Universität Leipzig
Faculty of Mathematics and Computer Science
Institute of Computer Science
Degree Programme Computer Science

Determining the Quality of Archived Web Pages from Screenshot Differences

Master's Thesis

Theresa Elstner

1. Referee: Jun.-Prof. Dr. Martin Potthast
2. Referee: Dr. Andreas Niekler

Submission date: April 25, 2021

Abstract

The number of web pages in web archives grows larger, and the archived web pages are often of reduced quality compared to their original web page. At the same time, there is still no automatic quality assessment tool that could assist the detection of low-quality archived web pages making a step towards solving the various complex problems archiving processes face. This thesis explores the possibility of automatic quality assessment of archived web pages through a visual comparison of screenshots of archived and corresponding original web pages. The visual comparison of the screenshots provides a pixel error, the number of differing pixels between archived an original screenshots, which is explored in terms of its ability to indicate the archived web pages quality. This thesis contributes a categorization of visual perceivable error types which can occur on archived web pages, a general framework to develop a tool for automatic quality assessment of archived web pages, and a first implementation of this framework. In this first implementation of the framework, the archived web pages' screenshots are reconstructed by shifting translated HTML elements back to their position in the original web pages' screenshots. This reconstruction process provides a less noisy pixel error compared to the pixel error provided by the unchanged archived web pages' screenshots. The two pixel errors are then compared in their ability to predict the quality of the archived web pages according to human annotations in a linear regression model. The results show that more accurate predictions about the quality of archived web pages can be made using the pixel error resulting from the reconstructed screenshots than using the pixel error from the unprocessed archived screenshots.

Contents

List of Figures

List of Tables

Glossary

archived screenshot A screenshot of a web page that was opened from the archive.

chroma Information on the color value of a pixel in a digital image, for example represented by the Cb and Cr components in the Y'CbCr color space.

corresponding pixel Two pixels with equal coordinates in two screenshots.

description of structure A file containing information on every element in the screenshot of an archived web page: (1) the text content (if present); (2) the XPath of the element; and (3) the position of the element as top left (a,b) and bottom right (c,d) coordinates.

error types See reproduction error.

image element A HTML image element (<img>). In contrast to image and screenshot, which describe the visual representations themselves, the image element describes the HTML element which embeds some visual representation into a web page.

luma Information on the brightness of a pixel in a digital image, for example represented by the Y component in the Y'CbCr color space.

original screenshot A screenshot of an online version of a web page.

pixel error The relative or absolute number of differently colored corresponding pixels in two screenshots.

reconstructed archived screenshot See reconstructed screenshot.

reconstructed screenshot A reconstructed screenshot of the archived web page in which the occurrence of specific reproduction errors has been reduced.

reproduction error Issue on the reproduced web page, which causes it to look or behave differently than the original page at the time of archiving.

screenshot A photograph of currently visible items on a display.

snapshot A collection of data about the web archive captured at a certain point of time. It can contain screenshots of archived web pages and corresponding screenshots of the original web pages, also there can be contained various metadata about the archived web pages and their relation to the original web pages, like for example the description of structure.

subimage An isolated visual representation of a specific element as found on a web page's screenshot.

unprocessed archived screenshot See archived screenshot.

web page reproduction Showing an archived page in a browser that looks and behaves like the original page at the time of archiving [16].

Chapter 1

Introduction

In an increasingly digital world, we strongly rely on the ubiquity of data on the internet. Not only do researchers of various disciplines rely on information exclusively available online, but also our daily lives take place on the internet, leaving big parts of our digital heritage solely stored on servers. When servers that offer this digital content go offline, the information they provided is no longer available.

One of many situations where the disappearance of digital content becomes critical is lost evidence when trying to prove a crime. The article "Gone, But Not Forgotten: Evidence from the Archived Internet" [5] describes a trial in which the illegal online distribution of drugs was proven through screenshots taken from a web archive because the original web pages were not online anymore. Keeping our digital heritage accessible for future use by preserving the ephemeral content of web pages becomes more and more important.

The *Gesellschaft für Informatik* even states the conservation of our digital culture as one of the five grand challenges in computer science [15]. The most prominent way to realize this is to store the web pages in a digital archive—a web archive. Consequently, there are multiple organizations dedicated to web archiving: Some are dedicated to preserving local online publications, like Pandora[1], a web archive established by Australia's National Library, which stores Australian online publications [20]. Others dedicate their work to documenting the change of information, like The Internet Archive's Wayback Machine[2], which pursues the goal of storing different versions of web pages over time. The latter crawled 458 billion web pages since its launch in 2001, which means about 3 million web pages are added hourly to the archive [7]. It is easy to see that this vast amount of data generated by the automated archiving process can hardly, or only with a great effort, be reviewed by humans.

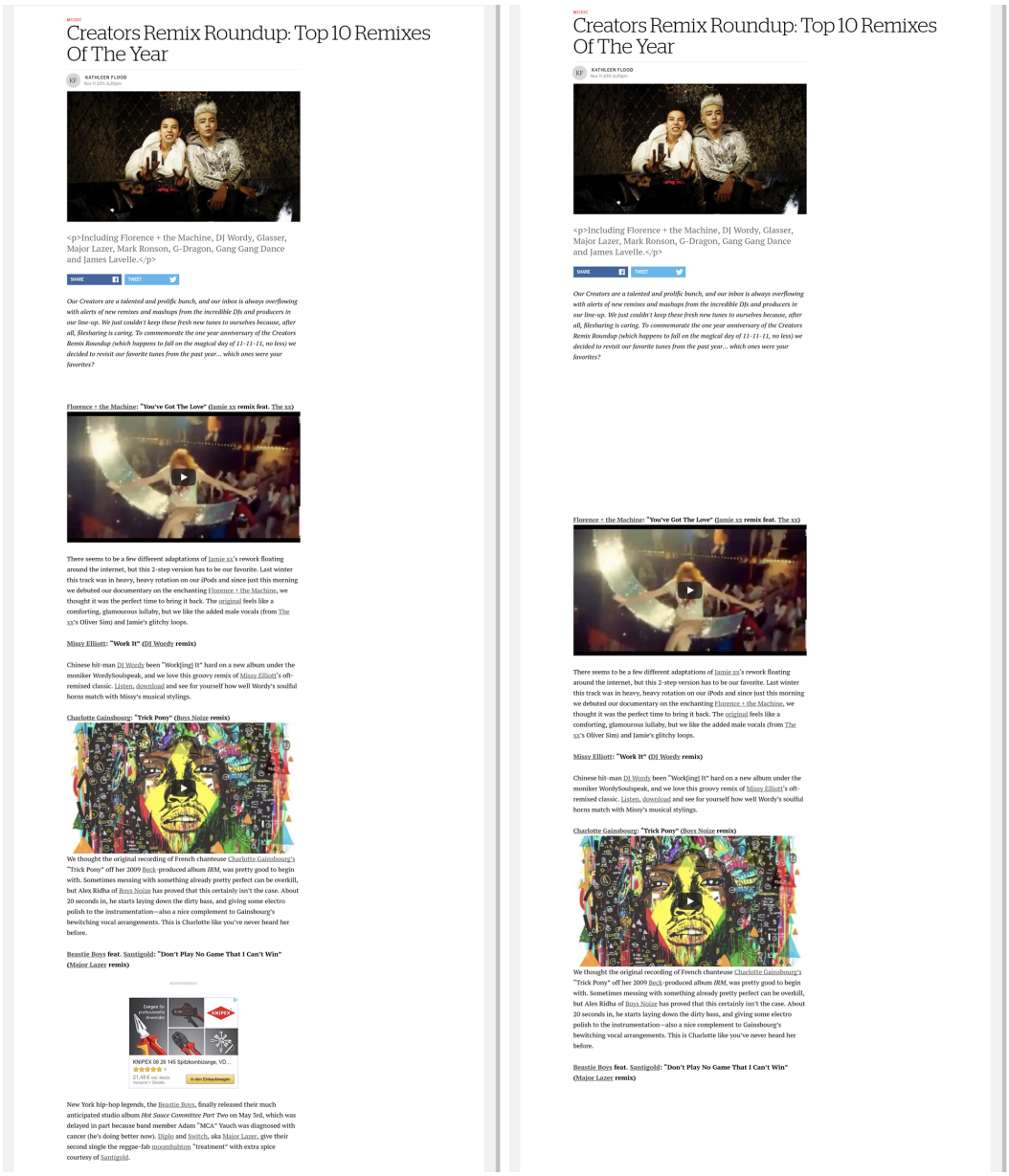
[1] `http://pandora.nla.gov.au/about.html`
[2] `https://web.archive.org`

The Problem. An archived web page can contain reproduction errors—differences compared to the original online web page—for example, missing image elements or wrong layout, which can render the archived content unusable. An example of an archived web page with reproduction errors is shown in figure 1.1. One can see that an advertisement is missing, also some articles are shifted to the bottom, and at the bottom of the page, text content is missing. The missing text content and the missing advertisement would be lost if the original web page were not available anymore, leaving the archive as the only source for this data.

In their paper "Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment" [16], Kiesel et al. state that web page reproduction in an archive "is still far from perfect," considering that human annotators assigned 11% of the archived web pages in their data set qualities of 3 and worse on a scale from 1 (best) to 5 (worst). A total of 3.6% of their archived web pages even received the worst quality score. At first sight, 11% might not sound much, but storing low-quality archived versions of web pages should still be avoided, as web archives consist of massive amounts of data: Translating the percentage of the lowest quality found in the data set by Kiesel et al. to the example of the Wayback Machine mentioned above, 3.6% would make up about 16.5 billion useless web pages with a quality of 5. This amount of potentially useless data calls for an automated quality assessment of the archived web pages, which would prevent storing useless data, and let the archive serve its purpose in providing information for the users.

Contribution. In this work, we will present an approach to improve the quality of web archives through the exploration of automatic quality assessment of archived web pages. We will build our research on the discoveries and data set provided by Kiesel et al. and presented in "Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment" [16]. We will present their discoveries about automatically assessing the quality of archived web pages using visual differences in screenshots in more detail in chapter 2.

Throughout this thesis, we will often use the term *quality of an archived web page*. However, we explicitly will not define the word *quality* in the context of web pages because the perceived quality can vary from user to user depending on their interest or information need. Instead, in this work, we use an indirect understanding of quality. We assume the quality is indicated by visual differences between screenshots of pairs of archived and corresponding original web pages. These differences will be measured and correlated with a quality score assessed by human annotators, who based their judgment on how *useful* an archived web page appeared to them. We assume these anno-



(a) **(b)**

Figure 1.1: Two screenshots of the same web page. (a) shows a screenshot of the original web page and (b) shows a screenshot of the archived web page, taken from the web page loaded from the web archive. The archived page appears different than the original page: some articles are shifted and an element is missing.

tations reflect the opinion of most users of an archive. With this in mind, our procedure will include the following subjects:

Measure and Categorize Reproduction Errors
: Visually perceivable reproduction errors generated on archived web pages will be measured using an adaptation of the edit distance. These visually perceivable reproduction errors will be listed and categorized to facilitate an overview of the bigger picture of the problems of automatic quality assessment of archived web pages. The reproduction errors considered in this work will not include animated or interactive elements like media players, buttons, or support chats because they are not detectable on screenshots of web pages.

Present a Framework
: We will present a framework for assessing an archived web page's quality. This framework includes a comparison between screenshots of original and archived versions of web pages. We suggest reducing the negative influence of reproduction errors on determining the quality from differences in the screenshots of archived pages by computing a reconstructed version of the archived web page which is as similar as possible to the original. We use the data gained through the reconstruction process as an indicator for the quality of the archived web pages. Lastly, we suggest designing a statistical model to assess the quality of the archived web pages by correlating the data from the reconstruction process with human annotations about the archived web pages' qualities.

Demonstrate an Implementation of the Framework
: We will demonstrate an implementation of the framework, which explores an approach to reduce the occurrence of a specific reproduction error. With regard to this specific reproduction error, we will explore and evaluate different possibilities of reconstructing the archived screenshot. Then we will use the data from the best reconstruction to design a linear regression model which predicts the quality of the archived web pages according to human annotations.

Evaluate the Implementation
: We will evaluate the implementation concerning its effectiveness and potential.

Chapter 2

Related Work

This chapter discusses research related to this thesis. First, the difficulty of archiving web pages will be elaborated on by explaining Ilya Kreymer's *encapsulation complexity* [18]. Then we will present various works in the broader field of quality assessment of web archives, discussing closely related work by Kiesel et al. [16] about automatic quality assessment of archived web pages and other research using a visual similarity metric to define the quality, but also elaborate on approaches with non-visual quality measures. Lastly, we will discuss research by Cormier et al. [14], which is not concerned with quality assessment but connects to this thesis' subject in terms of a visual approach to web page analysis.

2.1 On the Difficulty of Archiving Web Pages

Some web pages are more difficult to archive than others. An important reason for the general difficulty of archiving web pages is that most often, web pages are not served by a single server but also integrate data from other servers. The *2020 Web Almanac* by the HTTP Archive states that "At the median, pages make 20 JavaScript requests" [6], which puts a number on how often external content is embedded in a web page. Among other factors, this dynamic content renders web pages increasingly difficult to archive because the more resources are to be stored, the more likely it is to fail.

Ilya Kreymer, the Lead Software Engineer of the Webrecorder, suggests a methodology on different levels of archiving difficulty [18]. He describes four levels of web objects with increasing *encapsulation complexity*. By encapsulation, Kreymer describes the act of preserving a web object within limited boundaries, like files or containers. The least complex level to archive—level 0—contains web pages which only consist of one page without any ex-

ternal dependencies. These web pages can be archived using the browser's built-in function "Save Page As."

Level 1 web objects are all level 0 web objects plus web objects which load resources from a finite number of URLs. Kreymer states that they can be fully "encapsulated as a web archive in a single WARC or WACZ format, and can load directly in a browser" [18]. The finite number of URLs in level 1 web objects allows for an archiving strategy like web crawling because the number of web objects to be crawled is limited. Therefore a crawling process would be completed after some time.

This limitation does not apply to the next level: Level 2 web objects are all level 1 web objects plus those that can make dynamic URL requests to one or more fixed web servers and have a WARC/WARZ based web archive. For web objects of this level, the fixed web server must be fully functional under encapsulation. A crawling process would not be sufficient to encapsulate these web objects because an infinite number of URLs could be called. This would result in an infinite or incomplete crawling process. Instead of crawling, Kreymer suggests archiving web objects of level 2 using orchestrated web server containers combined with web archives or web server emulation.

The last level, called level 3+, contains web objects which are not possible to fully encapsulate. To this level, Kreymer adds web objects with dependencies (for example, databases) that either cannot be enumerated in a preservation system or have an unknown number of external dependencies. As an example for this level, he names web objects which make search requests on Google. He argues this requires dynamic requests to servers that are outside the control of the preservation system and, therefore, impossible to encapsulate at full fidelity. Kreymer has named possible tools which would provide the functionality to encapsulate the respective web objects, except for level 3+, for which he suggests a migration to lower levels before preserving them with a lower level tool.

Kreymer's analysis of encapsulation complexity provides a common ground and vocabulary for the field of web archiving, which could introduce more structure to future research on web archiving in general but also on automation of quality assessment. The data set for this thesis was crawled before the existence of this methodology. Hence, web pages of several levels might be included in the data set. Quality assessment research on web archives could profit from this methodology insofar as the set of possible errors varies in each level of encapsulation complexity: For example, level 0 web object cannot contain external sources; therefore errors like missing elements or CSS files could not occur and would therefore not have to be considered in quality assessment. In terms of this thesis, Kreymer's definition of encapsulation complexity could additionally be used to compute the quality of the archived web pages sep-

arately for each level. Hence it would contribute to debugging the archiving process itself: For example, if a web page of level 0 were not archived correctly, it would be clear that the archiving process was generally flawed.

2.2 On Quality Assessment of Web Archives

While there is a considerable amount of research that deals with the broader field of quality of web archives, there are only a limited number of approaches that consider an automatic quality assessment of archived web pages and even fewer that take into account visual approaches. The state of the art quality assessment presented on the support page of the Internet Archive's web archiving service Archive-It transfers the responsibility of reviewing the quality of an archived web page to the user of the archiving software. While also providing crawl reports which detail metadata on crawled hosts or file types, among other things, they suggest a manual quality assessment by stating: "Browse through your archived site(s), clicking links and activating dynamic media players in order to make sure that they were archived in accordance with your expectations." [11]

This thesis builds on the research done by Kiesel et al. and presented in the paper "Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment" [16]. Kiesel et al. contribute a data set with 10 000 archived web pages with annotations about the quality and offer a comparison between automatically assigned qualities for these web pages and the human annotations. The data set is gained through their archiving tool, and the human annotations of the archived web pages' qualities are crowd-sourced based on the data set. In this thesis, we use the data set presented by Kiesel et al. to further their research on automatic quality assessment. To automatically derive the quality of the archived web pages, Kiesel et al. tested three quality assessment strategies, one of which was outperformed by the other two: The method which was outperformed is a method proposed by Brunelle et al. in "Not All Mementos Are Created Equal: Measuring The Impact Of Missing Resources" [13] to measure the quality of an archived web page with the help of its missing resources' differing impacts. They use the notion that a smaller missing element has a less negative impact on the quality of a web page than a larger missing element (a notion which we also implicitly used in this thesis by measuring the quality through the number of differing pixels). Kiesel et al. suspect the cause for the worse performance of Brunelle et al.'s approach to be that it does not take any resources apart from the archived web page into account and, therefore, does not consider how the web page looked when

all resources were present[1]. The other two methods tested by Kiesel et al. were the following: They computed the correlation to the human annotations and the differences in the screenshot of an archived and corresponding original web page (1) using the *root-mean-square error* (RMSE) and (2) using a deep convolutional neural network. The much simpler—hence more intuitive—RMSE provided an r-value of 0.48. The deep convolutional neural network did not perform much better, considering its complexity, with an r-value of 0.57. Based on the result from Kiesel et al., the question arises if the correlation between the RMSE and the human annotations could be strengthened. Kiesel et al. state that the RMSE is not accurate, for example, because "(...) a single missing advertisement banner at the top of the page can shift up the entire web page, causing the RMSE to become unjustifiably high" [16]. This thesis explores the possibility of strengthening the correlation between the RMSE and the human annotations through reconstructing the archived screenshots by re-positioning shifted elements to their corresponding positions in the original web page's screenshot in order to decrease the differences between the archived and original screenshots.

Also related to this thesis and to the work of Kiesel et al. is the paper "Using Image Similarity Metrics to Measure Visual Quality in Web Archives" by Reyes Ayala, Hitchcock, and Sun [10]. Here, Reyes Ayala et al. created a data set of archived web pages' screenshots and original web pages' screenshots to measure their similarity. This visual approach and the inherent understanding of quality of an archived web page is very similar to the notion of quality used in this paper. Both this thesis and Reyes Ayala's work define the quality of an archived web page through its similarity compared to its original version based on a screenshot. Reyes Ayala et al. compare three different image similarity measures with concerning their ability to reflect the differences between pairs of screenshots of archived and original web pages: The Mean Squared Error (MSE), which measures the difference of color values of pixels without taking the position of the pixels into account, Structural Similarity Index (SSIM) which also compares color values of pixels but takes the position of the pixels into account and a distance measure by Reyes Ayala et al. called Vector distance which interprets the screenshots as vectors of pixels represented by their color values and then computes the difference between the archived screenshot's vector and its corresponding original screenshot's vector and reflects this difference as a percentage. They find that MSE and Vector distance are almost equivalent in their behavior but recommend using Vector distance instead of MSE because it is limited to a scale from 0 to 100. At the same time, the MSE does not have an upper limit and is, therefore, more challenging to

[1]Justin Brunelle stated in a blog post [12] that he agrees with this interpretation.

interpret. In this work, we used a similarity measure which is comparable to MSE, measuring the number of different pixels between archived and original screenshots on an absolute and on a relative scale because it was very intuitive to interpret on our data set. However, as Reyes Ayala et al. suggest, all three of their inspected similarity measures would be fit to use on screenshots of archived web pages and their corresponding original web pages' screenshots. It would be interesting to compare the performance of SSIM, MSE, or Vector distance concerning their correlation on the quality of archived web pages a human reader would assign. Apart from their inspection of similarity measures, Reyes Ayala et al. point out one problem which occurred while generating their data set of archived and original web pages screenshot pairs: They found that many archived pages used a banner to inform the users that they are visiting an archived web page, which rendered the screenshots less accurate. Assuming that here "less accurate" means that the screenshots of the archived web pages—if taken including the banner—were shifted compared to the original web pages' screenshots, the work of this thesis directly tackles this problem by shifting elements in the archived screenshot back to their original position before the visual comparison is executed.

In 2020, Reyes Ayala published the paper "Correspondence as the Primary Measure of Quality for Web Archives: A Grounded Theory Study" [9], where she analyzed support tickets for the Internet Archive's Archive-It service, in which clients state quality problems on archived web pages. She finds that "the degree of similarity or resemblance between the original website and the archived website" is "the most important facet of quality in web archives" [9]. Similarly, this thesis and Kiesel et al. rely on the similarity between archived and original web page as a quality indicator and measure the quality of an archived web page's screenshot by the degree of difference compared to the corresponding original web page's screenshot. Reyes Ayala further points out that the notion of quality in web archiving is interpreted differently by various researchers. Therefore she sees her work as a contribution to a unanimous understanding of the term.

Reyes Ayala's previous work was committed to the quality of web archives as well. In "Current Quality Assurance Practices in Web Archiving" [8], she and the co-authors survey quality assurance practices in web archiving to make a step towards finding a standard best practice. The authors find out that the quality assurance of web archives is mostly conducted manually and explain the need for an automated process, stating the manual procedure was time-consuming. The survey of Reyes Ayala et al. points out that there is a practical application in the archiving community to which this thesis wants to contribute.

A definition of quality, which is not based on the visual similarity between the archived and original web page, is used by Marc Spaniol et al. [26]. Instead of similarity, Spaniol et al. [26] use the metric of coherence of archived resources within web sites to analyze the quality of archived web sites. Their paper "'Catch me if you can': Visual Analysis of Coherence Defects in Web Archiving" introduces this time-based approach to quality: Coherence here means that a web site changed during the crawl so that resources on the site differ in topicality resulting in decreased quality of the web site. Spaniol et al. detected the incoherence through an analysis of change of web sites' resources between two crawls, using metadata like last-modified dates. Both—the work of Spaniol et al. and the research presented in this thesis—focus on detecting improperly archived web pages, but they concentrate on different aspects of quality. In contrast to Spaniol et al.'s approach, our work focuses on comparing a representation of an online page and an archived page, while the paper of Spaniol et al. compares two versions of an archived web page within one site. Unlike Spaniol et al., we do not consider entire web *sites* but web *pages*; this implies that the entire field of automatic quality assessment is much broader with more aspects of quality than we consider in this work. Moreover, we aim to detect any quality reduction which is visually perceivable on the appearance of the web page, while Spaniol et al. concentrate on detecting time-based incoherences which are derived from metadata. Further, the visual aspect of the work presented in this thesis and the work presented by Spaniol et al. do not share a common goal. Our visual approach builds on a comparison of pixels within screenshots of original and archived web pages to detect bad quality archived web pages. In contrast, the visual exploration presented by Spaniol et al. aims at visualizing bad quality archived web pages (or resources). The visual aspect of Spaniol et al.'s work does not contribute to the automatic quality assessment itself but instead provides a visual representation of automatically detected incoherences, which assist in the easier manual exploration of the quality of the archived sites in terms of temporal discrepancies.

The paper "Improving the Quality of Web Archives through the Importance of Changes" by Saad and Gançarski [23] is likewise based on the concept of coherence as an understanding of quality of web archives. Nevertheless, the authors additionally consider the metric of completeness which measures if web pages are missing in the archive. This understanding extends the research of Spaniol et al. but does not contribute to the work of this thesis because it does not share the same understanding of quality. Nevertheless, it is an example of how different aspects of quality might have to be considered to develop an automatic quality assessment tool.

2.3 Visual Analysis of Web Pages

In this thesis, we use motion detection to find translated elements in the archived web page. We expect that exploring the capabilities of web page segmentation with regard to finding translated elements bears promising results. A work that is not dedicated to improving upon the quality of web archives but provides an interesting approach to the field of web page segmentation is presented by Cormier et al. in their paper "Purely vision-based segmentation of web pages for assistive technology" [14]. The authors analyze the structure of web pages' elements only based on visual information. Their research aims at dividing parts of web pages into contextual segments in order to provide information valuable for rendering versions of web pages into alternate presentations more suitable for a diverse user group. The 2021 study "An Empirical Comparison of Web Page Segmentation Algorithms" by Kiesel et al. [17] compares multiple segmentation algorithms and finds out that the purely visual approach of Cormier et al. performed second best after another method called VIPS. VIPS would not be applicable to the data set used in this thesis because it is not based on screenshots of web pages but on their DOMs; Cormier et al.'s approach, on the other hand, is screenshot-based. Hence, this novel approach by Cormier et al. of segmenting web pages using screenshots could be used to compare archived and original screenshots concerning their different segmentations to find out if they indicated a change in quality of the archived web pages. This method would be interesting to compare to the results of the automatic quality assessment obtained during this thesis through reconstruction of archived screenshots using the H.264 video codec.

Chapter 3

Web Page Reproduction Errors

In this work, we explore an approach of automatically assessing the quality of an archived web page from the differences between screenshots of an archived and its corresponding original web page. In this chapter, we will present a measure of difference for the two screenshots to approximate the quality of an archived web page and discuss the shortcomings of this quality measure. In the last parts of this chapter, we will categorize web page reproduction errors—error types that occur on archived web pages—and elaborate on the theoretical limits to detecting all possible reproduction error types.

3.1 Computing the Pixel Error

In this section, we will describe a measure of difference for an archived and an original web pages' screenshot. We will explain and adapt the *edit distance* defined by Levenshtein [19] as a distance measure for our purposes.

A Distance Measure: The Edit Distance. We suggest an adaptation of the *edit distance* defined by Levenshtein [19] to measure the error as a pixel error. This distance measure originally describes the distance between two strings over the alphabet {0,1} through the minimal number of operations from the set of {`insertion, deletion, reversal`} to transform one string into the other [19]. In our case, we would apply these same operations not on strings over the alphabet {0,1} but on matrices which we interpret as two-dimensional strings over the alphabet of an RGB color space {(0-255,0-255,0-255)}. Another change we must introduce, due to the change of alphabets, is to modify the operation `reversal` to become the operation `substitution`.[1] Originally, this operation allowed to exchange one sign (e.g., 0) for the other

[1] In fact, `substitution` and `reversal` are idempotent under any binary alphabet.

in a binary alphabet; for our purposes, we would need this operation to allow the substitution of any color for another one, so we define:

Substitution changes a single RGB color value $x = (r_1, g_1, b_1)$ to another color value of the same color space $y = (r_2, g_2, b_2)$, so that $(r_1 \neq r_2) \vee (g_1 \neq g_2) \vee (b_1 \neq b_2)$.

Furthermore, we define that the operation `insertion`, other than its original usage, cannot insert a single value (which in our case describes one RGB colored pixel). Instead, it only operates on a per row basis of a two-dimensional matrix that contains RGB color values in every cell—a representation of the screenshot of a web page. As the width of the screenshots of the archived and the original web pages are fixed to the browser's window size, archived and original screenshot will always have the same width, so the operation `insertion` will not be applied to the columns of our color matrices. Nevertheless, archived and original screenshots could have different heights because in the archiving process, elements could vanish or appear, which can make a web page longer and increase or decrease the height of the corresponding screenshot as a result. Therefore we define the operation `insertion` as follows:

Insertion adds a single row of n RGB color values $x = (r, g, b)$ to an $m \times n$ matrix of RGB color values. Here, m defines the height of the larger screenshot in pixels and n the number of columns, hence the width of the screenshots in pixels.

This operation will be used for the process of inserting rows of uniformly colored pixels to the shorter screenshot to be able to better compare the elements in both screenshots. The last operator in use is the `deletion` operator that we describe as:

Deletion removes a single row of n RGB color values $x = (r, g, b)$ of an $m \times n$ matrix of RGB color values. Here, m defines the height of the larger screenshot in pixels and n the number of columns, hence the width of the screenshots in pixels.

It is easy to see that the `deletion` operator is the counterpart of the `insertion` operator and that it would result in reducing the bigger screenshot's height.

For any operation that removes or adds an element within the bounds of both screenshots (archived and original), we would always use the `substitution` operator because it does not modify the size of the screenshot.

The Pixel Error: A Quality Measure. The operations `substitution`, `insertion`, and `deletion` allow us to transform a given screenshot to another

screenshot according to any reference picture. We will use the minimal number of pixels changed through `substitution`, `insertion`, and `deletion` in order to generate a perfect replica of the original from the archived screenshot and to measure the difference between the two screenshots; we call this difference *pixel error*.

Absence of the Pixel Error: If a pixel-by-pixel comparison of the archived and the original screenshot of a web page results in the same color for every position, we define the pixel error to be absent, hence zero.

Presence of the Pixel Error: If a pixel-by-pixel comparison of the archived and the original screenshot of a web page does not result in the same color for at least one pixel pair, we define the pixel error to be present. Additionally, we define the pixel error to be one pixel for every pixel that differs in both screenshots. This includes pixel differences that could be fixed through `substitution` as well as errors that could be fixed through `insertion` or `deletion`. The total pixel error will be the count of all differing pixels.

Instead of the number of differing pixels, the root-mean-square error (RMSE) could be used to measure the differences between the screenshots. Nevertheless, we decided to use the number of differing pixels because it is more intuitive and easier to interpret than the RMSE.

Reflections on the Pixel Error. There are several details that diminish the meaningfulness of the suggested pixel error as quality measure. First of all, our distance measure does not differentiate between a large color difference and a small color difference of two corresponding pixels in the archived and original screenshot. It adds any pixel in the archived screenshot which differs from the original to the sum that composes the error. For example, we observed an element in the archived screenshot having an RGB color value of RGB(240, 240, 240) while the original screenshot displayed it as RGB(241, 241, 241). This difference cannot be perceived by the human eye. This gave us reason to assume that the color difference between corresponding pixels of the archived and the original web pages could have a varying negative influence on the pixel error to indicate the archived web page's quality. We assumed that a small color difference on the same element might not have the same destructive effect as a big color difference. However, there are cases in which the assumption of a large difference being worse than a small difference does not hold: One could imagine an error on the archived web page's screenshot that displayed a black and white image with inverted colors compared to the same image on the original web page's screenshot. The image would still be possible to

interpret, but the color difference would be at its maximum for every pixel as any RGB(0,0,0) would have been turned to RGB(255,255,255) and vice versa. This gave us reason to believe that the assumption about the influence of the color error would need to be evaluated before implementing it into the pixel error.

Another issue with the pixel error is that distances of elements that are shifted in the archived web page's screenshot compared to their position in the original do not only produce an error their own size but also the shift distance is not reflected in the pixel error. That is, elements that shifted only by one pixel would produce an equally large pixel error as elements that have shifted very far away. One could argue that larger shift distances might indicate a worse quality of the archived web page than small shift distances. For example, if an element is shifted very far to the bottom of the archived web page, it would be hidden for any user of the page who does not scroll down. On the other hand, there are cases in which a large shift does not necessarily imply a bad quality of an archived element: If a very long advertisement is left out during the archiving process and all elements below it move up correspondingly, the non-advertisement content of the original web page would still have been transported to the archive without loss, and the quality of the archive would not have taken damage. It would be necessary to evaluate both of these assumptions before adapting the distance measure accordingly. Unfortunately, this falls out of the scope of this thesis.

An issue of the pixel error that can cause inconsistent results is the occurrence of corresponding pixels that coincidentally have the same color in the archived and original screenshot, but belong to different elements. Those pixels do not increase the pixel error, even if they should. It is difficult to estimate how often this actually occurs because it depends on the context of the affected areas in the screenshots, like the diversity of colors. Therefore, this issue will not be discussed in this thesis.

