Automatic Transcription of Historical Newsprint by Leveraging the Kaldi Speech Recognition Toolkit

Patrick Schone, Alan Cannaday, Seth Stewart, Rachael Day, Jeremy Schone; Family Search, Salt Lake City, Utah.

Abstract

We present a method for transcribing historical newsprint images. To begin, for training and evaluation, we created a corpus of human-generated transcripts for almost 38,000 image snippets which contains nearly five million words. This may be one of the largest corpora of transcribed historical newsprint ever created. Then, we developed our automatic transcription process by leveraging the pattern recognition and statistical components of the state-of-the-art speech recognition toolkit, Kaldi. Specifically, we modified its language model behavior and we replaced Kaldi's speech-to-features transformation components with our own image-to-features process. Our replacement components include the use of word partials; image rotation; line segmentation which extends state-of-the-art methods; and customized feature generation. We conduct two evaluations of our technology: (a) an evaluation based on random selections of newspaper snippets; and (b) a diachronic evaluation of newspaper snippets by time frame. We compare our results of these evaluations to those of the commercial engines ABBYY Fine Reader Version 12 and OmniPage 18, as well as to the freely available system, Tesseract. We demonstrate that our process typically yields accuracies which are comparable to or exceed the accuracies of these other engines.

1. Background

Historical newspapers are significant sources of genealogical information. In recent years, genealogical organizations such as Ancestry, FamilySearch, and FindMyPast have gone to great lengths to obtain such documents. Optical Character Recognition (OCR) is commonly used as a technique for transcribing newspapers and providing some search access to these documents. Yet since OCR on historical newsprint yields poorer results than on modern documents, most OCR-enabled transcription is only used in keyword-search mechanisms.

Keyword search can definitely help find *name strings* of interest or words that have some proximity to each other. Yet, the typical genealogical patron usually seeks to identify a specific *person*. It is not enough for a user to say "I want a document with 'John' and 'Chapman' within 10 words of each other" for several reasons. For one, there may be many people by the name of "John Chapman," so refinement is required. For another, this kind of search also returns things like "Robert Chapman, John Smith,.." and "Robert Chapman attended St. John's Cathedral." It is also likely to miss the ancestor when referred to only as "J. Chapman."

To find a particular ancestor, a patron must use queries that accommodate name variations and must couple name-based queries with search parameters related to dates and places of the ancestor's vital facts and family relations. For example, if a person wants to find their specific ancestor "John Chapman," he would need to provide specifics about when and where John Chapman lived and to whom he was related. These valuable relationship-based kinds of searches cannot be achieved readily through the use of normal keyword searches.

Therefore, genealogical companies have sought volunteer or for-pay human labor forces that can distill out critical facts from the newspapers to then enable these relationship queries. This data-distillation methodology is expensive and time-consuming. If it were possible to automatically create word transcriptions with reasonable accuracies, it would then become feasible to economically extract the desired genealogical facts and associations through subsequent natural language processing techniques. Our efforts are motivated by this overarching goal.

We here present a system for transcribing historical documents which we believe can achieve a high performance bar in the near term. Our system leverages, as its core pattern recognizer, the HMM version of the Kaldi *speech* recognition toolkit [1] which has yielded some of the best results in the world on various *speech* recognition tasks [2]. Other researchers have tried to extend Kaldi to non-speech domains such as Arabic handwritten recognition [3] and Chinese handwritten text recognition [4]. Yet to our knowledge, Kaldi has not been applied to machine-set printed documents – and especially not to historical newsprint.

The application of Kaldi to historical newsprint is, by itself, novel. Yet we believe we have a number of other noteworthy elements in this effort. For one, we have created a corpus for training Kaldi which contains almost five million words of human transcription of historical newspaper image snippets. We are not aware of a larger corpus of this kind of data. Also, our process of transforming images and converting snippet transcriptions into data that is consumable by a speech recognition engine requires multiple steps. For one of these steps – line segmentation – we have extended recently-published work. Also, though word recognition-based systems often are thwarted by out of vocabulary constraints, we have implemented a partial-word process which helps to partially overcome some of these deficiencies.

In this work, we lastly compare our results to those of three different available systems: Tesseract [5], Abbyy Fine Reader 12 [6], and OmniPage [7]. Our evaluation is quite interesting in that it shows performance against two distinct data sets. The first evaluation is a large, but randomly-selected collection of newspaper image snippets consisting of about 48K test words. The second is a larger 74K-word diachronic evaluation set which contains typically 20 test images from each of 23 decade-like time frames that cover over 200 years and that were selected by individuals who were not involved in the system-building tasks. We show that, despite the fact that our system is still in its infancy and there are rough edges still to iron out, it can yield performance that is comparable to or which surpasses well-established systems.

2. Data Corpus Descriptions

Before providing details on the newsprint transcription process itself, we first describe the corpus on which we are building the tools. Our collection consists of almost 38K image snippets of American newspapers dating back to the 1730s and it represents almost *five million* human-transcribed words (30.5M characters including white space). This collection may be one of the largest training sets in existence for historical newspapers.

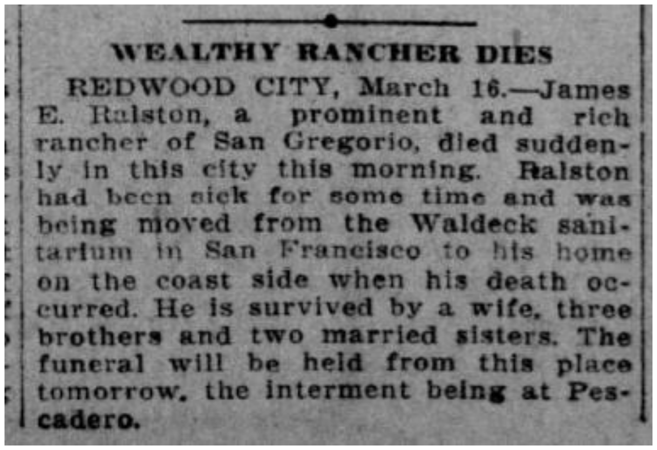
Although we are primarily interested in documents of a genealogical nature (which include obituaries, birth/marriage/death notices, immigration reports, etc.), our corpus was selected to provide broad coverage of newspaper articles. This is because even non-vital-record articles (such as advertisements, crime reports, or public events stories) can provide biographical information that is valuable to a genealogical patron. On the flip side, reasonable transcription of genealogically non-beneficial stories can be provided to tools for filtering out such articles.
The bulk of our collection contains images from United States newspapers including content from the Library of Congress's *Chronicling America* website [8]. Chronicling America is a website which has attempted to preserve history through the digitization, automated transcription, and word-based search of historical newspapers.

