentence where it was paused. This gesture is very useful when the user does not want to browse sentence by sentence and more interested in listening the story with as less interaction as possible.

We have introduced a gesture called "single finger tap and hold", which basically gives the user idea about the geometric positions of the sentences and paragraphs. The system starts reading the sentence when user taps and hold his/her finger for long (2 secs). The user can lift his/her finger anytime and the system will stop reading. Otherwise, the user can move his/her finger around the surface without lifting the finger. While moving the finger around the system will read all the sentence that comes under the finger. To be more specific suppose the system just started reading a sentence under the finger and the user just moved into a new sentence, then the system will stop reading the previous sentence and start reading the new one.

To pause the system we implemented a gesture called "two finger tap". To stop the system reading at any point of time, user can make two finger tap. It works different for different reading mode though. For example while reading one by one by swipe gesture, two finger tap will pause the system only, but for continuous mode reading two finger tap will toggle between pause and resume the reading.

The final and most important gesture we have designed is "pinch". The user can do pinch-in and pinch-out to control the speed of the skimming. In the core what it does is choose how much of the original text will be used to construct the summary. We tried to imitate the zoom-in/zoom-out gesture and feature which is very common in touch based devices these days. So while pinching-in we ask the system to provide more details of the story similar to zoom-in for an image zoom, and for pinch-out we want to read smaller story akin to zoom-out feature for an image.

In the user interface there is a slider that appears on the top of the browser which indicates the compression level of the summary. The left most end of the slider indicates the lowest compression level (0%), which means all of the original contents will be used to construct the summary for skimming. Similarly the right end of the slider indicates the highest compression level (100%), which means the least (at least the root word) (section 5.1.2) of the original content will be used in the skimming. So ideally user can set the slider at any point between two ends (0% to 100% compression level).

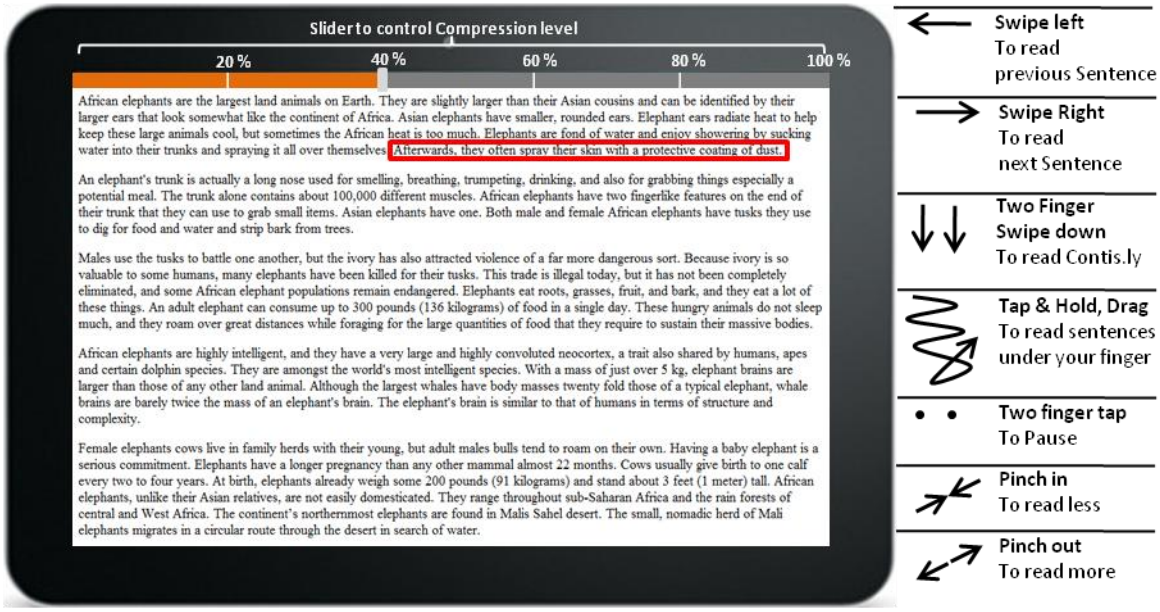


Figure 15. Touch-based skimming interface

But having a range with continuous points of compression ratio is much harder to control, compared to a range with discreet points. Also from the initial interview with our accessibility consultant provides insight on why we should have discreet points within a range to regulate compression levels. Therefore we decided to have five different compression levels which can be chosen by moving the slider in the top. The five different compression ratios are 20%, 40%, 60%, 80% and 100%. For sighted user it's very easy to set the slider to one of the five different compression ratios but moving the slider with their finger, where for blind users it's not. Therefore we used short length sounds of different pitches to let the user know which end they are approaching. For example as the user approaches to the leftmost side of the slider which

means lower compression ratio the system will play short-length sound with higher pitch. On the other hand as the user moves towards the rightmost/higher compression ratio, it will play same sound with lower pitch. So basically using the pinch-in or pinch-out gesture, the blind user controls the compression level slider. To summarize as the user pinches-in the slider will keep moving to left, decreasing compression level and playing higher pitch sounds and vice versa. The device and the interface illustrated in **Figure 15**.

There is a special module designed in the application to collect data during user evaluation. This particular module keeps track of all the users and the tasks they have done and collect user action data, time stamps for future analysis. According to your previous studies (section 3 and 4) and our accessibility consultant, we collect the number of different kinds of gestures made by different subjects, the compression ratios they are using and the time stamps. We have incorporated a file browser using which we can load html files and change the subjects id for individual experiment. The application creates separate data collection folder for each user and logs all the interactions made for each different files into that folder.

