Aida: Intelligent Image Analysis to Automatically Detect Poems in Digital Archives of Historic Newspapers

Leen Kiat Soh, Elizabeth Lorang, Yi Liu
University of Nebraska, Lincoln, NE
lksoh@cse.unl.edu, liz.lorang@unl.edu, yil@cse.unl.edu

Abstract

We describe an intelligent image analysis approach to automatically detect poems in digitally archived historic newspapers. Our application, Image Analysis for Archival Discovery, or Aida, integrates computer vision to capture visual cues based on visual structures of poetic works—instead of the meaning or content—and machine learning to train an artificial neural network to determine whether an image has poetic text. We have tested our application on almost 17,000 image snippets and obtained promising accuracies, precision, and recall. The application is currently being deployed at two institutions for digital library and literary research.

Introduction

Creating and making available digital libraries of historic materials continues to be a significant area of development for libraries, archives, and similar institutions. As storage has become cheaper, Internet and web access more ubiquitous, and users more likely to expect electronic access to information, the number, variety, and size of digital collections and digital libraries of historic materials has grown apace. It is now relatively inexpensive to create digital image copies of historic material, store those images, and make the images available. Yet, the work of describing and characterizing the materials to increase their findability and usability for researchers remains an expensive part of creating digital libraries, because of the human labor—and expertise—required to do the work effectively and well. As a result, the users of voluminous digital libraries often face significant difficulty finding materials of relevance for their research, because the types and nature of information they can query are limited, as are the ways that researchers can engage with the materials within digital libraries (Lorang et al. 2015; Green & Courtney 2015; DeRidder & Matheny 2014; Underwood 2014, "How to Find"). The challenge of locating materials of relevance is compounded when the collections are highly heterogeneous, both across items within a digital library and especially when individual items themselves are highly heterogeneous (e.g., scrapbooks, newspapers, magazines).

Artificial intelligence (AI) has already had important and lasting impacts on the development of digital libraries of historic materials with the advent and continued improvement of optical character recognition (OCR) processes. Applications of *computer vision* techniques are helping researchers to identify derivative works, such as reprintings, as well as visually similar works, and to recognize and identify faces in digital collections, among other applications (Bergel et al. 2013; Chung 2014; Oliveira, et al. 2017; Seguin et al. 2017; Smiths 2017). The information generated in these processes can then be used for deeper levels of information in large collections, aiding researchers in locating materials of relevance.

Other researchers are using *machine learning* to leverage information out of the electronic texts of materials in digital libraries. Information gleaned from these processes then can be added to the descriptive information about materials in digital collections or be made separately available as datasets. Underwood (2014, "Understanding Genre") describes using machine learning to identify genre at the page-level of each volume in a large digital library. And others have employed topic modeling as an avenue to explore the text of digital collections of cultural materials, both historic and current. The resulting word groupings and subsequent human labeling of topics can become another pathway into digital collections (Cain 2016; Hagedorn et al. 2011; Tze-I Yang et al. 2011).

For the most part, researchers have pursued image-based approaches to digital collections that are understood to be "visual" and have pursued text-based approaches to digital collections that are understood principally to be "textual." Yet, virtually all digital collections of historic materials include digital images of the objects they represent. A common use of these digital images is as input for OCR-

derived text. However, historic materials, particularly handwritten materials and highly heterogeneous and dense texts, pose significant challenges for OCR. Poor OCR prevents these materials from being identified, indexed, and subsequently analyzed to their full potential. The *Image Analysis for Archival Discovery*, or Aida (2017), research team is investigating the potential of intelligent image analysis as applied to *textual* materials—via the digital images that represent them—to advance the state of the field in applied AI in digital libraries, with broader applications that provide greater accessibility and usability of historic materials to non-expert users.