The image scans for these newspapers were made by external organizations, so our algorithms need to respond to whatever quality of image we might receive. The documents are often black and white TIFF images, but some of the images have used compression (jpg or jpeg2000) and a large number of the images are gray-scaled. The image resolution is believed to usually be 300 dpi, but given the multiplicity of sources from which this data came, the resolution may vary. Additionally, the language of most historical newspapers in the United States has been English, but other languages such as Spanish, Portuguese, French, Italian, and German (including German Gothic) also appear. Our complete collection of data contains representations from all of these languages. However, to fairly compare transcription engines, we limit the evaluation sets to consist of only English documents.

As mentioned, the collection is a set of newspaper snippets. These snippets are not whole pages of newspapers, but rather, they are regions of newspaper pages which ostensibly have a single flow of text. More specifically, in order to avoid confusion during the human and computational transcription processes, we created a rudimentary image zoning capability which was designed to identify regions of the newspaper page where the text of a snippet can be read in only one way (for example, left-to-right and top-to-bottom). Exceptions to this were tables, and we asked to have these transcribed row-by-row. Since our automatic image zoner was only rudimentary, it often proposed snippets that did not meet our desired criteria; so humans selected auto-zoned snippets that did meet the criteria.

The selected snippets could be readily read by humans. Yet due to the automatic nature of their creation, many of the snippets contained truncated words, sentences, and/or paragraphs. Therefore, the human transcription process needed to account for this through the use of word fragments and the incorporation of a special symbol (⌨) to represent illegible information. Figure 1 presents an example of how these snippets might appear. The image is legible but it is dark, has streaking, vertical bars on the left and right sides, and has weak inking in some spots. These kinds of phenomena – and others that are definitely more severe – are common in our collection. So we asked transcribers to just transcribe the words they see and disregard most blotches, salt-and-pepper backgrounds, stray marks, and other non-intentional image features. However, they could use symbols such as █ and ▓ to represent regions of blackness, and they were asked to use the vertical bar symbol (┃) to indicate the presence of vertical lines.

Figure 1. Snippet example (see [9])



WEALTHY RANCHER DIES
REDWOOD CITY, March 16.—James
E. Ralston, a prominent and rich
rancher of San Gregorio, died sudden-
ly in this city this morning. Ralston
had been sick for some time and was
being moved from the Waldeck sani-
tarium in San Francisco to his home
on the coast side when his death oc-
curred. He is survived by a wife, three
brothers and two married sisters. The
funeral will be held from this place
tomorrow, the interment being at Pes-
cadero.

In addition to marking image phenomena, textual font issues could be marked as well. Italicized words, small capitals, bold, script, Fraktur, and other kinds of font issues are marked by placing each such word or subword in an html wrapper. More specifically, to render the phrase "*this text is italics*," the transcribers would write "<i>this text is italics</i>." Interestingly, to meet demands such as texting, Unicode recently released the Mathematical Alphanumeric Symbols table (U+{1D400} through U+{1D7FF}) which represents standard text characters in bold, italics, script, and so forth. We therefore map these HTML-wrapped text regions into their corresponding Unicode equivalents.

From the transcripts, we randomly selected a development set and one of our evaluation sets. The two sets were selected to have approximately 50000 words in each. After noting that some of the randomly-selected *evaluation* files were non-English or multi-flow, these were culled out leaving an evaluation set with 344 documents having 47.5K words. The test set selected by this strategy will be referred to hereafter as the "Randomly-Selected Test Set." This set is probably most indicative of a system's actual performance in practice since it should roughly represent the actual distribution of newspapers in a broad historical inventory.

We are also interested in understanding how systems perform with newspapers across time, despite the frequency of occurrence. Thus, we created a second set of evaluation data to study this temporal aspect. This test set was selected by individuals who were not associated with the engineering aspects of the project. They mined newspaper pages that spread across over two-hundred years and they specifically selected multiple image snippets from each of 23 contiguous decade-like time bands. More specifically, they identified 10 snippets from pre-1800s, 10 snippets from 1800-1809, and then 20 for every subsequent decade until the 2010s. This set contains 74.4K words (where the number of words per time frame is indicated later in Table 3). This evaluation set we refer to as the "Diachronic Evaluation Set."

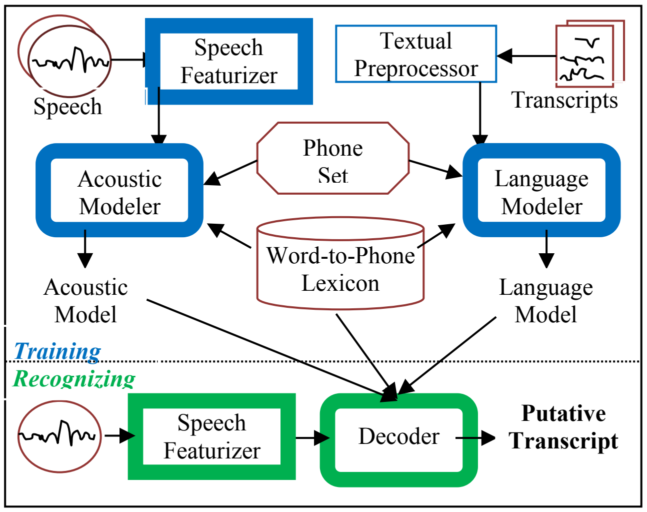
Our training data consists of the remaining 35K documents. We only use as *image* training a subset of the data upon which we have verified our preprocessing stages are functioning appropriately. Likewise, we use any of the images' *transcripts* for language training as long as the resultant model fits into memory. At the time of writing, we are using about 8K images and 23K transcripts for training. As our systems continue to mature, we expect to leverage all of the available training data.

3. Overview and Rationale

Our process for doing the actual newsprint transcription is accomplished by repurposing an existing *speech-to-text* toolkit, Kaldi [1]. Though this may seem unusual, automation to process speech and automation to process images have a number of similarities. To ensure that all readers have a common understanding of how speech-to-text recognizers work and why the use of such could be beneficial for the recognition of newsprint, we here provide an overview.

3.1 Speech Recognition Overview and KALDI

Figure 2. Speech-to-Text Training/Recognition Flow



The basic workings of a speech-to-text trainer (blue) and recognizer (green) are depicted in Figure 2. The externally-provided inputs to a speech recognition trainer (shown in brown) are (1) a collection of pre-diarized audio (i.e., audio where the regions of human speech have already been identified); (2) a collection of texts where each utterance of the diarized speech has a corresponding human-provided transcript; (3) a list of "phones" or units that are the individual sounds that form the basis of the words in the language; and (4) a dictionary which maps words in the language to their constituent phonetic representations.