So in summary the interface we designed supports all the typical navigation shortcuts used by the only mainstream touch based screen-reader VoiceOver [80]. On top of that in order to gain control over the reading speed, we incorporated the pinch-in/pinch-out gesture.

5.2.3. User Study : Touch based Skimming

5.2.3.1. Experimental Setup

As I have designed touch-based and shortcut-based interfaces for variable-speed skimming, I have compared their effectiveness with each other.

I have conducted user studies with blind people, who were asked to complete several information-oriented tasks on touch-screen devices. Subjects were asked to look for specific information in a body of text or skim through the information to get the gist of the text using gestures and/or dragging interfaces or keyboard shortcut based interface. I have used summaries generated from *VariableSummary* algorithm (Section 5.1.2.2) in these experiments.

In the experiments, I measured the time and the number of interactions, such as number of different types of gestures (swipe, tap and hold, drag, two finger tap, two finger swipe), number of keystrokes etc. required to accomplish these tasks. Statistical significance tests (e.g., two-tailed paired t-test and analysis of variance) were used to compare the effects of the interfaces on the dependent variables. The power analysis showed the minimum number of subjects I needed for the user study. I have conducted in-situ experiments to observe the behavior of blind users to see if they can find other applications for variable-speed skimming.

In this within subject experiment we have a 5 by 5 cell design, we ask each subject to skim/read through 5 different articles using touch-based skimming interface (without skimming **T**, with fixed speed skimming **TSF**, with variable speed skimming **TSV**) and keyboard interface (without skimming **K**, with fixed speed skimming **KS**). The tasks are counterbalanced across subjects to avoid bias. For each task we ask a question to the subject, which they have to find in the article. They were instructed not to use the keyword search feature because the experience with screen readers may vary across different subjects. Also to make sure subjects cannot find the word using keyword, we created the questions without using the actual words from the article.

The answer to the question lied in the last paragraph of the article, in order to record more time to find the answer so that we can clearly detect if there was any effect of skimming or not. If the answer lies in the first paragraph then the times required to find answer are likely to be the same with and without skimming, resulting in an ineffective user evaluation. Also if we would vary the location of the answers it would have increased the variation in task completion times. In addition in order to avoid the learning bias of the subjects, we introduced dummy tasks in between two consecutive real tasks, where the answers were present randomly in different paragraphs.

For this evaluation, we will record the time user spends to find the answer, the number of gestures they use. We let the user practice with a sample task first before doing actual experiment until he/she is comfortable with it. Each of the tasks took around 10 minutes.

We have asked some post completion questionnaires on the tasks (Section 5.2.3.5)

5.2.3.2. Power Analysis

Section 5.1.3.2 illustrated the basic overview of what power analysis is and discussed about the tool we are using for power analysis. According to the tool the following formula is used

```
pwr.t.test(n = , d = , sig.level = , power = , type = , alternative =)
```

For this user study we want to compare two groups of data (i.e., times required to find information on touch interface with and without skimming). The hypotheses we want to prove whether the task "searching for info" on touch-based skimming interface is faster (let's say two times) than that on 1) regular touch interface and 2) keyboard based skimming interface. We determine the parameters in the following:

```
mean1 = mean1 (no change in speed) (null hypo)

mean2 = 2 * mean1 (twice speed) (alternative hypohesis)

common std deviation = mean1 (taking worst case, large deviation)

effect size = d = |mean1-mean2| / mean1 = 1

sig.level = 0.05

power = 0.7

type = "paired"

alternative = "greater"
```

For power=0.7, we execute the following command

```
pwr.t.test(d=1,power=0.7,sig.level=0.05, type="paired", alt="greater")
```

The computed estimated sample size is n = 6.273823

So the minimum number of subjects required for the study is 7 ≈ 6.273823.

5.2.3.3. Participants

The subjects who participated in the evaluation (section 5.1.3.3) of *VariableSummary* Algorithm, also participated for the user evaluation of touch-based skimming interface. 2 of them

considered themselves expert user, 1 of them were very comfortable, 4 of them were comfortable, 8 of them were mildly comfortable with touch based device.

5.2.3.4. Hypotheses and Results

H1: Finding information on touch-based screen reading interface with variable speed skimming is faster than ad-hoc touch-based navigation.

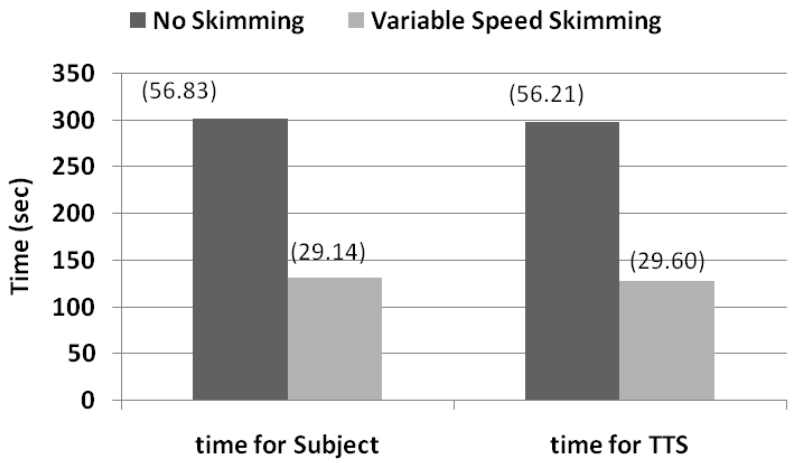


Figure 16. Average time to reach and understand the answer to a question with variable speed skimming vs. without variable speed skimming on a touch interface (St. Dev.)