Despite the flaws of the pixel error, we found it sufficiently reliable for a first analysis of the differences. For this work, we decided to use the pixel-by-pixel comparison without modifications and consider its output with care rather than introducing new unnoticed flaws through the implementation of insufficiently checked assumptions.

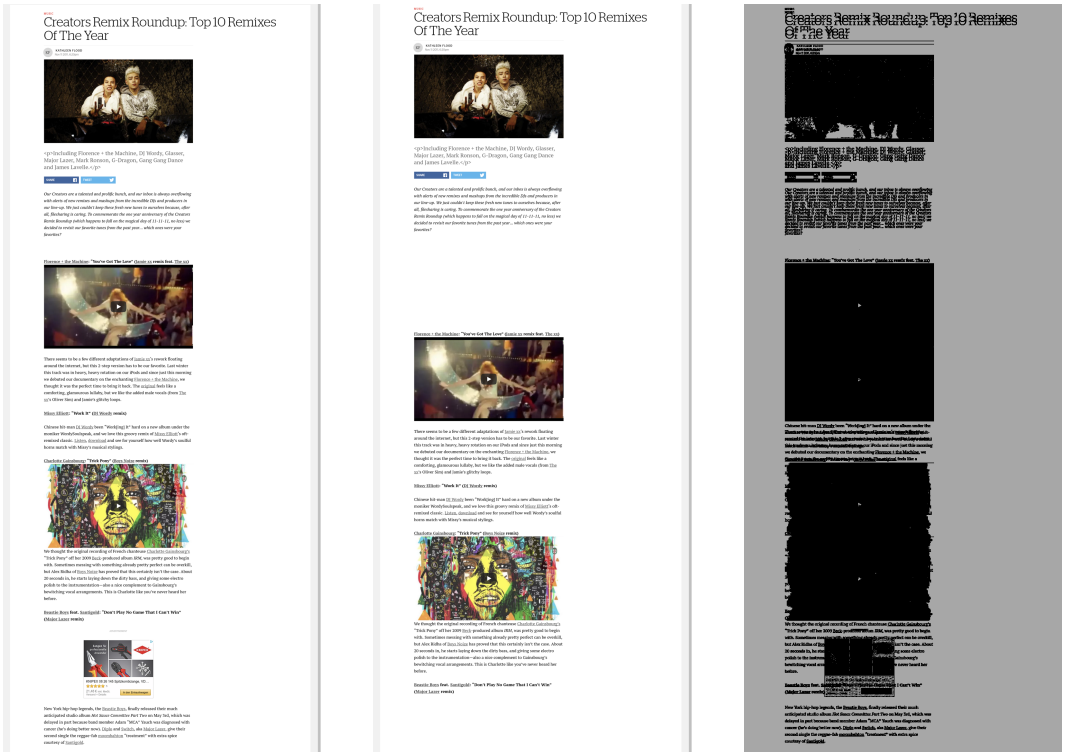
3.2 The Shortcomings of the Pixel Error

We show in this section that simple methods of judging the quality through the pixel error between the screenshot of the archived web page and the original web page are conceptually flawed, inducing the need for a more sophisticated

approach to quality assessment. For a big part, this flaw can be explained by noise introduced to the pixel error, especially through translated elements on the archived web page: They increase the pixel error, but usually they affect the quality that a human reader would assign much less.

The example, illustrated in figure 3.1, shows that the differing pixels between the screenshot of the archived web page and the screenshot of the original web page do not suffice as an indicator for the quality of the archived web page. The image on the right (figure 3.1c) displays the differences between the original (figure 3.1a) and the archived (figure 3.1b) web page's screenshot. Black areas mark differently colored pixels, and grey areas mark equally colored pixels between both screenshots. The differing pixels make up 32% of the right image. This could suggest that about 32% of the archived web page is unusable. Conversely, a human reader might assign the archived web page a very good quality because the archived version of the web page displays all content from the original web page. Only an advertisement element is missing, but the typical reader might not be interested in that, and therefore would not count this error when judging the quality of the page. At the same time, this missing ad element causes a pixel error of its own size, decreasing the quality indicated by the pixel error. In this example, the high number of differing pixels is mainly generated by shifted content, which causes the number of differing pixels to be at least the size of the shifted element.

These examples support the hypothesis that the pixel error is not a perfect indicator for the quality a human would assign because it is noisy. This assumption is based on the findings of Kiesel et al. [16]: In their paper, the authors present the relation of their data set containing archived and original web pages' screenshots and human annotations, rating the qualities of the archived web pages. They found that the normalized RMSE of the screenshots of the archived web pages and their corresponding original web pages produced a correlation of $r = 0.48$ with the quality the human readers assigned. The computation of the RMSE in Kiesel et al. "uses the screenshots taken during archiving and reproduction, computes the squared color difference for each pair of corresponding pixels, and then uses the root of the mean," [16] which is different from the pixel error which we focused on, but still based on the same differences between the screenshots. This means that we might get a different r-value from the results of our pixel-by-pixel comparison, but we would expect it to be within the same range. Of course, an r-value of $r = 0.48$ is not bad, but it could be better if errors (like the translation error) that increase the pixel error without lessening the quality of an archived web page were reduced. Removing the noise from the pixel error could make this measure a better indicator for quality. This could result in an improved tool for automatic quality assessment of archived web pages. In this thesis, we present our



(a) (b) (c)

Figure 3.1: Original (a) and archived (b) screenshot for one example web page and the pixels that are different between those two screenshots (c; in black). About 32% of pixels are different, which implies a worse quality of the archived web page than a human might assign to it.

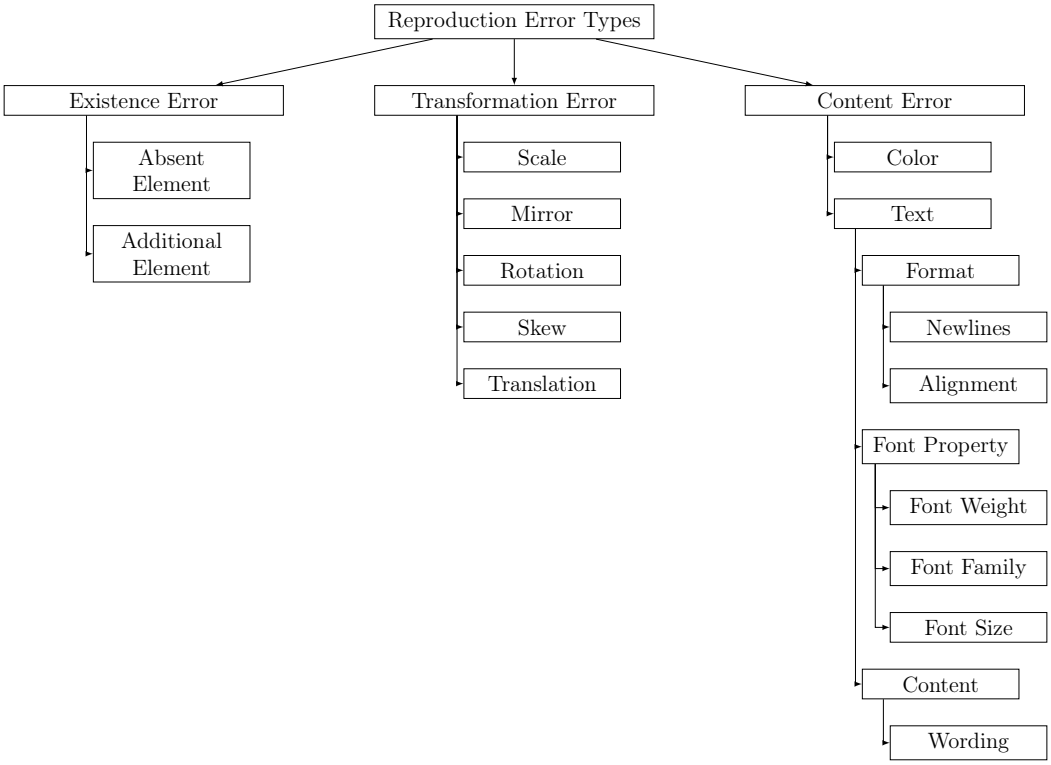
approach to developing such a tool by reconstructing the archived web page's screenshot with reduced errors in order to obtain a less noisy pixel error which better indicates the quality of the archived page.

3.3 Reproduction Error Types in Web Archives

In the last section, we have introduced the idea and motivation behind reducing the noise in the pixel error through reconstructing the archived web pages screenshot in order to obtain a better indicator for the quality of the archived web page. In this section, we want to explore how this can be done. Our approach of reducing reproduction errors on archived web pages that introduce noise to the pixel error requires knowledge about different error types that can occur on archived web page's screenshots that the pixel error cannot provide. In this section we will therefore list and categorize reproduction error types

that can be observed when comparing the archived and the original web pages' screenshots. Additionally, we will shortly elaborate on the influence of some of these error types on the quality of the archived web page.

A List of Reproduction Error Types. The following diagram depicts all reproduction error types that can be found by comparing two screenshots of an original web page and its archived version. Even though sometimes the original web page is not displayed correctly, we always consider the element in the archived version as the element containing the reproduction error and the original version as the point of reference. We assume, the original web page's elements have their correct and intended appearance. To give an example, the error type `additional element` would describe the difference between the screenshot of the original and archived element, where the archived version contains an additional element that is not contained in the original version.



For all error types two variants can be specified:

1. The error type occurs for the entire element.
2. The error type occurs for a part of the element.

Determining which of these variants is present could be crucial for the detection of corresponding elements in the archived and original web page's screenshots, as a pixel-by-pixel comparison of the elements would need this information to determine whether two elements that differ in one part could still be the same element or if they were not corresponding elements in the first place.

Influence of the Reproduction Errors on the Quality of an Archived Web Page. The above list of error types is organized by purely visual cues. Some of the above error types can cause visual differences between an archived and original web page's screenshot, while at the same time, they do not lessen the quality of the archived web page.

An example of such an error type is the `translation` error. If an element on an archived web page were merely translated (without additionally triggering other error types), the content of the web page would still be present, but the pixel-wise difference between archived and original screenshot would yield different results for all affected pixels.

In contrast to the `translation` error, some error types are more likely to lessen the quality of an archived web page, for example, the `absent element` error: This error type can cause important content to be lost on the archived web page. Of course, depending on the interest of the user of the archive, some content is more important than other content. Intuitively, lost advertisements are often considered irrelevant to the quality of the archived web page. However, articles on a news page would typically count as important and their absence reduces the quality of an archived web page. Conversely, depending on the interest of the reader of the page, even a missing advertisement can mean a reduced quality.

Other error types in the list can cause a degraded appearance of elements on a web page, for example, `skew` or `scale`. Their influence on the quality of the affected web page, on the one hand, depends on the degree of `skew` or `scale`, but also on the importance of the affected element with respect to the content of the archived web page. Again, the importance of the element's content is subjective to the interest of the user.

Similarly, automatic processing methods applied to the archive can be negatively impacted by specific archiving error types. For example, shifted or missing elements make layout analyses impossible, and missing text prevents successful training of language models. Additionally to these contemporary use cases of the archive, future uses of the archive data cannot be foreseen and as such, the influence of specific reproduction error types on the archive's usefulness is impossible to predict.

In this thesis, we follow the notion that the visual similarity of an archived web page's screenshot and its corresponding original web page's screenshot is

related to the quality of the archived page. We do not inspect the content of individual elements, and we do not take into account the individual interests of users.

Errors of Interactivity. The above classification of error types is missing some types that cannot be detected with our means at hand, consisting of one screenshot of each—archived and original web page—and the description of structure, including the positions of the elements in the archived version of the web page. Those error types concern audio, animated, or functional elements that are to be triggered through interaction, for example, broken links or missing audio or video in media players. The reason for the absence of these kinds of reproduction errors from the list above lies within the impossibility of capturing them in a screenshot: For example, clicking a link to another web page could not be captured with one screenshot depicting the web page.

Combinations of Errors Types. Within one element, most of these error types can occur in combination with each other. Those that cannot be combined with any other error type are:

1. Absent element (entire element is missing) (**A**)
2. Additional element (entire element is added) (**B**)

Another special case is the equivalence of certain combinations of errors: For example, a `translation` error (**C**) would be equivalent to the combination of a `rotation` error around two different axes; the element would have to be rotated around an external axis to affect the position and rotated around its own axis to affect the orientation.

Even without counting these special cases as distinct error types, there would still be a large number of combinations to examine for finding out all equivalences and impossible combinations among the error types. The following equation displays an example computation of how many combinations of error types would have to be examined closely in order to filter only those that do not display equivalences to and impossible combinations of other error types. In this example computation, the most obvious cases of equivalence (**C**) and impossible combinations (**A** and **B**) have already been taken into consideration.

Example Computation.

$$\begin{aligned}
&\mathcal{P}(\{\text{Error Types}\}) \\
&= 2^{28} > \mathcal{P}(\{\text{Error Types}\} \setminus \{\mathbf{A}\} \setminus \{\mathbf{B}\}) \cup \{\mathbf{A}\} \cup \{\mathbf{B}\} \\
&= 2^{26} + 2 > \mathcal{P}(\{\text{Error Types}\} \setminus \{\mathbf{A}\} \setminus \{\mathbf{B}\} \setminus \{\mathbf{C}\}) \cup \{\mathbf{A}\} \cup \{\mathbf{B}\} \\
&= 2^{25} + 2 = 33554434 \geq \text{number of possible combinations of error types,}
\end{aligned}$$

where $\mathcal{P}(\{\text{Error Types}\})$ denotes the power set over all error types listed above.

The large number of possibilities of combinations of error types that could occur within one element makes it difficult to analyze all of them separately. The analysis of all possible combinations, nevertheless, can be crucial to the process of detecting all errors and inferring the quality of the archived web page. Another difficulty for an automatic quality assessment tool is posed by the fact that certain combinations of error types could generate any element. This will be discussed in the following section.

3.4 The Impossibility of Perfect Detection

We have previously listed the reproduction error types which can be found on an archived web page's screenshot and will now answer the question if all of those error types can be found on a screenshot. We will describe the power of specific errors to generate any new element and the implications this has on the detection of errors, the coordination of correspondences, and the ability to reconstruct the original web pages screenshot from its corresponding archived version.

The Error Categorization Problem. Being able to always correctly categorize reproduction errors implies that a faulty element in the archived version of a web page would be detectable for all reproduction errors and combinations of them. As some of the reproduction errors are as powerful as to generate any element, the appearance of elements becomes ambiguous. This can be illustrated as follows:

The `color` error is an example of an error type that could produce an element that looks like any element on the original web page.[2] As illustrated in figure 3.2, it could easily be mistaken for the `absent element` error if the element in the archived version and the element in the original version of

[2]Of course also other errors could generate visual duplicates of elements from the original web page, for example a `scale up` error that is applied on a smaller element that has a bigger twin in the original web page.

the web page did not look similar enough. That is, the error type for this element cannot be categorized unambiguously. Obviously, this issue of not always being able to categorize error types unambiguously can lead to false categorizations. This can have severe implications on a process that tries to reconstruct an archived web page's screenshot so that it is as similar as possible to the original web page's screenshot. If a reconstruction process categorized an `absent element` error instead of the `color` error, the solution to fix this error would be to insert the missing element into the screenshot. This could result in a duplicate of the element affected by the color error and, as a result, produce a faulty reconstruction.

The categorization problem can even affect the detection of errors up to the point that an error cannot be detected at all. Figure 3.3 shows two web pages with seemingly the same element at the same position. A pixel-wise comparison of the two screenshots would falsely result in zero difference, if the element in the archived screenshot were an `additional element` and the true corresponding element to the original version were missing on the archived page. If the pixel-wise difference were used as an indicator of quality, this pixel error of zero would falsely result in the best possible quality. The implications of this on a web archive are obvious: An archived web page of seemingly perfect quality would be stored, but it could be useless to a user of the archive. As this error cannot be detected by purely visual means, in this work, we will assume that two visually identical elements with the same position are the same element.

The Correspondence Detection Problem. The difficulty of always correctly detecting correspondences between the archived and original web pages' elements will be described as an example which additionally is illustrated in figure 3.4. We suggest considering two elements in the original screenshot: One element is at the top of the original screenshot and is a visual duplicate of the lower half of the bigger element, located at the bottom of the original screenshot. In this example, the archived version contains only one element which looks exactly like the part both elements in the original screenshot have in common and which is located at the bottom of the archived screenshot. What would be the correct decision about which element of the two in the original is present in the archived version? One could argue, due to proximity to the matching parts, the bigger element of the original screenshot was the corresponding element to the archived version, but the archived version displayed a `missing element` error for part of the element. Simultaneously, a different valid argument could be made for the corresponding element being the other one, which found on top of the original screenshot: Both elements look exactly the same and nothing was missing in the archived version, only

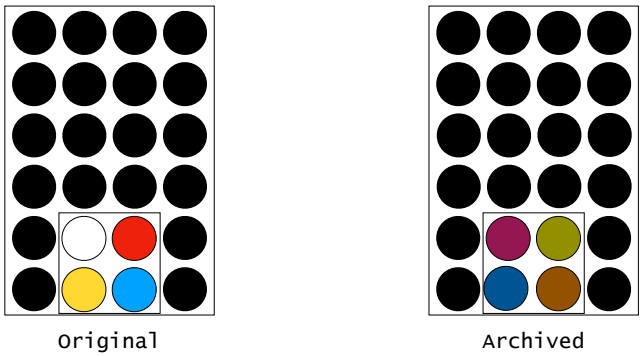


Figure 3.2: The left figure represents the original web page's screenshot containing one element. The right figure represents the archived web page's screenshot containing the same element with an `color` error for multiple parts of the element. This results in an `absent element` error for the entire element, as no correspondence was found. The errors cannot be unambiguously categorized.

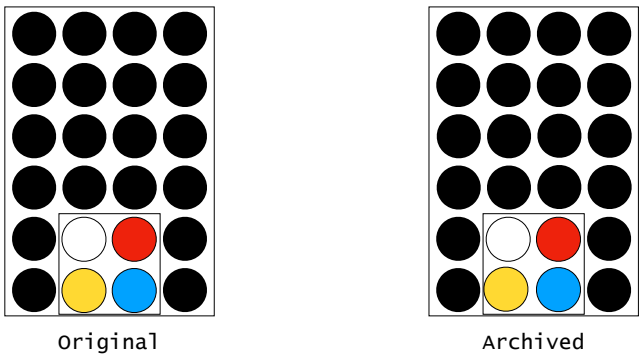


Figure 3.3: The left figure represents the original web page's screenshot containing one element. The right figure represents the archived web page's screenshot containing two errors: The `absent element` error resulted in a missing element corresponding to the original web page's element and the `additional element` error added an element that looks like the one missing. The errors cannot be detected.

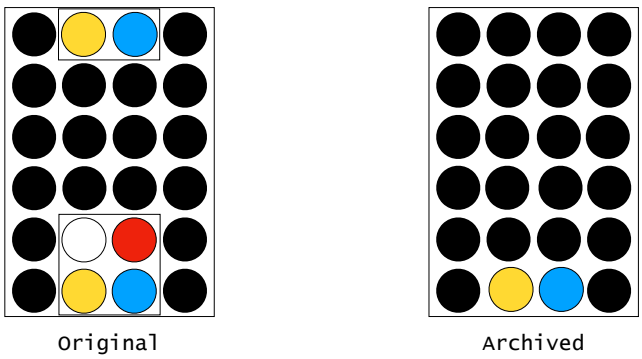


Figure 3.4: The left figure represents the original web page's screenshot containing two elements. The right figure represents the archived web page's screenshot containing one element which looks like the part both elements in the original screenshot have in common. The true correspondence cannot be found with certainty.

a `translation` error occurred, causing a shift on the corresponding element in the archived screenshot. To put it differently, it is unclear which one of the elements in the original screenshot corresponds to the element in the archived version.

Implications. Conclusively, we emphasize two main problems: Firstly, the categorization of errors is imperfect, which means that we can neither be sure to have found all errors that occurred during the process of archiving nor to have correctly categorized them in every case. Secondly, we underline the impossibility of unambiguous correspondence detection between two elements as a result from the error categorization problem. The implication of the impossibility of finding correspondences with one hundred percent certainty goes even further: It suggests that the ability to perfectly reconstruct the original web page's screenshot from an archived web page's screenshot is limited, as the archiving process could be subject to error types that are impossible to classify unambiguously and the occurrence of false correspondences cannot be excluded with certainty. An error that is not found or categorized correctly can lead to different results in the reconstruction. This suggests that tools invented to solve this problem will possibly not operate perfectly. Hence, they cannot promise to always correctly reconstruct any web page for any given error. However, the mere existence of these limits does not mean most of the web pages are affected by them. Therefore, for the web pages which contain errors that can be correctly detected, categorized, and on which correspond-

ing elements can be identified correctly, it does not seem to be out of reach to improve the archiving process through automatic quality assessment using reconstructed screenshots.

General Assumptions. The above implications suggest that for using the pixel error as a quality measure and reconstructing archived web pages' screenshots, the following assumptions will be considered throughout this work:

1. We consider the error which is detected first to be the correct and only error, even though, theoretically, other errors could have caused the visual difference between the archived and original web pages screenshot.

2. The element containing the error is always the element in the archived version of the screenshot, not the original.

3. If a pixel-by-pixel comparison results in no error for two elements, we consider the error absent.

4. Two elements that look the same are corresponding elements. If multiple elements look the same, the one closest to the original position is the corresponding element. If all elements are equally far away, one will be chosen randomly.

Chapter 4

A Framework for Quality Assessment

The aim of this thesis is to further the development of a tool for automatic quality assessment of archived web pages. In the previous chapter, we described a quality measure, the number of differing pixels between archived and original screenshot, to indicate the quality of an archived web page and explained that this pixel error is noisy and therefore does not reach its full potential for indicating an archived web page's quality. In this chapter, we introduce a generic framework for the development of a tool to automatically assess the quality of an archived web page. This framework aims at reducing the noise of the pixel error by reconstructing the archived screenshots. We provide an overview of all steps of the framework for automatic quality assessment and explain their purpose and methods in detail.

A Generic Framework for the Development of a Tool for Automatic Quality Assessment. Our method of measuring the quality of archived web pages is an iterative process that follows the steps presented in figure 4.1 on page 27. The framework displays the main steps *reconstruction*, *comparison* and *model*. The reconstruction is considered the starting point of the framework. It takes three inputs: A screenshot of the original web page (original screenshot), a screenshot of the archived web page (archived screenshot), and a browser-generated file describing the tree structure of all archived HTML elements found in the archived screenshot including their positions and contents (description of structure). The reconstruction process will use a tool that provides the reconstruction and the model with additional data about the changes the original web page has undergone during the archiving process. The framework is extendable so that it can be iterated with a different tool every iteration, reducing the occurrence of different error types (e.g., translation

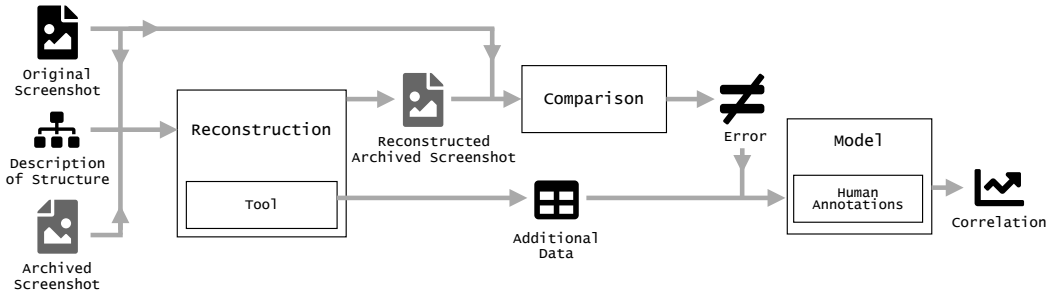
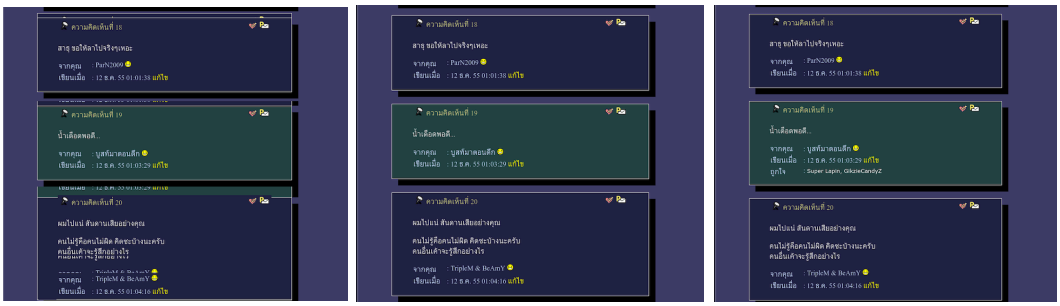


Figure 4.1: A generic pipeline displaying the procedure for assessing an archived web page's quality.

error). During reconstruction, the errors detected by the tool are removed in the archived screenshot (e.g., elements are shifted back to their original position). This results in a reconstructed version of the archived screenshot. The reconstructed screenshot may replace the archived screenshot as input for the reconstruction step in the next iteration. Together with the original screenshot, it then serves as input for the next step in the framework—the comparison. The comparison uses the quality measure explained in chapter 3, namely a pixel by pixel comparison of the reconstructed screenshot and the original screenshot. It outputs the sum of the pixels that differ between the screenshots as an absolute or relative error. The last step will apply the measured error and the data gathered by the tool to a regression model. The purpose of the regression is to find a correlation between the data drawn from the web pages and the human annotations made to the same web pages, grading their quality on a scale from 1 (best) to 5 (worst).

4.1 Step 1: Reconstructing the Archived Web Page's Screenshot

In this section, we will describe the first step of the framework that consists of the reconstruction of the original web page's screenshot based on the archived screenshot, the corresponding description of its structure, and the output of the tool. We will first discuss the benefit reconstruction offers to our process, and afterward, we will take a look at the general technical process of the reconstruction.



Reconstructed Screenshot | Archived Screenshot | Original Screenshot

Figure 4.2: A reconstructed screenshot (left) does not always transport content more clearly than its unprocessed archived (middle) or original screenshot (right).

4.1.1 The Image Reconstructor

Utility. Our approach of reconstructing the original screenshot's appearance from the archived screenshot by eliminating pixel errors aims to extract data that incorporates information about the error types and their specific form found in the archived screenshot. We suggest that this data can enrich the model and strengthen the correlation with the human annotations data set collected by Kiesel et al. [16].

At this point, we need to emphasize what we *do not* use the reconstruction process for: The reconstructed screenshot could visually make a worse impression on the human reader than the archived screenshot, for example, because related elements can be torn apart, as shown in figure 4.2. The Image Reconstructor merely reduces the noise in the pixel error in order to obtain a better indicator for quality. Its output, the reconstructed archived screenshot, will never be seen by a human reader of the archived web page.

Functionality. We developed a piece of software to obtain the reconstructed version of an archived screenshot which we called Image Reconstructor. The Image Reconstructor uses the following three inputs:

Original Screenshot
: The original screenshot is a screenshot of an online web page in the moment of archiving. The HTML source that this screenshot was taken from is an online web server hosting the original web page.

Archived Screenshot
: The archived screenshot is a screenshot of the archived version of a web page. The HTML source that this screenshot was taken from is a web server from the archive. There is a corresponding original screenshot for

every archived screenshot. Both of these screenshots document the same web page but from different sources.

Description of Structure
: The description of structure is a browser-generated text file that contains a list of the HTML elements from the archived web page and also provides their positions, element types, and contents. Thus, it contains the tree structure of the elements of the entire archived web page.

Moreover, the Image Reconstructor incorporates a tool that provides additional information about the differences between each archived and original screenshot pair. The tool is supposed to tackle a specific error type which manifests in differences of pixels between archived and original screenshot. Furthermore, since the tool may change for every iteration, the information it gathers would usually differ for every iteration as well. One example of an output of the tool—if used to tackle the translation error—is a list of detected shifts for each element in the archived web page. Later on, when discussing our implementation of the framework, we will return to this example in more detail. Using the three inputs and the additional information on the error type that the tool provides, the reconstruction step aims at eliminating or at least reducing the noise in the pixel error generated by this error type in the archived screenshot while at the same time gathering data about its occurrences. The actual reconstruction is then computed through the following steps:

- Read the original screenshot.
- Read the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output screenshot—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data for each element.
- Create a list of all elements in the element tree object sorted by depth in the HTML structure.

- For each element in this list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list (e.g. apply detected shifts to improve upon the translation error).
  - Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed archived screenshot).

The resulting output of the reconstruction is a reconstructed version of the archived screenshot that should display less pixel error compared to the original screenshot than the non-reconstructed archived screenshot.

4.2 Step 2: Measuring the Pixel Error

In this section, we will elaborate on the second step of the framework: the comparison. Here, we will explain why the comparison cannot replace the reconstruction step and then present the general technical functionality of the code responsible for the comparison.

Comparison versus Reconstruction. It may be counter-intuitive to use an imperfect reconstruction process when there is also the comparison, which uses the quality measure that can generate perfect copies of the original screenshots from the archived versions using `insertion, deletion` and `substitution` and seemingly eliminate all errors. The reason why the comparison cannot replace the reconstruction is that the Image Reconstructor is not supposed to create a perfect replica (a copy) of the original screenshot. Otherwise, all information on the error types would be overwritten during the reconstruction. Therefore the Image Reconstructor will only eliminate the noise in the pixel error but not the actual indicators of issues on the archived pages, which decrease their quality. The comparison will measure these differences using the quality measure. How this is done will be described in the next section.

4.2.1 The Image Comparator

Utility. We implemented a program, called the Image Comparator, to execute a pixel by pixel comparison between an archived web page's screenshot and its corresponding original web page's screenshot. The comparison outputs the pixel error of the two input screenshots. The input can be either

an archived screenshot and its corresponding original or the original screenshot and its corresponding reconstructed screenshot (the output of the Image Reconstructor).

Every iteration of the framework would reduce a different reproduction error type providing additional information about its specific occurrences. In the last step of the framework, all additional data on the reduced reproduction error types and the pixel error will be used together to train a statistical model which predicts the web pages' qualities according to the human annotations.

Functionality. To compute the error as described in chapter 3, we implemented the Image Comparator to execute our adaptation of the edit distance by Levenshtein, using the operators `insertion`, `deletion`, and `substitution`. The Image Comparator takes a screenshot of an original web page and a screenshot of its corresponding archived version in any form, be it unprocessed or reconstructed, as an input. The comparison is computed through the following steps:

- Read the original screenshot.
- Read the archived screenshot.
- Use the bigger screenshot's height to compute the dimensions of a placeholder image. This image, which we call heatmap, will be used to store the presence or absence of error for every pixel. The heatmap can also be used to visualize the differences. Note that the widths of archived and original screenshot are always the same (1 366 px), as defined by the archiving process implemented by Kiesel et al. [16].
- Compute the insertion or deletion error:
  - If the archived screenshot is shorter than the original by a number of x rows: Color the bottom x rows of the heatmap blue to mark the insertion and compute the insertion error as $x \cdot width = x \cdot 1366$ pixels.
  - If the archived screenshot is longer than the original by a number of y rows: Color the bottom y rows of the heatmap red to mark the deletion and compute the deletion error as $y \cdot width = y \cdot 1366$ pixels.
  - If both screenshots are equally sized: Do not apply any color to the heatmap and set the insertion and deletion error 0 pixels.
- Substitute differently colored pixels:

  - For all pixels in the heatmap that have not been colored: Compare the color of the pixel in the original to the color in the archived screenshot and assign the color black to the heatmap if they differ, or white if they are equal.
  - The substitution error is equal to the number of black pixels.

- Compute the total error as an absolute error by using the sum of all non-white pixels in the heatmap, which is equal to the sum of substitution, insertion and deletion error.

- Fill a .csv file with the following data:

  - absolute insertion (or deletion) error
  - absolute total error
  - number of pixels in the original
  - relative total error as percent of the number of pixels in the original
  - number of pixels in the heatmap (which always has the dimensions of the bigger screenshot)
  - relative total error as percent of the number of pixels in the heatmap

If the Image Comparator gets a reconstructed screenshot and compares it to the corresponding original, there will never be an insertion or deletion error, as both screenshots already have the same dimensions. This is because the height of the archived screenshot would already have been adapted through the Image Reconstructor (and the width would have been equal in both screenshots anyway).