Digital libraries of historic newspapers comprise millions of digital images (e.g., Chronicling America, the largest freely available digital collection of U.S. newspapers contains over 12 million digital images, as of September 2017). The archived newspapers are heterogeneous publications, with many different types of content as well as genres in single issues. Much like today, a single issue could potentially include reportage, editorials, advertisements, illustrations, obituaries, weather information, industry, and market information. Also, both poetry and fiction were common features in historic newspapers. The magnitude of the corpora, research demand, and the heterogeneity of the materials mean that digital libraries of newspapers are a fertile testing ground for developing automated techniques to aid in the identification, description, and researcher discovery of materials of interest.

The Aida team is, therefore, developing an intelligent image analysis system for aiding in this work, focusing on poetic content in historic newspapers. Poetic content, for our purposes, includes lyric verse of four or more lines that is visually distinctive from surrounding newspaper text. Our interest in poetic content is three-fold: millions of poems were published in historic newspapers; these materials are growing in interest to literary scholars; and poetic content is very visually distinctive in the newspapers, making it a good initial test case for our larger approach.

Methodology

Our application involves: (1) page segmentation, (2) pre-processing, (3) feature extraction, and (4) training and classification. Page segmentation divides the entire newspaper page into smaller image snippets. Pre-processing converts each snippet into a binary image of textual and background pixels. Feature extraction computes a vector of numerical attributes for each binary image, to describe the structural characteristics of its textual pixels. Finally, we train an artificial neural network (ANN) using these vectors to distinguish snippets with poems and those without.

Our underlying approach is rooted in capturing how humans use visual cues to quickly identify poetic contents in newspapers. Researchers working with historic newspaper pages look for visual structures and impressions to identify aspects of the page as well as content. For poetry, these features include jaggedness (or unjustified text) on the right-hand side of a column or gaps between stanzas to help them focus in on a region to confirm whether there is a poem. Our approach aims to represent these structures into numeric representations and train an ANN to identify them as visual cues critical for classification.

To begin extracting visual cues or features, it is more convenient to exploit the inherent columns available in historic newspaper pages, to obtain image snippets that are sufficiently small in order to highlight the visual cues. Even while millions of poems appeared in historic newspapers, their proportion of newspaper space they occupied, in relation to non-poem text, is small. Hence, page segmentation identifies the columns in a newspaper page and subsequently breaks each column into multiple overlapping image snippets with roughly the ratio of 3:2 (number of rows: number of columns), where the width of each image snippet is basically the width of the column found.

Page Segmentation

We begin image segmentation with some pre-processing steps. First, for a given image we find an average intensity of the input image pixels by adding up the numerical pixel values and then dividing by the total number of pixels. After that, it is binarized using Otsu's method (1979), which provides a robust automatic threshold selection method assuming that an image has a bimodal intensity histogram. (Binarization is discussed more in the next section.) The image then undergoes morphological cleaning (Dougherty 1992) to remove noise, where outlying object pixels in areas of background intensity are removed. We then identify breaks between the page's columns by looking for *pixel columns* that have mostly background pixels. We call these pixel columns *column breaks* and compute the average width between each successive pair of column breaks in the page. Finally, the algorithm divides the original newspaper page image into image snippets page-column by page-column: for each page-column with the width the same as the average width computed above, each image snippet of 3:2 ratio was extracted with overlapping areas between each image snippet and the next. Figure 1 shows an example of the page-columns found. Typically, a newspaper page image has about 6700 x 4900 pixels. Page segmentation generates ~70 image snippets for each newspaper page, with about 600 x 400 pixels per snippet.

Snippet Pre-Processing

Our pre-processing module consists of two steps. Step 1 is *binarization* to identify and focus on the object or textual pixels when we perform feature extraction at a later stage. Similar to that in page segmentation, we also base our binarization algorithm on Otsu's method. Note that we do *not* perform page-level binarization as a newspaper page

can have range effects where, for example, one region is darker than another region on the page, causing "global" binarization to do poorly. Instead, by performing binarization on image snippets, we essentially break down the process similar to dynamic local thresholding (Haverkamp et al. 1995). Step 2 is *consolidation*. Because of inherent noise in these image snippets, textual or object lines are usually corrupted, e.g., not solid, with "holes," and there are also "false-alarm" object pixels. Thus, this consolidation step is used to "clean" the binary image to make the textual pixels more prevalent.