Result: While recording time for task completion, we considered two aspects. First, the time user spends to navigate to the sentence containing answer and the TTS reads out the answer. Second, the time user spends to give the answer verbally. In **Figure 16** we referred to them "time for TTS" and "time for subject" respectively. While searching for information, subjects took on average 128.07 seconds (St. Dev. = 29.60) to reach the answer using variable speed skimming and 296.86 seconds (St. Dev. = 56.21) using ad-hoc touch based navigation (**Figure 16**). There was a significant difference in the speed of navigation ($t=14.47$, $df=14$, $p<0.0001$). The time it took the subjects to actually answer the questions was on average 130.93 seconds (St. Dev. 29.14 seconds) and 300.60 seconds (St. Dev. = 56.84 seconds) for skimming and reading respectively.

The one-tailed paired t-test ($t=14.22$, $df=14$, $p<0.0001$) found this difference to be statistically significant; thus, we accept H1.

Discussion: Subjects were able to navigate to the answer location 2.3 times faster using touch-based interface with skimming compared to using touch-based interface without skimming. However, to actually answer the questions, subjects were 2.2 times faster while using skimming compared without skimming. These results indicate that touch-based skimming interface can be effective in saving time while navigating long documents/web contents.

H2: Variable size skimming on touch based interface was easier than ad hoc touch-based searching.

Result: We found that, on average, people rated the difficulty of the tasks using variable speed skimming as 1.60 (St. Dev. = 0.63) and the difficulty of tasks without variable size skimming as 3.10 (St. Dev. = 0.79). Using the one-tailed paired t-test, we found skimming appeared to be significantly easier than ad hoc navigation ($t=8.88$, $df=14$, $p<0.0001$), rejecting the null hypothesis.

Discussion: This was a quite a unique finding, especially considering that the subjects have had very little experience with skimming. In our experiments with variable size summary (5.1.3) although the user found the full text S0 is the easiest one in terms of comprehension, while using the touch interface along with variable size skimming they found this combination a lot easier.

H3: Finding information using touch-based skimming interface is faster than keyboard-based skimming.

Result: The average time spent on reaching the answer using touch-based skimming is 148.06 seconds (St. Dev. = 29.60) and using keyboard-based skimming is 174.93 seconds (St. Dev. = 46.22). The difference in speed was significantly different ($t=3.11$, $df=14$, $p=0.0038$). For actually answering the questions, subjects took on average 150.33 seconds (St. Dev. = 29.14) for touch-based skimming interface and 176.93 seconds (St. Dev. = 45.77) for keyboard-based skimming. After doing one tailed paired t-test we found the difference of the speed was statistically significant ($t=3.087$, $df=14$, $p=0.0040$); thus we accept **H3.** (**Figure 17**)

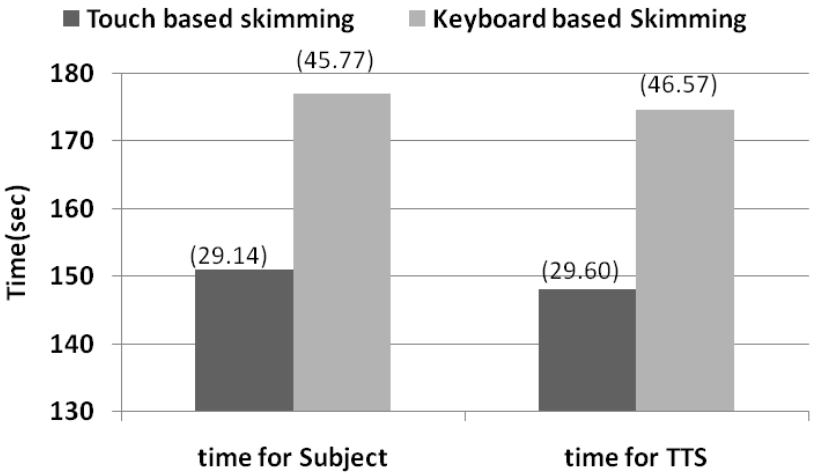


Figure 17. Average time to reach and understand the answer to a question with touch-based skimming vs. keyboard-based skimming (St. Dev.)

Discussion: Subjects were more comfortable in using touch based skimming interface because it's intuitive and unlike keyboard-based skimming you don't have to remember the position of keys or numerous shortcuts, it's just the gestures that you can make anywhere on the touch surface.

H4: Finding information using touch-based skimming interface is easier than keyboard-based skimming.

Result: We found that, on average, people rated the difficulty of the tasks with touch-based skimming as 1.96 (St. Dev. = 0.85) and the difficulty of tasks with keyboard-based skimming as 2.46 (St. Dev. = 0.83). Using the one-tailed paired t-test, we found touch-based skimming appeared to be significantly easier than keyboard-based skimming ($t=3.094$, $df=14$, $p=0.0040$), rejecting the null hypothesis.

Discussion: The touch based interface was easy to use compared to keyboard based interface, because the gestures are intuitive, the subjects did not have to spend time to recall the shortcuts or find the position of the keys.

H5: The difference between the time spent on reaching the answer and the time to actually answer the question using touch-based skimming is less than that of keyboard-based skimming.

Result: The time the subjects took to perceive the answer after reaching is 2.86 seconds (Std. Dev = 2.94 seconds) for touch-based skimming and 2.40 seconds (Std. Dev = 1.88 seconds). One tailed paired t-test shows the difference between these two times were not significant (t=0.4962, df=14, p=0.3137), thus rejecting H5.

Discussion: Although touch-based skimming is faster than that of keyboard-based, the perception time of the subjects did not vary because of the change in interface.

H6: Keyboard based skimming is faster than ad-hoc keyboard based searching.