4.3 Step 3: A Quality Assessment Model

We built a quality assessment model in order to be able to predict the true quality of an archived web page. As ground truth for the true quality of an archived web page, we used the data set presented by Kiesel et al. [16], which contains human annotations about the quality of all archived web pages in our data set. The quality was judged on a Likert scale of five options ranked by severity of decrease in quality of a web page after archiving compared to the original (online) web page [16]. On this scale, the score of 1 means that the archived web page has the same quality as the original (meaning that the archiving process had no effect on the quality), and 5 indicates that the effects were so severe that the web page's archived version was rendered unusable [16]. These human annotations should be correlated with the pixel error between

the archived, the reconstructed, and the original web pages' screenshots, and the additional data we gathered through the reconstruction process.

In order to find a correlation between the quality assigned by the human annotators and the data from the reconstruction process, we use a linear regression model. We chose linear regression because it is easiest to implement and interpret compared to other regression methods. Hence, it seemed like a good starting point for an analysis of the correlation.

Chapter 5

A Data Set for Implementing the Framework

To implement the general framework described in chapter 4 in the next step, we first needed to gather the necessary data. To do this, we merged parts of two different data sets—webis-web-archive-17 and webis-web-archive-18. Both hold information on the same web archive. The two data sets contain three snapshots of each web page in the archive, taken at different points in time: webis-web-archive-17 provides us with a snapshot of the web pages taken at the moment of archiving and another snapshot taken shortly after the archiving was done, while webis-web-archive-18 contributes the third snapshot of the archived versions of the web pages at a later point in time. The use of both data sets was necessary as the third snapshot, contained in webis-web-archive-18, includes a description of the structure of the archived web pages, which was missing in webis-web-archive-17. In this chapter, we will introduce in detail the data set used to implement the general framework. Additionally, at the end of this chapter, we will discuss some problems with older web archives that we encountered in the process of gathering our data set.

5.1 Combining Files from Multiple Data Sets

Because not all information was found in just one data set, necessarily some work went into merging two data sets to get all files we would need to further our process. In the following sections, we will describe all files that occur in our final data set.

Data Set 1: webis-web-archive-17. The data set used to obtain files on the original web pages is webis-web-archive-17, a web archive produced

in September 2017. It was collected by Kiesel et al. and is described in "Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment" [16]. This data set comprises 10 000 folders, each containing the following data for an archived web page (descriptions according to [16]):

`archive-*.warc.gz`
: The web archive file that contains all web page data. It was created using the webis-web-archiver[1] which scrolled down up to 25 screen heights during the archive process.

`replay.db`
: Index of the warc contents.

`archiving.png`
: PNG screenshot taken by the webis-web-archiver after scrolling down.

`archiving.html`
: HTML DOM snapshot taken by the webis-web-archiver after scrolling down.

`reproducing-custom,pywb,warcprox.png,html`
: Screenshots and snapshots taken by the webis-web-archiver after scrolling down during reproduction of the page from the warc.gz when using one of the three reproduction methods. These reproduction methods refer to different proxies that redirect the browser to the archived web page. While webis-web-archive-17 used three different proxies to redirect, namely `custom`, `pywb` and `warcprox`, the proxy used in the other data set (webis-web-archive-18) was always `custom`.

`normalized-rmse.txt`
: RMSE (root mean squared error) matrix when comparing the images (archiving and reproducing) using imagemagick's 'compare -metric rmse'.

`page-quality-scores.txt`
: The quality of the screenshot of the archived web page compared to the screenshot of its corresponding original web page's screenshot that was assigned by human annotators. The quality scores range from 1 ("no significant difference") to 5 ("unusable")

Data Set 2: webis-web-archive-18. To compute the translation error based on the screenshots from the archived web pages and the corresponding original screenshots, we need a description of the structure of the archived

[1] see https://zenodo.org/badge/latestdoi/107244409

screenshots, which is not contained in webis-web-archive-17. Therefore we use another data set, webis-web-archive-18, to provide us with this structure file and the corresponding screenshots of archived web pages. This data set was generated in February 2018 and includes a subset of 9 519 web pages related to the 10 000 web pages from webis-web-archive-17. To obtain these files, the same archived web pages which had already been used to produce the archived screenshots in webis-web-archive-17 were opened again, and the screenshots were taken anew. Additionally, a description of the structure of the newly opened archived web pages was generated by the browser. The result of this process were 9 519 new versions of screenshots of the archived web pages from the data set webis-web-archive-17 along with corresponding files for each of these pages containing a description of structure:

`archived-page.png`
: A screenshot of the archived web page.

`elements.txt`
: A text file containing the structure of the elements found in the archived web page. The structure data contains the ID of the element as position of the element in the HTML structure of the archived web page, the text content of the element and the position of the element as top left and bottom right coordinates in pixels, referring to a position of the element's surrounding rectangle in the screenshot.

Final Data Set: Merged webis-web-archive-17 and webis-web-archive-18. To summarize, the merged data set, which we use to implement the general framework for reducing the occurrence of the translation error, contains files on the 9 519 web pages that occur in both data sets webis-web-archive-17 and webis-web-archive-18.

From webis-web-archive-17, we use:

`archiving.png`
: The screenshot of the original web pages.

`page-quality-scores.txt`
: A text file containing the human annotations concerning the quality of the screenshots of original web pages and the screenshots of their corresponding archived versions.

From webis-web-archive-18, we use:

`archived-page.png`
: The screenshots of the newly opened archived web pages.

`elements.txt`
: The description of structure for each archived web page, containing each element's ID as XPath, the content of the element and its position in the corresponding screenshot (`archived-page.png`) as top left and bottom right coordinates in pixels.

5.2 An Analysis of the Combined Data Set

As the general idea of an archive is to perfectly conserve artifacts for future use, the notion that the appearance of an archived web page can change over time might not be intuitive. Therefore, we found it essential to be more informed about the kind of changes that can appear. This is why we took a closer look at the differences between the archived screenshots from the older data set webis-web-archive-17 and the archived screenshots from the data set webis-web-archive-18, which were reproduced about five months later from the same archived web pages. We noticed that 2 438 of the 10 000 web pages appeared to have no pixel error compared to the original web page in the older data set webis-web-archive-17 but displayed a pixel error compared to the original in the younger data set webis-web-archive-18. Among these pages, the relative pixel error resulting from a comparison with the corresponding original web pages was distributed as shown in figure 5.1. The histogram illustrates a clear majority of the 1 582 screenshots that display an error less or equal to 5% compared to the original screenshot, making up about 65% of all affected web pages.

Comparing random samples of screenshots from both data sets to find out more about the kind of errors that appeared in webis-web-archive-18 and webis-web-archive-17 in general, not only of the web pages with an error of less than or equal to 5%, it seems that one error is by far the most dominant one: Half of the manually reviewed screenshots (64 samples) display an error that is caused by different renderings of color gradients. We discovered that artifacts in raster images in the screenshots show a minor color difference of 1 color value per channel red, green and blue. An example of such a difference is shown in figure 5.2.

Another frequent error found among these manually selected web pages is a horizontal shift which most often affects the entire page and is usually caused by a scrollbar being present in the younger screenshot but absent in the older. In the rare cases for which the horizontal shift does not occur on the entire page but on single elements, the shift is caused by a dysfunctional time offset: The proxy which calls the web page from the archive is set to simulate the date and time of the moment it archived the original web page. This is supposed to have the effect that any date displayed dynamically on

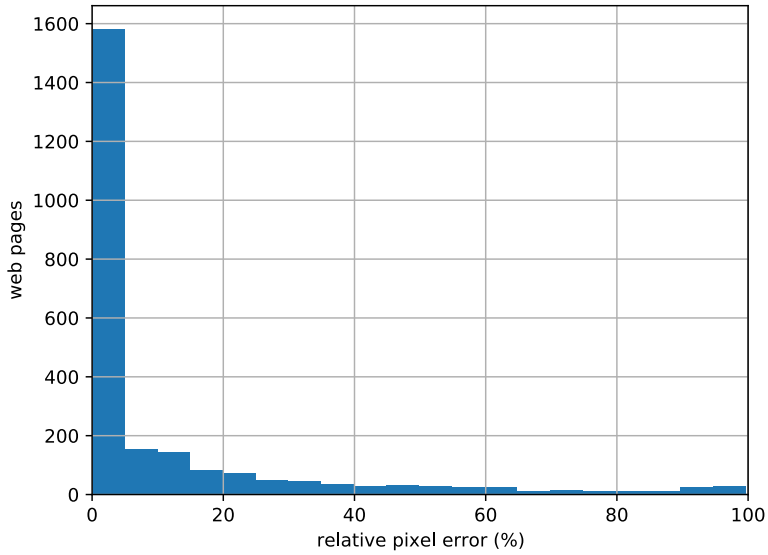


Figure 5.1: Distribution of the relative pixel error in the screenshots of 2 438 archived web pages from webis-web-archive-18: The screenshots of the same archived web pages in webis-web-archive-17 had no pixel error, but the screenshot of the archived web pages from webis-web-archive-18 had a pixel error larger than zero.

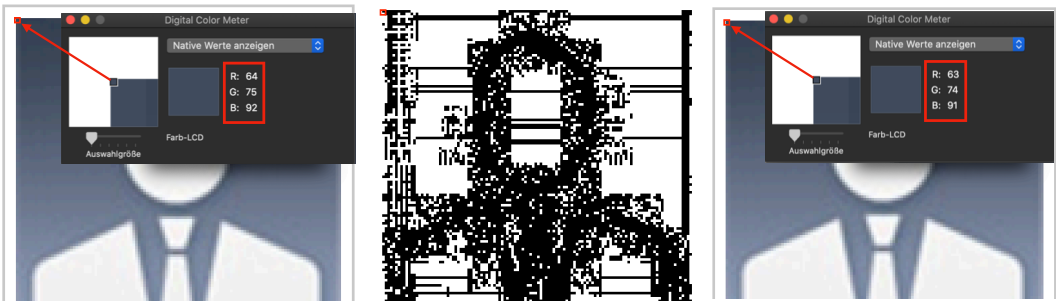


Figure 5.2: The different color values of the same pixel on an archived page from webis-web-archive-17 (RGB(63,74,91)) and webis-web-archive-18 (RGB(64,75,92)) are caused by different renderings of color gradient effects.

the archived web page should match the date on the screenshot of the original web page. Unfortunately, this does not always work: For example, in one of the pages, a horizontal shift of single elements was caused when the page displayed "Today" as the entry date for a blog post on the screenshot from webis-web-archive-17, but in the screenshot from the archived web page from webis-web-archive-18 read "09/22/17" as entry date for the same blog post. Due to the change of characters, the shift was only caused for the elements containing said difference.

An error that was not found very often, but that usually caused huge differences between the screenshots of the younger and older data sets, was a difference in the height of the two compared screenshots.[2] These differences were produced by a malfunctioning resize of the browser window, resulting in the screenshot being too short. In the observed examples, this malfunction only affected pages from the younger data set. This is because our sample contained exclusively pages with no error compared to the original in the older data set but displayed some error in the younger one.

At the beginning of this exploration, we were hoping to find the first signs of degeneration of the archive, manifesting in errors like missing elements that had been loaded from external pages which had gone offline or any other error caused by the relation of the archive and the "outside world". Instead, we were confronted with the idea that not the archive itself showed the first signs of aging, but the technology we recorded it with. Almost every error we observed, especially the omnipresent error caused by artifacts in raster images being computed differently, and the horizontal shifts seemed to be caused by different browser versions used for the two "recordings" of the archive. While we have indeed observed some missing elements in pages (an error which was almost only affecting Facebook's Like-Button), this kind of error type was very rare compared to the ones mentioned before.

In summary, we can say that when dealing with a long-term archive and measuring its quality based on screenshot differences, it would be essential to guarantee the exact same technological basis in order to be able to review the archive and assess its quality correctly. The alternative would be to prepare for a longer reviewing process of analyzing the different errors caused by different versions of the technologies and removing those errors from the quality computation. Conversely, the first suggestion would hardly be possible to realize in the long run, for example, due to missing hardware support for very old browser versions. However, the second suggestion could be similarly difficult to put into practice as error detection might not work perfectly for all possible errors that occur due to a change of technology.[3]

[2]Of course, such an error can also cause small differences, if only one row of pixels was missing in one screenshot.

[3]On the problems of error detection, see chapter 3.

Chapter 6

An Implementation of the Quality Assessment Framework

In this chapter, we will present an implementation of the framework, which we introduced in chapter 4. This framework builds upon four distinct parts: (1) a web archive data set with quality annotations (see chapter 5); (2) a component to reconstruct the original web page's screenshot from the archived screenshot; (3) a component to calculate the error; and (4) a model to compute the correlation between the error and the quality of the archived web page. We introduced part (1), the web archive data set with quality annotations, in chapter 5; in the following sections, we will explain in detail the other parts (2), (3), and (4).

Following the general idea of the framework, which is to reduce the noise in the pixel error generated by certain error types, we implemented this framework[1] with the aim of reducing the noise generated by the translation error. To do this, we used video encoding to move translated elements back to their respective positions in the original screenshot. Among the many reproduction error types and their combinations, we decided the translation error was the best choice for the first implementation of the framework. Not only does it often occur in our data set, but also it can cause much noise in the pixel error (as explained in section 3.2). Therefore, we deemed the reduction of the translation error to be most effective in terms of strengthening the correlation between the pixel error and the quality of the web page.

[1] The repository containing the software of this implementation is stored at `https://git.webis.de/code-teaching/theses/thesis-elstner`. Throughout this thesis preexisting code of this implementation was adapted and extended substantially.

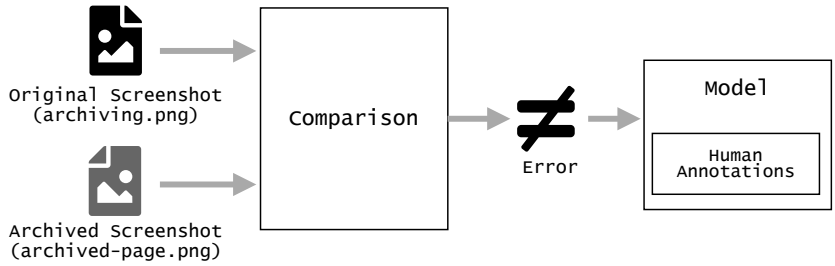


Figure 6.1: Procedure for computing the baseline error to be used in the model later on.

6.1 Definition of the Baseline Error

We call the pixel error between the screenshot of the unmodified archived web pages and their corresponding original web pages the *baseline* or *baseline error*. This pixel error is called baseline because it will later be compared with the pixel error and other information gained from the reconstruction of the archived screenshots. We computed this pixel error as the adaptation of the edit distance explained in chapter 3, using our implementation of the Image Comparator, described in section 4.2.1. As we explained, the unprocessed screenshots of the archived web pages are drawn from webis-web-archive-18 and called `archived-page.png`, and the screenshots of their corresponding original web pages that stem from webis-web-archive-17 are called `archiving.png`. These files serve as input to the baseline computation. The baseline error will later serve as an independent variable for the model. It defines the error of the unprocessed archived screenshots and provides a reference point for the reconstructed screenshots' pixel errors. Figure 6.1 illustrates the general process of the baseline error computation and its later use in the model.

6.2 Image Reconstruction

Before describing the process of reconstructing archived screenshots with a reduced translation error, we will define the meaning of the term *Translation Error* in this context.

Existence of the Translation Error: *For an element in the original web page's screenshot* $\mathcal{E}_{original} = (a, b, c, d)$ *that reoccurs in the archived web page's screenshot as element* $\mathcal{E}_{archived} = (a', b', c', d')$*, we call the translation error to be present, if there is a tuple* (x, y)*, with* $x, y \in \mathbb{Z}$ *and* $(x \neq 0 \vee y \neq 0)$*, for which the following statement is true:*

$$(a + (x, y) = a') \wedge (b + (x, y) = b') \wedge (c + (x, y) = c') \wedge (d + (x, y) = d')$$

Here, (a, b, c, d) *are absolute pixel positions in the original web page's screenshot describing the coordinates of the four corners of a rectangle in* $\mathbb{R}^2$ *with ((topleft.x, topleft.y), (topright.x, topright.y), (bottomleft.x, bottomleft.y), (bottomright.x,bottomright.y)). Respectively,* (a', b', c', d') *describe the absolute pixel positions in the archived web page's screenshot describing the coordinates of the four corners of a rectangle in* $\mathbb{R}^2$ *with ((topleft.x, topleft.y), (topright.x, topright.y), (bottomleft.x, bottomleft.y), (bottomright.x,bottomright.y)).*

We will now explain our method of detecting and reducing the occurrence of shifted elements in archived screenshots using video compression. Thus, we will elaborate on the process of motion detection in video encoding and then explain various implementations for reconstructing the archived screenshots and re-positioning the shifted elements. Not only does this reconstruction process provide a reconstructed screenshot with a less noisy pixel error, but we can use the data on the detected translations as features for an estimate of the reproduction quality.

6.2.1 Video Encoding

The first step in detecting the translation error is to determine which elements in the screenshot of the archived version of the web page have shifted from their position in the original screenshot. For this, we employ video encoding technology. For the elements in the archived web pages' screenshots, we know the positions (as explained in chapter 5), but for the original web pages' screenshots, this information is missing. Hence, we need to derive these positions in order to gain information on the shifts. Video encoding solves this problem: It uses motion estimation to detect shifted pixel blocks from one frame in another.

We transformed the screenshot of the original web page and the screenshot of the corresponding archived web page into an H.264 encoded video file with exactly two frames—the first being the original web page's screenshot and the second the screenshot of the archived version. The H.264 standard aims to compress the video through motion estimation by not duplicating but just referring to the moved parts in another (in our case, the previous) frame. This mechanism uses motion vectors to signal if a block of pixels has moved compared to the preceding frame [27]. The H.264 standard describes multiple ways to compress a video file—defined in so-called profiles—using motion vectors. The profiles differ, among other things, in the size of the pixel blocks for

which one motion vector is computed or the detail of color, which is taken into account in the compressed version of the video input file [28]. From the H.264 encoded video, we were able to obtain motion vectors for the second frame, which hold information on the translation of pixel blocks in the archived screenshot compared to the original screenshot. Due to our knowledge about each element's position in the archived screenshot (as described in chapter 5), we can relate the motion vectors to specific elements in the archived screenshot. As a result, we could restore shifted elements' positions according to the corresponding motion vectors. Here, we decided to use the metric of a majority vote for triggering a shift: Since all motion vectors for one element do not always point in the same direction or even display the same length, we translated an element only if at least two thirds its motion vectors indicated the same shift direction and distance. Indeed a different threshold of motion vectors would have resulted in different shifts for the elements and thus in other results for the reconstructed screenshots. Unfortunately, thorough experimentation with different parameters falls outside the scope of this thesis.

Configuring ffmpeg's Encoding Parameters. To generate the H.264 encoded video from the archived and original screenshots, we use ffmpeg, which is a media converter framework. It offers, among other things, libraries for the H.264 encoding standard [2, 3]. As we are working on Ubuntu 18.04, we use ffmpeg version 3 even though the newer version 4 exists because version 3 is the latest stable release for our system. From the profiles, which are defined in this standard, we use the `High 4:4:4 Predictive Profile` to encode our videos. Below, we will explain the effects of this profile to justify our choice.

`4:4:4` refers to the degree of chroma subsampling used to compress the information about the color in the video. `4:4:4` means that each of the channels in the `Y'CbCr` color space is sampled in a separate plane, so that all colors can be decoded without loss. In other words: No chroma subsampling is used, and all chroma and luma information from the uncompressed frame is kept in the compressed video [29]. Further, the `High 4:4:4 Predictive Profile` provides the maximum amount of 14 bits per chroma sample compared to other predictive profiles and allows for motion prediction through inter frames[2] [28].

Inter frame prediction refers to a technique in video compression which searches for pixel blocks that have already been encoded in a preceding frame or will be encoded in a future frame. If a corresponding block of pixels is found in the preceding frame (original screenshot), instead of storing the block itself for the second frame (archived screenshot), a motion vector will be stored,

[2]In contrast to other profiles, for example all `Intra Profiles`, which encode only intra frames [28].

pointing to the corresponding pixel block in the reference frame [30].[3] In our case, as we only used two frames, the prediction was made for the second frame (the archived screenshot), referring only to the preceding frame (the original screenshot). In the video, the original screenshot was encoded as an I-frame, meaning a decoder does not need additional information to compute this frame. The archived screenshot was encoded as a P-frame; these frames usually are smaller than an I-frame because some of the information they hold is only stored in reference to the preceding I-frame [30]. The encoding standard H.264 allows for various sizes of pixel blocks for which the inter frame prediction is computed: 16 x 16, 16 x 8, 8 x 16, 8 x 8, 8 x 4, 4 x 8 and 4 x 4 px [28].

The general process of computing the motion vectors is called motion estimation and is done during inter frame prediction [31]. While the specifications of the H.264 standard do not lay out any recommendation of an algorithm for motion estimation, we used the algorithms implemented in the libx264 library that result in the highest number of motion vectors [27, 1].[4] These algorithms use "advanced diamond zonal search for 16x16 blocks and half-pixel refinement for 16x16 blocks [...] plus advanced diamond zonal search for 8x8 blocks, half-pixel refinement for 8x8 blocks, and motion estimation on chroma planes [...] plus extended 16x16 and 8x8 blocks search" [1]. Unfortunately, the documentation of ffmpeg and libx264 does not further explain these algorithms. However, other sources (especially [4]) suggest the following connections: The advanced diamond zonal search seems to be based on the diamond search algorithm proposed by Zhu and Ma [24] and is a block matching algorithm which applies diamond shaped search patterns to blocks of pixels in order to find the point with the least distortion compared to the reference point [24, 4]. The motion vector for a pixel block, resulting from diamond search, is the vector between the reference point and the point of least distortion. Half-pixel refinement seems to refer to a method for sub-pixel motion estimation, which interpolates color values between the neighboring pixels at the endpoint of a motion vector which was returned by diamond search. If a better match with even less distortion is found, half-pixel refinement sets the endpoint of the motion vector to the half-pixel position between the original endpoint of the motion

[3] If a block of pixels has moved, that is if its motion vector has a length different from zero, it is also possible to compress this block even more to make the video smaller. This practice is possible due to the incapability of the human eye to perceive detail in moving objects [22]. To regulate the amount of detail stored in translated pixel blocks, ffmpeg allows for a calibration of the constant rate factor up to the point of lossless compression for these blocks [3].

[4] There are other libraries for H.264 video encoding in ffmpeg [1]. Using another library might result in different algorithms for motion compensation and therefore produce different motion vectors than we found using libx264.

vector and the pixel with the better match. This procedure is said to produce more precise motion vectors than the application of diamond search without half-pixel refinement [4]. Finally, we suspect the last feature in the motion estimation setting, promoting "extended 16x16 and 8x8 blocks search" [1], to describe that the search for corresponding pixel blocks is not only performed on 16 x 16 and 8 x 8 pixel blocks, but also on all variations of these block sizes (16 x 8, 8 x 16, 8 x 4, 4 x 8 and 4 x 4 px) [28] mentioned above. A note in the ffmpeg documentation, stating that motion estimation was also computed on chroma planes, confirmed our choice to use the `High 4:4:4 Predictive Profile` because this profile supported our aim to keep most of the information in the frames so that the motion vectors can be computed as accurately as possible.[5] We assumed that using another profile instead, which compresses the video more by using fewer color planes in the encoded video would result in less accurate motion vectors.

Another configuration is the so-called *level* in ffmpeg. Levels in H.264 specify the performance a decoder must meet in order to be able to release correctly decoded frames. We used level 6.2, which is the highest of all levels and supports the highest resolutions per frame rate compared to other levels [28]. We made this decision based mainly on the fact that the screenshots of the web pages in our data set displayed very large heights (up to 16 384 px), and we hoped the highest level would be most supportive of that.

Lastly, we want to examine some reasons for inaccurate motion vectors, because gaining the most accurate set of motion vectors for the archived web pages is essential to reducing the translation error: Different motion vectors will result in different reconstructed versions of screenshots of the archived web pages. First of all, the accuracy of the motion vectors depends strongly on the settings that are used for the video encoding. Since many factors can influence the motion estimation, there may be a better setting for encoding the original and the archived web pages' screenshots that would produce either more accurate or a larger amount of motion vectors. One other possible reason for inaccurate motion vectors is that video compression is typically used on videos with smaller dimensions than the screenshots we used as an input—our input web pages can have a height of up to 16 384 px. Usually, video encoding would be used on standard videos, for which a very high resolution lies within the bounds of 8K. This resolution, describing images with dimensions of 7 680 x 4 320 px, is much smaller than some of our screenshots. Videos with such high resolutions likely exceed the format of standard video files, leading ffmpeg's encoding to produce faulty motion vectors. Also, it is possible that

[5] Computing the reconstructed screenshots under lossless compression by setting the constant rate factor to zero might provide more detail resulting in even more accurate motion vectors for our video.

the algorithms for motion estimation in ffmpeg's libx264 do not consider far jumps of elements from one frame to the other and only try to find a match in closer distance to the reference point. We deem this likely as the farthest vertical shift was only 674 pixels. This is questionable because the farthest horizontal shift distance was 1 189 pixels. Because many of the web pages have larger vertical than horizontal dimensions and that the farthest horizontal shifts reach the whole width of the screenshots, but the farthest vertical shift does not even come close to the average height of the web pages, we suppose that the H.264 encoding process is not able to compute the motion vectors for such massive frame sizes. We suspect the reason for this observation to lie within performance aspects of video encoding: A typical use case of video compression is a transmission of a video file for which not only the small size of the encoded video is a performance benchmark but also a quick encoding process. We deem it likely that in order to meet the speed criterion in the encoding process, the diamond zonal search is not executed on pixels with too large a distance from the reference block.

Reconstruction through Video Encoding in a Nutshell. To summarize, we used ffmpeg's libx264 library to produce an H.264 encoded video made up of the original screenshot as the first frame and the archived screenshot as the second frame. In the second frame of the H.264 encoded video, ffmpeg generates motion vectors for every block of pixels. These motion vectors show the difference of each pixel block's position in the second frame compared to its position in the first frame. We used the information from the motion vectors to compute a reconstructed version of the screenshot of the archived web page. In this reconstructed screenshot, all elements for which a shift was detected were shifted back to the position the motion vectors indicated. As the motion vectors for one element do not always point in the same direction or have the same length, we decided to only translate the element if two thirds of its motion vectors implied the same shift direction and distance. Through this process of reconstruction, we gained additional data for our web pages: A reconstructed version of the archived screenshot, in which the occurrence of the translation error was reduced, and a file containing the shift distance for each shifted element on each web page. Figure 6.2 shows the embedding of video encoding in the reconstruction step and the related output in the generic framework.

6.2.2 Image Reconstruction Methods

Gaps. When shifting elements to reduce the translation error, the shifted elements leave a gap in the reconstructed screenshot. Sometimes this gap is

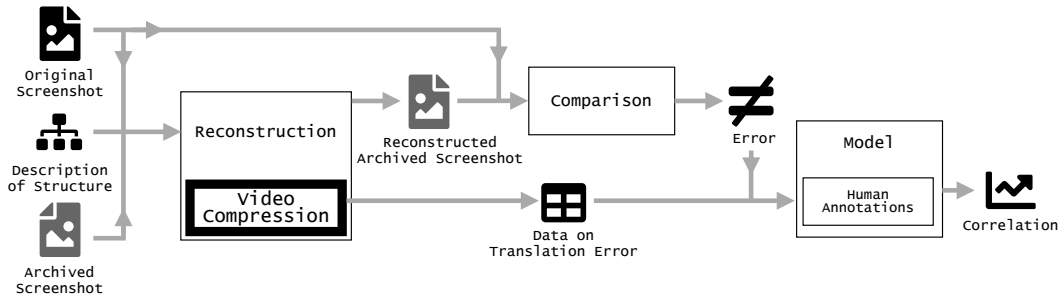


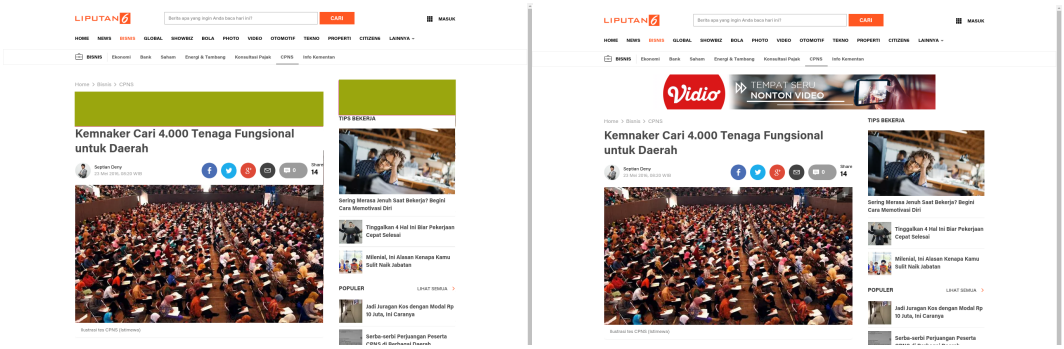
Figure 6.2: An implementation of the generic framework which shows the procedure for assessing an archived web page's quality by reducing the occurrence of the translation error through video compression. The reconstruction uses the motion vectors produced during the encoding process to shift the translated elements in the archived screenshot. The output of the reconstruction is (1) a reconstructed version of the archived screenshot in which translated elements were moved back to their respective position in the original screenshot and (2) data on the translation error for each element for which a translation was detected.

covered by another shifted element in the drawing process, but often the gap is left blank. Figure 6.3 shows an example of such gaps, colored green. It is not a trivial task to color the pixels in the gap because neither the archived screenshot nor the description of structure contains reliable[6] information about the pixels behind a shifted element. However, the color of the gap influences the pixel error, measured in the comparison step, since the colors of the pixels in the gap might match or differ from the pixels in the original version. The following sections will discuss the different methods for coloring the gaps during the reconstruction process in more detail.

Reconstruction with Duplicates

Method 0: Duplicates The first method we used to reconstruct the archived screenshots did not leave any gaps: The shifted elements in the archived screenshot were copied, and the translation error was only fixed for one duplicate. In contrast, the other duplicate of the element was not moved. A detailed description of this reconstruction method can be found in the appendix on page 112. Figure 6.4 shows the result of this reconstruction method, where 6.4a displays the reconstructed version of the archived web page and 6.4b the unprocessed

[6]The color information is not reliable, because even if the background color information of the parent element can be detected it does not mean the pixels in the gap would be colored in the background color of the parent (the pixels in the gap could have contained another element which is missing or was also shifted). Additionally, sometimes there even is no colored parent element: It happens regularly that the entire body element is shifted.



(a) (b)

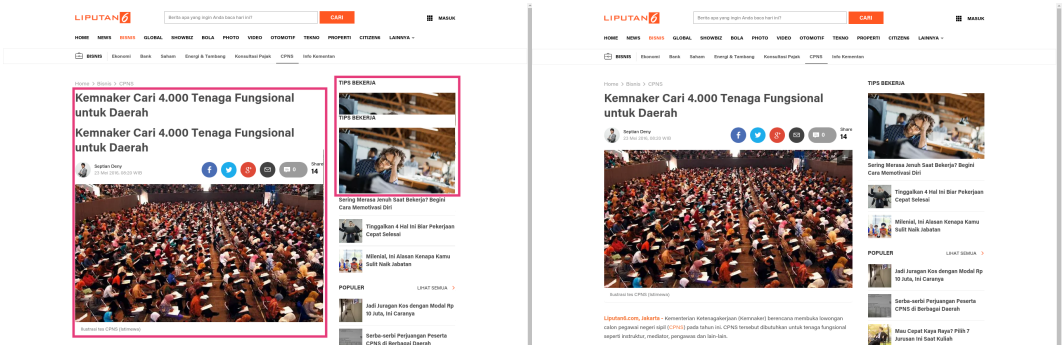
Figure 6.3: A reconstructed screenshot of the archived web page with a reduced translation error (a). The green rectangles mark the gaps the shifted elements leave behind. Compared with the original web page's screenshot (b) the gaps would decrease the positive effect the translation of the shifted elements produced on the pixel error.

archived screenshot for better detection of the changed parts. (Figure 6.3b shows the corresponding original screenshot.)