Figure 1. A newspaper page with page-columns (red vertical lines) automatically detected by our page segmentation module prior to dividing the image up into multiple snippets.

Binarization. To convert an image snippet to binary, we base our approach on Otsu (1975). Otsu's method compares the quality of a threshold value using between-class variance. This method assumes that the histogram of the input image is a typical bimodal distribution, and the optimal threshold should appear at the valley between two peaks when the value of the between-class variance is maximized. Algorithm 1 demonstrates our application of Otsu's method for automatically selecting a threshold to segment an image into object and background pixels.

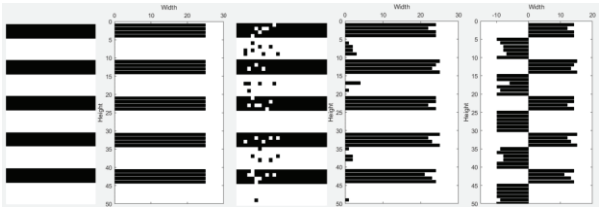
Consolidation. Our consolidation algorithm is based on first finding the Region of Text (ROT) histogram and then using the histogram to clean up the binary image further (Algorithm 2). In ideal conditions, given an image snippet (as shown in Figure 2(a)), the bar chart of the number of text pixels for each row looks like a square wave (Figure 2(b)). However, in practice, there are noisy pixels resulting in noisy histograms (Figures 2(c)-(d)). Now, if one subtracts the bar chart with a proper threshold, the residual non-background pixels are moved to below the threshold (or the so-called "zero" axis), creating a much clearer delineation between background rows and text rows, as shown in Figure 2(d). Based on this intuition, our Region of Text (ROT) algorithm (Algorithm 2a) uses the *horizontal summation* idea from (Kumar et al. 2016) to count the number of the object/textual pixels and then to identify real textual rows from noisy or faulty textual rows.

To counter noisy pixels, we leverage the successive adjacent textual pixel rows. At each row, we compute the height of a "block" starting at that row, by scanning the ROT histogram for a consecutive number of rows that have positive values. Given the starting and ending row, we compute the average percentage of textual pixels in all regions, pct_r. Then, we go through the block again, from left to right. A pixel is considered a true textual point if its region ($height_{block_r}$ x $height_{block_r}$) has a percentage of textual pixels greater than pct_r. Our Region-Enhanced Text (RET) algorithm (Algorithm 2b) is similar to adaptive noise filtering (Kuan et al. 1985).

Algorithm 1. Binarization

Inputs: Snippet s.
Output: Binarized Snippet s_b
1. Construct a gray level histogram for s, generating h_i, where $i = [1, L]$ where L = number of gray levels.
2. Convert the histogram to its relative distribution function representation: for all i, $p_i = h_i/size(s)$.
3. Loop through all gray levels k
   a. $\omega(k) \leftarrow \sum_{i=1}^{k} p_i$, prob. of region before k
   b. $\mu(k) \leftarrow \sum_{i=1}^{k} i \times p_i$, the mean level of region before k
   c. $\mu_T \leftarrow \sum_{i=1}^{L} i \times p_i$, the mean level of whole histogram
   d. $\sigma_B^2(k) \leftarrow \frac{[\mu_T \omega(k) - \mu(k)]^2}{\omega(k) \times [1 - \omega(k)]}$, between-class variance
4. Optimal threshold $k^* \leftarrow arg\max_k \sigma_B^2(k)$.
5. Set threshold of s using k^* to generate s_b.

End of Algorithm



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

Figure 2. (a) example of an ideal situation where the dark pixels are object/textual pixels; (b) corresponding histogram of (a); (c) example of a noisy situation; (d) corresponding histogram of (c); (e) the ROT histogram minus threshold.