Result: Subjects took on average 174.53 seconds (St. Dev. = 46.21) to reach the answer using keyboard based skimming and 288.27 seconds (St. Dev. = 89.81) using ad-hoc searching. One tailed paired t-test (t=6.541, df=14, p<0.0001) shows significant different in speed. It took on average 176.93 seconds (St. Dev. 45.77 seconds) and 298.8 seconds (St. Dev. = 89.21 seconds) for skimming and reading respectively, for the users to actually answer the questions. The speed difference was statistically significant as well (t=8.201, df=14, p<0.0001).

Discussion: This particular hypothesis was verified in **H8** in section 3.1.4.2. Since this time we ran it using different subjects, we could not assume that keyboard based skimming was faster than ad-hoc searching. Therefore we ran this same experiment again to ensure the hypothesis H8 in section 3.1.4.2 is not also violated for the participants of the current experiment. This also ensured users properly answered post completion questionnaires (section 5.2.3.5) on comparison between keyboard and touch interface, because without experiencing the utility of keyboard based skimming it would be hard to compare them.

5.2.3.5. Post Completion Questionnaires

At the end of the experiments, we read statements about the variable speed skimming experience using touch interface to the subjects and asked them to rate the statements on a 5-point Likert scale (1=Strongly Disagree to 5=Strongly Agree) – **Table 10** summarizes the ratings. The ratings in the questionnaires shows that majority of the subjects agrees that skimming touch-based skimming is faster than regular touch-based navigation or keyboard based skimming. They also express their high interest to use the touch interface in the future.

General Statements	Avg. (St. Dev.)
I wish I could look through articles faster than I can with a screen reader	4.60 (0.63)
I experience difficulties in fast navigation in an article with regular touch interface	4.13(0.74)
Touch based skimming made reading articles faster than regular touch navigation	4.67(0.48)
Touch based skimming is easier than keyboard based skimming	4.13(1.12)
I want to use the touch based skimming feature in the future	4.67(0.48)

Table 10. Post Completion questionnaire on Touch-Based skimming

5.3. Testimonials

To complete the research, we are offering these verbatim quotes of our subjects commenting on the skimming:

"This is my first experience with touch. Once you are familiar with the touch interface it's pretty easy. With the touch interface I got impression to move faster than keyboard based interface. I think 1-2 weeks of practice would do to get used to the touch interface."

"For keyword search skimming is the second option to look for info. For getting overview skimming is the 1st option. I think it's faster and touch can be equal good as keyboard."

"it picks up keywords and main points…I prefer touch more than keyboard based because you don't have to remember the key locations. It's easier to remember gestures…I loved the way various speed skimming is implemented. It gives me control on how much information I need…I don't think much practice is necessary"

"I like the touch based interface much more compared to keyboard based interface. I wish I never had to use the keyboard other than typing. On the other hand, touch interface is intuitive (flipping page, pinch out pinch in). Touch based skimming is really cool."

"Usually I don't like compressed article but even with compression it's better because I have independent control of the speed….. It's useful. Skimming is faster is some situation, but reading

this way is not very much fun. It's not for serious reading or study. But it's deifnitely useful to get a quick idea, when you are in short of time. ....I like touch more than keybaord, because it's easier. Does not have to learn the command keys and shortcuts."

"My brain was synthesizing the bits and pieces of the article and forming up complete picture of the story. .....I like the touch commands. .....It's faster more efficient. Touch command is very intuitive and easier to remember. I had to put less effort to get used to them. I did not have to search for key locations or remember the shortcuts like the way I do in keyboard based interface."

"Skimming is good, simple. Swiping is intuitive. I did not know touch is faster than keyboard. If I can practice it, I will make the keyboard as alternative device"

"I want to use touch interface.....Touch interface helps me to stay focused. By the time I find answer in keyboard, I fall asleep, I become bored and lost. Skimming is something that gets me to the point and using touch based skimming made it even more efficient."

"Skim is much faster, concise way to the way I typically read. Touch based interface is more convenient, quicker than key based. In key based approach I have to remember all the shortcuts and finds out key locations"

"I use a lot of software so I have to remember all the shortcuts they have, when I use them. But because in touch device you don't need to remember any shortcuts, it's just general intuition.....It makes reading a lot faster. I feel like I don't have to sit and get bored with info that I am not interested in.....I am more willing to skimming using touch, I have already learned a lot of keyboard shortcuts. I don't want to remember new ones"

One of the subjects said *"Skimming feature made the reading a lot faster. I love keyboard more than touch based skimming because I have dexterousness problem".* This particular subject had a surgery and dexterousness issue in the hand, because of which he/she preferred keyboard than touch-based interface.

6. CONCLUSIONS AND FUTURE WORK

6.1. Contribution

Novel user experience Non-Visual Skimming: From the literature review I made in this dissertation, I have not come across any tools or technology that create skimming experience for the people with visual impairment. In this work I have introduced a new tool which enables blind people navigate through long content faster than they usually do with a regular screen reader, without sacrificing the basic understanding of the content.

Keyboard Shortcut based Skimming interface: The first kind of interface I designed and evaluated in this dissertation is a keyboard shortcut based skimming interface. This is a very generic interface. It can be implemented along with any keyboard based screen reading interface available at present in the market. This interface functions like an invisible layer, which can be turned on/off at any point of time of screen-reading in order to skim/normal reading respectively. To my knowledge there has not been any such interface for blind individuals.

Touch based Skimming interface: Using Touch interface we pushed our keyboard based interface once step further towards real skimming experience. In this work I have presented a gesture based skimming interface built on top of a touch-based screen reader. The idea of using touch interface for skimming is a novel one, in terms of user interface. Visual impaired users will use their fingers instead of eyes to enjoy skimming akin to a way sighted people do.