The idea behind this form of handling the gap was lead by the notion that every pixel that held no error before the reconstruction should not display one afterward. This means that gaps which were left after the translation of an element were still colored the same way as they were in the unprocessed archived screenshot. Therefore, the pixel error would not be different for those parts of the reconstructed screenshot that were unknown after shifting the elements because every pixel in the unknown part which held a pixel error before the reconstruction would still have the same color—hence the same pixel error—afterward. Conversely, this also means that it would very rarely be possible to produce a perfect reconstruction, even if all shifted elements were re-positioned correctly. Therefore we deemed it necessary to explore other procedures which do not produce duplicates to handle the gaps in the reconstructed versions of the archived screenshots. These procedures will be explained in the following sections.

Reconstruction Without Duplicates

Regardless of the gaps, the decrease in pixel error we could maximally reach by reducing the occurrence of translated elements is limited by the accuracy and completeness of the set of detected shifts, which we gained from the motion estimation through ffmpeg. We computed the upper and the lower bound of how much influence the color of the gap could have on the pixel error to better

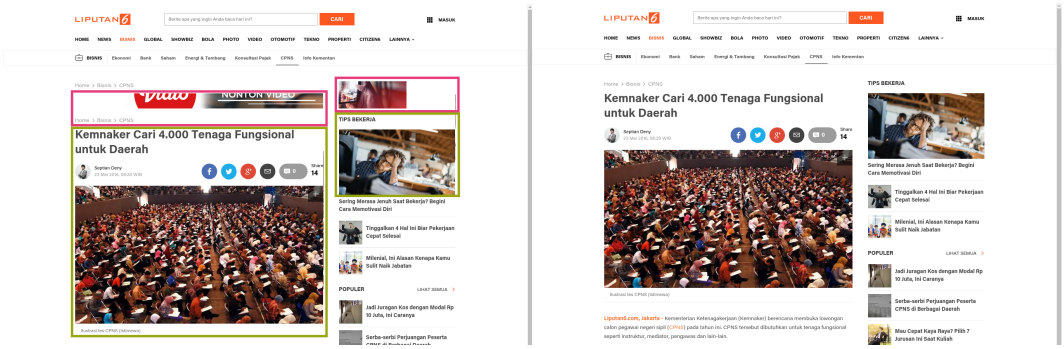


(a) (b)

Figure 6.4: A reconstructed screenshot of the archived web page with duplicates of shifted elements (a). The pink rectangles in (a) mark the duplicates and the shifted elements which leave them behind. In contrast to the unprocessed archived web page's screenshot (b), the pixel error between this reconstructed screenshot and the original screenshot would not change at the position of the static (not shifted) duplicates compared to the baseline error, but the pixels in the shifted duplicates would not produce a pixel error anymore. Therefore, the overall pixel error for this reconstructed version of the archived screenshot would be reduced.

understand how much influence the reduction of the translation error had on the pixel error. In the following paragraphs, we will first explain the upper and lower bound methods. After that, we will explain two reconstruction methods that use different strategies for coloring the gaps and produce a pixel error within the upper and lower bounds that can be used to indicate the quality of an archived web page.

Method 1: Upper Bound. The upper bound computation aims to minimize the pixel error the gaps would produce in the reconstructed screenshots. This would help us to gain knowledge about the quality we would be able to achieve with the underlying set of detected shifts in a reconstruction process. To compute the upper bound, we filled the gaps which were left after translating the elements by copying the corresponding pixels of the original screenshot. A detailed description of this reconstruction method can be found in the appendix on page 113. This computation resulted in a new set of reconstructed screenshots, in which not only information of the archived screenshot but also information taken from the original screenshot was used. To illustrate the effect of this procedure, figure 6.5 displays the output of this computation, where 6.5a shows the reconstructed version and 6.5b the unprocessed archived screenshot.

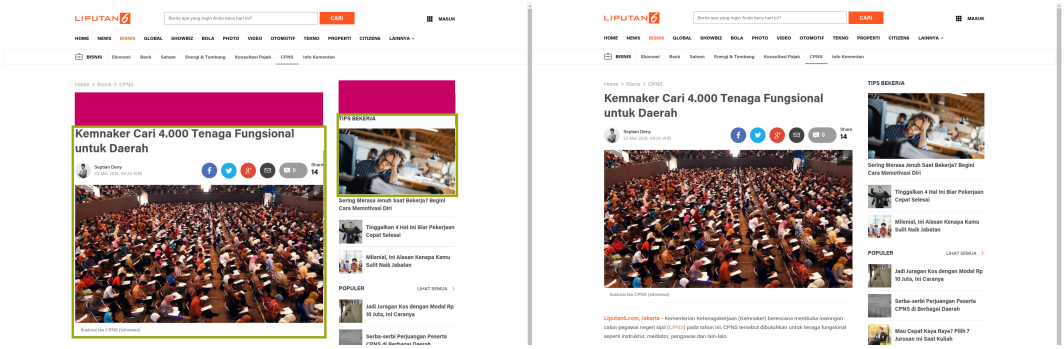


(a) (b)

Figure 6.5: A reconstructed screenshot of the archived web page with copied pixels from the original web page's screenshot at the gaps the shifted elements left behind(a). The pink rectangles in (a) mark the gap areas in which the pixels were copied from the original screenshot; the green rectangles mark the shifted elements which produced the gap. In contrast to the unprocessed archived web page's screenshot (b) the pixels contained in the gaps would not produce a pixel error when compared to the original screenshot. Therefore, the overall pixel error for this reconstructed version of the archived screenshot would be reduced to the minimal value for the underlying set of detected translations.

It is necessary to elaborate on the reason why the reconstructed screenshots which were produced by the upper bound computation cannot be used to estimate the quality of the archived web pages. Using this method, we gained reconstructed versions of the archived screenshots with pixels taken from the original screenshot. Therefore, those reconstructed screenshots could incorporate visual representations of elements that did not exist in the archived web page. This means information on the actual quality of the archived web page is overwritten by this reconstruction method. For example, if the shift of an element in the reconstructed screenshot led to a gap at a place where an element was missing in the archived screenshot, the gap would have been filled with the missing element. This would result in the false information that this element was present in the archived web page while it was really being absent.

Method 2: Lower Bound. The lower bound computation aims to maximize the pixel error the gaps would produce in the reconstructed screenshots. To compute the lower bound, we implemented a slight change to the computation of the upper bound: Instead of copying parts from the original screenshots to fill the gaps, we colored every pixel in the gap different from the color the corresponding pixel had in the original screenshot. We would write a pink pixel to the gap if the corresponding pixel in the original screenshot were not



(a) (b)

Figure 6.6: Two screenshots of an archived web page. (a) displays the reconstructed version of the unprocessed archived web page's screenshot (b). The pink rectangles in (a) mark the gap areas in which the pixels were colored differently than the corresponding pixels in the original screenshot; the green rectangles mark the shifted elements which produced the gap. In contrast to the unprocessed archived web page's screenshot (b) the pixels contained in the gaps would produce the largest possible pixel error when compared to the original screenshot. Therefore, the overall pixel error for this reconstructed version of the archived screenshot would be increased to its maximum possible value for the underlying set of detected translations.

colored the same pink; else we would color it violet to produce a pixel error for every pixel in all gaps. A detailed description of this reconstruction method can be found in the appendix on page 114. The output of this process is shown in figure 6.6, where 6.6a displays the reconstructed screenshot containing the lower bound (all pink pixels) and 6.6b the unprocessed archived screenshot.

Method 3: Most Frequent Color in Archived Screenshot. The idea behind the method of coloring the gap in the most common color of an archived screenshot was to generate the smallest possible pixel error in the reconstructed screenshot without losing information about the archived screenshot's quality. The intuition behind using the archived screenshot's most common color to fill the gaps was motivated by two scenarios: (1) A gap is at a position of an element that is missing in the archived screenshot but present in the original. In this situation, we would like the pixel error for the gap to be present to let the information about a missing element in the archived screenshot be reflected in the pixel error. Not only missing elements' pixel errors would have been preserved, but also other errors, like a missing CSS file, defining the background color, are cases for which the pixel error should be preserved. This latter example of a missing CSS file demonstrates that using the most common color of the original screenshot instead of the archived screenshot would not

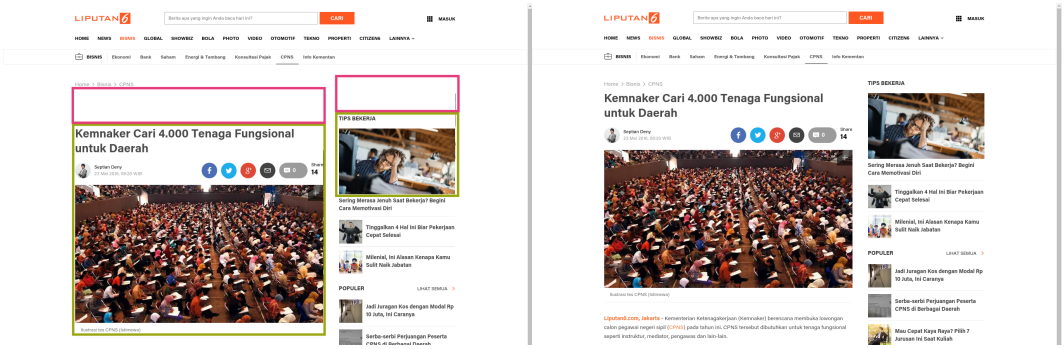
have been a better choice because then the information about a missing CSS file would have been overwritten in the reconstructed screenshot. (2) A gap is placed at a position where the archived screenshot and the original screenshot display the colors of the element that lay behind the shifted element. In this situation, we would like the pixel error for the gap to be absent in order to preserve the information that the translation error was the only problem for the pixels at the coordinates of the gap and that this problem was solved by shifting the element.

However, it is not an easy task to know which of the two scenarios is the case because, besides the position of the gap and the colors of the corresponding pixels in the original screenshot, we do not have any information on the gap. Therefore, we would need to derive the fitting scenario from other information: We found coloring the gap in the most common color of the archived screenshot a method with promising features for both scenarios. The most common color in the archived screenshot was likely also to be the most common color in the original screenshot, and therefore least likely to produce a pixel error in the gaps if scenario (2) was the case. We hoped that with this method, we would receive reconstructed versions of the archived screenshots in which gaps produced a pixel error at places where the archived screenshot had missing elements, like scenario (1) suggests.

A detailed description of this reconstruction method can be found in the appendix on page 115. An example of the reconstructed screenshot and corresponding archived screenshot is shown in figure 6.7. The gaps' positions in the reconstructed screenshot (figure 6.7a) are positions of elements that are missing in the archived screenshot. Here, the pixel error would not be zero at the positions of the gaps and the information of a missing element in the archived screenshot would be persistent in the reconstructed version of the screenshot. In contrast to the unprocessed archived web page's screenshot (6.7b), the pixels composing the shifted element would not produce an error when compared to the original screenshot, but the pixels contained in the gaps would. Under the assumption all shifted elements have been correctly re-positioned, the overall pixel error between this reconstructed version of the archived screenshot and the original screenshot would be an approximation of the errors produced by other error types than the translation error.

This reconstruction method had one major problem: When a gap was printed in the most common color in the archived screenshot but was positioned on an element, which was colored differently, the gap would always produce a pixel error, no matter if scenario (1) or (2) were the case.

Method 4: Most Frequent Color Around Gap in Archived Screenshot. To improve upon this problem, we implemented a context-dependent



(a) (b)

Figure 6.7: A reconstructed screenshot of the archived web page with pixels in the most common color in the archived screenshot at the gaps the shifted elements left behind (a). The pink rectangles in (a) mark the gap areas in which the pixels were colored in the most common color in the archived screenshot; the green rectangles mark the shifted elements which produced the gap.

method for computing the color of the gaps. We did this by coloring the pixels in the most common color, which was found around the gap in the archived screenshot. To do this, we decided to sample one row (respectively column) of pixels immediately bordering the gap (the bounding box pixels). Then we would color the pixels in the gap with the most common color in the sample. This process is illustrated in figure 6.8.

We put some thought into handling the gaps which lay at the borders of the archived screenshot, because obviously, for these pixels outside the border of the archived screenshot, we did not have color values. Our first intuition was to skip the sample for these edges and compute the most common color only for edges of the gap which lay inside the archived screenshot. However, we did not use this idea because it did not work for common cases where this scenario of a bordering gap would usually apply, for example if the entire HTML body element were shifted. In this case, the gap would be bordering only pixels outside the archived screenshot.

Instead of skipping the sample from the edges outside of the archived screenshot, we sampled the pixels composing the border of the shifted element itself to determine the color of the gap. We did this only for the edges around the gap which lay outside of the archived screenshot (the other edges we sampled as planned from the bounding box edges of the gap). This process is shown in figure 6.9. This way, we were able to sample a color not only for the gaps that were produced by a shifted HTML body element, but also in the other cases, where footer elements or header elements were shifted: Usually, only one edge of the bounding box of a shifted footer or header includes

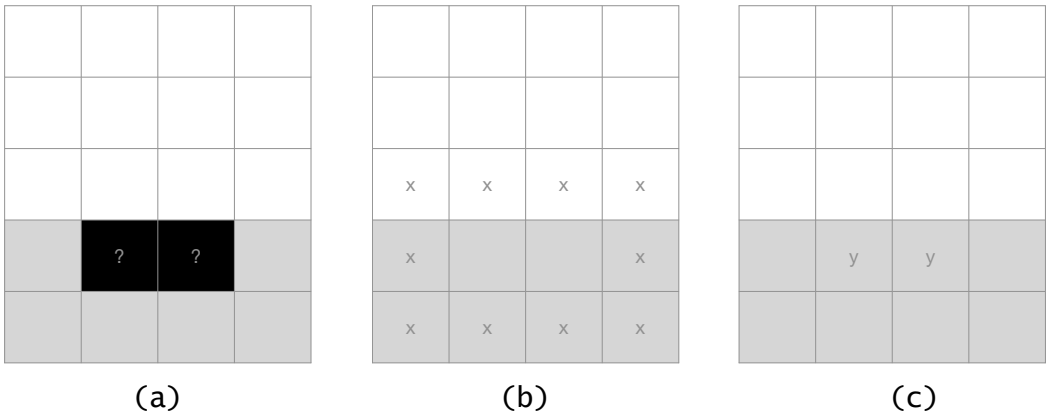


Figure 6.8: The figure shows an example for determining the color of the gap if the gap is not positioned at the borders of the screenshot. The position of the gap during the reconstruction process of archived screenshot is displayed in black (a). To determine the color the gap will have in the reconstructed version of the archived screenshot, the pixels around the gap marked with "x" are sampled as shown in (b). Pixels marked with "y" in (c) display the most frequent color among the sampled pixels "x". This will be the color of the gap in the reconstructed version of the archived screenshot.

pixels inside the bounds of the archived screenshot. Conversely, the edges of the shifted header or footer themselves are likely to hold the most valuable information on the surrounding color near the upper and lower borders of the archived screenshot.

A detailed description of this reconstruction method can be found in the appendix on page 117. An example of the reconstructed screenshot with the method of coloring the gaps according to the most common color around the gap is shown in figure 6.10a. Figure 6.10b shows the reconstructed screenshot produced by reconstruction method that colors the gap in the most common color in the archvied screenshot. One can see that the gaps in figure 6.10b were all colored white, but the colors of the gaps computed from the sample of pixels around the gap shown in figure 6.10a adapted to the environment of the gap. We expected this method used in 6.10a to produce a lesser pixel error than the method in 6.10b, but we will see later, when we present the results of the pixel error of each reconstruction method, that this was not the case.

6.3 Definition of the Reconstruction Error

We call the pixel error between the reconstructed screenshots and their corresponding original screenshots the *reconstruction error*. In contrast to the base-

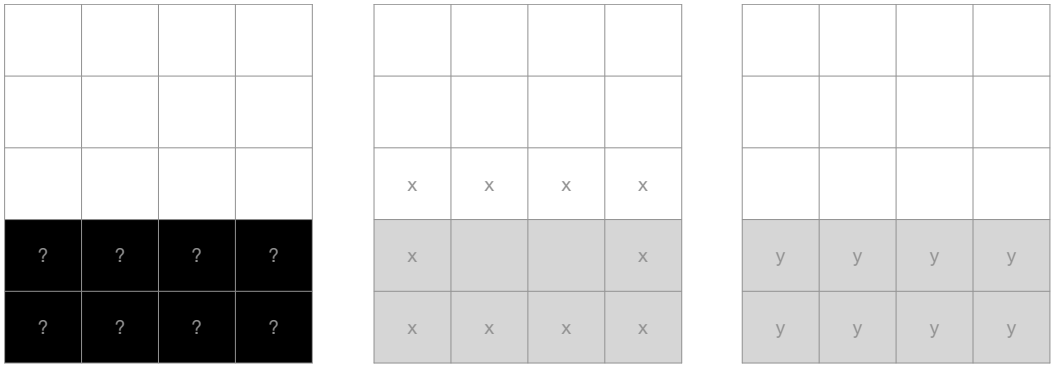


Figure 6.9: The figure shows an example for determining the color of the gap if the gap is positioned at the borders of the screenshot. The position of the gap during the reconstruction process of archived screenshot is displayed in black (a). To determine the color the gap will have in the reconstructed version of the archived screenshot, the pixels around the gap marked with "x" are sampled as shown in (b). Pixels marked with "y" in (c) display the most frequent color among the sampled pixels "x". This will be the color of the gap in the reconstructed version of the archived screenshot.

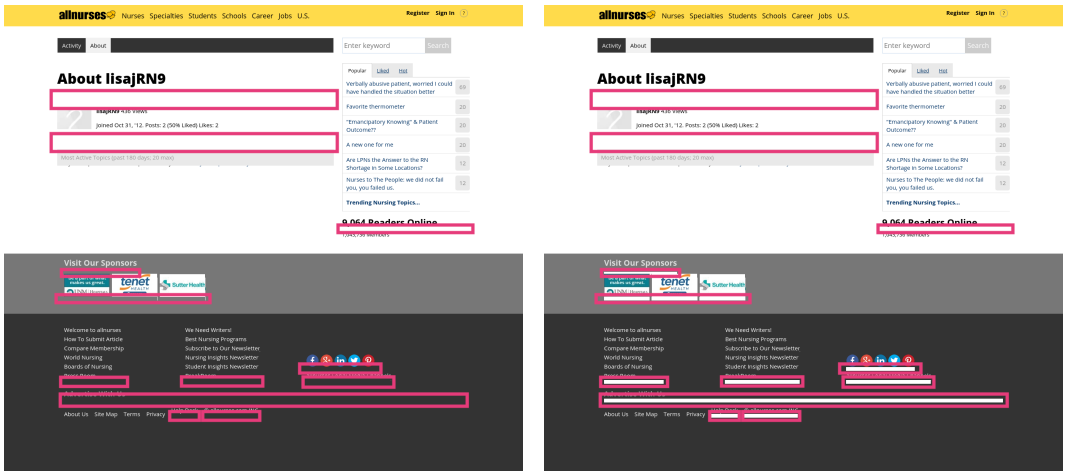


Figure 6.10: Two reconstructed screenshots. The reconstructed screenshot in (a) shows the output of the reconstruction method which colored each gap in the most common color of the pixels around the gap. The reconstructed screenshot in (b) shows the output of the reconstruction method which colored each gap in the most common color of the archived screenshot. The pink rectangles mark the gap areas. The gaps in (a) adapt to the color of their environment, while the gaps in (b) are all colored the same, but often match the background.

line error, which results from a comparison between the unprocessed archived screenshots and their corresponding original screenshots, the reconstruction error compares the reconstructed versions of the archived screenshots with the original screenshots. We computed this error as an adaptation of the edit distance as explained before, using our implementation of the Image Comparator, described in section 4.2.1, as we already did for the computation of the baseline error. The reconstructed screenshots are the output of the Image Reconstructor; these files serve as input for the Image Comparator, together with their corresponding original screenshots. The reconstruction error is the output of the Image Comparator and will later serve as part of the feature vector containing the independent variables for the model. The reconstruction error defines the pixel error of the reconstructed archived web pages' screenshots. Together with the baseline error (which is also part of the feature vector), the reconstruction error transports information on the changes that had been made on the archived web page during reconstruction. Especially the absolute and relative number of changed pixels will be transported by using both the baseline error and the reconstruction error as features in the model.

6.4 A Linear Regression Model to Assess the Quality of Archived Web Pages

In this section, we will describe the procedure of computing a linear regression model for our data set. This model correlates the human annotations about the qualities of the archived web pages to the data which we generated by reconstructing the archived web pages. We will first describe what was necessary to prepare the data for the regression, then we show how we generated the data set for the model. Finally, we will present how we computed the linear regression and which measures we used to derive information on the correlation's strength.

6.4.1 Preparing the Data Set for the Regression

The data that is used for linear regression consists of

- The baseline error (the differing pixels between the archived screenshots from webis-web-archive-18 and the original screenshots from webis-web-archive-17 computed by the Image Comparator)
- The reconstruction error (differing pixels between the reconstructed and original screenshots computed by the Image Comparator and based on the archived screenshots from webis-web-archive-18)

- The shifts detected by ffmpeg for each element on the basis of each archived web page from webis-web-archive-18
- The human annotation for each archived web page from webis-web-archive-17

Apples and Oranges. The data used for the linear regression ultimately came from two sources: Firstly, the screenshots of archived web pages that the human annotators saw originated from the data set webis-web-archive-17. Secondly, the data we based our reconstruction on—which resulted in the errors and shifts that should be the independent variables for the model—was from webis-web-archive-18. The two data sets were made from the same archived web pages, but the screenshots captured for webis-web-archive-17 were made in September 2017, while the screenshots of the archived web pages for webis-web-archive-18 were made in February 2018. Not only the time but also the proxies, which redirect the browser to the archive instead of an online web page, differed partly between the two data sets. The screenshots of the archived web pages that the human annotators saw (from webis-web-archive-17) were redirected by either one of the proxies `custom`, `warcprox` or `pywb`. In contrast, the proxy used for redirecting to the archived web pages in for webis-web-archive-18 was always `custom`. It was necessary to use different data sets because all the information needed was not present in just one: The reconstruction step needed the description of the structure of the HTML elements, which was generated only in February 2018 and therefore was absent in the earlier data set, which the human annotators based their judgments on. This means that the screenshots of the archived web pages that were assessed by the human annotators might have differed from the screenshots of the archived web pages, on which we based the detection of shifts and the reconstructions. Consequently, to not have the model "compare apples and oranges," we felt it was necessary to guarantee a certain similarity between the screenshots that were to be included in the model. When preparing the model, we decided to exclude those web pages that displayed a large pixel error: First, we manually reviewed some pairs of archived web pages and evaluated the relation between the error and the similarity of the screenshots from webis-web-archive-17, which the human annotators saw, and from webis-web-archive-18, which the reconstruction was based on. Supported by the outcome of this review, we decided that 5% differing pixels was an adequate threshold for the similarity between those two, as this amount of differing pixels was mostly not perceivable. This threshold allowed us to use about 65% of the data set for the regression. As illustrated in figure 6.11, the filtering step guaranteed that the difference between the archived screenshot used for the reconstruction and the archived screenshot

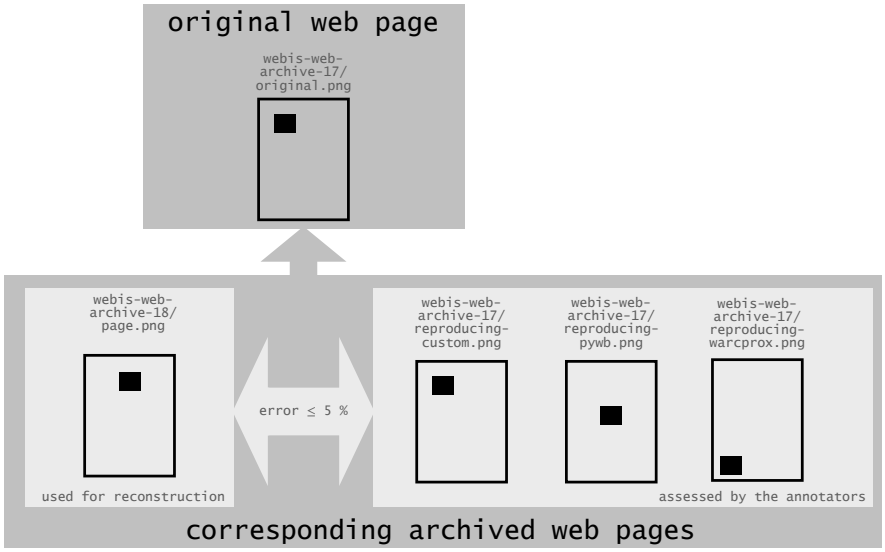


Figure 6.11: The threshold on pixel error allowed on the different data sets webis-web-archive-17 and webis-web-archive-18.

that was assessed by the human annotator never exceeded 5%. This way, we could guarantee that the quality score the human annotators assigned to an archived web page was also a valid quality score for the archived screenshot used for the reconstruction.

6.4.2 Generating the Data Set for the Model

For the computation of the linear regression, we used the reduced data set containing only the filtered web pages (as mentioned in the section before) about which we added the information from the four contexts: (1) the baseline error, (2) the reconstruction error, (3) the data on the shifts, and (4) the human annotations. Specifically, adding the information from the four contexts resulted in a vector X of independent variables and a vector y containing the dependent variable for each web page. An overview of the specific variables is shown in table 6.1.

We call all independent variables in table 6.1 that do not belong to the group "baseline error" *reconstruction data*. They were relevant to the reconstructed screenshots but not to the unprocessed archived screenshots. Furthermore, we need to point out that the *reconstruction error* is not equivalent, but only a subset of the *reconstruction data*. In the following, we will shortly detail the independent variables with regard to how they were obtained.

	Baseline Error: pixels added through insertion/deletion, total number of changed pixels, pixels total in original, changed pixels as percent of pixels in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image
	Reconstruction Error: pixels added through insertion/deletion, total number of changed pixels, pixels total in original, changed pixels as percent of pixels in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image
X	**Shifts General:** total number of elements on archived web page
	Shifts All: total number of shifted elements on archived web page, shifted elements to the top, to the top right, to the right, to the bottom right, to the bottom, to the bottom left, to the left, to the top left
	Shifts Small: total number of shifted elements on archived web page with small shift distance, shifted elements to the top, to the top right, to the right, to the bottom right, to the bottom, to the bottom left, to the left, to the top left
	Shifts Large: total number of shifted elements on archived web page with large shift distance, shifted elements to the top, to the top right, to the right, to the bottom right, to the bottom, to the bottom left, to the left, to the top left
y	**Human Annotation Data:** quality score of the archived web page

Table 6.1: Dependent and independent values for each archived web page to be used in the linear regression model.

The independent variables about the baseline and the reconstruction error were the output of the Image Comparator, which was further explained in section 4.2.1. We decided to use the reconstruction error the Image Comparator produced for the reconstructed screenshots whose gaps were colored the most common color occurring in the archived web page. This reconstruction error was the best choice because it seemed to have reduced the most noise as it produced the lowest pixel error among the different reconstruction methods (discussed in section 6.2.2).

While for the baseline and the reconstruction error, we counted differences in pixels, we chose to measure the data on the shifted elements as the number of elements (instead of the number of shifted pixels). Since the number of shifted pixels would implicitly be contained in the baseline and reconstruction error, we deemed the number of elements to provide further relevant information.

The data about the shifts was mostly gathered through ffmpeg's video encoding, which is used during the reconstruction step realized by the Image Reconstructor (see 6.2). As explained in section 6.2.1, using ffmpeg, we generated an encoded video for every web page. This video contains motion vectors for every pixel block. With the Image Reconstructor, we applied those motion vectors to elements on the archived web page and computed their new positions. For every element on every web page, we then noted its web page's ID, XPath, top left coordinates before the shift, bottom right coordinates before the shift, top left coordinates after the shift, and bottom right coordinates after the shift. From this data, we could deduce and store for every element the data on their shift distances and directions.

We decided to store the data on the shift distances for the elements on each web page for the eight directions top, top right, right, bottom right, bottom, bottom left, left, and top left. Also we grouped the shifted elements by their respective shift distances, setting up three groups (1) any shift distance, (2) small shift distances and (3) large shift distances. We counted the number of elements in each group per direction. The groups (2) small and (3) large shift distances were defined by the absolute number of pixels that elements were allowed to have shifted in either direction. Here, we defined distances larger than 5 px to the top or bottom to be large shifts and shorter shifts in these directions to be small. For the directions left and right, we defined shifts further than 8 px to be large and shifts shorter or equal to 8 px to be small. The thresholds for large or small shift distances were chosen with respect to the data we gathered on the frequencies of shifts, and partly, they were based on our intuition of how small or big the dimensions of an element containing important content could be. The idea behind grouping shift distances was motivated by the notion that a shift can be caused by missing elements: For example, a shift to the top could be caused by a missing element above the

shifted one, thereby connecting the translation error to the missing element error.[7] We suspected that elements with very tiny dimensions, either in width or in height, were likely to also cause tiny shifts in corresponding directions while likely not being relevant content of the web page. From the data on the frequency of the shift distances, we found out that in vertical direction, there was a small drop in frequency after the 5 px mark. As we suspected the height of elements that were unlikely to contain relevant content (or content at all) to be somewhere between 0 and 10 px, we took the drop in frequency after the ±5 px mark (in both directions) as a possible hint to draw the threshold for the small shifts there. For the horizontal shifts, we proceeded equally: We assumed the maximum width for an irrelevant element might be somewhere between 0 and 10 px and found a drop in frequency after ±8 px.[8] Conclusively, we assumed for an element with maximally 5 px height or 8 px width, it might not have contained important content and it missing should therefore not decrease the quality of the archived web page. The shift distances in the diagonal directions top right, bottom right, bottom left, and top left were only sorted into the categories small or large if both of the directions (top, left, bottom, right) fell into the same category. Obviously, this decision was necessary because a diagonal shift could have been triggered by two missing elements. If one was assumed to be large (possibly content relevant) and the other was tiny (possibly not content relevant), the corresponding diagonal shift might land in the wrong category no matter if sorted into the large or small shift group. However, it would not be missed in the data set because it would, in any case, show up in the category which counted all shifted elements—no matter their distance.

Finally, the independent variable *total number of elements on the page* is used to put the number of shifted elements into perspective. As those are absolute frequencies, we needed to take into account how many elements are found on the entire web page to access the relative number of shifted elements per page. (We assumed it would make a difference if a web page only consists of 5 elements and 3 of them were shifted, or if a web page consisted of 5 000 elements and 3 of them were shifted.) We computed the total number of elements on each page by counting the lines in the description of structure, which was included in the data set webis-web-archive-18 (see chapter 5); each line in the description of structure represents a distinct element on the archived web page.

[7] We have further explored this notion, and will present our findings later, in chapter 8 of this work, as they did not directly contribute to the implementation of the framework which we discuss here.

[8] We are aware that this drop in frequency does not define the threshold for all content relevant and irrelevant elements.

6.4.3 Designing the Linear Regression Model

Measuring the Strength of the Correlation. To measure the strength of the correlation, we used the R^2 score and the accuracy. The R^2 score is part of the sklearn library and implements the coefficient of determination through the following equation:

R^2 Score ([25]): *If $\hat{y}_i$ is the predicted value of the i-th sample and y_i is the corresponding true value for total n samples, the estimated R^2 is defined as:*

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

The R^2 score measures how well the hyperplane computed by the linear regression fits the data. The sklearn documentation states, "It represents the proportion of variance (of y) that has been explained by the independent variables in the model. It provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model, through the proportion of explained variance" [25]. The maximum value of the R^2 score is 1, which means that the predicted values were always correct; it can get infinitely worse, depending on how often and with which distances the predictions differed from the true values. If the model was always predicting the expected value of the true values, the R^2 score would be zero [25].