Algorithm 2. Consolidation

Inputs: Binarized Snippet, s_b
Output: Consolidated Snippet, s_c.
1. $hist_{ROT} \leftarrow ROT(s_b)$
2. For each textual block, $tblk_r$, in $hist_{ROT}$, where r is the first row of the block.
   a. Corresponding textual block in $s_c \leftarrow RET(tblk_r)$

End of Algorithm

Algorithm 2a. ROT

Inputs: Binarized Snippet, s_b,
Output: the ROT histogram, $hist_{ROT}$
1. Compute the number of text pixels for each row *r*: ntp_r.
2. Compute the mean of all ntp_r: $\overline{ntp}$.
3. For each row *r*, perform $nap_r = ntp_r - \overline{ntp}$
4. $hist_{ROT} \leftarrow nap$ // Histogram shown in Figure 2(e)

End of Algorithm

Algorithm 2b. RET

Inputs: Binarized Snippet, s_b, row r, and height of the block, $height_{block_r}$.

Output: Consolidated Snippet, s_c.

1. Compute percentage of textual pixels in each region of size $height_{block_r}$ x $height_{block_r}$, starting with the region starting at (column, row) = (0, r), until the region at (image width - $height_{block_r}$, r).
2. Compute adaptive threshold = the mean of the textual pixels percentages for the above regions with the starting row r, $\overline{pct_r}$.
3. Identify the leftmost column, α_{left}, of the block to be filled as the top-left corner pixel of the leftmost region that has percentage of textual pixels $\geq \overline{pct_r}$.
4. Identify the rightmost column, α_{right}, of the block to be filled as the bottom-right corner pixel of the rightmost region that has percentage of textual pixels $\geq \overline{pct_r}$.
5. Label all pixels between (α_{left}, r), and $(\alpha_{right}, r + height_{block_r} - 1)$ as object/textual pixels.

End of Algorithm

Figure 3 shows three examples of binarization and consolidation. Our consolidation step "solidifies" rows of textual characters or letters into rows of textual blocks. Note that only the image snippets at the top and bottom rows have poetic texts in these examples.

Feature Extraction

After the pre-processing module, we are ready to extract features from each binarized, consolidated image snippet. To do so, Algorithm 3 first counts both the length of background pixels prior to the first object pixel and the length of background pixels after the final object pixel in a row, for each row, generating a list of *leftColumnWidths* and *rightColumnWidths*. Then, the algorithm counts the number of each sequence of continuous background pixels in each column and stores the values in a 2D integer matrix, generating a list of *rowDepths*.

We identify four groups of features: margins, jaggedness, stanza gap, and length. First, we compute margin statistics using the list of *leftColumnWidths*. We include this feature in our algorithm because poetic content typically was typeset with wider left and right margins than non-poetic text. At present, our design focuses on left margins only, since we evaluate qualities of the right side of the poem in relation to jaggedness. In general, poetic text should have wider left margins than non-poetic text.

Second, we compute the jaggedness using the list of *rightColumnWidths*. The jaggedness algorithm measures the number of background pixels after the final object pixel in each row. In general, poetic text should have a higher variance in its jaggedness value than non-poetic text.

Third, we extract feature attributes that determine the presence of stanzas by looking for gap between stanzas using a list of *rowDepths*. Cutting along each column, poetic text should have numerous sequences of similar-length background pixels (i.e., spacing between lines of text) interleaved by a longer sequence of background pixels (i.e., gap between two stanzas), while non-poetic text should have only sequences of similar-length background pixels.

Figure 3. Column (1) original image snippets, (2) binarized results, and (3) consolidated results.