During training, the textual input data is typically converted into statistical *n*-gram word counts and probabilities. In terms of acoustic processing, the training and recognition phases both begin by transforming audio files into numerical *feature vectors* upon which they each do subsequent statistical computations. In training, the information from these feature vectors passes through multiple layers of analysis to yield a model that represents the acoustic portions of the data. During recognition, comparisons are made to these feature vectors, in concert with the linguistic probabilities, to hypothesize a putative transcript for newly-presented audio files.

In Figure 2, the thick boxes are key components that are included in the speech recognition toolkit, Kaldi. The toolkit not only provides core system-building functionality, but it includes start-to-finish recipes for how to build speech recognizers for key types of audio (broadcast news, telephony, etc.). The builders of Kaldi have tried to incorporate the current best practices for each stage of statistical analysis. For this key reason, we, like others, have sought to use Kaldi for applications outside of speech recognition. We leverage Kaldi's "CallHome" recipe as our system's flow.

3.2. Why use Kaldi for Printed Images?

We previously indicated that Kaldi, to the best of our knowledge, has not been applied to the recognition of printed images, and it is certain to be the case that the *HMM* version of Kaldi has not been applied to historical newsprint. This noticeable omission may largely be due to the assumption that the recognition of printed images is straightforward and that character-by-character OCR can yield high performance for well-printed documents. Moreover, character-based recognition has the advantage that as long as the images upon which it is to run stay within a specified language, there will be no out-of-vocabulary concerns (i.e., there will be no words that the recognizer is not able to produce).

Unfortunately, historical newsprint often fails to meet these basic constraints. Scanned historical newspapers are not guaranteed to be clear, well-printed, well-aligned on the page, or well-scanned. Figure 1 from earlier, for example, contains the words "some time" which are easily read as words when viewed in context. Yet when processed as individual characters, these regions could easily be mis-transcribed as "som**o** tim**o**." For many of the pre-1900 image snippets in our set, there are regions of text which almost cannot be transcribed without some understanding of word usage. This suggests that recognition at a word level, particularly through a robust system which handles noise and huge variations of symbol sets, could be advantageous. (It should be mentioned, though, that nothing precludes us from using Kaldi to do character-by-character recognition.)

There is another value to us in the use of Kaldi. Our eventual goal (beyond the focus of this paper) is to build a system for transcription of historical images which is agnostic to whether or not the image was originally printed, handwritten, or a mix of both. For genealogical purposes, this has huge benefit in that a recognizer could conceivably run and distill out all the relevant textual information independent of the kind of text in the image on which it is being run. Since we are using the elements of the Mathematical Alphanumeric Symbols (MAS), it becomes almost seamless to transcribe mixed initiative documents because one can use regular alphabets for printed documents and use the script characters of the table to transcribe handwritten words.

A last and perhaps best reason for the use of KALDI is that it is a well-created toolkit. Its developers have created recipes which allow one to use various recipes for processing from raw speech all the way to recognition. The toolkit also readily leverages whatever parallel process the user might be able to access. The toolkit likewise benefits from continued development – such as the recent incorporation of neural network functionality which augments its original hidden Markov modeling. Kaldi performs forced alignment as a normal process, so there is no need to give the system any more than transcribed lines of text and corresponding image fragments and Kaldi will determine how the individual symbols span across those fragments.

To transform Kaldi into a newsprint recognizer, we only need to swap out a few components identified in Figure 2 with our own customizations while at the same time adding in the core elements that are required of any system. In particular, we need to supply images in lieu of audio along with their transcripts; we need to use printed phenomena like punctuation, letters, numbers, etc. as substitutes for phonemes; and we need to supply our own signal featurization and text preprocessing. The "heavy lifting" which is in the pattern recognition and statistical analysis can then be offloaded to Kaldi. After the models are built, one need only featurize a new incoming image and feed these features and the models into Kaldi and the output will be newsprint transcription.

4. Transforming Kaldi for Newsprint

Each component of the transformation process warrants some detail – particularly given that some of these steps provide novelties over the existing state-of-the art. We already described the data transcription process, so we present here the details of the "phoneme" set, multi-step featurization process, and vocabulary development. At all points in our transformation processes, we have elected to use general-purpose techniques that are designed to work on the image regardless of whether the image is printed or handwritten.

4.1 Phonemes to Print Phenomena

In speech recognition, the basic unit would be a "phone" or a "phoneme." For printed documents, our phonemes can be any symbol we wish to use. Our "phoneme set" contains all the alphanumeric characters as regular characters, usual punctuation marks, and the various noise symbols indicated previously. We also include accented characters to cover the language space of the historical US newspapers; long s (or ſ) and its italic equivalent (*ſ*) which are observed frequently in historic newspapers; and a number of characters representing various oft-used fractions. Also, as mentioned before, our system incorporates italicized alphabetic characters, bold characters, Fraktur (German gothic) alphabetic characters, and small capitals; and it can optionally incorporate script and other MAS character sets.

In addition to the characters themselves, we have allowed other characters which are augmented with special symbols representing left or right bindings. We use these to reduce the size of the recognizer's dictionary while still being able to properly compose words after recognition as will become clearer later on (see Section 4.4.2). For the moment we will just say that the phoneme "2↠" means the right-bound numeral two, "↞2" is a left-bound numeral two, and "↞2↠" means a two that is bound on both the left and right. The binding means that the symbol cannot occur in isolation so it must consume the space character in the direction in which it is pointing. So if the recognizer eventually reports symbols together such as "2↠ ↞7," the final result will be interpreted as "27."

Finally, as a normal part of Kaldi, phone sets are augmented to account for word position. More specifically, a phone X can be augmented with an attachment indicating whether it occurred at word beginning (_B), ending (_E), in the middle (_I), or as an independent character (_S). Our system allows for all of these variations (except for low-frequency phenomena, which are only allowed to be generated in the "_S" mode).

4.2 Substitution for Diarization

Speech that feeds into a speech recognizer must be diarized, either through a human or an automatic process. Diarization has the role of indicating the start and end times where a speech utterance actually occurs in the audio signal. Since images are two-dimensional, the image analog of diarization would be to indicate the regions of the image where individual rows in the text exists, as well as any rotation, skew, and odd warping information about those rows. Since we are processing snippets of newspapers, which are scanned flat, perspective distortion issues are not a major concern. We expect, likewise, that localized warping issues will have to be accounted for by the "acoustic" models (in particular, the Gaussian mixture models and hidden Markov models that Kaldi will produce). This means we must provide capability primarily for rotation and line segmentation.