We measured the accuracy using the following equation:

Accuracy: *If $\hat{y}_i$ is the predicted value of the i-th sample and y_i is the corresponding true value for total n samples and $y_{correct}$ denotes a pair of $(\hat{y}_i, y_i)$, where $\hat{y}_i = y_i$, we define the accuracy of the model M as :*

$$Accuracy(M) = \frac{|y_{correct}|}{n}$$

We decided to use both—the accuracy and the R^2 score—because even if both measure the quality of the predictions, the R^2 score includes information on the distance between the predicted and the true quality, which the accuracy does not consider. This implies that the R^2 score would decrease more the larger the difference between the predictions and the true values were, while the accuracy would not be influenced by these distances.

Computing the Linear Regression Model. For the linear regression, we used Python's sklearn library (version 0.24.0) [21]. We implemented the standard 10-fold cross-validation, separating our data set into ten folds, of which one fold was used as testing set while the remaining nine were used as training set to fit the model. This process was iterated ten times with a different testing fold each time. This generated ten results (one for each iteration), which were then averaged to one single result, providing information on the strength of the correlation between the feature vectors and the quality of the archived web pages from the human annotations.

General Process: Baseline Regression and Reconstruction Regression. The general workflow of our regression analysis produced two models: (1) a linear regression model for the baseline error, which we call *baseline regression* from here on, and (2) a linear regression model for the reconstruction process, which will be called *reconstruction regression* from here on. We needed the results from the baseline regression as a comparison for the results we gained from the reconstruction regression in order to determine changes in the strength of the correlation to the human annotations. The feature vector for the reconstruction regression was already explained and shown in full length in table 6.1 in section 6.4.2. The feature vector for the baseline regression was a subset of the features for the reconstruction regression and was likewise listed in table 6.1.

We first needed to determine if the data set should be stratified by comparing the accuracy and R^2 scores of the regression model using stratified and not stratified data. Proceeding with the data set that provided the better results, we determined the best rounding method to use for the predicted qualities. This was necessary because the linear regression model does not predict categories but real numbers as qualities. If these qualities were compared to the true values, which are integer numbers (1 to 5), the accuracy would be very low. This problem was resolved by mapping—using a rounding method—the real number predictions to the true qualities. Then we composed varying combinations of features to determine which features result in the most accurate predictions about the quality of the archived web pages. Finally, we evaluated the effectiveness of the reconstruction approach to determine the quality of the archived web pages using the results from the features which provided the best predictions. All this is discussed in detail in the next chapter.

Chapter 7

Results of the Implementation of the Framework

In the previous chapter, we explained in detail the implementation of the general framework. We described our approach of reconstructing archived screenshots and computing the pixel error between an original screenshot and its corresponding archived screenshot or reconstructed screenshot. Also, we elaborated on the method of computing the correlation between the human annotations about the quality of the archived web pages and the data gained through the reconstruction and comparison process. In this chapter, we will present the results which were obtained from the implementation of the framework. These results comprise the baseline error (6.1), the reconstruction errors (6.3) for the different reconstruction methods (6.2), and the information we gained through the linear regression model (6.4).

7.1 Results for the Baseline Error

The baseline error represents the amount of differing pixels between the standard (non-reconstructed) archived screenshot and its corresponding original screenshot. We found a wide range of values for the baseline error, but it was very surprising that we found some archived web pages with a relative baseline error of even 100%. Moreover, we suspected some archived web pages to be completely identical to their corresponding original versions, but finding about 2 700 screenshots (so about 29%) to have a relative baseline error smaller than 1% was still more than we expected. We did expect more web pages to display a larger baseline error because we compared screenshots of archived web pages that were reproduced months after their corresponding original screenshots. As we discussed in chapter 5, errors can occur more frequently in an older archive. The exact distribution of the baseline error is shown in figure 7.1.

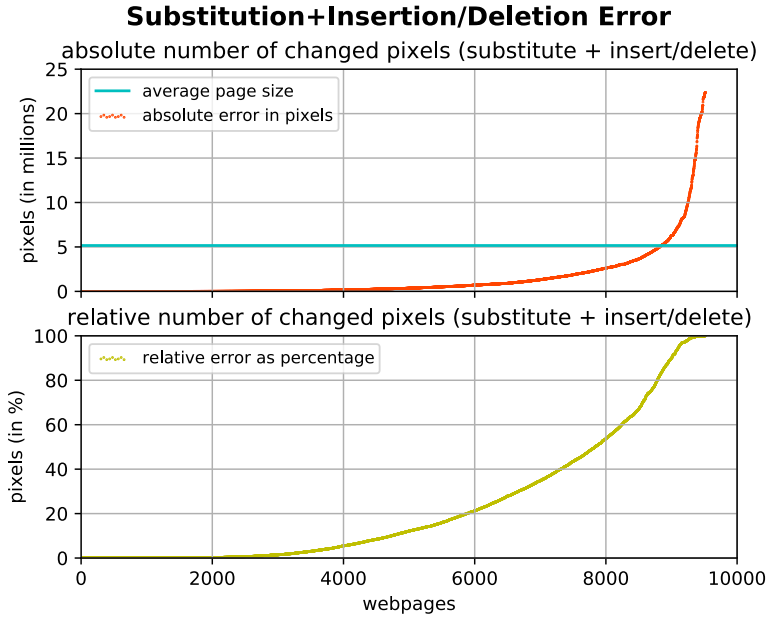


Figure 7.1: Absolute baseline error (red) and average archived pages' size (turquoise) in pixels (top) and the relative baseline error (olive) as percentage of pixels applied to the larger screenshot (archived or original) (bottom). The average size of the larger web page is 5 161 624 pixels.

The plot shows all 9 519 web pages from our data set sorted by increasing size of baseline error.

The average absolute error, which comprises the absolute number of inserted or deleted and substituted pixels, lies at 1 468 167 pixels, while the average page size is 5 161 360 pixels. The average relative error is the percentage of changed pixels applied to the bigger screenshot and measures 23.113%. The average values of the baseline are additionally displayed in table 7.1. Using the bigger screenshot as a reference for the relative error instead of the original screenshot was necessary because the archived and original screenshot can be of different heights. Here, it would have been possible that the original screenshot was smaller than the archived screenshot, which would result in a relative error exceeding 100%. The distribution of the baseline error shows a steep increase for about two thirds of the data set: While 50% of the web pages show a relative error smaller or equal to 10.26%, 70% of the web pages do not exceed the 30% mark of the relative error. There are 10% of web pages that have a relative error larger than 70.34%. Moreover, 5% of the web pages display an error larger than 91.71%, and 2% of the pages even exceed 99% relative error. Table 7.2 displays the quantile values in more detail.

Average Absolute Error	Average Page Size	Average Relative Error
1 468 167.263 px	5 161 360.102 px	23.113%

Table 7.1: Average baseline error.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 74.336%
20%	$\leq$ 0.206%	92%	$\leq$ 78.457%
30%	$\leq$ 1.187%	93%	$\leq$ 83.740%
40%	$\leq$ 4.292%	94%	$\leq$ 87.969%
50%	$\leq$ 10.261%	95%	$\leq$ 91.713%
60%	$\leq$ 18.161%	96%	$\leq$ 96.177%
70%	$\leq$ 29.967%	97%	$\leq$ 97.756%
80%	$\leq$ 45.399%	98%	$\leq$ 99.265%
90%	$\leq$ 70.345%	99%	$\leq$ 99.883%

Table 7.2: Distribution of the relative baseline error among the web pages through quantiles.

Method	Average Absolute Error	Average Page Size	Average Relative Error	Decrease in Baseline Error
none	1 468 167.263 px	5 161 360.102 px	23.113%	-
0	1 313 780.684 px	5 123 258.060 px	21.145%	1.968%
1	1 289 719.040 px		20.807%	2.306%
2	1 326 756.970 px		21.344%	1.769%
3	1 307 732.313 px		21.074%	2.039%
4	1 310 445.110 px		21.114%	1.999%

Table 7.3: Average absolute and relative pixel error, page size and decrease in pixel error for the baseline and the different reconstruction methods reconstruction with duplicates (0), upper bound (1), lower bound (2), most frequent color in archived screenshot (3), most frequent color among gaps' edges (4). Within the borders the upper and lower bound set, the most decrease in pixel error was produced by method 3.

7.2 Results for the Reconstruction Error

The reconstruction error represents the amount of differing pixels between the reconstructed screenshots and their corresponding original screenshots. We tested several methods of handling the gaps in the reconstructed versions of the archived screenshots, as we described in detail in section 6.2.2. In the following, we will discuss the main properties of the reconstruction errors for each reconstruction method, presented in table 7.3. This table and the following explanations will refer to the different reconstruction methods by number: Method 0 refers to reconstruction with duplicates, method 1 refers to the upper bound method of coloring the gaps, method 2 refers to the lower bound method of coloring the gaps, method 3 refers to coloring the gap in the most frequent color in the archived screenshot, and method 4 refers to coloring the gap in the most frequent color around the edges of the gap. We will present a detailed view of the reconstruction error that is used as feature in the linear regression. The detailed plots and tables with the results for the reconstruction errors produced by the other methods can be found in the appendix of this work. Afterward, we will elaborate on the general impact of the reconstruction on the pixel error compared to the baseline.

7.2.1 Reconstruction Errors by Reconstruction Method

Method 0: Reconstruction with Duplicates. Reconstructing the archived screenshot by translating copies of the shifted elements (method 0) produced duplicates of shifted elements. At positions where these duplicates were not hidden by other elements, they produced the same pixel error as the baseline error. Therefore, this method produces a reconstruction error which is most comparable to the baseline error regarding the improvements introduced through shifting elements, compared to the other methods which were manipulating the gaps.

Table 7.3 shows the average absolute error, average page size, average relative error, and decrease with respect to the baseline for the unprocessed and reconstructed screenshots. Compared to the baseline, the relative pixel error was decreased by about 1,97%. It is the lowest decrease in pixel error among all reconstruction methods[1]. This seems reasonable to us because pixels at positions of gaps that produced an error before the reconstruction (in the baseline) still produced an error afterward. Therefore this reconstruction method decreases the pixel error less than others.

Method 1: Upper Bound. Reconstructing the archived screenshot by translating the shifted elements and coloring every pixel in the gap in the color the original screenshot displayed at this position produced a reconstruction error, which we call the upper bound (method 1). Table 7.3 shows that the average absolute error now lies at 1 289 719 pixels, which means it is lower than the baseline and also lower than the reconstruction error produced by method 0. As intended, the decrease in pixel error of 2.306% compared to the baseline error is the largest among all reconstruction methods. Nevertheless, the decrease seems within a small range from the baseline error. From the relatively small impact the upper bound computation had on the error, we conclude one or both of two things: Firstly, we might not have detected enough translated elements to make a significant change, and secondly, there are many other error types in the screenshots that would need to be considered in other reconstructions in order to reduce the reconstruction error even more. The same explanations may account for the minor changes of baseline error produced by the lower bound method, which is explained next.

Method 2: Lower Bound. Reconstructing the archived screenshot by translating the shifted elements and coloring every pixel in the gap in a different color than the corresponding pixel in the original screenshot produced

[1]We do not consider the upper and the lower bound to be true reconstruction methods, because they produce pixel errors which cannot be used in the linear regression model.

a reconstruction error which we call the lower bound (method 2). Table 7.3 shows that the average absolute error now lies at 1 326 757 pixels, which means it is lower than the baseline error but higher than the pixel error produced by any other reconstruction method. This outcome was expected and intended, as method 2 was designed to produce the largest possible amount of differing pixels in the gaps compared to the original screenshot. However, similar to the results of the other methods, the general impact of the lower bound computation was relatively low. After having seen the results the upper bound reconstruction produced, which neither caused a large decrease of the baseline error, we were not suprised by this outcome.

Method 3: Most Frequent Color in Archived Screenshot. Reconstructing the archived screenshot by translating the shifted elements and coloring every pixel in the gap in the most common color in the archived screenshot (method 3), produced the most decrease in baseline error within the upper and the lower bound. This is the reason why we discuss the results of this method in more detail. Table 7.3 shows that the average absolute error resulting from this reconstruction method lies at 1 307 732 pixels. Because this method produced the highest decrease of 2.039% in pixel error compared to the baseline and did not use information from the original screenshot to fill the gaps, this reconstruction error served as an input feature for the linear regression later on.

Figure 7.2 shows that the overall distribution of the reconstruction error has barely changed compared to the baseline. From table 7.4, which shows the quantiles of the web pages which displayed certain amounts of pixel errors, we learn that the overall distribution of pixel errors is nearly the same as for the baseline. The decreased pixel error shows a larger effect on web pages which already had medium-sized pixel errors in the baseline, than on those web pages with very low or very high baseline errors. This could be explained if very large pixel errors for a web page were caused by other reproduction error types than translated elements. For example, a missing ad element that is accompanied by an overlay, which causes the web page to become entirely opaque[2], would cause a pixel error of 100%. However, it could not be fixed by translating shifted elements. Additionally, if there were shifted elements, they might have become undetectable for the video encoding because the pixel blocks would have been colored too differently to find corresponding ones. An example of a reproduction error type that could not be fixed through translating elements for web pages with very small baseline errors is the pixel error caused by color

[2]A specific form of the reproduction error type *color*.

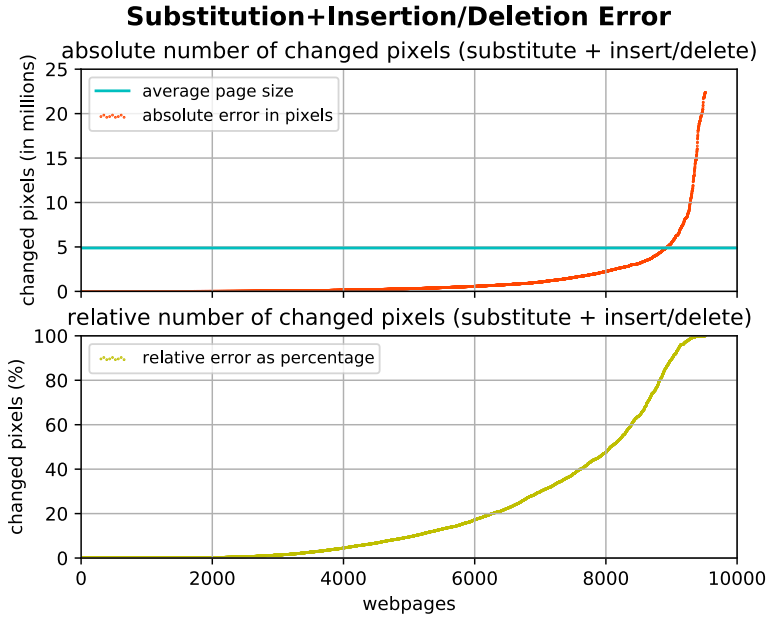


Figure 7.2: Absolute reconstruction error (red) and average reconstructed screenshot's size (turquoise) in pixels (top) and the relative reconstruction error (olive) as percentage of pixels in the original screenshot for reconstruction method 3 (bottom). The average size of a web page is 5 123 526 pixels.

differences in color transitions which were differently rendered, as discussed previously in section 5.2.

All in all, we were surprised by the outcome that method 3 performed best, as we would have expected the more dynamic method of coloring the gap in the most frequent color around the edges of the gap (method 4) to produce the lowest reconstruction error. We will elaborate on possible causes after presenting the results of method 4 in the following.

Method 4: Most Frequent Color Around Gap in Archived Screenshot. Reconstructing the archived screenshot by translating the shifted elements and coloring every pixel in the gap in the most common color found around the gap in the archived screenshot (method 4) produced a reconstruction error within the upper and the lower bound (similar to method 3). It did not use information from the original screenshot that could overwrite information about low-quality parts of the screenshot (like missing elements). Table 7.3 shows that the average absolute error now lies at 1 310 445 pixels which means it is higher than the ones produced by method 1 and method 3, and lower than the ones produced by the baseline, method 0, and method 2.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 71.319%
20%	$\leq$ 0.199%	92%	$\leq$ 76.063%
30%	$\leq$ 1.107%	93%	$\leq$ 81.512%
40%	$\leq$ 3.723%	94%	$\leq$ 87.058%
50%	$\leq$ 8.230%	95%	$\leq$ 91.488%
60%	$\leq$ 14.324%	96%	$\leq$ 95.777%
70%	$\leq$ 24.614%	97%	$\leq$ 97.572%
80%	$\leq$ 40.086%	98%	$\leq$ 99.265%
90%	$\leq$ 66.319%	99%	$\leq$ 99.883%

Table 7.4: Distribution of the relative reconstruction error among the web pages through quantiles for reconstruction method 3.

We were surprised by the result that method 4 produced a higher average reconstruction error than method 3 because, in contrast to method 3, this procedure of coloring the gap reacted dynamically to the color context around the gap. Method 3, on the other hand, did not sample the colors in the gaps' immediate surroundings but only used the overall most frequent color of the archived screenshot. An explanation for the worse performance of method 4 compared to method 3 might be found in the position of the gaps: A visual inspection of the screenshots has shown that gaps most commonly are placed near the screenshot's borders. Therefore the sampling process of method 4 must mostly rely only on the color that the shifted element itself provides, as there are no pixels outside the screenshot's borders. Resulting from this notion, we understand that method 3 is more stable in its performance regardless of the position of the gap.

7.2.2 General Impact of the Reconstruction on the Pixel Error

What first stands out when looking at the average page size shown in table 7.3 is that it has decreased for the reconstructed screenshots compared to the unprocessed screenshots. The reason for this decrease is that the reconstructed screenshots are all sized to the original screenshots' dimensions, cutting all

archived screenshots that are larger than the original, and consequently reducing the average size of the screenshots.[3]

The relatively minor changes to the distribution of the pixel error were surprising at first sight. As we found many shifted elements in the unprocessed archived screenshots from general visual inspections, we would have expected the impact the reconstruction would have on the pixel error to be larger. Table 7.3 displays the percentage by which the pixel error decreased compared to the baseline for all reconstruction methods. The maximum decrease is achieved by the upper bound (method 1), through which a 2.1% decrease of the baseline error was reached. The lowest decrease was gained through the lower bound (method 2), which reached a 1.7% decrease in average relative pixel error compared to the baseline.

Our intuition was that knowledge on the general number of pixels that were changed during the reconstruction could put the low decrease of pixel error into perspective. To find out the number of pixels that were changed through reconstruction, we computed the average percentage of differing pixels for all pairs of reconstructed screenshots and corresponding archived screenshots. Because the reconstruction methods differed, we decided to compute this percentage for the two methods that would yield the largest difference considering this percentage: Method 0 would provide insight on the amount of pixels that were changed due to the translation alone while method 2 would provide insight on the number of pixels that were changed due to the translation including the coloring of the gaps.[4] Through this comparison, we found out that about 13% of the pixels in the reconstructed versions of the archived screenshots differed from the unprocessed archived screenshots. More precisely, 13.39% of the pixels were changed on average by method 0, and 13.91% were changed on average when using method 2. Resulting from this, we learned that while we changed a large number of pixels (13%) in the reconstruction, we gained a much less reduction of the pixel error (maximum 2.306%). We believe the main reasons for this imbalance to be the following: First, the reconstructed screenshots contain gaps, either filled with a specific color or a copy of the shifted element, which differ from the original and therefore add to the pixel

[3]The average page size of original and archived screenshot pairs is only computed on the larger ones to be coherent with the relative pixel error, which is also computed in relation to the screenshots with the larger dimensions.

[4]Method 0, which produced duplicates, did not differ from the unprocessed archived screenshot at the positions of the gaps, therefore a comparison at the positions of the gaps would result in zero differing pixels. On the other hand, a comparison with the lower bound (method 2) would most likely approximate the highest amount of difference, because the gap was colored in two rare colors. Compared to all other methods, we hoped with method 2 the pixel error would also be largest when compared with the unprocessed archived screenshot and not only when compared to the original (as per design).

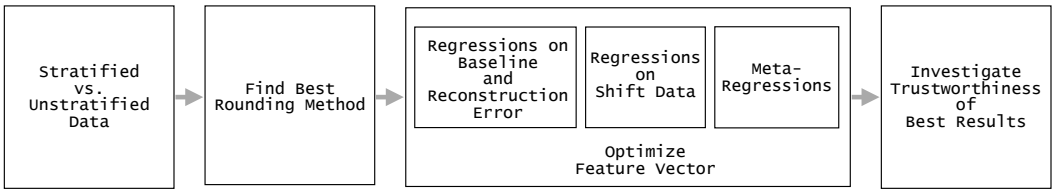


Figure 7.3: The process of the experiments which are presented and evaluated in this section. The setup of each experiment's best result will be the input setup for the next one in the framework.

error. A second reason could lie in the majority voting of the shift detection: The Image Reconstructor moves an element if two thirds of its motion vectors point to the same position. This can cause elements that contain shifted elements to likewise—but falsely—be translated as well.

7.3 Assessing the Quality of Archived Web Pages with a Linear Regression Model

In order to find out how reconstructing the archived web pages' screenshots influenced the correlation between the human annotations and the quality of the archived web pages, we conducted multiple experiments, which will be discussed in this section. The experiments build on each other: The setup that produces the best results will be used in the next experiment. Figure 7.3 displays this process to give an overview of this process and this section's structure. Each experiment will be described and evaluated in the following sections.

7.3.1 Testing Linear Regression on Stratified and Unstratified Data

As explained in Section 6.4.3, we performed the linear regression by using 10-fold cross-validation. We stratified all folds using the standard method of random oversampling in order to even out the appearances of true qualities in the training and test data. To implement random oversampling, we computed the most frequent quality for each fold. For each quality score that was less frequent, we duplicated random entries with said quality score until it occurred as frequently as the most frequent quality. The frequencies of all quality scores in the unstratified data set are displayed in table 7.5. The best quality score 1 is the most frequent, followed by the quality score 2. The worse quality scores

Quality Score	1	2	3	4	5
Pages	3 821	2 038	235	231	206

Table 7.5: Frequency for each quality score in the unstratified data set.

3, 4, and 5 are underrepresented, making up only about 3% each of all web pages in the data set.

Table 7.6 shows the R^2 score for the baseline regression and reconstruction regression if (1) the data was stratified using the standard method of random oversampling or (2) the data was kept in its regular state (unstratified). To clarify the configuration of the regression and processing of the data, we also added the information that no rounding method was used, even if this configuration will only be the subject of a later experiment.[5] We were surprised by the fact that the stratified data produced a worse R^2 score than the unstratified data. We suspect this to be caused by the very large difference in frequency of the quality scores 1 and 2 and the worse scores 3, 4, and 5. As mentioned before, the scores 3, 4, and 5 each make up only about 3% of the web pages in the data set. Therefore, stratifying the set with the random oversampling method might have caused too many copies of the same entry. Using undersampling as a stratification method was not an option, unfortunately, because then our data set would have decreased to only 1 030 entries, which we consider too small to produce representative results.

Another unexpected observation was that the R^2 score of the baseline error was better than the one of the reconstruction regression: The main intention of this work is to make better predictions about the quality of the archived web pages using a less noisy pixel error gained through reconstructing the archived screenshots. We would have hoped for the outcome of the reconstruction regression to be better than the outcome of the baseline regression. However, we expected that only a subset of features from the reconstruction regression introduced a bad influence on the correlation to the human annotations. To find out which of the features in the reconstruction data results in a decrease of the strength of the correlation compared the baseline regression, we isolated multiple features from the reconstruction regression, and tested their predic-

[5] As explained earlier, the accuracy for the results is irrelevant and was therefore not considered because the linear regression model does not predict categories but real numbers as values as qualities. If these quality scores were compared to the true qualities, which are integer numbers (1 to 5), the accuracy would be very low.

Stratified Data	Rounding Method	R^2 Baseline	R^2 Reconstruction
yes	none	0.267	0.244
no	none	0.382	0.365

Table 7.6: Best and worst R^2 scores and accuracies of the reconstruction regression and the baseline regression on stratified and unstratified data. The stratified data set produced a worse R^2 score than the unstratified data set.

tion power by combining different features. This will be evaluated later in this chapter.

The R^2 score of the baseline regression in table 7.6 shows a value of 0.365, which is a rather weak score. Yet, we want to reiterate that the lowest value for the R^2 score is not 0.0: This score would be reached by a model which would always output the expected value. Considering that the R^2 score can reach values infinitely smaller than 0.0, the score of the baseline regression reaching 0.365 might be interpreted as acceptable in terms of strength of correlation, even if it could be better.

7.3.2 Testing Different Rounding Methods on the Predicted Values

Because the linear regression predicted quality scores, which were real numbers, we needed to round the predicted values to integer numbers in order to be able to compute the accuracy. We computed the R^2 score and accuracy for the baseline regression and the reconstruction regression with each of the following rounding methods using the unstratified instead of the stratified data set because it had the better R^2 score:

- none: use no rounding method
- round: round up to the next integer if the first decimal place is at least .5 else round down
- ceil: always round up
- floor: always round down
- int: cut the number at the integer place[6]

[6]The rounding methods int and floor differ only for negative numbers, which we found among the predicted values.

Stratified Data	Rounding Method	R^2 Baseline	R^2 Reconstruction	Accuracy Baseline	Accuracy Reconstruction
no	none	0.382	0.365	0.0	0.0
no	round	0.306	0.291	0.720	0.721
no	ceil	−0.174	−0.185	0.230	0.239
no	floor	0.123	0.120	0.673	0.667
no	int	0.123	0.121	0.673	0.667
no	int + 1	−0.174	−0.184	0.230	0.239

Table 7.7: Accuracy and R^2 score of the reconstruction regression and the baseline regression for different rounding methods used on the predicted quality scores.

- int + 1: cut the number at the integer place and add 1 (because we noticed that the highest quality was almost never predicted)

The results in table 7.7 show that the best rounding method is *round*, but even this method shows a lesser R^2 score than the regression that used no rounding method. This is not surprising, considering the R^2 score takes differences between the true and predicted values into account. That is, the larger these differences are, the more the R^2 score drops. Obviously, these differences between the true and predicted values are caused by the linear regression because the linear function outputs real numbers. In contrast, the domain of the true values are integers, and rounding can make the difference between a false prediction and the true value even larger.

Of course, rounding causes positive effects on the accuracy if the right method is chosen: The accuracy of unrounded predicted values is 0, but rounding with the best method *round* increased the percentage of correct predictions to 72.0% for the baseline regression and 72.1% for the reconstruction regression. Rounding with *ceil* and *int + 1* performed worst, reaching an accuracy of only 23%. We think a reason for the bad performance of these rounding methods is that the models' predictions tended to be too high, rather than too low in the first place, as the far better (but still mediocre) results of the performance of the rounding methods *floor* and *int* suggest.

In the next section, we will evaluate the influence of different features and combinations on the prediction accuracy and R^2 score.

7.3.3 Testing Different Sets of Input Features

Previously, we computed the linear regressions for the baseline error and for the reconstruction data using all available features for each regression. Now, we explore the effects on the outcome of the linear regressions using only subsets of the available features, in order to find combinations that would produce better results. The experiments in which features of the baseline regression were changed were conducted for both the baseline regression and the reconstruction regression. Obviously, those experiments which only concern features from the reconstruction regression were not conducted as a baseline regression because they would have had no effect there.

We will evaluate the influence of different features and combinations of features on the prediction accuracy and R^2 score. All experiments will be explained in the following, but only the most interesting results will be presented in this chapter; the other results can be found in the appendix. Because the following experiments included at least 64 linear regression computations each, the tables will only display the best result regarding the R^2 score, the best result regarding the accuracy, and likewise the worst result regarding the R^2 score, and the worst result regarding the accuracy. All tests were made using unstratified data and the rounding method *round* because these parameters showed the best result in the experiments before.

Overview of the Experiments.

- **Regressions on Baseline Error and Reconstruction Error**
  We combined all features from the baseline error (A) and the reconstruction error (B). In total, 64 regressions were performed in (A) as well as in (B), and each regression was based on a different set of features. We computed the accuracy and the R^2 score for each of the combinations to determine the feature (or combination of features) that produces the best results concerning the correlation between the human annotations and the reconstruction error. Experiment (B) additionally provided information about the differences in the power of prediction regarding equivalent features in the baseline error. The features used for the regressions in this experiment were:

  (A) Regressions on each item in the powerset from the features of the baseline error:

  * X_{baseline}
    - Powerset(baseline error): a different item for each regression

    * $X_{reconstruction}$
        - Powerset(baseline error): a different item for each regression
        - reconstruction data (reconstruction error + data on the shifted elements)

  (B) Regressions on each item in the powerset from the features of the reconstruction error

    * $X_{reconstruction}$
        - Powerset(reconstruction error): a different item for each regression

- **Regressions on Shift Data**
  The data about the number of shifted elements used in the reconstruction error was divided into three groups: (1) total number of shifted elements separated by direction, (2) number of shifted elements separated by direction with small shift distances, and (3) number of shifted elements separated by direction with large shift distances. In the following experiment, we computed the reconstruction regression using the features which composed group (1) (experiment (C)), group (2) (experiment (D)), and group (3) (experiment (E)). Through these experiments, we wanted to find out which feature or set of features in the shift data was the best predictor. We chose to additionally use the total number of elements on the web page as a feature in each experiment in order to relate the absolute number of shifted elements counted by the features in this group to the total number of elements on the page in some combinations. This resulted in 2^{10} regressions per experiment because there were nine features in each group and one feature in *Total Number of Elements*. The features used for the regressions in this experiment were:

  (C) Regressions on each item in the powerset from the features of the total number of shifted elements (without regarding shift distances)

    * $X_{reconstruction}$
        - Powerset(shifts all + total number of elements on the archived web page): a different item for each regression

  (D) Regressions on each item in the powerset from the features of the number of shifted elements with small shift distances

    * $X_{reconstruction}$
        - Powerset(shifts small + total number of elements on the archived web page): a different item for each regression

(E) Regressions on each item in the powerset from the features of the number of shifted elements with large shift distances

* $X_{\text{reconstruction}}$
  - Powerset(shifts large + total number of elements on the archived web page): a different item for each regression

- **Meta-Regressions**
  The features in the reconstruction data can be divided into contextual groups: (1) baseline error, (2) reconstruction error, (3) shifts general, (4) shifts all, (5) shifts small, and (6) shifts large. (see table 6.1 in section 6.4.2.) In experiment (F), we computed the powerset of all of these groups. Then, we computed the reconstruction regression for the complete set of features in each subset of the powerset of these groups. This resulted in 64 reconstruction regressions.[7]

  In experiment (G), we combined the knowledge about the best groups gained from experiment (F) with the knowledge about the best individual features per group: We computed the reconstruction regressions with the features that produced the best R^2 scores and accuracies, gained through experiments (A), (B), (C), (D) and (E).[8] The feature vectors for these experiments were constructed as follows:

  (F) Regressions on each item in the powerset from the features of the reconstruction data grouped by context

  * $X_{\text{reconstruction}}$
    - Powerset([(1),(2),(3),(4),(5),(6)]): a different item for each regression, but using all features from each group

  (G) Reconstruction regression using the combination of the best features of the best groups

  Best R^2 score regression:

[7] Of course, two reconstruction regressions were superfluous: One being computed on the empty set of features and the other being computed only on the baseline error, which would be equivalent to the baseline regression on the full set of features, which was already computed in (A). Therefore only 62 reconstruction regressions of this experiment were of interest.

[8] We conducted this experiment knowing that features which perform well if isolated do not necessarily perform equally good or better if grouped, because they influence each other. Nevertheless, we conducted this experiment, because we did not want to miss an easy opportunity to find a better result than the ones produced by the previous experiments.

  - $X_{reconstruction}$
    - Powerset([max_{R^2}(group (1)), max_{R^2}(group (2)), max_{R^2}(group (3)), max_{R^2}(group (4))]): using only the features with the highest R^2 score from each group (1), (2), (3) and (4)

  Best accuracy regression:

  - $X_{reconstruction}$
    - Powerset([$max_{accuracy}$ (group (2)), $max_{accuracy}$ (group (5))]): using only the features with the highest accuracy from each group (2), and (5)

Discussion of the Experiments' Results. In the following, we will discuss the results obtained by the above experiments.