Algorithm 3. Feature Extraction

Inputs: Consolidated Snippet, s_c

Output: a computed feature set, *attributeList*

1. *leftColumnWidths*, *rightColumnWidths* ← *computeColumnWidths*(s_c)
2. *rowDepths* ← *computeRowDepths*(s_c)
3. *attributeList* ← *attributeList* + *computeMarginStats*(s_c)
4. *attributeList* ← *attributeList* + *computeJaggedStats*(s_c)
5. *attributeList* ← *attributeList* + *computeStanzaStats*(s_c)
6. *attributeList* ← *attributeList* + *computeLengthStats*(s_c)

End Algorithm

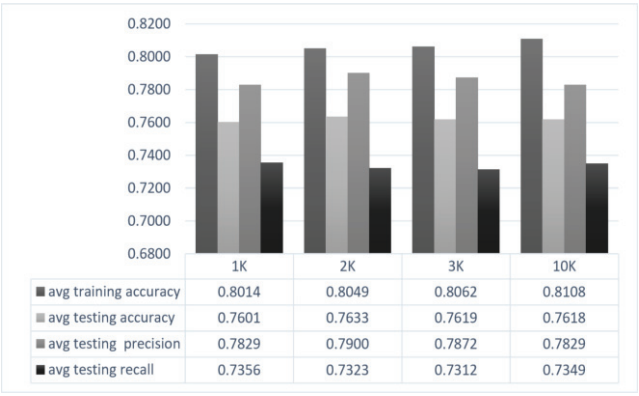
Fourth, we also compute the length of the first contiguous sequence of object pixels for each row. In general, non-poetic text should have longer rows of object pixels than poetic text. This feature complements the margins and jaggedness particularly for image snippets that are noisy where a row of object pixels might be broken up into multiple segments. In cases where these rows are contiguous from left to right, this feature would in turn work in tandem with especially the jaggedness feature.

Note that the visual cues the system derives are based on left margin whitespace; whitespace between stanzas; and content blocks with jagged right-side edges, which are the result of varying line lengths in poetic content and are in contrast to justified blocks of much newspaper content.

Since one of our team members had elsewhere cataloged approximately 3,000 newspaper poems by hand, we began with the features that seemed the most relevant in that work (Lorang 2010). Reassuringly, we identified some overlapping features with the Visual Page project (Houston 2014), which might further suggest the relevance of these features. At present, an analysis of jaggedness is unique to our project, likely because jaggedness emerges as a feature in comparison to the justified prose text of newspapers.

ANN Training and Classification Results

The ANN model used in our system has three layers: (1) input layer, (2) hidden layer, and (3) output layer. The input layer contains 20 input nodes, each for one of 20 feature values derived from the consolidated, binary image snippets. Each of the four feature groups (i.e., margin, jaggedness, stanza gap, and length) has five statistics (mean, standard deviation, minimum, maximum, and range), yielding 20 feature values in total. The hidden layer contains 11 nodes. The output layer consists of one classification node that labels an image true if it has a poem, or false otherwise. We use the ANN implementation from the WEKA Workbench (Eibe et al. 2016).



	1K	2K	3K	10K
avg training accuracy	0.8014	0.8049	0.8062	0.8108
avg testing accuracy	0.7601	0.7633	0.7619	0.7618
avg testing precision	0.7829	0.7900	0.7872	0.7829
avg testing recall	0.7356	0.7323	0.7312	0.7349

Figure 4. Training and testing results on ~17K image snippets of an ANN with a hidden layer of 11 neurons, using the 10-fold cross-validation approach.

We used a 10-fold cross-validation training process. For our investigation, we use 16,928 image snippets from the Library of Congress' Chronicling America repository, of newspapers during the 1836-1840 years. Half of the snippets have poetic text while the other half do not. (All snippets were manually labeled in order to perform this study.) We also explored different numbers of ANN training iterations: 1K, 2K, 3K, and 10K. Figure 4 shows that, overall, the average training accuracies were above 80% while the average testing accuracies were above 76%, indicating some level of overfitting in training. As more iterations were run during training, accuracies in training improved marginally while testing accuracies decreased slightly after the 2K-iteration version. The average precision and recall on the test sets were above 78% and 73%, respectively.

Looking more closely at the poetic and non-poetic image snippets, we find that on average, poetic snippets were correctly labeled between 77.74%-78.98% (for 1K, 2K, 3K, and 10K iterations) of the time, while non-poetic snippets were correctly labeled between 82.20%-83.29%. The slightly lesser recall performance on poetic snippets indicates that the classifier is slightly more conservative in labeling snippets as poetic ones.