4.2.1. Rotation

We use common techniques to tackle image rotation. In general, rotation of image snippets, if any, is only modest due to the likelihood that the newspaper pages at digitization time were carefully laid out to be oriented properly. That said, there are single-flow subimages in newspapers which flow differently than the page as a whole. For example, words can be rotated 90 degrees and read from top-to-bottom, left-to-right. Hence, in our rotation analysis, we constrain rotation to be within 2.5 degrees of the main cardinal directions. In practice, this is adequate although there are a number of snippets from amidst the entire collection which actually read at a 45 degree angle; so we will opt to mistranscribe those rather than risking the chance of over-rotating other images.

To compute the angle of rotation, we compute the Hough transform of the image [10] with half-degree increments from which we identify the N largest transform values (where we choose N=20). We compute the weighted average of the angles associated with those top N values and use those values as the weights. Then the angle from amongst the N maxima which is closest to the weighted average is selected as the rotation angle.

4.2.2. Line Segmentation: Baselines

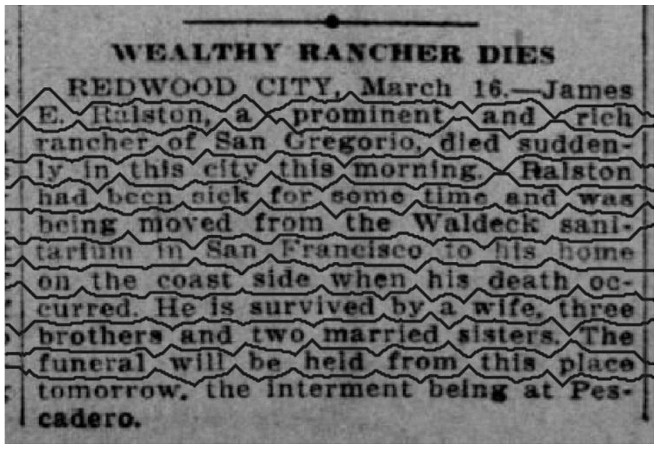
For line segmentation, we extend the work of Arvanitopoulos and Süsstrunk [11] which they presented for seam carving of historical gray scale images. Since their technique was designed to work with handwritten documents, this seems appropriately general for a system designed to run on newsprint and, eventually, on handwriting or mixed initiative documents.

The first goal of their algorithm is to identify the image's *medial seams.* One could think of these as drawing a connected arc through each line of text which passes through the centers of mass of the rows as closely as possible. Another way of thinking of them is as finding the peaks of "activity" or "energy" mountain ranges where energy is the amount of blackness in the image. Once medial seams are identified, the Arvanitopoulos-Süsstrunk (or "A-V") system uses a dynamic program to walk backward between paired "activity mountain ranges" trying to stay in the areas of lowest blackness (an "activity valley"). These activity valleys they call *separating seams*.

Their medial-seam finding process treats an image as a set of V vertical strips. They compute a Sobel filter on the image to create character halos. Then they compute each vertical strip's *profile* which is the amount of blackness in each row of the strip. Each strip's profile is smoothed, and the maxima are then identified. Medial seams are computed piecewise by connecting the row possessing each maximum, m, in the particular strip to the maximum in the subsequent strips whose row value is closest to m.

After the medial seams are identified, Arvanitopoulos and Süsstrunk compute the separating seam pixel by pixel by cutting a path between two medial seams. This process uses a dynamic program to attempt to identify a path with least energy-transition costs. We refer the interested reader to their paper for the specific details beyond what has been described here. Figure 3, though, shows their matlab code optimized for and applied directly to the

Figure 3. A-V System code applied to Figure 1's image.



WEALTHY RANCHER DIES
REDWOOD CITY, March 16.—James
E. Ralston, a prominent and rich
rancher of San Gregorio, died sudden-
ly in this city this morning. Ralston
had been sick for some time and was
being moved from the Waldeck sani-
tarium in San Francisco to his home
on the coast side when his death oc-
curred. He is survived by a wife, three
brothers and two married sisters. The
funeral will be held from this place
tomorrow, the interment being at Pes-
cadero.

image snippet from Figure 1. As can be seen, their algorithm – though designed for handwritten documents – clearly has potential application to newsprint as well. Yet there are a number of limitations to their algorithm which we need to overcome for use with newsprint documents. For example, note that in Figure 3, the first and last lines did not get separated – a common phenomenon. In fact, their approach has challenges in the following situations which potentially yield the errors listed in parentheses below:

a) textual lines have limited content at the beginning, middle or end of the line (no seam);
b) lines use multiple fonts (extra medial seams);
c) document uses both large and small fonts (extra medial seams)
d) the image only contains single big characters (myriad extra seams);
e) the image is framed in some manner (extra medial seams);
f) there are lots of stray marks on the page (extra medial seams and/or separating seam process crosses medial bounds);
g) though not an error, their separating seams are highly jagged which adversely affects featurization which will be discussed in section 4.3; and
h) blurry images, bleed through, and dark images (extra or not enough seams).

4.2.3. Line Segmentation: Extensions

We have not remedied *all* of the various situations that occur in their algorithm, but our techniques have removed significant amounts of error and some jaggedness. We attempt here to identify our modifications:

Linear projections: Perhaps one of the most commonplace errors of the A-V system is failure to cut seams for textual components that do not span much of the page (see "(a)" above). This can occur at the end of a paragraph, some isolated words in a centered document, or words such as page or author information on the right side of the page. In these situations, the vertical strips that must be profiled in order to find medial seams are largely empty. To remedy this, we perform interpolated linear projections from the regions that do have content into the regions that do not. This allows for medial seams which span the entire image.

Connected components: We noted that connected components in the image can be used interchangeably as a surrogate for energy profiles. Yet these can also strengthen segmentation. When two or more medial seams pass frequently through the same sets of connected components, we know that only one seam is needed. We use the top-most of such medial seams to identify the separating seam that should be above the medial; and we use the bottom-most one to help identify the separating seam below the line of text. This process helps with cases (b) through (e).

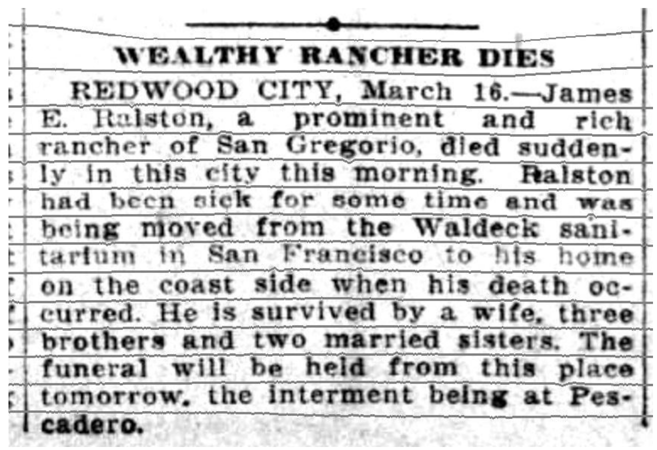
Image Bleaching: although the A-V approach was designed to work without binarization, bleed through and other darkness artifacts do have impact on the algorithm. We first apply conservative histogram normalization to the images to bleach the image [12]. This process helps with case (h), but it also helps establish better connected components.