These two interfaces have been tested in realistic user scenarios to validate their effectiveness and usability.

Fixed speed Skimming Algorithm: From literature review, examination of visual speed-reading technique and experiments with the gold-standard summaries, I have identified the ideal characteristics a summary should have to support the skimming interface. Based on the finding, I have designed a summarization algorithm that is suitable to feed the skimming interfaces. This

algorithm analyses the semantic structure, exploits grammatical relations in a sentence and used a machine learning approach to create summaries. This algorithm does not depend on the whole document (i.e. document length), so it can be used for small snippets event for a single sentence (which is a frequent case for web pages especially front pages).

Variable Speed Skimming Algorithm: I have also designed a variable size summarization algorithm in order to enable screen reader user experience variable speed skimming. Variable speed skimming is better than fixed speed skimming because it allows user to control the information flow, so that they can speed up or slow down their reading at any point of time. Skimming became more effective when Variable Speed Skimming is combined with touch interface. The combination was tested by controlled and in-situ [71] real-world experiments.

Other: In this dissertation, I have reviewed a number of related works in web accessibility area. I have interviewed a number of web accessibility experts and screen reader users in order to get an insight into the existing problem they face while using screen readers. In summary, my work opens up a new direction of fast reading technique (interface and algorithm) for the screen reader users.

6.2. Impact

Mass Benefit: The interface and the algorithm I designed can be easily implemented with any commercial screen readers existing now. Because of visual disability, blind people are always a step behind when it comes to browsing web, reading documents. With the existing screen reading technologies, visually impaired users face significant problems in fast reading. As more and more people with visual disability start using screen readers every day, this becomes more potential field where my research will have impact on.

Wider areas: The interfaces and the technologies I described here are widely applicable to people with different kind of disabilities/conditions (i.e. mobility impairments). For example instead of reading out the words, we can highlight the important words depending on the high or low compression ratio user chooses. So when people read articles he/she will choose how many important words he/she wants to be highlighted. These techniques can also be help sighted

individuals. For example if a person's hand is busy (i.e. driving, on a sunny day when you can barely see things on our touch-based device) but he can listen to articles (i.e. news, books) in a similar way blind person listens to.

Educational Impact: I have supervised and worked with 9 Master's students in total, who have gained software development experience through this project over the last four and a half years.

6.3. Future Work

Heterogeneous Content: A comment – "I wish skimming could be applied to non article page as well, where you can give a general name to a group of page elements. And I wish it could strip a web page into important parts similar to skimming" – reveals another possible direction of research that would allow users to skim other types of content, for example a list of links, menu items, headings, images. Other summarization approaches would be necessary to enable this type of skimming.

Speed Based Skimming: In my work I have enabled the skimming text contents on a touch based interface. The feature variable speed of the skimming is controlled by the screen-reader user. The compression level he/she chooses using pinch in/out is used to skim the content. To make the experience closer to that of sighted people, we should associate it with the speed of finger motion. For example when a sighted person moves his/her eyes over a long text, he can control the amount of information flow, by moving his/her eyes faster or slower. The faster the/she moves his/her eyes, the less information he/she gets and vice versa. Similarly on a touch interface, instead of moving eyes, people with visual impairment will move their finger on it and depending on the speed of their finger, we will present more or less information.

Haptic Skimming: Furthermore, a haptic (embossed) overlay will help blind users relate the position on the touch-screen with the information displayed there. While haptic overlays may give an immediate improvement in usability of web browsing, experimentation with such interfaces will help us formulate the specifications for haptic touch-screen interfaces in anticipation of the hardware making such interfaces possible.

Summarization algorithms: There are opportunities to improve the summarization algorithm (which drives the interface) in this dissertation. For example we can extend or customize other keyword extraction algorithms to be used with the skimming interfaces described in this dissertation. We can increase our corpus by involving more subjects to label the data or collaborate with other institutions to create more.

6.4. Conclusion

The problem we proposed in this dissertation is, the information overload that people with visual impairment, face during reading large volumes of information. We suggested that the solution lies in skimming. Unlike the sighted people visually impaired ones, will use a specialized screen reader interface for skimming. Though controlled experiments we have proved the importance of this problem, introduced novel non-visual skimming interfaces: 1) keyboard based and 2) touch based.

The algorithms proposed in this dissertation can summarize one sentence at a time without the dependence on the entire document. It can extract meaningful word combinations from most sentences and preserve the original order of the extracted text. I can also produce summary of different length depending on a user defined parameter. All these properties make the proposed summarization approach more suitable for supporting a non-visual skimming interface. Our experiments give reasons to believe that non-visual skimming can enable screen-reader users to browse textual content faster.

There was highly positive feedback from the study participants, which validates the usability of the interface. Furthermore, since subjects were able to perform several browsing tasks significantly faster with skimming, and skimming did not impede subjects' understanding of the content, skimming proves to be a very useful feature for any screen-reading software.

7. REFERENCES

[1] *Blackboard*. Available from: http://www.blackboard.com/.

[2] AFB. *Facts and Figures on American Adults with Vision Loss*. 2011 [cited 2011]; Available from: http://www.afb.org/Section.asp?SectionID=15&TopicID=413&DocumentID=4900.

[3] Ahmed, F., Y. Borodin, Y. Puzis, and I.V. Ramakrishnan, *Why Read if You Can Skim: Towards Enabling Faster Screen Reading*, in *International Cross-Disciplinary Conference on Web Accessibility*. 2012.

[4] Ahmed, F., Y. Borodin, A. Soviak, M.A. Islam, I.V. Ramakrishnan, and T. Hedgpeth, *Accessible Skimming: Faster Screen Reading of Web Pages*, *to be appeared in UIST*. 2012, ACM: Cambridge, Massachusetts, USA.