Application

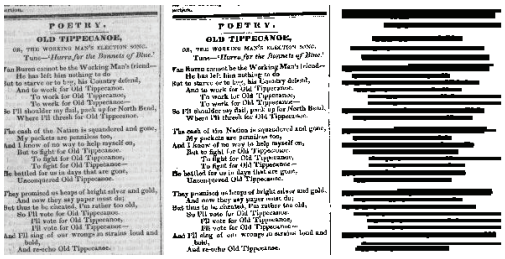
Based on the above methodology, we have developed the first generation of our Aida application in Java. As aforementioned, we use the ANN implementation from Weka (Eibe et al. 2016). We have also developed other ancillary tools as part of the application: (1) scripts to download newspaper pages from the Library of Congress' Chronicling America repository, (2) web-based scripts to display and facilitate visual comparative studies of images and intermediate imagery results, and (3) data parsing and aggregation scripts to collect numeric outputs of the classification to produce table of results (e.g., accuracies, recall, and precision). The application is being used at the University of Nebraska applied to newspapers from Chronicling America to facilitate investigations into digitization, discovery, and data reintegration in massive digital libraries, as well as explorations of historic newspapers and newspaper verse, and at the University of Virginia applied to the Burney Collection of British Library newspaper collection to facilitate development of a digital archive of 18th-century British newspaper verse, with corresponding metadata and meta-tagging and dissemination strategies.

Conclusions and Future Directions

We have described an emerging application called Aida that automatically identifies poetic content in historic newspaper pages. This application uses a set of computer vision and machine learning techniques. It segments newspaper pages into image snippets, pre-processes the image snippets into binary images that contain textual blocks and background, and extracts features from those blocks to capture visual cues resembling human expert behaviors. It uses an ANN mechanism to take these features as input and learn to classify or label image snippets as containing poetic text or not. Our first-generation application is capable of achieving promising accuracies, precision, and recall on ~17,000 image snippets. Our goal is to improve the accuracies to above 90% and the precision and recall to above 80% for the application to be widely useful.

In terms of next steps, we aim to address two particular issues. First, our approach needs to better account for the "jagged" nature of the left-hand side of blocks of poetic

content. Figure 5 shows an example of a poem that has significant jaggedness at both the beginnings and ends of lines within stanza blocks. As also shown in Figure 5, the newspaper page is skewed, and so is the image snippet. These peculiarities caused the classifier to label this snippet incorrectly as non-poem. Second, there are also page segmentation challenges posed by features such as considerable *bleed-through of text from the other side of a newspaper page* that obscures column breaks, *skewed page orientation* that hinders the search for contiguous background pixel columns, *excessive low contrast* (too dark or too bright) that prevents the ROT algorithm to find a proper valley to threshold the image, and *column spanning headlines* (and also advertisements and drawings) that make column breaks difficult to locate. In the longer term, a next step is to extend the approach to develop classifiers for other types of newspaper content as well, such as advertisements, which make up a significant percentage of content in historic newspapers.


POETRY.

OLD TIPPECANOE,

OR, THE WORKING MAN'S ELECTION SONG.

Tune—*Harrea for the Buttons of Blue.*

Figure 5. The classifier incorrectly labels as non-poem due to orientation skewness.

Acknowledgments

This project is supported in part by the Institute of Museum and Library Services (IMLS) and has received previous support from the National Endowment for the Humanities (NEH). Any views, findings, conclusions, or recommendations expressed in this publication do not necessarily reflect those of IMLS or NEH. Maanas Varma Datla, Spencer Kulwicki, and Grace Thomas made early contributions to this project that shaped its development.

References

Aida (Image Analysis for Archival Discovery) (2017). [online] Available at: http://projectaida.org/ [Accessed 15 Nov. 2017].