- **(A) and (B) Regressions on Baseline and Reconstruction Error**

  Table 7.8 shows the features from the baseline error that resulted in the best and worst R^2 score and accuracy for the baseline regression in experiment (A). One can see that the features which caused the best R^2 score differed from the features which caused the best accuracy. Likewise, the features differed for the worst results. This can be explained by the different handling of errors: While the R^2 score takes the distance of an incorrect prediction into account, the accuracy weights all incorrect predictions the same independently of their distance to the true value. This means that a particular set of features that gained the best R^2 score did not necessarily cause the highest number of correct predictions. Additionally it means, that the incorrect predictions were not as far from the true values as the incorrect predictions caused by the set of features that produced the highest accuracy. For the set of features which caused the highest accuracy, one can say it achieved the highest number of accurate predictions. (In reverse, this observation can be applied to the worst R^2 score and accuracy.) From this table, we also learn that the best and the worst results differ a lot for the R^2 score and for the accuracy, which suggests that varying the features has a large impact on the strength of the correlation for the baseline regression.

  In contrast, the results of the reconstruction regression were not so much impacted by varying the features of the baseline error because neither accuracy nor R^2 score showed large ranges from best to worst result. This might be explained by the fact that the reconstruction regression uses many other features to determine its outcome and is, therefore, less impacted by changes of the subset of baseline features. (The exact results

ID	Used Features	R^2 Score	Accuracy
A1B *best* R^2	**Baseline Error:** pixels added through insertion/deletion, total number of changed pixels, number of pixels in bigger image, changed pixels as percent of pixels in bigger image	**0.309**	0.721
A2B *worst* R^2	**Baseline Error:** pixels total in original	**-0.182**	0.354
A3B *best Accuracy*	**Baseline Error:** pixels added through insertion/deletion, total number of changed pixels	0.217	**0.748**
A4B *worst Accuracy*	**Baseline Error:** pixels added through insertion/deletion, pixels total in original, number of pixels in bigger image	-0.156	**0.315**

Table 7.8: Best and worst R^2 scores and accuracies of the baseline regression, which was computed for all combinations of features from the baseline error.

for this experiment's reconstruction regression can be found in table B.5 on page 125 in the appendix.)

Table 7.9 shows the features which resulted in the best and worst R^2 score and accuracy for the reconstruction regression (B). The worst results on both the accuracy and the R^2 score were produced using only the feature 'Reconstruction Error: pixels added through insertion/deletion'. The reason behind this is clear: The insertion and deletion was only performed on screenshots with unequal heights (the widths were always the same). As the reconstruction error was only computed on reconstructed and corresponding original screenshots, the insertion-deletion-error was always zero because the reconstructed screenshots were resized to the original screenshot's size. Regarding the features which produced the best R^2 scores and accuracy, one can see that they differ. The best R^2 score is produced by 'Reconstruction Error: pixels total in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image' while the best accuracy was triggered through 'Reconstruction Error: total number of changed pixels'. Clearly, both sets of features include information on how many pixels were changed. Nevertheless, it is surprising that the accuracy was highest using only the absolute number of changed pixels without any reference to the total amount of pixels in

ID	Used Features	R^2 Score	Accuracy
B1 *best* R^2	**Reconstruction Error:** pixels total in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image	**0.306**	0.728
B2 *worst* R^2	**Reconstruction Error:** pixels added through insertion/deletion	**-0.190**	0.313
B3 *best Accuracy*	**Reconstruction Error:** total number of changed pixels	0.251	**0.766**
B4 *worst Accuracy*	**Reconstruction Error:** pixels added through insertion/deletion	-0.190	**0.313**

Table 7.9: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of features from the reconstruction error while no other features belonging to the reconstruction data were used.

the screenshots. We would have expected the total number of changed pixels to transport more information in combination with the size of the screenshot the changes were performed on because the number of pixels in the screenshots varied largely. The same unexpected result is to be seen on the features which produced the best R^2 score. The number of changed pixels is contained as both, an absolute and a relative number. Indeed, we would have expected the percentage alone to contain enough information about the relation between the size of the screenshot and the changed pixels in it.

- **(C), (D) and (E) Regressions on Shift Data**

  In general, the results of the experiments (C), (D), and (E) were not good at all and very similar to each other. A detailed view of the best and worst results is presented in the appendix on tables B.6, B.7, and B.8 starting on page 126. We will present them only shortly here: The best R^2 score of all experiments (C), (D), and (E) is smaller than zero. This means that the predictions were worse than always predicting the expected value no matter the input. The best accuracy is of the three experiments is always the same and is equal to the worst accuracy from experiment (B). For all three experiments, the best and worst accuracies are very close together; the difference of 0.004 is the maximum difference and was produced by experiment (C).

The shift data did not seem to provide usable results when isolated. However, we have to point out that the bad performance of using just one group of features does not mean that the entire group was useless considering its power of prediction in general—it could be that a group of features that performs poorly on its own could be an excellent indicator of the quality when combined with another group of features. The combination between groups of features will be the subject in the next section.

- **(F) and (G) Meta-Regressions**

  Table 7.10 displays the results from experiment (F)—the best and worst combination of groups of features that were examined individually in the experiments before. The features from the groups were not varied while the powerset of the groups was iterated (but the table only displays the group names). The complete set of features in each group can be found in table 6.1 in section 6.4.2. In this experiment, the worst accuracy and worst R^2 score were both produced by the same combination of groups of features. While the input of the number of shifted elements with large shift distances (group (6)) seemed to have a negative influence, because it did only appear in connection with the worst accuracy and the worst R^2 score, the influence of the groups (4) Shifts All and (5) Shifts Small did not appear clearly positive or negative: Both of these groups appeared in the best and worst contexts. Group (1) Baseline Error and group (2) Reconstruction Error appeared only in good results. While the reconstruction error appeared in both the best accuracy and best R^2 score, group (1) Baseline Error was only part of the combination of groups of features that produced the best R^2 score.

  For the negative influence of group (6), we find two possible explanations: The first explanation is that we might have chosen a wrong threshold for the definition of large shifts. The second explanation might be that there simply is no connection between large shifts and the quality of the archived web page. It seems reasonable for us that good results for the R^2 score are gained through a combination of group (1) Baseline Error and group (2) the Reconstruction Error, even though we wonder why this combination did not also work well regarding the accuracy. We find the good result for these two groups reasonable because the difference between the baseline error and the reconstruction error bears information on how many pixels were shifted and therefore provides information about how the state of the screenshots was before and after the reconstruction.

ID	Used Features	R^2 Score	Accuracy
F1 *best* R^2	(1) Baseline Error, (2) Reconstruction Error, (3) Shifts General, (4) Shifts All	**0.314**	0.725
F2 *worst* R^2	(4) Shifts All, (5) Shifts Small, (6) Shifts Large	**-0.267**	0.308
F3 *best Accuracy*	(2) Reconstruction Error, (5) Shifts Small	0.306	**0.726**
F4 *worst Accuracy*	(4) Shifts All, (5) Shifts Small, (6) Shifts Large	-0.267	**0.308**

Table 7.10: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of groups of features. For each group all features were used.

In experiment (G), we hoped using the selected features would provide better results than the other experiments, but this was not the case. We were not surprised by this outcome, as the features can influence the predictive power of one another so that even features that perform badly if isolated can increase their power of prediction if grouped with other features. This experiment has shown that vice versa, this mutual dependency affects features that perform well if isolated. The exact results of experiment (G) can be found in the appendix on page 129 in table B.9.

To summarize, the overall best results for accuracy and R^2 score were both obtained from the reconstruction regression. The highest R^2 score was obtained by experiment F1 from experiment (F), which tested the combination of groups of features. The highest accuracy was obtained through experiment B3 from experiment (B), in which all combinations of features from the reconstruction error were tested. We have summarized the best results obtained by the baseline regression and reconstruction regression in table 7.14 for further investigation of the results of the linear regression, which we will proceed with later in this chapter.

Investigating the Trustworthiness of the Absolute Features. It was surprising that experiment B3 for the reconstruction and A3B for the baseline regression provided the highest accuracy, not only because they were using very few features but also because these features were absolute measures. The

reconstruction and baseline error also provided the number of changed pixels as a percentage with respect to the number of pixels in the screenshot. We were expecting a relative measure to be more predictive than an absolute measure because we thought it would make a difference in quality if all pixels or just a few pixels of an entire web page were wrong. To further inspect the trustworthiness of these absolute features, we examined their performance on isolated groups of small, medium, and large web pages. We believed that if the results from the reconstruction regression were constantly better than the results for the baseline, it would indicate the reconstruction feature was not generating random results.

Therefore, we divided the test data into three different size classes so that each of the classes contained approximately the same number of web pages. Figure 7.4 shows the distribution of the web pages' sizes when no grouping was applied (figure B.6 on page 131 in the appendix shows the distribution of web pages in each size class after grouping). Figure 7.4 shows that grouping was necessary to get equally large size classes because the amount of smaller web pages was larger than the amount of larger web pages in each class if no grouping was applied. Unequally large groups could have lead to less representative results when testing the linear regression model because the risk of only having unrepresentative test cases would be larger compared to size classes with higher sample sizes (more web pages). The small group contains web pages with a total amount of pixels up to 2 300 000 pixels, medium considers all web pages larger than small with a total amount of up to 4 800 000 pixels, and all other web pages were categorized as large.

With our model, we predicted the quality for the archived web pages, using only the feature 'Reconstruction Error: total number of changed pixels' to examine the accuracy of the predictions compared to the predictions that were made using only the baseline features with the best accuracy (gained through the features 'Baseline Error: pixels added through insertion/deletion, total number of changed pixels' in experiment A1B). The accuracies for every combination of the three size classes small, medium, and large are presented in table 7.11. The results show that the accuracy gained by the reconstruction feature was always higher than the accuracy gained by the baseline features. For both reconstruction and baseline regression, the models' predictions became worse with increasing page sizes. The decrease in prediction accuracy could indicate that the absolute features perform worse on larger web pages due to their absolute measure. But as shown for the accuracy of the reconstruction regression in table 7.12, the relative feature 'Reconstruction Error: changed pixels as percent of pixels in bigger image' also performs worse the larger the web pages are.

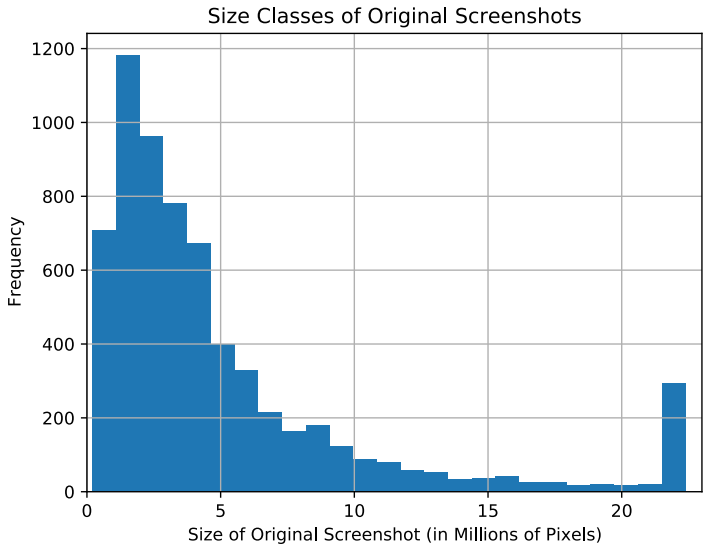


Figure 7.4: Frequencies of web pages with a certain size.

This leads to the understanding that there might be a different reason for inaccurate predictions on larger web pages than the predictive power of the examined absolute features. We argue that the worse accuracy of predictions about larger web pages does not so much depend on the power of prediction of the absolute features but rather on factors in the data set: Figure 7.5 shows the frequencies of certain quality scores for each size class of web pages. Among the group of small web pages are much more web pages with qualities of 1 and 2 than web pages with worse qualities. Among the group of medium and large web pages, the quality scores 1 and 2 are still the most frequent, but the difference between the frequencies of these web pages and the frequencies of the web pages with qualities 3, 4, and 5 is less than in the group of small web pages, while the group of large web pages contains the least difference in frequencies concerning the good and bad qualities. In order to verify or discard the intuition that the decrease in accuracy might not be triggered by the incapability of the absolute features to react to page sizes, but rather by the varying difficulty to predict certain qualities, we tested the performance of our model on different quality scores separately for each size class. The results, which are displayed in table 7.13, agree with the hypothesis that the accuracy was better for the groups with small and medium-sized web pages, only because they contain far more web pages with qualities scores of 1 and 2 than web pages with quality scores 3 to 5. The accuracy becomes very bad in general if qualities 1 and 2 are excluded from the test sets for every group.

Used Size Classes	Accuracy Baseline (A3B)	Accuracy Reconstruction (B3)
small	0.785	0.804
medium	0.640	0.642
large	0.349	0.371

Table 7.11: Accuracy of the reconstruction regression and baseline regression for combinations of the three size classes small, medium and large.

Used Size Classes	Accuracy Reconstruction	Used Feature
small	0.776	**Reconstruction Error:** changed pixels as percent of pixels in bigger image
medium	0.445	
large	0.328	
small and medium	0.748	
small and large	0.749	
medium and large	0.395	

Table 7.12: Accuracy of the reconstruction regression for combinations of the three size classes small, medium and large under use of the relative feature Reconstruction Error: changed pixels as percent of pixels in bigger image.

However, it is interesting to see that it performs worst for the small web pages, which received the best accuracy among the groups before removing web pages with good qualities. Better results are gained with the group containing the large web pages. We understand that this suggests that the accuracy of the predictions made by the absolute measure is valid and that the decreased accuracy on larger web pages was caused by the smaller amount of web pages with qualities of 1 and 2, which are easier to predict.[9]

Having found a trustworthy combination of features for the reconstruction error (in experiment F1 and B3), we will now evaluate our model by comparing the R^2 score and accuracy of the reconstruction regression with the respective results from the baseline regression. From the baseline regression we will con-

[9]One reason for an easier prediction of web pages with quality 1 is that for the cases where the pixel error is 0, the quality must be 1 because the archived (or reconstructed) and original web page's screenshots do not differ at all. The quality is more difficult to determine if the pixel error is different from zero.

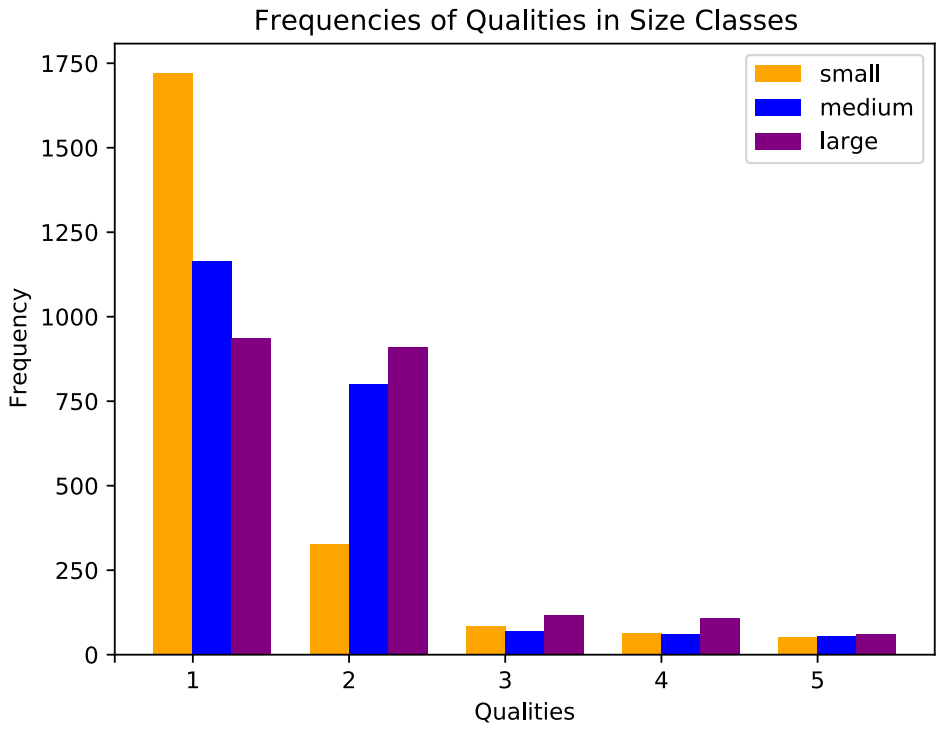


Figure 7.5: Frequencies of qualities of archived web pages in the size classes of the archived web pages.

Size Class	Accuracy for Quality Score	
	1+2	3+4+5
Small Baseline	0.835	0.0
Small Reconstruction	0.855	0.0
Medium Baseline	0.875	0.0
Medium Reconstruction	0.878	0.0
Large Baseline	0.481	0.238
Large Reconstruction	0.516	0.254

Table 7.13: Accuracy of the reconstruction regression and baseline regression for combinations of the three size classes small, medium and large and with reduced test data, containing either exclusively web pages with qualities 1 or 2 or exclusively web pages with qualities 3, 4 or 5.

sider the results of experiment A1B and A3B in table 7.8, which performed best regarding the accuracy and R^2 score.

7.3.4 Comparing the Best Baseline Regression and Best Reconstruction Regression

Comparing the best results reached by the baseline regression with the best results reached by the reconstruction regression, summarized and presented in table 7.14, we find that if the right features are chosen, the reconstruction regression performed slightly better than the baseline regression in predicting the quality of the archived web pages. While both results do not differ much, it must be considered that the changes made through the reconstruction were relatively small as well. By manually reviewing the reconstructed screenshots, we found that a noticeable amount of translated elements were missed in the reconstruction process. This suggests that if more translated elements were found and considered in the reconstructed screenshots, then the correlation between the reconstructed archived web pages' screenshots and the quality could be improved, and the true quality could be predicted better.

The dimension of the accuracy and R^2 score, in general, is not so bad as to suggest the model only outputs random data, but it could be improved upon. It is possible that the data is not linearly distributed, and using a different model, like a neural network, a polynomial regression, or even a categorical algorithm could provide better results.

Further, we found that all data on the shifts other than the reconstruction error performed poorly when isolated, but using the entire feature set of the group which counted all shifts disregarding their shift distance together with the reconstruction error and the baseline error provided the best result concerning the R^2 score as seen in experiment F1.[10]

All in all, we find that the reconstruction regression performs better regarding both the accuracy and the R^2 score, presuming the right features were chosen. The experiments have shown that choosing the right features from the reconstruction data is critical to the performance of the linear model, to the point where certain features not only performed worse than the baseline but even worse than a constant model that would always predict the expected value. While overall relatively few changes were made on the screenshots dur-

[10] As we did not compute all combinations of features from the reconstruction data due to time restrictions, there might be a combination of features which provides better results than the combination of features which performed best in our experiments for the reconstruction regression. But as we tested all combinations of features for the baseline error, we can be sure that the best features we found are better than the best combination of features from the baseline error alone.

Regression Type	Best R^2 Score	Best Accuracy
Baseline	0.309 (A1B)	0.748 (A3B)
Reconstruction	0.314 (F1)	0.766 (B3)

Table 7.14: Best results among all experiments regarding the R^2 score and accuracy of the reconstruction regression and baseline regression.

ing the reconstruction process, we would expect to improve upon these results even more if all translated elements could be detected and re-positioned.

The general effect of reducing the features showed significant effects on the domain of the predicted qualities. Figure 7.6 displays the domain differences if all features ((a), (b)), only the features which provide the best accuracy ((c),(d)), and only the features which provide the best R^2 score ((e),(f)) for baseline and reconstruction regression were used. Using all features leads to the farthest stretch of domain for the reconstruction shown in 7.6(b), where in some cases even quality scores of 0, 6, and 7 were predicted. The baseline regression with the full feature set shown in figure 7.6(a) did not predict any quality worse than 4 but in one case assigned 0, which is also out of range. Concerning the domain, the feature set which provided the best accuracy performed best; figure 7.6(c) and 7.6(d) show that, for the reconstruction regression and for the baseline regression, the domain of the predicted qualities was the same as for the true qualities. Using the features which provided the best R^2 score for baseline (figure 7.6(e)) and reconstruction regression (figure 7.6(f)) resulted in mediocre results regarding the domain: Quality 5 was never predicted, but the reconstruction regression predicted a score of 0, which is out of range, for one page. From figure 7.6, we also see that score 1 seems to be the easiest to predict because the cloud of points is the largest at (1,1). The reason for this is most likely the existence of many web pages with a pixel error of 0, for which the quality automatically could be assigned 1 (the best quality) because no reproduction error was detected on the archived and original screenshots (all corresponding pixels are colored the same).

Tables 7.15 and 7.16 show the confusion matrices for the predicted and true values for the baseline and reconstruction regression using the features which provided the best accuracy. From these tables, we can deduct how the predictions have changed for the reconstruction error compared to the baseline error using the best accuracy feature. While the chances for accurately predicting the quality score of 1 for a web page were hardly improved (but it was already quite good for the baseline regression), the main improvement gained by the reconstruction regression is that it better differentiates between web pages of qualities 1 and 2. It accurately assigned quality 2 to 115 more

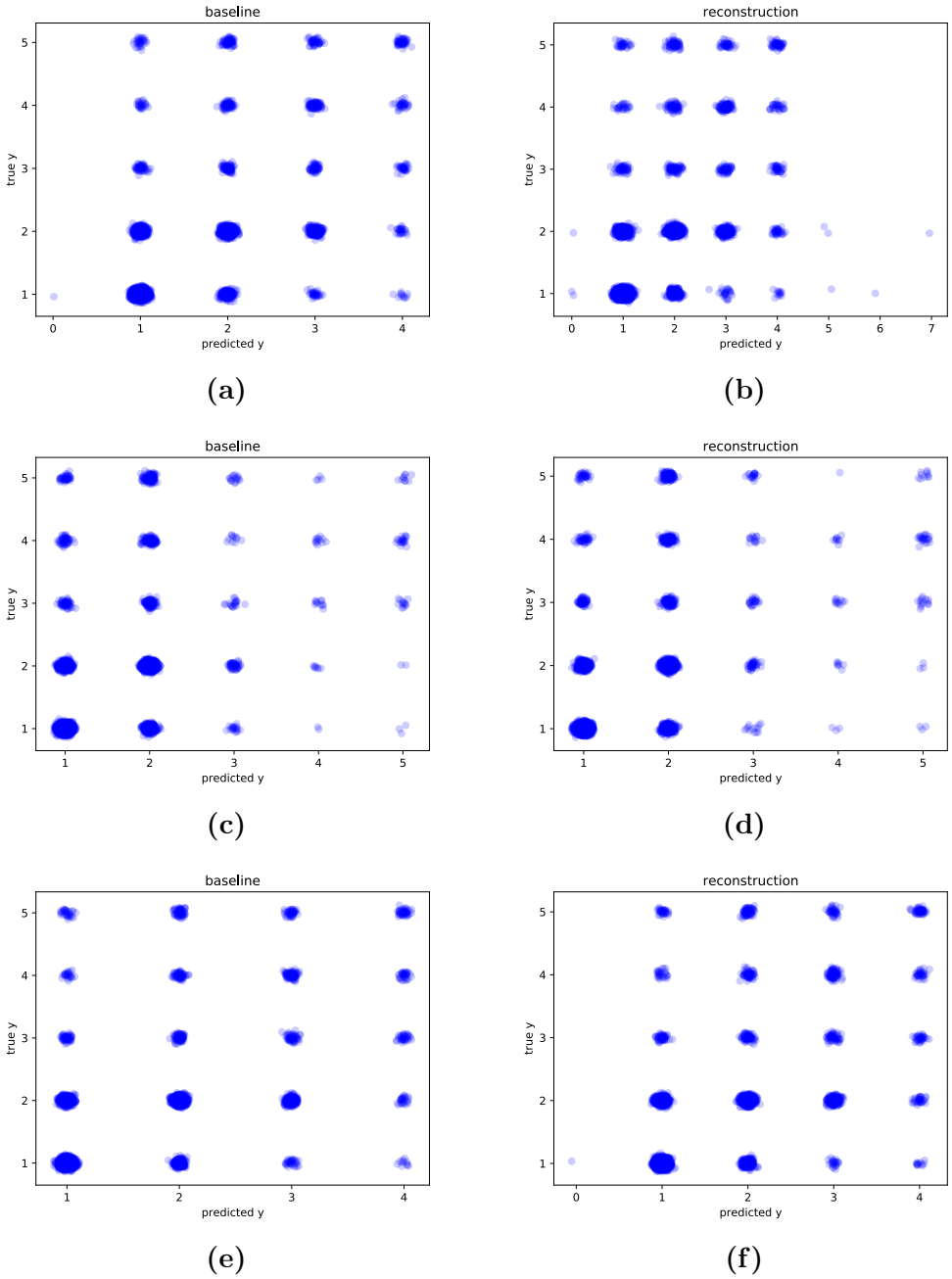


Figure 7.6: Distribution of predicted qualities for a given true quality when using the full feature set for the baseline (a) and reconstruction regression (b) in contrast to only using the features which provide the best accuracy ((c) and (d)) and best R^2 score ((e) and (f)) for each regression. When using the reduced feature set for the best accuracy for the baseline and reconstruction regression, the domain for the predicted values is equal to the domain of the true values (scores 1 to 5).

pages than the baseline regression and falsely assigned 1 for 36 less pages. The reconstruction regression's predictions for quality 3 were only marginally better than the predictions gained by the baseline regression, as four more web pages were accurately assigned the quality score 3. On the other hand, the baseline regression performed marginally better correctly predicting quality score 4, but it also confused it more often with the score of 1 than the reconstruction regression. We understand that incorrect predictions are worse the more they differ from the true score.[11] Generally speaking, both models' predictions for qualities 4 (and 5) are not good. The accuracy for web pages with quality score 5 was the same for both baseline and reconstruction regression, and also the confusion with other quality scores was about the same: Both regressions assigned scores 1 and 2 instead of 5 to 176 web pages and assigned 20 web pages false scores of 3 and 4. Nevertheless, the baseline regression predicted the closer quality score 4 in three more cases than the reconstruction regression. The distribution of falsely assigned qualities' distances to their corresponding true value is more narrow for the better scores 1 and 2 for both regressions.

We suggest there is a difference in importance of qualities of certain levels: We assume a quality score of 1 and 2—if correctly assigned—belongs to good enough pages which would be rather useful to a typical human reader, while quality scores of 3, 4, and 5 might indicate the web pages are rather not useful. It is therefore crucial to the usefulness of an automatic quality assessment tool to identify these coarse groups correctly and not to confuse them. We consider accuracy especially important for quality scores 3 and 2, as they might define the threshold of whether a page was *rather* useless (scores 3, 4, and 5) or just useful *enough* (scores 1 and 2).

Applying these two assumptions to the results presented in the last two columns of tables 7.15 and 7.16, we can see that the first criterion is improved: The reconstruction regression differentiates the "good" quality classes 1 and 2 slightly better from the "bad" ones 3, 4, and 5 than the baseline regression (allowing confusion between the scores within one group). Regarding the accuracy for categories 2 and 3, we can see that even though the reconstruction regression categorized web pages with a quality score of 2 more accurately than the baseline, most of the accuracy was gained through correctly differentiating between quality scores of 1 and 2 and not 2 and 3. We consider the difference between quality scores of 1 and 2 to be not as relevant, because we assume that archived web pages with these qualities would both be useful *enough*. Nev-

[11] The R^2 score provides information about the performance regarding this aspect. Nevertheless, it is hard to interpret within the mediocre ranges that our best results show. From the results it is not clear if a model made many slightly false predictions or few large mistakes. This is why we base our analysis mainly on the accuracy, as it is easier to interpret and compare.

ertheless, even though the progress is minor, we see that the reconstruction regressions accuracy concerning quality scores of 2 and 3 was higher than the accuracy gained through the baseline regression.

	Predicted						
	1	**2**	**3**	**4**	**5**	**1+2**	**3+4+5**
True							
1	3 452	345	19	2	3	3 797	24
2	519	1 390	48	7	2	1 909	57
3	59	133	19	11	13	192	43
4	54	133	11	12	21	187	44
5	43	133	16	4	10	176	30

Table 7.15: Confusion matrix for true and predicted quality scores resulting from the baseline regression using the features which provided the best accuracy.

	Predicted						
	1	**2**	**3**	**4**	**5**	**1+2**	**3+4+5**
True							
1	3 453	349	13	3	3	3 802	19
2	483	1 505	40	7	3	1 988	50
3	61	126	23	11	14	187	48
4	44	140	14	9	24	184	47
5	36	140	19	1	10	176	30

Table 7.16: Confusion matrix for true and predicted quality scores resulting from the reconstruction regression using the features which provided the best accuracy.

Chapter 8

Connecting the Translation Error and Missing Elements

We have implemented the framework presented in chapter 4 with the aim of reducing noise in the pixel error, that is introduced by shifted elements. However, there are many more reproduction errors (as seen in chapter 3), that inflict noise to the pixel error, decreasing its ability to indicate archived web pages' qualities. One of these reproduction errors that we find especially interesting is the missing element error: When we manually inspected archived web pages with shifted elements, we often noticed a missing element in close proximity to some of the shifted elements. Especially advertisement elements were missing often and seemed to be causing shifts on the archived pages. We consider the advertisement on a page to be of lesser use to a human reader than the content on the archived web page, at least for common use cases of the web pages. A missing element that is not content-relevant introduces noise to the pixel error. Indicators that would help telling missing elements that are content-relevant from those that are not content-relevant would help to remove this noise in the pixel error. In this chapter, we explore the ability of shifted elements to be used as such an indicator.

8.1 Missing Elements and Gaps in Reconstructed Screenshots

In chapter 6, we have explained that during the reconstruction of the archived screenshot, shifted elements are re-positioned and often cause gaps in the reconstructed screenshots. Our initial understanding was that gaps would only appear at places where elements were missing in the archived screenshot. Fig-

(a) (b) (c)

Figure 8.1: Three screenshots of a web page. The screenshot of the original web page (a) shows the headline "El PAÍS" at a lower place than the screenshot of the archived version of the web page (b). The gap, resulting from the shift of the headline in the reconstructed version of the archived web page's screenshot (c) is colored pink and corresponds directly to the shifted headline element.

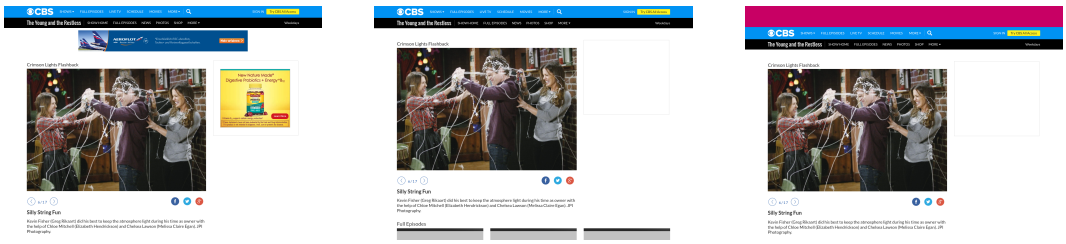
ure 8.1 displays an example of such a gap, corresponding to the element which is missing in the archived screenshot but present in the original screenshot.

The first idea was to use the coordinates from the gaps to copy subimages of missing elements corresponding to these positions from the original screenshot. We expected that we would be able to classify the missing elements' visual representations by type, that is to tell content-relevant from not content-relevant elements. An example of a content-relevant element is an article in a blog; not content-relevant elements can be advertisements or social media buttons, among other things.

However, when we examined the results of the reconstruction process in more detail, we learned that the gaps present in the reconstructed screenshots, even though they were often triggered by a missing element, mostly neither appeared at the place nor with the size of the corresponding missing elements. Figure 8.2 displays an example of such a gap, not corresponding to the element which is missing in the archived web page but present in the original web page. This mismatch between the gaps' positions and the missing elements is often caused by the tree structure of the HTML: Elements are layered by their depth, and sometimes, the element containing the shifted element is re-positioned.[1] In the example in figure 8.2, the missing ad element at the top caused the below image element in the archived screenshot to be shifted upwards. To align the image element to the original screenshot, the reconstruction process shifted the containing element—the HTML body. This caused the gap which should have appeared at the position of the missing ad element below the header of the page to be positioned above it.