[5] Ahmed, F., M.A. Islam, Y. Borodin, and I.V. Ramakrishnan, *Assistive web browsing with touch interfaces*, in *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*. 2010, ACM: Orlando, Florida, USA. p. 235-236.

[6] Amitay, E. and C. Paris, *Automatically summarising Web sites: is there a way around it?*, in *Proceedings of the 9th International Conference on Information and Knowledge Management*. 2000, ACM: McLean, Virginia, United States.

[7] Arons, B., *SpeechSkimmer: Interactively Skimming Recorded Speech*, in *User Interface Software and Technology*. 1993: Atlanta, Georgia. p. 187-196.

[8] Berger, A.L. and V.O. Mittal., *Ocelot: a system for summarizing web pages*, in *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. 2000, ACM Press. p. 144–151.

[9] Berghel, H., *Cyberspace 2000: Dealing with Information Overload*. Communications of the ACM, 1997. 40(2).

[10] Boguraev, B., R. Bellamy, and C. Swart, *Summarisation Miniaturisation: Delivery of News to Hand-Helds*, in *NAACL Workshop on Automatic Summarization.* . 2001, Association for Computational Linguistics: New Brunswick, New Jersey.

[11] Borodin, Y., F. Ahmed, M.A. Islam, Y. Puzis, V. Melnyk, S. Feng, I.V. Ramakrishnan, and G. Dausch, *Hearsay: a new generation context-driven multi-modal assistive web browser*, in *Proceedings of the 19th international conference on World wide web*. 2010, ACM: Raleigh, North Carolina, USA.

[12] Borodin, Y., J.P. Bigham, G. Dausch, and I.V. Ramakrishnan, *More than meets the eye: a survey of screen-reader browsing strategies*, in *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*. 2010, ACM: Raleigh, North Carolina. p. 1-10.

[13] Borodin, Y., J.P. Bigham, R. Raman, and I.V. Ramakrishnan, *What's new?: making web page updates accessible*, in *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*. 2008, ACM: Halifax, Nova Scotia, Canada.

[14] Borodin, Y., J. Mahmud, I.V. Ramakrishnan, and A. Stent, *The HearSay non-visual web browser*, in *Proceedings of the 2007 International Cross-Disciplinary Conference on Web Accessibility (W4A)*. 2007, ACM: Banff, Canada.

[15] Buyukkokten, O., H. Garcia-Molina, and A. Paepcke, *Seeing the Whole in Parts: Text Summarization for Web Browsing on Handheld Devices*, in *Proceedings of the 10th International World Wide Web Conference*. 2001.

[16] Chang, C.-C. and C.-J. Lin, *LIBSVM : a library for support vector machines*, in *ACM Transactions on Intelligent Systems and Technology*. 2011. p. 27:1-27:27. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[17] Charmtech. *Charmtech Labs LLC.* 2011 [cited 2011]; Available from: www.charmtechlabs.com.

[18] Chieko, A. and C. Lewis, *Home page reader: IBM's talking web browser*, in *Closing the Gap Conference Proceedings*. 1998.

[19] Christopher D. Manning, P.R., Hinrich Schütze, *Introduction to Information Retrieval*. 2008: Cambridge University Press.

[20] Chuang, W.T. and J. Yang, *Extracting sentence segments for text summarization: a machine learning approach*, in *Proceedings of the 23rd annual international ACM SIGIR*

conference on Research and development in information retrieval. 2000, ACM: Athens, Greece. p. 152-159.

[21] Corston-Oliver, S., *Text compaction for display on very small screens*, in *NAACL Workshop on Automatic Summarization*. 2001, Association for Computational Linguistics: New Brunswick, New Jersey.

[22] DeJong, G., *Skimming stories in real time: An experiment in integrated understanding*, in *Research Report 158*. 1979, Department of Computer Science, Yale Univesity.

[23] Delort, J.Y., B. Bouchon-Meunier, and M. Rifqi, *Enhanced web document summarization using hyperlinks*, in *Proceedings of the 14th ACM Conference on Hypertext and Hypermedia*. 2003, ACM: Nottingham, UK. p. 208–215.

[24] Easton, V.J. and J.H. McColl's, *Statistics Glossary*. 1.1 ed. 2010.

[25] Edwards, A.D.N., *Outspoken software for blind users*, in *Extra-ordinary human-computer interaction*. 1995, Cambridge University Press. p. 59-82.

[26] Elhadad, N. and K.R. McKeown, *Towards Generating Patient Specific Summaries of Medical Articles*, in *NAACL, Workshop on Automatic Summarization.* 2001: New Brunswick, New Jersey: Association for Computational Linguistics. .

[27] Farhoomand, A.F. and D.H. Drury, *Managerial information overload*, in *Commun*. 2002, ACM. p. 127–131.

[28] Firefox. *Firefox Addon*. Available from: https://addons.mozilla.org/en-US/firefox/.

[29] Fischer, G. and C. Stevens., *Information access in complex, poorly structured information spaces*, in *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*. 1991, ACM Press. p. 63–70.

[30] Gentleman, R. and R. Ihaka. *R*. Available from: http://www.r-project.org/.

[31] Goldstein, J., M. Kantrowitz, V.O. Mittal, and J.G. Carbonell, *Summarizing Text Documents: Sentence Selection and Evaluation Metrics*, in *In Research and Development in Information Retrieval*. 1999. p. 121-128.

[32] Gong, Y. and X. Liu, *Generic text summarization using relevance measure and latent semantic analysis*, in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. 2001, ACM Press. p. 9–25.