Energy thresholding: some spurious medial seams are produced in regions of low activity. This can be alleviated to some degree by setting a minimum threshold for the specification of a profile maximum. This can also help with (b)-(d) as well as with (f) and, to some degree with (e). Additionally, we compute the statistical spread of data along each medial seam and throw out those that have high variance but low distributional activity.

Balancing between seams: situation (g), where separating seams are jagged as in Figure 3, is not necessarily an error. Yet for the purposes of our featurization, we would like somewhat smoother curves. To accommodate this, we add an element into the energy cost issue of the separating seam dynamic program. To be concrete, suppose we are trying to draw a separating seam from the right side of the page to the left and suppose we are at pixel (x,y). The A-V algorithm asks, "Do we expend less energy to move from (x,y) to (x-1,y-1), to (x-1,y), or to (x-1,y+1)." In many cases, the cost would be equivalent regardless of which place we would step – and this yields the signature jaggedness. We add a cost to each which increases as those points are further from the midpoint of the medial seams on either side of the separating line we are trying to draw.

These various enhancements are shown in Figure 4. Figure 4 is our processed version of the image from Figure 1. As can be seen in Figure 4, the enhancements yield much smoother lines which tend to be fairly well-centered between the medial seams. We also see the outcome of the bleaching (which was not required for the segmentation, but which helps featurization). Though this image was selected randomly and happened to have interesting line segmentation properties, the positive outcomes shown here do occur as general improvements throughout our data sets.

Figure 4. Our line segmentation applied to Figure 1.



WEALTHY RANCHER DIES
REDWOOD CITY, March 16.—James
E. Ralston, a prominent and rich
rancher of San Gregorio, died sudden-
ly in this city this morning. Ralston
had been sick for some time and was
being moved from the Waldeck sani-
tarium in San Francisco to his home
on the coast side when his death oc-
curred. He is survived by a wife, three
brothers and two married sisters. The
funeral will be held from this place
tomorrow, the interment being at Pes-
cadero.

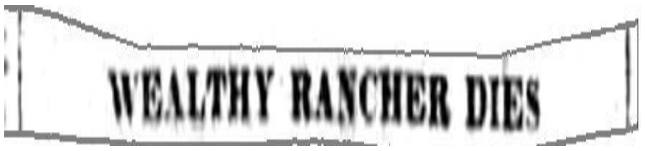
Even so, there still remain line segmentation issues that we have not fully resolved. The key offenders include situations of (a) multiple fonts coupled with centered verbiage intermixed with horizontal lines; (b) text which is very blurry; and (c) text which is very close together. These kinds of phenomena do occur in our test set and result in lesser accuracies than if the line segmentation were done perfectly.

4.3 Image Featurization

In order for Kaldi to function, we need to convert these images into vectors of numerical values – feature vectors. To do this, we make use of both the medial seams (MS[i]) and the separating seams (SS[j]) provided in the line segmentation step. Our feature vectors contain 41 dimensions whose contents we will here describe.

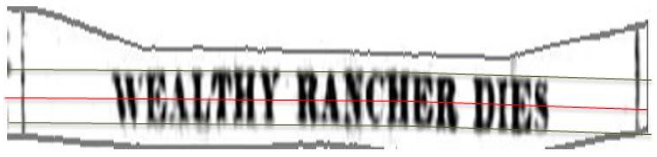
For simplicity, we will first say that SS[0] is the top of the image and SS[N] is the bottom of the image. Also, we declare B[k] to be the kth band (for k=1,...,N) which is an image swath that is bounded below by SS[k-1] and below by SS[k]. If the image width is 2W, then for B[k] we will be creating W feature vectors. Figure 5 shows the B[1] image swath of Figure 4B.

Figure 5. The B[1] swath from Figure 4B.



We seek to trim the band so that its edges are smoother and upper and lower white space has been removed. Specifically, for each band, we compute its total density, or the amount of blackness in the band. We also compute the density along the medial seam that passes through the band (D[0]), the density of elements one pixel above (D[-1]) and below the seam (D[1]), two pixels above (D[-2]) and below the seam (D[2]), and so on until we have reached the top and bottom of the band. Suppose D[u] is the density at the uppermost pixel distance from the medial seam and D[l] is the density at the lowermost pixel distance. We select the lesser of D[u] and D[l] and throw away all points that correspond to that particular distance from the medial seam; we reset the uppermost and lowermost points accordingly; and we repeat. This pruning process continues until the density of the remaining band is 99.9% of the original band. This is depicted in Figure 6, where the lines above and below the characters indicating putative trimming boundaries.

Figure 6. The swath from Figure 5 with trimming regions.

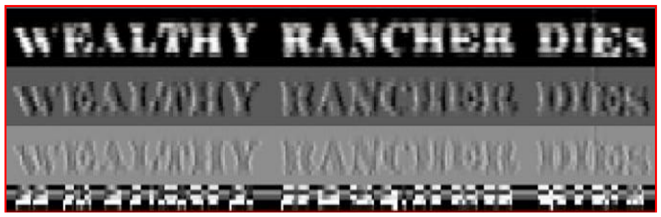


We compute the features from this remaining band. We create a sliding, rectangle-like window that we pass over the band which has width of three pixels and a height that is determined by the band's upper and lower bounds at the given columns. We step the window forward in two-pixel increments.

At each stop of the sliding window, we partition the window vertically into 12 equally-spaced cells, and compute the maximum density across all cells. Then, as some of the features, we compute the per-cell ratio of **density** to maximum density, the **derivative** of this quantity, and the **double derivative** as well. The practice of creating feature vectors by using base values coupled with first and second derivatives comes directly from speech recognition techniques. The above information, then, constitutes 36 of the 41 dimensions of our feature vectors.

We derive the other five features using descriptions in the literature. Chherawala, et al [13] identify a number of different kinds of features (citing the work of others) that one might use for Arabic handwriting recognition using neural networks and they report the performance of these by classes (distributional, concavity, etc.). We identified five features from among those which we felt could be applied to a hidden Markov model based system. One of these is the **distance** between the uppermost and lowermost black pixels in each window. The next two are the points at which the **uppermost** and **lowermost** black pixels occur. The last two should be beam height invariant: (1) the rate of **alternation** of black to white pixels within the window, and (2) the **average density** of the window. The composite of these features is shown in Figure 7.

Figure 7. Pictographic representation of feature vectors



There are two additional components that we have noted which have been beneficial to the featurization process. One of these is the use of projection techniques. We use principal component analysis on the main 36 features to project them into a 20-dimensional subspace. Though this provides only very small gains in performance, it does result in much smaller feature vectors and storage costs.