This is why we were not able to deduce information about the missing elements from the gaps. Consequently, the idea to include information about

[1]Because we use a majority vote of two thirds of the motion vectors of an element to trigger a shift.



(a) (b) (c)

Figure 8.2: Three screenshots of a web page. The screenshot of the original web page (a) shows the image element at a lower place than the screenshot of the archived version of the web page (b). The shift of the image element in the reconstructed screenshot (c) to the position of the same element in the original screenshot produces a gap colored in pink in the reconstructed screenshot which does not correspond to the position of the missing element causing the shift.

the missing elements' types and their respective influence on the quality of the archived web pages could not be realized.

8.2 Identifying Missing Elements through Shift Distances

Learning about missing elements from the gaps alone was not possible because the positions of the gaps were not a reliable source of information. However, the information about the shifts was: We assumed one missing element at the top of the web page would cause the elements below to shift upwards by the height of the missing element.[2] In order to derive information on the most frequently missing elements' heights from the shift distances, we computed the frequency of certain shift distances. We did this by grouping the absolute number of shifted elements of a certain distance of shift by pages in which at least one such distance occurred for a shifted element. The grouping was necessary to prevent shifted elements affected by the same missing element from increasing the frequency of a specific shift distance because this would unjustifiably increase the perceived importance of some shift distances.

Figure 8.3 shows these frequencies of web pages with at least one element of a specific shift distance for all occurring shift distances in four diagrams. Each diagram represents one of either direction top, bottom, left, or right. The x-axes display the shift distances as an absolute number of pixels, and

[2]Even if the shift was detected not for the element below the missing element but assigned to the element containing it, the shift distance would still represent the missing element's height.

the y-axes show the number of web pages in which at least one element with the respective shift distance occurred. Each of the blue lines in the histograms represents one separate number of pixels—one specific shift distance. The coordinate system for each screenshot starts at position $(0, 0)$ in the top left corner; positive x-axis describes a shift to the right, the negative x-axis a shift to the left, positive y-axis a shift to the bottom, and negative y-axis a shift to the top. Of course, elements can be shifted in two directions. However, as we were primarily interested in elements that are positioned below a potentially missing element, it was sufficient to look at each shift direction separately per axis. We found that the shifts to the top, as shown in figure 8.3d, were most interesting, not only because they occurred in the highest number of pages but also because they displayed multiple peaks. Each peak represents a high number of web pages, which (if our assumption of a connection between missing elements' heights and shifted elements' distances were valid) might be missing an element of the height indicated by the corresponding shift distance. The other diagrams, displaying the directions other than the top of the archived web page, do not indicate such varying peaks.

We found the shifts to the top of the screenshot to be the most interesting direction of shifts, not only because this shift direction is the most common among all shifted elements[3], but also because the shifts to the top of the web page displayed the most diverse distances. Among these diverse distances, we noticed that some shift distances occurred more often than others. Out of those often occurring shift distances, we found three peaks especially interesting: (1) the shift distance of 90 px to the top because we recognized the shift distance of 90 px as a standard height for advertisement banners; (2) the shift distance of 20 px to the top because we recognized the shift distance of 20 px as a standard height of many social media buttons (like Facebook's Like-Button or Instagram's Follow-Button); and (3) the shift distance of 1 px to the top (which we suspected to be caused by tracking pixels) because it was the most common shift distance. In the following sections, we will describe our findings about the causes for these shift distances.

8.2.1 Relating Missing Elements to Shift Distances

Finding missing elements and relating those missing elements to the occurrence of shifted elements can be challenging if done by a screenshot comparison with the bare eye. Especially if pages are very crowded, finding a missing element

[3]About 50% of all shifted elements are shifted to the top, top left or top right. About 41% of all shifted elements are shifted to the left, top left or bottom left. Only about 12% of all shifted elements are shifted to the bottom, bottom left or bottom right and even fewer shifted elements, namely about 5% are shifted to the right, top right or bottom right.

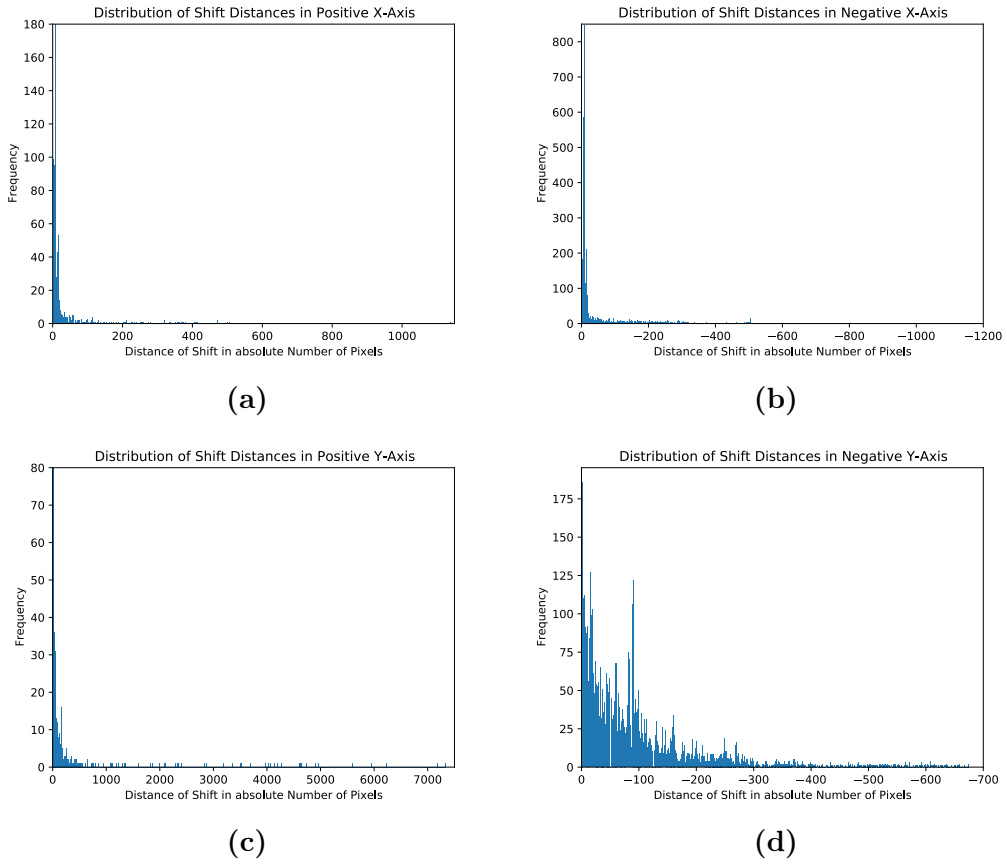


Figure 8.3: Four diagrams displaying the frequency of web pages with at least one element shifted to any of the four directions right (a), left (b), bottom (c), top (d) by an absolute amount of pixels. The number of pages with at least one element shifted x pixels to the right of the archived web page is displayed by (a), the number of pages with at least one element shifted x pixels to the left of the archived web page is displayed by (b), the number of pages with at least one element shifted x pixels to the bottom of the archived web page is displayed by (c) and the number of pages with at least one element shifted x pixels to the top of the archived web page is displayed by (d).

can take a long time and is error-prone. To reduce the complexity of this task, we proceeded as follows:

Finding Shifted Elements By Shift Distance. By computing the shifted elements for the reconstruction process in a previous step, we gained a file containing old and new positions of all shifted elements for every web page in our data set. From this file, we extracted the information about the shift distance. We generated three new files containing only elements with a particular shift distance (90 px, 20 px, or 1 px) and direction (to the top of the archived screenshot) per file. For each element in these three files, we also noted the page ID of the web page it was found on, the XPath of the shifted element and its old top left and bottom right coordinates as well the new (shifted) top left and bottom right coordinates.

Redrawing the Original Screenshots with fewer Elements. In order to reduce the complexity of the visual comparison between the original screenshot and the reconstructed screenshot to identify missing elements of a particular type in the archived screenshots, we redrew the original screenshots to contain only the relevant elements for this analysis: Here, we used the Image Comparator to compare each original screenshot with its corresponding reconstructed version. If a pixel had the same color in both screenshots, it was not drawn to the output screenshot, and if it differed, it was drawn in the color it had in the original screenshot. The resulting screenshot will be called *reduced screenshot* from here on. In the reduced screenshot, missing elements would show because they were not contained in the archived screenshot. We chose the reconstructed screenshots from the lower bound computation (see section 6.2.2) as input in case the position of a missing element was overlapping the position of a gap.[4] The reduced screenshot would make it easier to see missing elements because there are fewer elements in the screenshot, one of which would show a missing element. (Obviously, the reduced screenshot does not only contain missing elements: Elements which were not missing but incorrectly shifted, or elements which were affected by other reproduction error types would still be present in the reduced screenshot.) At the end of this redrawing process, we gained three new data sets, one for each shift distance 90 px, 20 px, and 1 px containing reduced screenshots only for web pages for which the archived version had at least one element with the respective shift distances 90 px, 20 px, and 1 px.

[4]Because the pixels in the gaps would not randomly match the colors of the missing elements pixels.

Finding the Missing Elements. For finding the missing elements, we inspected the reduced screenshots in order to find the missing element for each of the three data sets (90 px, 20 px, 1 px). For the 90 px data set, we suspected advertisement elements to be displayed in the proximity of the position of the topmost shifted element with a 90 px shift of this page. For the 20 px shift, we were looking for social media buttons, and for the 1 px shift we suspected tracking pixels to be causing the shift. If such an element we suspected to be missing was found in the reduced screenshot, we confirmed it to be the cause of the shift of the element in its proximity by comparing the raw archived and original screenshots in this area. For example, if we found a missing ad element above the topmost 90 px shift in the reduced screenshot, we would confirm it was the cause for the shift of this shifted element if we then did not find a 90 px ad element in the unprocessed version of the archived screenshot.[5] Similarly, we proceeded to confirm the shift causes of the 20 px shifts and the 1 px shifts to determine if they were triggered by missing social media buttons, or tracking pixels.

8.2.2 Connecting Missing Ad Elements and the Occurrence of a 90 px Shift

We observed two dominant types of ad elements in our data set: One type is placed horizontally above and between the content of the web page and is a rectangle with larger width than height. The other type is placed next to the content either on the left or the right side of the web page and is shorter in width than in height. We call the first type *horizontal* and the second type *vertical* ad element. We only explored the relation between shifts to the top and horizontal ad elements. The relation between shifts to the left or right and missing vertical ad elements is not as interesting because they do not trigger shifts as often as horizontal ads. This is due to the fact that those vertically placed ad elements mostly do not mix into the content of the page but are placed at the blank space to the left or right of the web pages' contents. Horizontally placed ad elements are different because they usually are mixed with the content of the web pages and therefore trigger a chain of shifts of elements, which lie below them.

We observed a connection between missing ad elements of 90 px height and elements that shifted to the top of the archived web page: In total, there

[5]It was important to confirm an ad element was really missing in the archived screenshot, because ad elements are not always static: An ad containing animated content would show as a pixel error if the screenshot was taken at two different points in time. Such an ad element would show in the reduced screenshot even though it was not missing.

were 122 web pages affected by a 90 px shift distance in at least one element.[6] Among these affected web pages, we were able to identify a missing ad element causing the detected shift in 112 (92%) of the web pages. A detailed analysis of the ten web pages in which we did not find a corresponding missing ad element causing the shift of 90 pixels showed that three of these shifts were caused by missing content. Seven of them were caused by missing elements other than ads, but neither relevant to the content (see table B.10 on page 130 in the appendix for details).

While it seems promising to use the occurrence of at least one element with a 90 px shift to the top of the archived screenshot as an indicator for a missing advertisement element, there is still a risk of misclassification: The analysis of errors causing the shift showed that the hypothesis "A missing ad element was the cause for the 90 px shift." was confirmed in $\frac{112}{122} \cdot 100 \approx 91.8\%$, so for about 92% of the cases. However, there were seven web pages among the ten for which a missing ad element did not cause the 90 px shift, but we found that instead, a missing element not relevant to the content of the archived web page was causing the shift. Therefore, we come to the following conclusion: A generalization of the hypothesis, stating "A 90 px shift to the top of the archived screenshot is caused by non-content elements.", is valid for $\frac{112+7}{122} \cdot 100 \approx 97.5\%$, so for about 98% of the web pages in our data set.

8.2.3 Incorporating the Findings into the Data

The easiest way to incorporate these findings into the data would be to subtract the number of pixels of the missing element from the pixel error. However, the width of the missing element is not known to us. The only option of estimating the number of pixels to be subtracted would be to use the standard width of ad elements with 90 pixels in height. Unfortunately, this would not be the correct dimension for all not content-relevant elements which cause 90 px shifts but could be a good approximation of the noise caused by these elements.[7]

Another possibility to incorporate these findings into the data might be to flag all shifts with a boolean stating if this shift was caused by missing content. This approach could allow the model to incorporate statistics not only about the number of shifts but an estimation of the cause of the shift and the amount

[6] It suffices to find only one element with the shift distance of 90 px per page, because we assumed the missing ad element of 90 px height would at least affect the element directly below or around the missing ad. It was only important to find if an element was missing, not how many elements were affected by the resulting shift, therefore, it was irrelevant if also other elements were affected by the missing ad.

[7] Not all ad elements which are 90 pixels high also use a standard width. Also, as shown before, some elements, which are not ads, but neither content relevant, cause 90 px shifts; their dimensions would neither be estimated correctly with this approach.

of negative influence on the quality of the archived web page. If an archived screenshot had no other shifted elements than those with a 90 px shift to the top, we could assume with about 98% certainty (at least for our data set) that the shift was caused by a missing element that was not relevant to the content of the web page. This could be a marker for good quality on the human quality assessment scale.

Because we only found relatively few shifts with the information about a missing ad element, we expected that implementing our findings just yet would not change the reconstruction error's correlation with the human annotations. We think it would need more data about other connections to content relevant (or irrelevant) missing elements in order reduce the noise in the pixel error noticeably. This is why we did not use our findings about missing ad elements and the occurrence of a 90 px shift for the linear regression model.

8.2.4 Connecting Missing Social Media Buttons and the Occurrence of a 20 px Shift

We also analyzed the connection between the occurrence of a 20 px shift of elements in the archived screenshots and missing social media buttons, such as Facebook's Like-Button or Instagram's Follow-Button. For this analysis, we used the same method as for the analysis of the 90 px shifts.

Unfortunately, our results for the 20 px shifts were not as positive as the results we received for the connection between the 90 px shift and the missing advertisement elements. We found 103 web pages with at least one element that was shifted 20 px to the top of the archived screenshot. For only eight of these 103 web pages (approximately 9%), a missing social media button of 20 px height seemed to have caused the shift. The remaining 95 web pages displayed diverse causes for the shift. We found missing links, missing buttons (not social media but content-related), false correspondences with a coincidental distance of 20 px, and missing padding to be the most frequent. Due to the weak relation between the 20 px shift to the top and missing social media buttons, we find the hypothesis that 20 px shifts to the top of the archived web page are caused by missing social media buttons in the archived web page does not hold to be true.

8.2.5 Connecting Missing Tracking Pixels to the Occurrence of a 1 px Shift

There are not many element types on a web page that commonly have a height of 1 pixel. We were suspecting tracking pixels or a missing line element to be causing the shift of 1 px in the archived web page. As these elements are

especially hard to detect with the bare eye, we hoped to be able to identify them most easily on web pages that do not have other missing elements because then the reduced screenshot would only display a single row of pixels, which would be the missing line element or tracking pixels. We hoped to find those rather empty reduced screenshots among those pairs which display a 1 px size difference, where the archived web page is 1 px shorter than the original. There were 186 web pages that exhibited at least one element, which was shifted by 1 px to the top of the archived web page. We found that, among the 186 web pages, 28 web pages' heights differed in 1 px, where the archived web page was shorter than the original web page. Among the reduced screenshots of these 28 web pages, we found 24 web pages in which a single line of pixels in close proximity to the first shifted element was visible. Comparing the corresponding unprocessed archived and original screenshots, we were able to identify the presence of these lines of pixels in the original and its absence in the archived screenshot for 21 of the 24 pages. We deem it likely that tracking pixels are composing these lines of pixels missing in the archived web page even though we could not ultimately prove it, lacking a description of structure of the original web page.

We need to emphasize that we only looked at the web pages with a height difference of 1 px because we expected the lines to be easier to detect in the reduced screenshots. Of course, this was just an assumption to reduce the amount of web pages to compare. It does not mean that the 158 web pages, which had at least one element shifted by 1 px, but a height difference different from 1 px do not contain such lines in their reduced screenshots. It simply means that we did not expect those lines to be seen easily and therefore did not sample those pages for further inspections.

8.2.6 General Remarks on the Connection between Shift Distances and Missing Elements

We were only looking at web pages in which at least one occurrence of a particular shift distance (90 px, 20 px, 1 px) was found. This does not mean that we detected all web pages with missing ad elements with a height of 90 px, or all web pages in which a social media button was missing: The shift distances most often do not stand isolated, that is, multiple missing elements might cause a vertical shift distance equal to the sum of their height. Those combined shift distances become more difficult to match to certain missing elements' heights the more elements are involved in the cause of the shift—present elements as well as missing ones: If, for example, two elements were missing, one of size 90 px and one of size 20 px, we might also need to look out for elements which shifted $90\ px + 20\ px = 110\ px$. Already for these two missing elements,

it would not be clear from only observing a 110 px shift if the cause was one missing element of 110 px height or multiple elements of heights that can be combined to 110 px. If one considers the possibility of the *additional element* reproduction error on an archived page, the problem becomes even more complex. Consequently, the search for all missing ad elements or social media buttons cannot be computed for all web pages. It is a requirement that the shift distance and the height of the missing element correspond to each other.

Chapter 9

Conclusion

In this thesis, we explored the possibility of an automatic quality assessment of archived web pages. We used a variation of the webis-web-archive-17 data set gathered by Kiesel et al. [16], which provided screenshot pairs of archived and original web pages as well as human annotations about the quality of each archived web page. We have first presented the pixel error, computed by and adaptation of the edit distance, as a measurement of the differences between archived and original screenshot pairs, categorized reproduction errors, and shown that there are limits to the task of detecting corresponding elements screenshot pairs. Then we suggested a framework for automatic quality assessment of archived web pages and implemented it. The implementation of the framework was based on the reconstruction of archived screenshots by shifting translated elements back to their original position. The shift distances and direction were computed using motion estimation in video encoding software. Throughout the implementation process, we explored various possibilities of reconstructing archived screenshots and deducing information from shifted elements about missing elements. We also found solutions to problems related to incoherent information in the data set, which was caused by merging information of two different data sets. Solving this problem resulted in a closer examination of the more general issues regarding the durability of a web archive: We saw that time has a degrading impact on the quality of the archived web pages as well as on tools that assess their quality. The aim of the reconstructive approach to automatic quality assessment was to decrease the amount of noise—generated by specific reproduction errors—in the differing pixels between archived and original screenshot pairs. With this approach, we explored whether the less noisy pixel error, which was computed by comparing the reconstructed and original screenshots, would be a better indicator for the quality of an archived web page. For this, we trained linear regression models with the noisy and the less noisy pixel error to predict the quality of the

archived web pages according to human annotations. The results have shown that the pixel error gained through the reconstruction process, which reduced the noise generated by translated elements, is a more accurate predictor of the quality than the pixel error resulting from a comparison between the unprocessed archived screenshots and their corresponding original screenshots. Still, the prediction accuracy of 0.766 gained by the reconstruction error is not perfect. Compared to the baseline error's accuracy of 0.748, the increase in accuracy is only minor. Considering that we did not find all shifted elements through video encoding and knowing that there are other error types that cannot be fixed by shifting elements, we expect the accuracy of the reconstruction error to increase if the presented framework was further iterated on. Generally speaking, we find the number of changed pixels to be a good estimator for the quality and believe it can be improved upon by better detection of shifted elements and by eliminating further noise from the pixel error introduced by other reproduction error types.

9.1 Future Work

We emphasize that this thesis only explored one of many possible ways to approach automatic quality assessment of archived web pages. We focused on a visual comparison of screenshots of archived web pages and their corresponding original web pages' screenshots and presented an extendable framework, which can be applied to other information that might not be derived from the screenshots themselves but from metadata found in the web archive. This future work section focuses on the possibilities of improving upon screenshot-based data.

Quality Measure. In section 3.1, we mentioned that our quality measure, the pixel error, does not differentiate between large or small color deviations of corresponding pixels, and neither does it distinguish between large and small shift distances of translated elements. Later in section 6.4.2, we used information on the shifted elements' distances in the linear regression model by including a feature, which counted the number of shifted elements with small and large shift distances for each archived page. These features turned out to result in bad predictions concerning the quality of the archived web pages, as seen in chapter 7. Nevertheless, it is possible that we did not find the right threshold. Therefore, we encourage to further test the assumption that large shifts might indicate a reduced quality compared to small shifts and to find other statistics on shifted elements to consider.

Regarding the assumption that small color deviations of corresponding pixels between the archived and original screenshot might influence the quality of an archived web page less, we find it promising to inspect the impact of a tolerance for deviation on the pixel error. The baseline and reconstruction error, both computed by the Image Comparator introduced in chapter 4, do not consider the amount of difference between two corresponding pixels. Therefore minor color deviations are weighted equally to major deviations. In chapter 5, we pointed out that in some cases—possibly due to rendering differences of different browser versions—a small color deviation was introduced between a pixel in the archived and original version. Under the assumption those minor color deviations are not relevant to the quality, we consider those differences to be noise in the pixel error, which could weaken the correlation to the human annotations. Applying a tolerance for minor deviations in the Image Comparator might show an effect on the pixel error and could possibly reduce the noise of the pixel error regarding the correlation to the human annotations. Figure 9.1 shows the decrease in pixel error on the baseline error given a tolerance of 1 ($\delta = 1$) in each color channel of an RGB color space compared to the pixel error we used. The decrease seems large enough for us to suspect it could improve upon the correlation. Therefore, we recommend implementing this tolerance to the presented framework.

Moreover, it would be interesting to compare different quality measures concerning their performance on the correlation to the human annotations. An interesting measure in this regard is the Longest Common Subsequence (LCS) which could be used to measure the longest common subsequence of color values between the (reconstructed) archived and original web pages' screenshots. In contrast to the pixel error we used in this thesis, this measure would not become noisy from the translation error. Still, LCS would introduce new problems which the pixel error does not have; for example, it would not distinguish between translated elements and differing elements. Comparing different quality measures, like LCS or the pixel error, concerning their abilities to reflect the quality of various types of archived web pages, could contribute to a more sophisticated capture of quality with a focus on the different reproduction error types. Nevertheless, as shown in this thesis, the accuracy of the reconstruction error gained improved results compared to the baseline error, which is why we think it would be equally interesting to further reduce noise in the reconstruction error.

Video Encoding. As mentioned in section 6.2.1, we used ffmpeg's version 3 to compute the motion vectors from which we derived the shifts for each element and based our reconstructed screenshots on. After having used ffmpeg 4 on a few examples, we now have reason to believe that the motion vectors

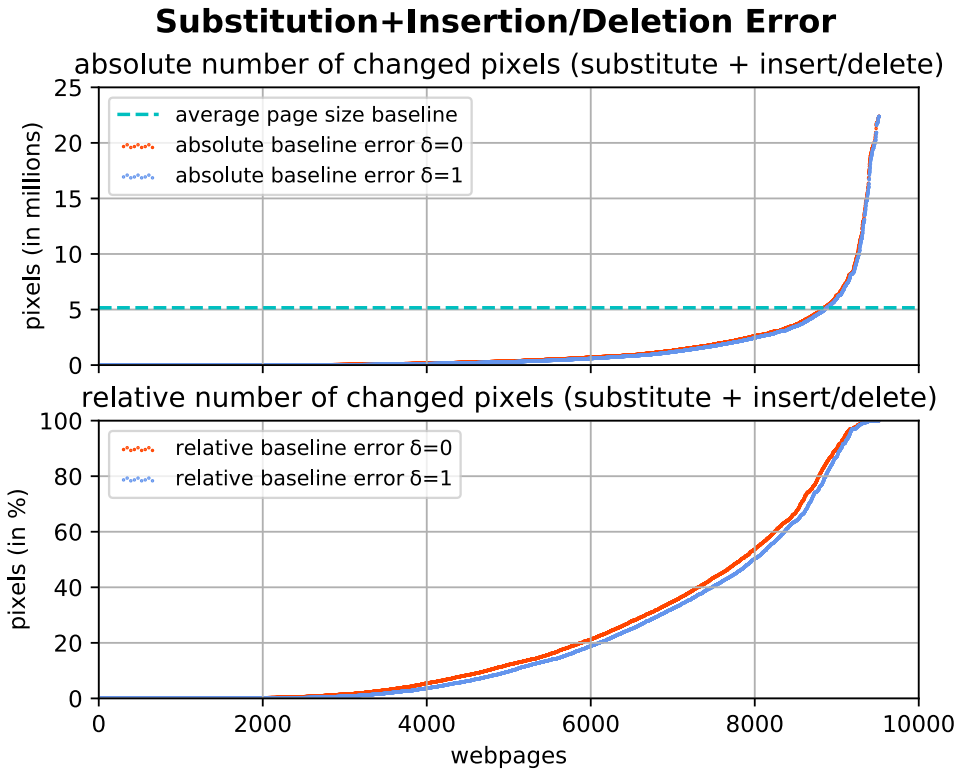


Figure 9.1: The plots show the absolute (top) and relative (bottom) baseline error as used in this thesis (red): Minor color differences are weighted equally to large color differences of corresponding pixels in archived and original screenshot. The blue plot shows a decreased baseline error when an allowance for minor differences of corresponding pixels is introduced. Here the allowance is 1 for each of the channels red, green and blue of the RGB color space ($\delta = 1$).

computed with ffmpeg 3 are far less accurate than those computed with ffmpeg 4. We expect that the more accurate the motion vectors are, the more complete the detection of shifted elements would be. As a result, this would lead to better reconstructed versions of archived screenshots with even more reduced noise in the pixel error. As a consequence, we suggest using ffmpeg 4 for further implementations because this is likely to result in more accurate predictions about the quality of the archived web pages.

Another suggestion concerning the video encoding technology is using a different codec. In this work, we only generated motion vectors from video encoding in H.264, but there is a more recent and possibly more powerful codec, H.265. We find this codec worth exploring given its improved ability to detect shifted elements because it might handle the extreme height of the encoded web pages in our data set better than H.264.

Image Reconstruction. Regarding our implementation of the framework, we think it would also be interesting to explore if the translation error could further be reduced by choosing a different threshold for applying the shift to an element. Our current implementation of the Image Reconstructor uses a majority decider which shifts an element only if two thirds of the motion vectors for an element indicate the same shift, as explained in section 6.2.1. However, there are cases in which an element is not shifted even though it is affected by a translation error because less than two thirds of its motion vectors indicate the same shift. Likewise, sometimes elements that are not meant to be shifted are nevertheless moved: For example, if an image element was shifted and all its motion vectors correctly indicated the shift, the surrounding element might be shifted as well if it was small enough. Using another threshold would result in different reconstructed screenshots than the ones generated during this thesis. It is not a trivial task to find an appropriate threshold because it depends strongly on the structure of the HTML pages and the ability of the chosen motion estimation software to detect shifted elements; hence, the threshold would be dependent on the data set. Nevertheless, we think the best threshold would balance out the tradeoff between moving too many elements without translation error and not shifting enough elements with translation error. This would result in the best reconstruction possible with the given motion vectors.

Description of Structure. Using video encoding to detect translated elements blackboxes the process of detecting shifted elements, but the amount of detected translations is crucial to the quality of the reconstructed screenshots and therefore influences the noise in the pixel error directly. Detecting all translated elements and moving them back to their position in the original

screenshot would result in the best possible reduction of noise caused by the translation error and very likely result in more accurate predictions about the quality of the archived web pages. Unfortunately, using video encoding technology to detect the shifts makes it impossible to know if all shifted elements were detected or if the detected ones were correctly identified. (It is possible that very similar parts of the screenshots were interpreted as corresponding elements with a translation error, even though the element stayed in place.) We would be very interested in finding out if the detection of shifts could better be pursued if the data set also included a description of structure for the original web pages, similar to the one stored for the archived web pages. Comparing both descriptions of structures would result in a transparent system to detect shifted elements because there would be no motion estimation or encoding process involved. It would also be possible to detect missing elements from a comparison of corresponding description of structure files. Generating a description of structure for the original web page would require a re-crawl of the web pages. The re-crawled pages would unfortunately not fit the human annotations about the quality of the web pages anymore. This means that in order to fit a new model with the new data set, new crowdsourced data about the quality would be needed.

Reducing the Occurrence of Other Error Types. Another option to reduce the noise in the reconstruction error is to iterate the framework presented in this thesis with a different tool specialized in detecting other reproduction errors than translation errors. We suggest such a new tool to reduce or even eliminate the pixel error produced by an opaque overlay on web pages. This overlay, caused by an ad element or alert box, almost always concerns the entire web page resulting in a very large baseline error. We suspect this error can be easily detected because, except for the alert box itself, all pixels affected by the overlay should display a constant difference. We suspect that reconstructing the archived web page by reducing or even eliminating opaque effects caused by alert boxes or advertisement elements would increase the accuracy of quality estimation for web pages with a very large baseline error. Of course, we encourage finding other tools to reduce other reproduction error types on archived web pages to further reduce the noise inherent in the pixel error.

Non-Linear Models. We suggest fitting a new model to the existing research provided through this thesis. It is possible that the data is not linearly distributed, hence the Linear Regression Model we trained would be limited in its ability to predict the archived web pages' qualities appropriately. In this case, a categorical algorithm or a neural network, for example, might provide better results.

Lastly, we consider the following approach, which does not use the framework presented in this thesis, to deliver valuable insights on problems with error types on archived web pages: This approach centers around the idea of generating an additional data set with human annotations from the screenshots in webis-web-archive-17, in which corresponding elements in the archived and original web pages' screenshots are mapped if present in both or marked as additional or missing if only present in one. These mappings could serve as input to a deep learning model to predict the annotations about the quality of the archived web pages from webis-web-archive-17. We would be especially interested in this approaches' results as a comparison to the results gained through the reconstruction error used in the framework presented in this thesis. Also, such an annotation process could provide insights on the general problems of correspondence detection and error type identification on archived web pages, which are not only difficult to solve for automatic processes but to which not even humans find a simple answer.

Appendix A

Implementation

In the following we will present a detailed description of all reconstruction methods discussed in chapter 6. The code for these methods is stored in our repository at `https://git.webis.de/code-teaching/theses/thesis-elstner`.

Reconstruction Method 0: Duplicates

The duplicate-producing method of image reconstruction was implemented as follows (steps which were introduced in chapter 4.1.1 are colored light):

- Read the original screenshot.
- Read the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output image—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool ffmpeg. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data, that is the direction and distance of the shift, for each element:
  - Compute whether the element is shifted by counting the fraction of motion vectors for the visual representation of the element pointing in the same direction and having the same length: If two thirds of the motion vectors agree upon shift distance and direction, mark the element as shifted, else mark the element as not shifted and proceed to the next element in the element tree object.

- Create a list of all shifted elements in the element tree object sorted by depth in the HTML structure:
- For each element in the list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list:
    * Make a copy (the duplicate) of the visual representation of the element.
    * Move the duplicate to the position the shift for this element indicates.
  - Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed screenshot).

Reconstruction Method 1: Upper Bound

The upper bound method of image reconstruction was implemented as follows (steps which were introduced in chapter 4.1.1 are colored light):

- Read the original screenshot.
- Read the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output screenshot—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool ffmpeg. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data, that is the direction and distance of the shift, for each element:
  - Compute whether the element is shifted by counting the fraction of motion vectors for the visual representation of the element pointing in the same direction and having the same length: If two thirds of the motion vectors agree upon shift distance and direction, mark the element as shifted, else mark the element as not shifted and proceed to the next element in the element tree object.