[33] Google. *Android*. Available from: http://www.android.com/developers/.

[34] Grefenstette, G., *Producing Intelligent Telegraphic Text Reduction to Provide an Audio Scanning Service for the blind*, in *Workshop on Intelligent Text Summarization*. 1998, American Association for Artificial Intelligence Spring Symposium Series: Menlo Park, California. p. 111-117.

[35] Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten, *The WEKA Data Mining Software: An Update*, in *SIGKDD Explorations*. 2009.

[36] Harper, S. and N. Patel, *Gist summaries for visually impaired surfers*, in *Proceedings of the 7th International ACM SIGACCESS Conference on Computers and Accessibility*. 2005, ACM: Baltimore, MD, USA.

[37] Mozilla Firefox. *SummaryFox* 2009; Available from: https://addons.mozilla.org/en-US/firefox/addon/summaryfox-47140/?src=search.

[38] Hirohata, M., Y. Shinnaka, K. Iwano, and S. Furui, *Sentence Extraction-based Presentation Summarization Tecniques and Evaluation Metrics*, in *IEEE*. 2005.

[39] Hori, C., S. Furui, R. Malkin, H. Yu, and A. Waibel, *A statistical approach to automatic speech summarization.* EURASIP J. Appl. Signal Process., 2003. **2003**: p. 128-139.

[40] Islam, A.M., Y. Borodin, and I.V. Ramakrishnan, *Mixture Model based Label Association Techniques for Web Accessibility*, in *UIST*. 2010: New York, NY, USA.

[41] IVONA. *IVONA multi-lingual speech synthesis system*. 2005; Available from: http://www.ivona.com/.

[42] Jacobs, P.S., *To parse or not to parse: relation-driven text skimming*, in *Proceedings of the 13th conference on Computational linguistics - Volume 2*. 1990, Association for Computational Linguistics: Helsinki, Finland. p. 194-198.

[43] JAWS, *Screen reader from Freedom Scientific*. 2011. http://www.freedomscientific.com/products/fs/jaws-product-page.asp.

[44] JAWS_Skimming. *Skim Reading*. 2011; Available from: http://www.freedomscientific.com/Training/Surfs-up/Skim_Reading.htm.

[45] Jeffries, R., J.R. Miller, C. Wharton, and K.M. lJyeda, *User Interface Evaluation in the Real World: A comparison of Four techniques.*, in *CHI*. 1991, ACM. p. 119–124.

[46] Kane, S.K., J.P. Bigham, and J.O. Wobbrock, *Slide rule: making mobile touch screens accessible to blind people using multi-touch interaction techniques*, in *Proceedings of the*

10th international ACM SIGACCESS conference on Computers and accessibility. 2008, ACM: Halifax, Nova Scotia, Canada. p. 73-80.

[47] Kessler, T. *How to use Safari's new 'Reader'*. 2010 [cited 2011]; Available from: http://reviews.cnet.com/8301-13727_7-20007195-263.html.

[48] Kirsh, D., *A Few Thoughts on Cognitive Overload*, in *Intellectica*. 2000 p. 19–51.

[49] Klein, D. and C.D. Manning., *Accurate Unlexicalized Parsing*, in *Proceedings of the 41st Meeting of the Association for Computational Linguistics*. 2003. p. 423-430.

[50] Kurtus, R. *Reading Faster*. 2001; Available from: http://www.school-for-champions.com/grades/reading.htm.

[51] Lamb, A. and L. Johnson. *Skimming and Scanning*. 1999; Available from: http://42explore.com/skim.htm.

[52] Litvak, M. and M. Last, *Graph-based keyword extraction for single-document summarization*, in *Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*. 2008, Association for Computational Linguistics: Manchester, United Kingdom. p. 17-24.

[53] Liu, H. and P. Singh, *ConceptNet - A Practical Commonsense Reasoning Tool-Kit.* BT Technology Journal, 2004. **22**(4): p. 211-226.

[54] Lopes, R., D. Gomes, Lu\, \#237, s. Carri\, and \#231, *Web not for all: a large scale study of web accessibility*, in *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A)*. 2010, ACM: Raleigh, North Carolina. p. 1-4.

[55] Mahmud, J.U., Y. Borodin, and I.V. Ramakrishnan, *CSurf: a context-driven non-visual web-browser*, in *Proceedings of the 16th International Conference on World Wide Web*. 2007, ACM: Banff, Alberta, Canada.

[56] Marneffe, M.-C.d., B. MacCartney, and C.D. Manning, *Generating Typed Dependency Parses from Phrase Structure Parses.*, in *LREC* 2006.

[57] Matsuo, Y. and M. Ishizuka, *Keyword Extraction From a Single Document Using Word Co-occurrence Statistical Information.* International Journal on Artificial Intelligence Tools, 2004. **13**(1): p. 157-169.

[58] Matsushita, N., Y. Ayatsuka, and J. Rekimoto, *Dual Touch: A Two-Handed Interface for Pen-Based PDAs*, in *User Interface Software and Technology*. 2000, ACM: San Diego, CA, USA.

[59] Merlino, A. and M.T. Maybury, *An Empirical Study of the Optimal Presentation of Multimedia Summaries of Broadcast News*, in *Advances in Automatic Text Summarization*, MIT Press: Cambridge, Massachusetts. p. 391-401.

[60] Milgram, J., M. Cheriet, and R. Sabourin, *Estimating Accurate Multi-class Probabilities with Support Vector Machines*, in *International Joint Conference on Neural Networks*. 2005. p. 1906 - 1911.

[61] NVDA, *Non-Visual Desktop Access*. 2011. http://www.nvda-project.org/.