The second of these has proven to be extremely helpful: automatic resizing at evaluation time. The optimal row width is about 36 pixels (+/-5 pixels) from top to bottom. When the width is narrow than that, recognition tends to fail. Consequently, we have implemented an automatic resizing of the image which multiplies the original image by whatever magnification is required to get the image to have 36-pixel widths. In addition to providing more information for the feature vectors to leverage, this rescaling makes the line segmentation much smoother and more accurate.

4.4 Special Handling of Linguistic Data

As a typical component in speech recognition, one needs to create a lexicon with the words of the language and their one or more corresponding pronunciations. For example, in a speech recognizer, one might find the dictionary entry for "hat" as:

hat	h-æ-t .

For images, the dictionaries can be significantly easier to create because there is usually a one-to-one correspondence between the word and it pronunciation. That is, for images, we could use:

hat	h-a-t .

4.4.1 Space Handling

In addition to handling words of the language, we also need to account for *spacing*. A simple way to compensate for spacing might be to put spaces directly into the transcript, but this has negative impact on the language model. That is, the probability of "hat" will be much higher if we know the previous words had been "cat in the" than it would have if we had merely seen the three preceding symbols of "<space> the <space>". This suggests that we want to incorporate the spaces directly into the word pronunciations.

Kaldi supports the idea of probabilistic dictionary entries. So we assume that each word will typically be followed with a "<space>" phoneme if it occurs in a chain of words. We then compute the probability that a given word like "hat" appears in this chain in the middle of a line (hat <space>), how often it will start the line (<space> hat <space>), and the frequencies of non-spaced situations. Our lexicon will then have four representations of most words, which increases the complexity of the model but allows spacing to be handled in a fairly straightforward way. For "hat," our word lexicon reports:

hat	0.771225	h a t <space>
hat	0.146796	h a t
hat	0.069837	<space> h a t <space>
hat	0.012141	<space> hat

Our *core lexicon*, which is based only on transcripts for which we are also using the images, consists of 88K unique terms. Yet the probabilistic spacing elements increase the number of variations to 264K.

It should be noted, though, that we can opt to transform our transcripts into characters only and use a character-based system in addition to a word-based system. A character-based system alleviates concerns about space, since space can be represented directly as a symbol to recognize. On the other hand, this significantly weakens the language model since much of the context is lost when one only considers *n*-grams of characters instead of words.

4.4.2. Bound Word Fragments

The 88K unique terms in the word lexicon seems like a significant number. For speech recognition of telephony or other controlled-vocabulary scenarios, this may be adequate. Yet given that we are trying to apply our algorithm to historical newsprint across centuries, 88K is a fairly small number. A word-based recognizer with this size of lexicon is likely to end up with a significant out-of-vocabulary (OOV) problem. This is especially true as it relates to personal names – which are the kind of information we are most interesting it identifying.

One way to increase the vocabulary is to leverage a look-aside corpus. In our case, we are fortunate to have tens of thousands of transcripts which have not been included because we were not yet confident that our preprocessing of the corresponding images was optimal. We could incorporate all the terms from these auxiliary transcripts. Though larger, this might still be too small of a set.

If we think about the remaining collection of text, we will note that many of the words occur exactly once – and, frankly, they may never occur again. So it does not make sense to add all of these words to the lexicon. Yet we need to be able to hypothesize such words when they arise. So we add all words that occur twice or more, and then we handle the singletons in a special way.

We compute the prefixes and suffixes of the singleton set. For each singleton, we add its longest non-singleton prefix to the dictionary augmented with the right-bounding marker, ↠, which was referenced earlier. We then replace the prefix in the word with a left-bounding marker, ↞. We can continue to decompose the word by pulling off suffixes as well as ↞-starting prefixes and ↠-ending suffixes. This process makes it so that subcomponents of singletons are added in ways that we might hope can later stitch together sub-pieces to form a word we had never before seen.

For example, since our collection is a historical one, "Barack," the given name of the current president of the United States, does not occur in the lexicon. However, "Bar↠" does appear in the word-partial lexicon as does "↞ack" :

Bar↠	0.035258	<space> B a r↠
Bar↠	0.964742	B a r↠
↞ack	0.910181	↞a c k <space>
↞ack	0.089819	↞a c k

With these word partials, if the system hypothesizes the term "Bar↠" as opposed to simply "Bar," it is saying that there should be no space after the ↠. So "Bar↠ X" will be rendered as BarX; or, in the case above, Bar↠ ↞ack will be converted into "Barack." Our final extended dictionary contains 467.1K terms of which 168.8K are unique. Of these, 140.8K are bounded prefixes, 28.8K are bounded suffixes, and 12.9K are bounded infixes.

5. Experimental Results

The main question now is: does this mechanism for historical newsprint recognition actually provide a beneficial functionality? In particular, as sub-questions, can recognition be performed this way? How does the recognizer compare to commercial and other open-sourced OCR systems when applied to the same historical data sets?

5.1 Existing Available Systems

If this system were the only one of its kind, then even modest performance would be useful. Yet since OCR functionality has existed for decades, it only makes sense to compare its performance to that of other existing systems. There are a number of available commercial systems, but Abbyy Fine Reader [6] and Nuance's OmniPage [7] are strong players in the commercial arena. In the open source community, Tesseract [5] is a frequently-used system which is trainable but also comes equipped with its own well-worked models of English. We have acquired copies of each of these three systems and we use them to compare their performance to our system. In particular, we use Abbyy 12, Build 12.0.101.467; OmniPage 18.0; and Tesseract 3.02. Of course, it should be made clear that these systems were designed for general OCR, so they could potentially be at a disadvantage in a head-to-head evaluation restricted specifically to historical newsprint. On the other hand, this increases the value of building a customized system for historical documents such as ours.

There are other available systems, but we do not use them in this evaluation. Ocular [14], for example, is a newly-released system which was designed to address issues of historical newspapers and shows promise for our domain. Its creators applied it to two small historical document collections – one collection having 10 images and the other having 20 images. We have downloaded a copy of this system's source code, but time constraints have not yet afforded us the opportunity to compile and test its performance.

Additionally, Schwartz, et al [15] presented the idea of converting a speech-to-text engine into OCR as far back as 1996 when they used the BBN BYBLOS speech recognition to do OCR of Chinese documents. They had a number of system iterations and were successful in porting their system to Arabic, English, Pashto, Hindi, and Japanese [16]. In their cases, if our understanding is correct, they ported their speech recognition system to do character-based recognition specifically. We do not have access to their system, so we cannot test the performance of their English system on our historical texts.