- Create a map gapsByDepth, to later store pixels that should fill the gaps produced by a shifted element and the depth of the element in the HTML structure. The map is sorted by depth.
- Create a list of all shifted elements in the element tree object sorted by depth in the HTML structure:
- For each element in the list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list:
    - Before applying the shift: Look up the top left and bottom right coordinates of the element (the position in the archived screenshot).
    - Copy the pixels contained in the rectangle at these coordinates from the original screenshot (the subimage of the gap).
    - Store the subimage of the gap and the depth of the current element in the HTML structure in the map gapsByDepth.
    - Move the element to the position the shift for this element indicates.
  - Draw the visual representations of each gap from root to leaves to the output image (the reconstructed screenshot).
  - Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed screenshot).

Reconstruction Method 2: Lower Bound

The lower bound method of image reconstruction was implemented as follows (steps which were introduced in chapter 4.1.1 are colored light):

- Read the original screenshot.
- Read the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output screenshot—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool ffmpeg. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data, that is the direction and distance of the shift, for each element:

  - Compute whether the element is shifted by counting the fraction of motion vectors for the visual representation of the element pointing in the same direction and having the same length: If two thirds of the motion vectors agree upon shift distance and direction, mark the element as shifted, else mark the element as not shifted and proceed to the next element in the element tree object.

- Create a map gapsByDepth, to later store pixels that should fill the gaps produced by a shifted element and the depth of the element in the HTML structure. The map is sorted by depth.
- Create a list of all shifted elements in the element tree object sorted by depth in the HTML structure:
- For each element in the list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list:
    * Before applying the shift: Look up the top left and bottom right coordinates of the element (the position in the archived screenshot).
    * Look up the pixels contained in the rectangle at these coordinates from the original screenshot (the subimage of the gap) and create a subimage with pixels colored in pink if their corresponding pixel in the subimage of the gap is not pink, else violet.
    * Store the subimage of the gap and the depth of the current element in the HTML structure in the map gapsByDepth.
    * Move the element to the position the shift for this element indicates.
  - Draw the visual representations of each gap from root to leaves to the output image (the reconstructed screenshot).
  - Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed screenshot).

Reconstruction Method 3: Most Frequent Color in Archived Screenshot

The method of image reconstruction, coloring the gaps in the most frequent color in the archived screenshot, was implemented as follows (steps which were introduced in chapter 4.1.1 are colored light):

- Read the original screenshot.
- Read the archived screenshot.

- Find the most common color in the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output screenshot—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool ffmpeg. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data, that is the direction and distance of the shift, for each element:
  - Compute whether the element is shifted by counting the fraction of motion vectors for the visual representation of the element pointing in the same direction and having the same length: If two thirds of the motion vectors agree upon shift distance and direction, mark the element as shifted, else mark the element as not shifted and proceed to the next element in the element tree object.
- Create a map gapsByDepth, to later store pixels that should fill the gaps produced by a shifted element and the depth of the element in the HTML structure. The map is sorted by depth.
- Create a list of all shifted elements in the element tree object sorted by depth in the HTML structure:
- For each element in the list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list:
    - Before applying the shift: Look up the top left and bottom right coordinates of the element (the position in the archived screenshot). This is the position of the gap.
    - Assign each pixel in the gap the most common color in the archived screenshot. This visual representation is the subimage of the gap.
    - Store the subimage of the gap and the depth of the current element in the HTML structure in the map gapsByDepth.
    - Move the element to the position the shift for this element indicates.
  - Draw the visual representations of each gap from root to leaves to the output image (the reconstructed screenshot).
  - Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed screenshot).

Reconstruction Method 4: Most Frequent Color Among Gaps' Edges

The context-dependent method of image reconstruction, coloring the gaps in the color which was most frequent around its edges, was implemented as follows (steps which were introduced in chapter 4.1.1 are colored light):

- Read the original screenshot.
- Read the archived screenshot.
- Instantiate the output image according to the dimensions of the original screenshot.
- Add the archived screenshot as background to the output screenshot—padded if it is smaller than the original or cut if it is bigger.
- Create an element tree object from the description of structure, the archived screenshot and the additional data from the tool ffmpeg. The element tree object stores a visual representation (a subimage) for each element that is found in the description of structure. The subimages are copied from the archived screenshot. The element tree object also stores the additional data, that is the direction and distance of the shift, for each element:
  - Compute whether the element is shifted by counting the fraction of motion vectors for the visual representation of the element pointing in the same direction and having the same length: If two thirds of the motion vectors agree upon shift distance and direction, mark the element as shifted, else mark the element as not shifted and proceed to the next element in the element tree object.
- Create a map gapsByDepth, to later store pixels that should fill the gaps produced by a shifted element and the depth of the element in the HTML structure. The map is sorted by depth.
- Create a list of all shifted elements in the element tree object sorted by depth in the HTML structure:
- For each element in the list:
  - Apply the changes which are implied by the additional data gathered by the tool to each element in the list:
    - Before applying the shift: Look up the top left and bottom right coordinates of the element (the position in the archived screenshot). This is the position of the gap.

    - Sample the color of each pixel around the element in the archived screenshot. If the row falls outside of the borders of the archived screenshot, sample the row of pixels in the according edge from the element. Do not sample corner pixels twice.
    - Assign each pixel in the gap the most common color in the samples of the edges around the element. This visual representation is the subimage of the gap.
    - Store the subimage of the gap and the depth of the current element in the HTML structure in the map gapsByDepth.
    - Move the element to the position the shift for this element indicates.
- Draw the visual representations of each gap from root to leaves to the output image (the reconstructed screenshot).
- Draw the visual representations of each modified element from root to leaves to the output image (the reconstructed screenshot).

Appendix B

Results

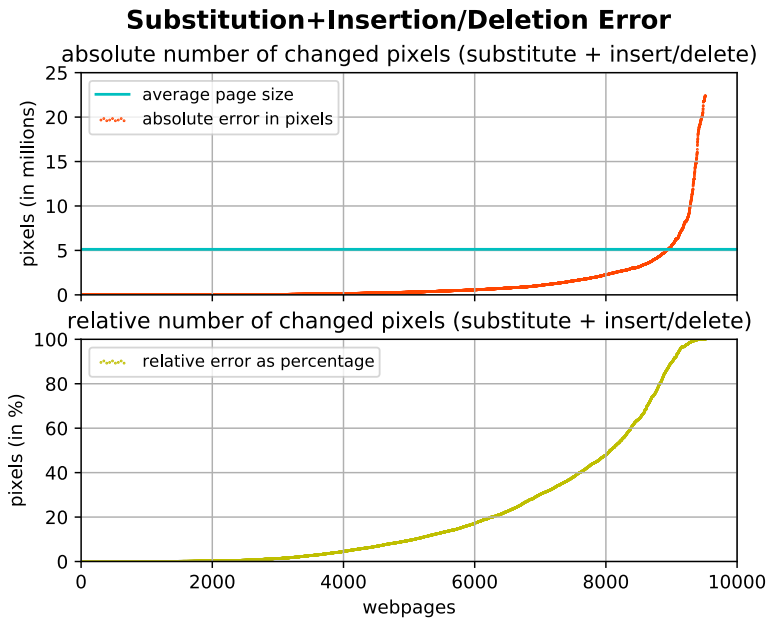


Figure B.1: Absolute reconstruction error (red) and average reconstructed screenshot's size (turquoise) in pixels (top) and the relative reconstruction error (olive) as percentage of pixels in the original screenshot (bottom) for reconstruction method 0. The average size of a web page is 5 123 526 pixels.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 71.528%
20%	$\leq$ 0.199%	92%	$\leq$ 76.248%
30%	$\leq$ 1.123%	93%	$\leq$ 81.699%
40%	$\leq$ 3.747%	94%	$\leq$ 87.018%
50%	$\leq$ 8.231%	95%	$\leq$ 91.488%
60%	$\leq$ 14.471%	96%	$\leq$ 95.777%
70%	$\leq$ 24.835%	97%	$\leq$ 97.573%
80%	$\leq$ 40.221%	98%	$\leq$ 99.280%
90%	$\leq$ 66.519%	99%	$\leq$ 99.883%

Table B.1: Distribution of the relative reconstruction error among the web pages through quantiles for the reconstruction method 0 which produces duplicates of shifted elements.

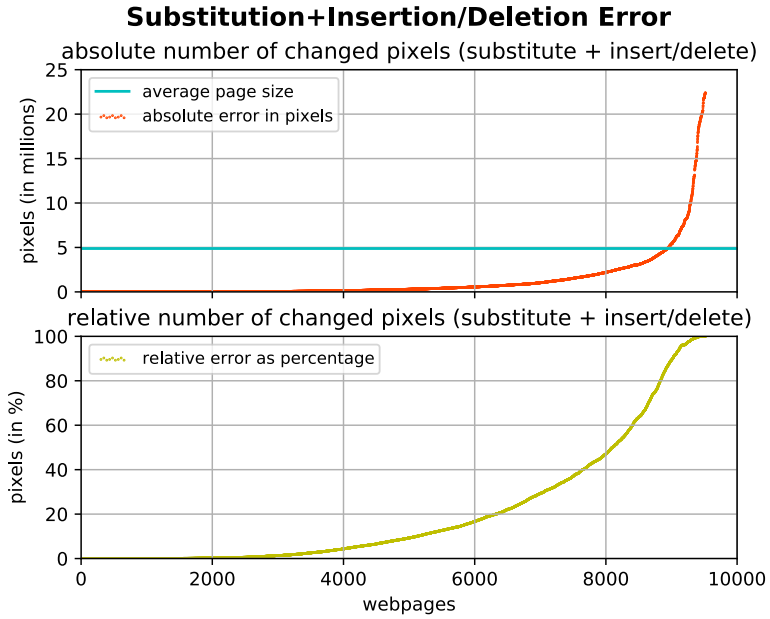


Figure B.2: Absolute reconstruction error (red) and average reconstructed screenshot's size (turquoise) in pixels (top) and the relative reconstruction error (olive) as percentage of pixels in the original screenshot for the reconstruction method 1, the upper bound (bottom). The average size of a web page is 5 123 526 pixels.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 70.837%
20%	$\leq$ 0.196%	92%	$\leq$ 75.287%
30%	$\leq$ 1.085%	93%	$\leq$ 81.326%
40%	$\leq$ 3.588%	94%	$\leq$ 86.986%
50%	$\leq$ 7.980%	95%	$\leq$ 91.276%
60%	$\leq$ 14.080%	96%	$\leq$ 95.214%
70%	$\leq$ 24.154%	97%	$\leq$ 96.900%
80%	$\leq$ 39.329%	98%	$\leq$ 98.881%
90%	$\leq$ 65.791%	99%	$\leq$ 99.846%

Table B.2: Distribution of the relative reconstruction error among the web pages through quantiles for reconstruction method 1 the upper bound.

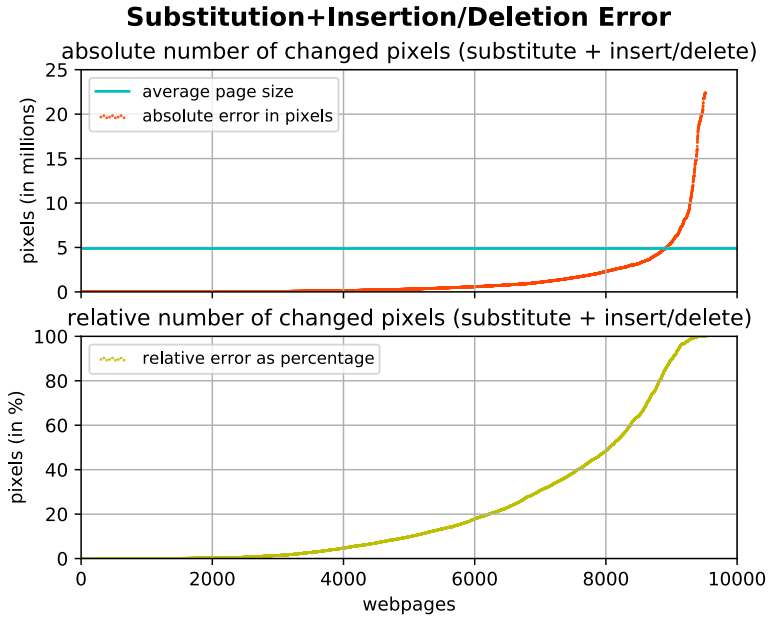


Figure B.3: Absolute reconstruction error (red) and average reconstructed screenshot's size (turquoise) in pixels (top) and the relative reconstruction error (olive) as percentage of pixels in the original screenshot for reconstruction method 2 the lower bound (bottom). The average size of a web page is 5 123 526 pixels.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 71.631%
20%	$\leq$ 0.204%	92%	$\leq$ 76.265%
30%	$\leq$ 1.160%	93%	$\leq$ 81.933%
40%	$\leq$ 3.892%	94%	$\leq$ 87.095%
50%	$\leq$ 8.448%	95%	$\leq$ 91.488%
60%	$\leq$ 14.831%	96%	$\leq$ 95.777%
70%	$\leq$ 25.255%	97%	$\leq$ 97.573%
80%	$\leq$ 40.660%	98%	$\leq$ 99.280%
90%	$\leq$ 66.703%	99%	$\leq$ 99.883%

Table B.3: Distribution of the relative reconstruction error among the web pages through quantiles for reconstruction method 2 the lower bound.

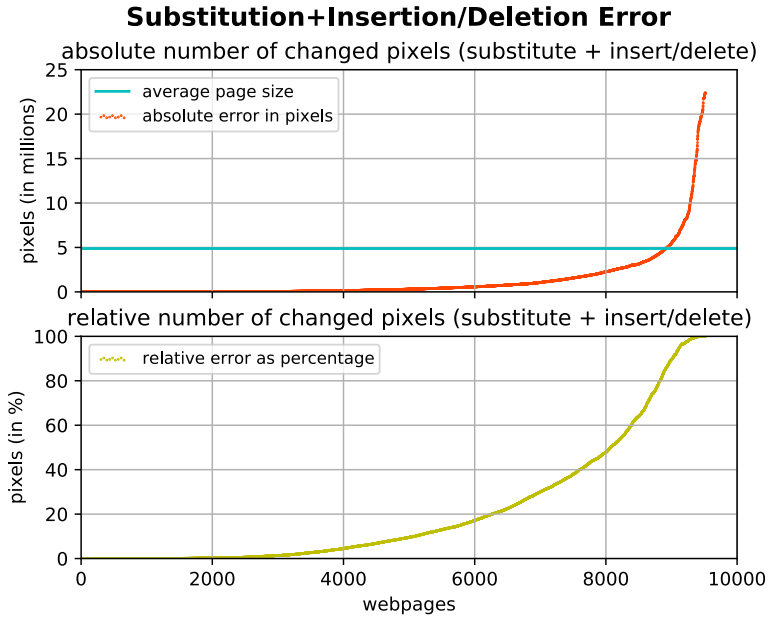


Figure B.4: Absolute reconstruction error (red) and average reconstructed screenshot's size (turquoise) in pixels (top) and the relative reconstruction error (olive) as percentage of pixels in the original screenshot for reconstruction method 4 (bottom). The average size of a web page is 5 123 526 pixels.

Affected Pages	Relative Error	Affected Pages	Relative Error
10%	$\leq$ 0.002%	91%	$\leq$ 71.521%
20%	$\leq$ 0.199%	92%	$\leq$ 76.248%
30%	$\leq$ 1.118%	93%	$\leq$ 81.584%
40%	$\leq$ 3.733%	94%	$\leq$ 87.068%
50%	$\leq$ 8.244%	95%	$\leq$ 91.488%
60%	$\leq$ 14.426%	96%	$\leq$ 95.777%
70%	$\leq$ 24.754%	97%	$\leq$ 97.572%
80%	$\leq$ 40.182%	98%	$\leq$ 99.280%
90%	$\leq$ 66.363%	99%	$\leq$ 99.883%

Table B.4: Distribution of the relative reconstruction error among the web pages through quantiles for reconstruction method 4.

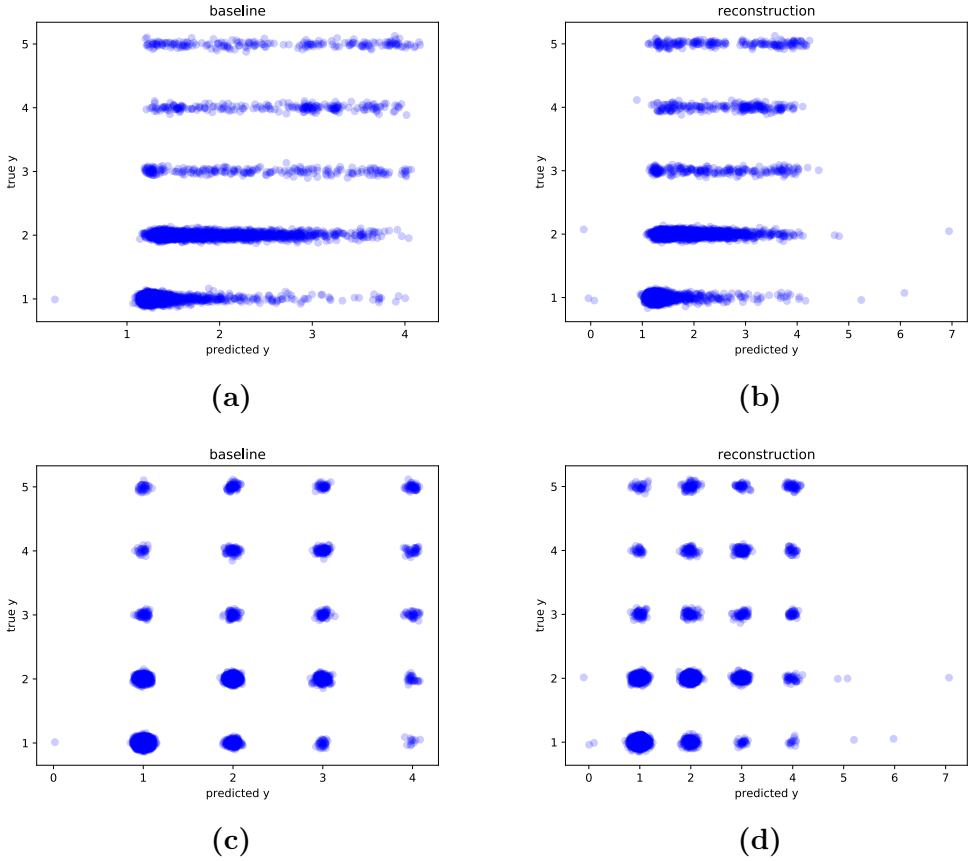


Figure B.5: The domain differences in the relation between the distribution of the predicted values with respect to the true values before and after rounding with the method *round*. (a) shows the unrounded predictions and their corresponding true value for the baseline and (b) for the reconstruction. (c) displays the rounded predictions and their corresponding true value for the baseline and (d) for the reconstruction. Larger clouds of points indicate more frequent predictions, smaller clouds of points indicate less frequent predictions for a value.

ID	Used Features	R^2 Score	Accuracy
A1R *best* R^2	**Baseline Error:** total number of changed pixels	**0.299**	0.727
A2R *worst* R^2	**Baseline Error:** total number of changed pixels, pixels total in original, changed pixels as percent of pixels in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image	**0.241**	0.720
A3R *best Accuracy*	**Baseline Error:** total number of changed pixels	0.299	**0.727**
A4R *worst Accuracy*	**Baseline Error:** total number of changed pixels, pixels total in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image	0.258	**0.717**

Table B.5: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of features from the baseline error while all other features belonging to the reconstruction data were also used but not varied. The results of the reconstruction regression were not so much impacted by varying the features of the baseline error, as neither accuracy nor R^2 score show large ranges for best and worst result. This might be explained by the fact that the reconstruction regression uses many other features to determine its outcome and is therefore less impacted by changes of the subset of features from the baseline error. The large set of unvaried features in the reconstruction regression and with it the stable appearance of the results might also be a reason for the best accuracy and best R^2 score being associated with the same feature.

ID	Used Features	R^2 Score	Accuracy
C1 *best* R^2	**Shifts All:** number of shifted elements on archived web page shifted to the to bottom right	**-0.190**	0.313
C2 *worst* R^2	**Shifts All:** total number of shifted elements on archived web page, shifted elements to the top, to the top right, to the right, to the bottom, to the bottom left, to the left, **Shifts General:** total number of elements on archived page	**-0.205**	0.310
C3 *best Accuracy*	**Shifts All:** number of shifted elements on archived web page shifted to the right, shifted elements to the left, **Shifts General:** total number of elements on archived page	-0.195	**0.313**
C4 *worst Accuracy*	**Shifts All:** total number of shifted elements on archived web page, shifted elements to the top, to the top right, to the right, to the bottom, to the bottom left, to the left	-0.202	**0.309**

Table B.6: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of features which concerned the amount of shifted elements without respecting small or large shift distances. The best results are very low compared to other experiments and do not differ much from the worst results of this experiment.

ID	Used Features	R^2 Score	Accuracy
D1 *best* R^2	**Shifts Small:** total number of shifted elements on archived web page with small shift distance	**-0.190**	0.313
D2 *worst* R^2	**Shifts Small:** total number of shifted elements on archived web page with small shift distance, shifted elements to the top, to the right, to the bottom right, to the bottom, to the bottom left, **Shifts General:** total number of elements on archived page	**-0.205**	0.310
D3 *best Accuracy*	**Shifts Small:** total number of shifted elements on archived web page with small shift distance, shifted elements to the top, to the right, to the bottom, to the left, **Shifts General:** total number of elements on archived page	-0.224	**0.313**
D4 *worst Accuracy*	**Shifts Small:** total number of shifted elements on archived web page with small shift distance, shifted elements to the top right, to the right, to the bottom right, to the bottom, to the bottom left, to the left, to the left	-0.210	**0.312**

Table B.7: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of features which concerned the amount of shifted elements with small shift distances.

ID	Used Features	R^2 Score	Accuracy
E1 *best* R^2	**Shifts Large:** number of shifted elements on archived web page with large shift distance shifted to the bottom right	**-0.190**	0.313
E2 *worst* R^2	**Shifts Large:** total number of shifted elements on archived web page with large shift distance, shifted elements to the top, to the top right, to the right, to the bottom right, to the bottom left, to the top left	**-0.230**	0.311
E3 *best Accuracy*	**Shifts Large:** number of shifted elements on archived web page with large shift distance shifted to the bottom right, **Shifts General:** total number of elements on archived page	-0.191	**0.313**
E4 *worst Accuracy*	**Shifts Large:** total number of shifted elements on archived web page with large shift distance, shifted elements to the top, to the top right, to the right, to the bottom right, to the left, to the top left	-0.215	**0.310**

Table B.8: Best and worst R^2 scores and accuracies of the reconstruction regression, which was computed for all combinations of features which concerned the amount of shifted elements with large shift distances.

ID	Used Features	R^2 Score	Accuracy
G1 *R^2*	**(1) Baseline Error:** pixels added through insertion/deletion, total number of changed pixels, number of pixels in bigger image, changed pixels as percent of pixels in bigger image, **(2) Reconstruction Error:** pixels total in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image, **(3) Shifts General:** total number of elements on archived web page, **(4) Shifts All:** number of shifted elements on archived web page shifted to the to bottom right	**0.313**	0.725
G2 *Accuracy*	**(2) Reconstruction Error:** pixels total in original, number of pixels in bigger image, changed pixels as percent of pixels in bigger image, **(3) Shifts General:** total number of elements on archived page, **(5) Shifts Small:** total number of shifted elements on archived web page with small shift distance, shifted elements to the top, to the right, to the bottom, to the left	0.307	**0.728**

Table B.9: R^2 scores and accuracies of the reconstruction regression, which was computed for the best combinations of groups of features. For each group only the features with the best results when used isolated regarding the R^2 score and accuracy were used. This experiment did not yield the overall best results.

Occurrence	Missing Content	Description
1/10	yes	Two image elements which were missing in the archived screenshot, each shortening their surrounding div element by 45 pixels, which resulted in a 90 px shift distance for elements below the second div with a missing image element.
1/10	yes	The 90 px shift was caused was a faulty browser resize, which produced a screenshot of the archived web page that was 90 pixels too short.
3/10	no	The shift was caused by a 90 px blank space below the header of original web pages that were missing in the archived web pages.
2/10	no	The shift was caused by a 90 px blank space above the header of original web pages that were missing in the archived web pages.
2/10	no	The shift was caused by a false detection of a corresponding element between the archived and original screenshot which coincidentally lay 90 px apart.
1/10	yes	The reason for the 90 px shift was a form element which was not loaded in the archived version of the web page and shortened the surrounding element by 90 px, so that the elements below this shortened one shifted 90 px upwards.

Table B.10: Findings about the ten reduced screenshots in which a 90 pixel shift distance was found but no corresponding missing element of 90 pixels height could be detected. The label *Missing Content* means that no content was missing related to the 90 px shift, but content unrelated to the 90 px shift could have been missing but is not relevant to this analysis.

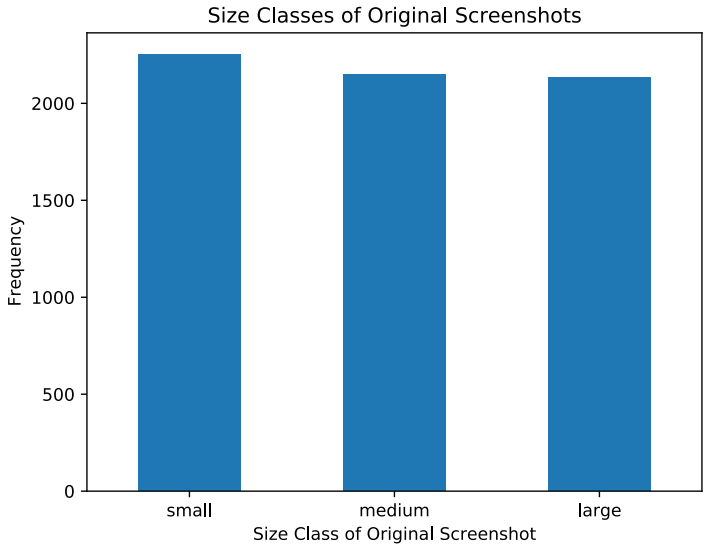


Figure B.6: Frequencies of web pages in each size class small, medium and large. The small group contains web pages with a total amount of pixels up to 2 300 000 pixels, medium considers all web pages larger than small with a total amount of up to 4 800 000 pixels and all other web pages were categorized as large.

Bibliography

[1] ffmpeg Codecs Documentation. https://ffmpeg.org/ffmpeg-codecs.html. last called December 2020.

[2] ffmpeg Documentation. https://ffmpeg.org/ffmpeg-all.html. last called November 2020.

[3] H.264 Video Encoding Guide. https://trac.ffmpeg.org/wiki/Encode/H.264. last called November 2020.

[4] Motion Estimation and Intra Frame Prediction in H.264/AVC Encoder. https://courses.cs.washington.edu/courses/csep590a/07au/lectures/rahullarge.pdf. last called December 2020.

[5] Robert J. Anello and Jane E. Bobet. Gone, But Not Forgotten: Evidence from the Archived Internet. March 2018.

[6] HTTP Archive. Request count. https://almanac.httparchive.org/en/2020/javascript#request-count. last called Febuary 2021.

[7] The Internet Archive. Wayback machine. https://web.archive.org. last called August 2020.

[8] B. Ayala, M. Phillips, and L. Ko. Current quality assurance practices in web archiving. 2014.

[9] Brenda Reyes Ayala. Correspondence as the primary measure of quality for web archives: A grounded theory study. In Mark Hall, Tanja Merčun, Thomas Risse, and Fabien Duchateau, editors, *Digital Libraries for Open Knowledge*, pages 73–86, Cham, 2020. Springer International Publishing.

[10] Brenda Reyes Ayala, Ella Hitchcock, and James Sun. Using image similarity metrics to measure visual quality in web archives. In *JCDL 2019: Web Archiving and Digital Libraries (WADL) workshop*. ACM, 2019.

[11] Karl-Rainer Blumenthal. Quality assurance overview. `https://support.archive-it.org/hc/en-us/articles/208333833-Quality-Assurance-Overview`. last called Febuary 2021.

[12] Justin Brunelle. Let's compare memento damage measures! `https://ws-dl.blogspot.com/2018/09/2018-09-03-lets-compare-memento-damage.html`. last called Febuary 2018.

[13] Justin Brunelle, Mat Kelly, Hany Salaheldeen, Michele Weigle, and Michael Nelson. Not all mementos are created equal: Measuring the impact of missing resources. volume 16, 09 2014.

[14] Michael Cormier, Karyn Moffatt, Robin Cohen, and Richard Mann. Purely vision-based segmentation of web pages for assistive technology. *Computer Vision and Image Understanding*, 148:46–66, 2016. Special issue on Assistive Computer Vision and Robotics - "Assistive Solutions for Mobility, Communication and HMI".

[15] Gesellschaft für Informatik. Grand Challenges. `https://gi.de/grand-challenges`. last called August 2020.

[16] Johannes Kiesel, Florian Kneist, Milad Alshomary, Benno Stein, Matthias Hagen, and Martin Potthast. Reproducible Web Corpora: Interactive Archiving with Automatic Quality Assessment. *Journal of Data and Information Quality (JDIQ)*, 10(4):17:1–17:25, October 2018.

[17] Johannes Kiesel, Lars Meyer, Florian Kneist, Benno Stein, and Martin Potthast. An Empirical Comparison of Web Page Segmentation Algorithms. In Djoerd Hiemstra, Maria-Francine Moens, Josiane Mothe, Raffaele Perego, Martin Potthast, and Fabrizio Sebastiani, editors, *Advances in Information Retrieval. 43rd European Conference on IR Research (ECIR 2021)*, Lecture Notes in Computer Science, Berlin Heidelberg New York, March 2021. Springer.

[18] Ilya Kreymer. Web object encapsulation complexity (part i). `https://webrecorder.net/2020/11/09/encapsulation-complexity.html`. last called Febuary 2021.

[19] Vladimir Iosifovich Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966. Doklady Akademii Nauk SSSR, V163 No4 845-848 1965.

[20] National Library of Australia. Pandora. `http://pandora.nla.gov.au/about.html`. last called August 2020.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[22] Werner Robitza. Crf Guide (Constant Rate Factor in x264, x265 and libvpx). `https://slhck.info/video/2017/02/24/crf-guide.html`. last called May 2020.

[23] Myriam Ben Saad and Stéphane Gançarski. Improving the quality of web archives through the importance of changes. In Abdelkader Hameurlain, Stephen W. Liddle, Klaus-Dieter Schewe, and Xiaofang Zhou, editors, *Database and Expert Systems Applications*, pages 394–409, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[24] Shan Zhu and Kai-Kuang Ma. A new diamond search algorithm for fast block-matching motion estimation. *IEEE Transactions on Image Processing*, 9(2):287–290, 2000.

[25] sklearn. R2 score, the coefficient of determination. `https://scikit-learn.org/stable/modules/model_evaluation.html#r2-score`. last called January 2021.

[26] M. Spaniol, A. Mazeika, Dimitar Denev, and G. Weikum. "Catch me if you can": visual analysis of coherence defects in web archiving. 2009.

[27] International Telecommunication Unit. H.264. Technical report, International Telecommunication Unit, 2019. `https://www.itu.int/rec/T-REC-H.264`.

[28] Wikipedia. Advanced Video Coding. `https://en.wikipedia.org/wiki/Advanced_Video_Coding`. last called November 2020.

[29] Wikipedia. Chroma subsampling. `https://en.wikipedia.org/wiki/Chroma_subsampling`. last called November 2020.

[30] Wikipedia. Inter frame. `https://en.wikipedia.org/wiki/Inter_frame`. last called November 2020.

[31] Wikipedia. Motion estimation. `https://en.wikipedia.org/wiki/Motion_estimation`. last called December 2020.

Erklärung

Ich versichere, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann. Ich versichere, dass das elektronische Exemplar mit den gedruckten Exemplaren übereinstimmt.

Leipzig, April 25, 2021

.....

Theresa Elstner