Bergel, G., Franklin, A., Heaney, M., Arandjelovic, R., Zisserman, A., and Funke, D. 2013. Content-Based Image Recognition on Printed Broadside Ballads: The Bodleian Libraries' Image-Match Tool. In *Proc. IFLA World Library & Information Congress*, 2013. http://library.ifla.org/209/

Cain, J.O. 2016. Using Topic Modeling to Enhance Access to Library Digital Collections. *J. Web Librarianship* 10(3):210-225.

Dougherty, E. R. 1992. *An Introduction to Morphological Image Processing*, SPIE.

Chung, J. S., Arandjelovic, R., Bergel, G., Franklin, A., and Zisserman, Z. 2014. Re-presentations of Art Collections. In *Workshop on Computer Vision for Art Analysis (Visart), ECCV*.

DeRidder, J. L. and Matheny, K. G. 2014. What Do Researchers Need? Feedback on Use of Online Primary Source Materials, *D-Lib Magazine* 20(7/8).

Eibe F., Hall, M. A., and Whitten, I. H. 2016. *The WEKA Workbench. Online Appendix for Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 4th Edition.

Green, H. E. and Courtney, A. 2015. Beyond the Scanned Image: A Needs Assessment of Scholarly Users of Digital Collections, *College & Research Libraries* 76(5):690–707.

Hagedorn, K., Kargela, M., Noh, Y., Newman, Davd. 2011. A New Way to Find: Testing the Use of Clustering Topics in Digital Libraries. *D-Lib Magazine* 17(9/10).

Haverkamp, E., Soh, L.-K., Tsatsoulis, C. 1995. A Comprehensive, Automated Approach to Determining Sea Ice Thickness from SAR Data, *IEEE Trans. Geo. & Rem. Sensing*, 23(1):46-57.

Houston, N. 2014. The Visual Page white paper. DOI: http://dx.doi.org/10.17613/M60S9G

Kuan, D. T., Sawchuk, A. A., Strand, T. C., and Chavel, P. 1985. Adaptive Noise Smoothing Filter for Images with Signal-Dependent Noise, *IEEE Trans. Pattern Analysis & Machine Intelligence* 7(2):165-177.

Kumar, S. S., Rajendran, P, Prabaharan, P., and Soman, K. P. 2016. Text/Image Region Separation for Document Layout Detection of Old Document Images Using Non-linear Diffusion and Level Set, *Procedia Computer Science* 93:469-477.

Lorang, E. 2010. American Newspaper Poetry from the Rise of the Penny Press to the New Journalism. Doctoral dissertation, University of Nebraska-Lincoln.

Lorang, E., Soh, L.-K. Datla, M. V., and Kulwicki, K. 2015. Developing an Image-Based Classifier for Detecting Poetic Content in Historic Newspaper Collections. *D-Lib Magazine* 21(7/8).

Oliveira, S. A., Kaplan, F. & de Lenardo, I. 2017. Machine Vision Algorithms on Cadaster Plans. In *Digital Humanities 2017*, 145-148. Montreal, Quebec: ADHO.

Otsu, N. 1975. A Threshold Selection Method from Gray-Level Histograms, *Automatica* 11(285-296):23-27.

Seguin, B., de Lenardo, I. & Kaplan, F. 2017. Tracking Transmission of Details in Paintings. In *Digital Humanities 2017*, 588–591. Montreal, Quebec: ADHO.

Smiths, T. 2017. Illustrations to Photographs: Using Computer Vision to Analyze Pictures in Dutch Newspapers, 1860-1940. In *Digital Humanities 2017*, 602-603. Montreal, Quebec: ADHO.

Tze-I Yang, Andrew J. Torget, and Rada Mihalcea. 2011. Topic modeling on historical newspapers. In *Proc. 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities* (LaTeCH '11), Stroudsburg, PA, 96-104.

Underwood, Ted. 2014. How to Find English-Language Fiction, Poetry, and Drama in HathiTrust. *The Stone and the Shell* [research blog]. https://tedunderwood.com

Underwood, Ted. 2014. Understanding Genre in a Collection of a Million Volumes, Interim Report. figshare. DOI: https://doi.org/10.6084/m9.figshare.1281251.v1