5.2. Evaluation Setup

To score word recognition, we need define what a "word" is. For some researchers, a word might be whatever is between white space. Yet we are interested in feeding the output of this system into a natural language processing (NLP) system, so we would like to make sure the system identifies NLP-style tokens. In NLP, most punctuation will be treated as tokens in its own right (with some exceptions being common abbreviations like Mr., Mrs., multiple dashes, etc.), so we follow suit here. We therefore transform a sequence like "John Smith, 30, died" into the token sequence 'John Smith ↞, 30 ↞, died' and we do this both for the evaluation data and for all system outputs.

In addition, we convert long punctuation repeats into a "three-or-more repeats" token. For instance, we transform _______..._ into ___+ and we do the same for sequence of periods, dashes, or equal signs. If vertical bars (representing black page borders) are in the transcript, we strip these out for evaluation. If the transcript uses italics, etc., we convert these symbols into their base letters in order to fairly evaluate the various systems. Lastly, we treat long S symbols (common in 18th century newspapers) as just "s."

To score systems, rather than doing so line by line (which would be our preference to see per-line performance) we instead concatenate all lines together into one long line per document. The rationale for this is that systems may have produced fewer or more lines of text than the original transcribers. Although it is possible to modify the system results so they perfectly align with the original transcripts, that process is expensive. The concatenative approach is an alternative for removing misalignment concerns.

There are also some optimizations to scoring which we make to all automatic systems with the goal of improving their individual accuracy numbers. For example, we allow the system to convert regions where a system proposes individual or paired characters (such as "W OR D S") into a concatenated token "WORDS." Also, the systems sometimes produce symbols as output which do not exist or are rare in the test set, so we eliminate these when they are generated. The result of these changes is small (usually less than 1/2 %), but we wanted to do our best to make sure that systems are evaluated as fairly as possible.

5.3. Evaluation #1: Random Selection Based

In Section 2, we described the data sets upon which we could evaluate the various algorithms. Recall that the first was a set that was selected randomly from the non-training images of our corpus. It consists of 344 images and 47.5K words, so it is a substantial set and demonstrates the power of these systems. The second test set was described as a diachronic test covering over 200 years. In this section, we will first talk about performance on the random set.

In Table 1 below, we report the system performances as applied to the Random Evaluation set. In the table, we first indicate each of the external systems along with their word accuracies (100% minus word error rate) and their character accuracies (100% minus character error rate). Lastly in the table, we show our system's performance, which we refer to as **Athilos** (for "**A**utomatic **T**ranscription of **H**istorical **I**mages **L**everaging **O**pen-sourced **S**peech-to-text.").

Table 1: System performances on Random Evaluation set

System	Word Accuracy	Character Accuracy
Abbyy 12	75.14%	90.50%
OmniPage 18	73.53%	89.76%
Tesseract 3.02	24.04%	73.33%
Athilos (our system)	83.04%	92.08%

The Abbyy system has been developed since at least the 1990s (see [17]), and OmniPage had its foundations in the late 1980s (see [18]), so these systems have had many years for improvement. Yet the primary focus for these commercial systems has likely been to provide exceptional accuracy for the *modern* business place. Table 1 suggests, though, that they both provide admirable quality even for non-modern documents. Tesseract, which is a common staple in the research community, did not perform as well as the other systems; but it is a trainable system which likely would achieve higher accuracy if it were trained as opposed to being run out-of-the-box. Our system has word and character accuracies that exceed the performance of the other systems on this particular historical newsprint test set, which suggests that it might be able to play a beneficial role in automatic transcription when it comes to more historical texts.

5.4. Evaluation #2: Diachronic Evaluation Set

We can also score the various systems diachronically to see how they might perform on the US newspapers across time. We see the results of the diachronic evaluation in Table 2.

Table 2: System performances on Diachronic Evaluation Set

System	Word Accuracy	Character Accuracy
Abbyy 12	71.09%	88.94%
OmniPage 18	66.68%	83.59%
Tesseract 3.02	18.09%	66.41%
Athilos (our system)	76.24%	87.54%

As mentioned, the image snippets for this diachronic set were chosen by individuals who were not associated with the engineering efforts. Consequently, some of the articles could be particularly challenging or the image resolution or quality may be different from what we have seen in the bulk collection. The 4-7% absolute drop in word accuracy for each system when applied to this set is suggestive of some limited quality differences. Still, Athilos provides the optimal word accuracy under this condition. Character accuracies also degrade significantly for all systems. In this case, though, Abbyy 12 degrades the least – resulting in it becoming the best performer for character accuracy.

We can explore this set deeper to get an improved sense of what is happening for each system on this collection. Recall that the diachronic test set contains 20 images in all but the first two time frames (on the other two time frames each have 10 snippets), so there may be scarcity issues resulting in a system perform surprisingly well in one decades and especially poor in another. Yet this analysis still reveals rough trends in performance across time. Table 3 illustrates each time frame, the number of words for each time frame, and the word-level accuracies of each system per time band.

There are a number of observations that we can make. We note that our system, Athilos, performs comparably if not better than the other systems from the earliest time bands until about the third quarter of the 20th century. Yet in the last quarter of the 20th century and beyond, Abbyy 12 and Omni 18 begin to really shine.

Table 3: System Word Accuracies by Diachronic Time Frame

	# words	Abbyy 12	Omni Page 18	Tesseract 3.02	*Athilos* (ours)
1700-99	858	57.99	56.11	-3.25	66.67
1800-09	1550	49.20	46.51	-10.94	64.38
1810-19	3271	55.89	49.06	-29.18	67.63
1820-29	5468	76.64	74.31	2.78	78.27
1830-39	3830	55.31	55.34	9.00	76.07
1840-49	4544	68.13	60.28	12.84	76.07
1850-59	3948	57.66	59.57	-3.50	72.56
1860-69	3437	64.38	60.23	0.90	75.34
1870-79	4910	77.27	78.98	30.63	77.03
1880-89	1739	58.42	61.89	6.61	74.53
1890-99	4086	64.11	47.07	-11.81	77.99
1900-09	2705	63.89	52.86	-0.52	77.32
1910-19	3396	72.99	74.96	27.59	72.65
1920-29	4281	69.26	72.19	8.49	83.70
1930-39	3726	79.11	77.30	40.50	79.89
1940-49	2914	66.94	57.32	16.05	67.42
1950-59	3430	73.89	63.64	29.41	78.86
1960-69	3031	73.90	72.97	45.66	70.48
1970-79	3283	71.22	64.48	32.10	74.74
1980-89	2089	89.93	86.57	71.23	81.70
1990-99	3835	91.79	78.67	58.03	78.98
2000-09	1732	85.24	84.79	52.53	83.18
2010+	2328	90.83	91.93	83.05	86.90

Tesseract's word accuracy also performs well for documents created in the last 5 years. In all cases, though, we see that the accuracy levels are currently fairly low for attempting to do the kinds of natural language processing (NLP) techniques we described earlier. It is our hope, though, that by adding in the remaining training data we have available, by improving our featurization which is still very much in its infancy, and possibly through system combination, we will be able in a short time to produce sufficiently high accuracies through Athilos to allow follow-on text processing.