[62] NYTIMES. *Twitter*. 2010; Available from: http://topics.nytimes.com/top/news/business/companies/twitter/index.html.

[63] Parmanto, B., R. Ferrydiansyah, A. Saptono, L. Song, I.W. Sugiantara, and S. Hackett, *AcceSS: accessibility through simplification & summarization*, in *Proceedings of the 2005 International Cross-Disciplinary Workshop on Web Accessibility (W4A)*. 2005, ACM: Chiba, Japan. p. 18-25.

[64] Pluribo. *Pluribo: Instant Summaries*. 2008; Available from: https://addons.mozilla.org/en-US/firefox/addon/pluribo-instant-summaries/?src=search.

[65] Popovici, V., W. Chen, B.G. Gallas, C. Hatzis, W. Shi, F.W. Samuelson, Y. Nikolsky, M. Tsyganova, A. Ishkin, T. Nikolskaya, K.R. Hess, V. Valero, D. Booser, M. Delorenzi, G.N. Hortobagyi, L. Shi, W.F. Symmans, and L. Pusztai, *Effect of training-sample size and classification difficulty on the accuracy of genomic predictors.* Breast Cancer Research, 2010.

[66] Project, R. *A Power Analysis LIbrary for R*. Available from: http://www.statmethods.net/stats/power.html.

[67] Radev, D.R., E. Hovy, and K. McKeown, *Introduction to the special issue on summarization.* Computational Linguistics, 2002. **28**(4): p. 399-408.

[68] Rahman, A.F.R., H. Alam, R. Hartono, and K. Ariyoshi, *Automatic Summarization of Web ConUent to Smaller Displlay Devices*, in *IEEE*. 2001.

[69] Raman, T.V., *Emacspeak - direct speech access*, in *Proceedings of the second annual ACM conference on Assistive technologies*. 1996, ACM: Vancouver, British Columbia, Canada.

[70] RealSpeak. *RealSpeak Solo Direct Voices for Freedom Scientific Products*. 2011; Available from: http://www.freedomscientific.com/downloads/RealSpeak-Solo-Direct-Voices/RealSpeak-Solo-Direct-Downloads.asp.

[71] Rogers, Y., K. Connelly, L. Tedesco, W. Hazlewood, A. Kurtz, R.E. Hall, J. Hursey, and T. Toscos, *Why it's worth the hassle: the value of in-situ studies when designing Ubicomp*, in *Proceedings of the 9th international conference on Ubiquitous computing*. 2007, Springer-Verlag: Innsbruck, Austria. p. 336-353.

[72] S., J.P., *To parse or not to parse: relation-driven text skimming*, in *Proceedings of the 13th conference on Computational linguistics - Volume 2*. 1990, Association for Computational Linguistics: Helsinki, Finland. p. 194-198.

[73] Schiffman, B., I. Mani, and K. Concepcion, *Producing Biographical Summaries: Combining Linguistic Knowledge with Corpus Statistics*, in *39th Annual Meeting of the Association for Computation Linguistics*. 2001: New Brunswick, New Jersey: Association for Computational Linguistics. p. 450-457.

[74] Serenson. *Speed Reading Methods*. Available from: http://www.write-better-english.com/speed-reading-methods.aspx.

[75] Smith, G.M. and M.C. Schraefel, *The Radial Scroll Tool: Scrolling Support for Stylus- or Touch-Based Document Navigation*, in *User Interface Software and Technology*. 2004, ACM: Santa Fe, New Mexico.

[76] Stent, A., A. Syrdal, and T. Mishra, *On the intelligibility of fast synthesized speech for individuals with early-onset blindness*, in *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*. 2011, ACM: Dundee, Scotland, UK. p. 211-218.

[77] SuperNova, *Screen reader from Dolphin*. 2010. http://www.dolphincomputeraccess.com.

[78] Sweller, J., *Cognitive load during problem solving: Effects on learning*, in *Cognitive Science*. 1988.

[79] Team, G. *GreatSummary* 2008; Available from: https://addons.mozilla.org/en-US/firefox/addon/greatsummary/?src=ss.

[80] VoiceOver, *Screen reader from Apple*. 2010. http://www.apple.com/accessibility/voiceover.

[81] Waibel, A., M. Bett, M. Finke, and R. Stiefelhagen, *Meeting Browser: Tracking and Summarising Meetings*, in *DARPA Broadcast News Workshop*. 1998.

[82] Wharton, S.W., *An analysis of the effects of sample size on classification performance of a histogram based cluster analysis procedure, Pattern Recognition*, in *Pattern Recognition*. 1984. p. 239-244. http://www.sciencedirect.com/science/article/pii/0031320384900621.

[83] WHO. *Visual impairment and blindness*. 2011 [cited 2011]; Available from: http://www.who.int/mediacentre/factsheets/fs282/en/.

[84] Window-Eyes, *Screen Reader GW Micro*. 2010. http://www.gwmicro.com/Window-Eyes.

[85] Young, S. and P. Hayes, *Automatic classification and summarization of banking telexes* in *The Second Conference ok Artificial Intelligence Applications*. 1985, IEEE Press. p. 402-208.

[86] Zajicek, M., C. Powell, and C. Reeves, *A Web navigation tool for the blind*, in *Proceedings of the third international ACM conference on Assistive technologies*. 1998, ACM: Marina del Rey, California, United States. p. 204-206.

[87] Zechner, K., *Fast Generation of Abstracts from General Domain Text Corpora by Extracting Relevant Sentences*, in *COLING '96 Proceedings of the 16th conference on Computational linguistics - Volume 2*. 1996.

[88] Zhang, Y., *World Wide Web Site Summarization*, in *Technical report, Faculty of Computer Science*. 2002, Dalhousie University.