5.5. Statements on Out of Vocabulary

We mentioned in Section 4 that out of vocabulary (OOV) issues would be of concern for a word-based recognizer. It turns out that the random test set has 2738 of 13630 *unique* tokens that Athilos has never seen. Allowing for word reuse, 2929 of the 59828 tokens were never seen or 4.90% of the data. Better said, using Athilos' full-term dictionary alone, one could not hope to get better than 95.1% word accuracy. Worse still, there is a general observation that every OOV influences the choices of word that is hypothesized to the right and left of it, yielding 1.5 to 2 *additional* errors per OOV [19]. If we assume the 1.5 factor, the OOV's would likely limit our maximum accuracy to be 87.75% -- a number which would be too low for NLP use.

The look-aside corpus reduces the out-of-vocabulary to 1511 unique words or 1597 by frequency, which is 2.67% of the data. This is a significant improvement, but it still means that the likely maximum performance for the Athilos would be 93.38%.

Fortunately, the partial words are able to cover the vast majority of the remaining words. Of the remaining OOVs, 174 of them are covered by a pair of partial words that are in the lexicon. An additional 306 are covered by the composition of three partial words in the lexicon; and 375 more are covered by four partials. Assuming the system could properly hypothesize a sequence of two, three, and four word partials, that would mean that 656 words would be missing – which is 696 OOVs by frequency, or 1.2%. This means that the likely maximum for Athilos could be as high as 97.1%. Such a level of performance, if achievable, would definitely lend itself to NLP and to the automatic extraction of genealogical relationships that were described at the beginning of this paper.

6. Comments and Synopsis

We have demonstrated that by leveraging the state-of-the-art speech recognition toolkit, Kaldi, and incorporating our own image transcriptions and customized image preprocessing steps (some which extend the works of recent publications), we are able to create a word-based tool for transcribing historical newsprint. We have demonstrated that our system performs comparably or better than commercial OCR engines that have been in development for decades; yet we have been able to assemble our system in a matter of months. We still have a ways to go before one could leverage our system to perform downstream NLP, but we are hopeful that with our additional data and system tuning we can attain these high performance levels. We also believe that if other researchers follow the processes we describe here, including the leveraging of Kaldi, they, too, should be able to assemble high-quality newsprint transcription systems.

7. Acknowledgements

The authors wish to express their appreciation to Jake Gehring and Rose Pierson for their efforts to acquire funding for and oversee the transcription of the newspaper image snippets used in this evaluation. Thanks are also extended to the anonymous reviewers and other editors for excellent comments and feedback.

References

[1] D. Povey, A. Ghosal, G. Boulianne, L. Burgat, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, YM Qian, P. Schwarz, J. Silovsky, G. Stemmer, K. Vesely, "The Kaldi Speech Recognition Toolkit," in IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, Big Island, Hawaii, 2011.

[2] C. Gaida, P. Lange, R. Petrick, P. Proba, A. Malatawy, D. Suendermann-Oeft, "Comparing Open-Source Speech Recognition Toolkits," OASIS, 2014.

[3] F. Stahlberg, S. Vogel, "The QCRI Recognition System for Handwritten Arabic," in Lecture Notes in Computer Science, pp 276-286, 2015

[4] R. Messina, J. Louradour, "Segmentation-free Handwritten Chinese Text Recognition with LSTM-RNN," in ICDAR, IEEE, 2015.

[5] R. Smith, "An Overview of the Tesseract Engine," in ICDAR 2007, pp. 629-633, Paraná, Brazil, 2007

[6] ABBYY, "ABBYY FineReader ," https://www.abbyy.com/finereader accessed 14 August 2015.

[7] Nuance, "OmniPage for PC", http://www.nuance.com/for-individuals /by- product/ omnipage/index.htm, accessed 14 August 2015.

[8] *Chronicling America*: Historic American Newspaper site, Library of Congress, chroniclingamerica.loc. gov

[9] J. Vonderlin, *ScreenShot4719*, Creative Commons/ Flickr: https:// creativecommons.org/licenses/by-nc/2.0/legalcode, 2010

[10] R Duda, P. Hart "Use of the Hough transformation to detect line and curves in Pictures," *Communications of the ACM*, 15:1, pp 11-15, ACM, New York, NY, 1972.

[11] N. Arvanitopoulos, S. Süsstrunk, "Seam Carving for Text Line Extraction on Color and Grayscale Historical Manuscripts," in ICFHR, Crete Island, Greece, 2014

[12] R. Gonzalez, Woods, E. *Digital Image Processing*, 3ed, 2008.

[13] Y. Chherawala, P.P. Roy, M. Cheriet, "Feature design for offline Arabic handwriting recognition: handcrafted vs automated," ICDAR 2013, Washington, DC, 2013

[14] T. Berg-Kirkpatrick, D. Klein, "Improved Typesetting Models for Historical OCR", in 52nd ACL, Baltimore, MD, 2014.

[15] R. Schwartz, C. LaPre, J. Makhoul, C. Raphael, Y. Zhao. "Language-independent OCR using a Continuous Speech Recognition System," in 13th International Conference of Pattern Recognition, Vienna, Austria, 1996.

[16] P. Natarajan, E. MacRostie, M. Decerbo. The BBN Byblos Hindi OCR System. Guide to OCR for Indic Scripts, pp. 173-180, 2009

[17] Wikipedia. "ABBYY", en.wikipedia.org/ABBYY, accessed 15 August 2015.

[18] Wikipedia. "OmniPage", en.wikipedia.org/OmniPage, accessed 15 August 2015.

[19] P. Geutner, M. Finke, P. Scheytt, A. Waibel, H. Wactlar. Transcribing Multilingual Broadcast News Using Hypothesis Driven Lexical Adaptation, " http://www.itl.nist.gov/iad/mig/publications/ proceedings/darpa98/html/bn60/bn60.htmDARPA, 1998.

[20] B. Taylor, V. Karasev, S. Soatto. "Causal Video Object Segmentation From Persistence of Occlusions," CVPR 2015, Boston, MA, 2015

Author Biography

The lead author and presenter of this paper, Patrick Schone, received a PhD in Computer Science in 2001 from the University of Colorado at Boulder. He has been working in research for over twenty years. His research has principally been focused on human language technologies throughout that time frame, including analysis, processing, and search of speech, text, and images.