WORLD LIBRARY AND INFORMATION CONGRESS: 78TH IFLA GENERAL CONFERENCE AND ASSEMBLY

IFLA 2012 Helsinki

LIBRARIES NOW! INSPIRING SURPRISING EMPOWERING

Optical Character Recognition Software in Library Mobile Apps

Jim Hahn
University of Illinois at Urbana-Champaign
E-mail: jimhahn@illinois.edu

&

Nathaniel Ryckman
University of Illinois at Urbana-Champaign
E-mail: nathanielryckman@gmail.com

Session:

Session 103 — *Open source technology and mobile devices use and accessibility* — Information Technology with Library and Research Services for Parliaments

Abstract:

In order to reach out to the digital generation, libraries can develop mobile applications that more efficiently connect campus assignments with library resources. One approach in making this connection is through the use of optical character recognition (OCR) software. Optical character recognition is software that allows a computer to create text strings out of scanned images or, in our case, an image that is captured with a library user's smart phone. Research and development tasks include studying how mobile applications powered by OCR software and a recommendation database will help students connect and integrate library resources into their work.

Use Cases

The initial use case is for a course assignment page. Student users of the app will have the option to scan a course-assignment page. The mobile app will use modified OCR software and check a suggestion database to identify library resources that will support the assignment or topical interest, check against a database of course reserves for the class and other relevant sources of data (course specific help guides), and suggest library resources and research databases that are relevant. This information package can be viewed on the phone, or, since students value collaboration and social aspects of research, students will have the option of sending the information to themselves via email or other social networking platforms.

We theorize there are many additional use cases that must necessarily include any textual data. This research will also plan investigate affordances of the app that allow users to scan a course syllabus, scan a citation or parts of a bibliography, scan the contents of a book page, or scan a shelf of books in the library. After scanning these elements we would work to connect the student with relevant library information such as relevant titles, relevant research databases, help pages, and sections of the library that may be relevant to this scanned interest. This involves a computing backend of image recognition and image preprocessing, detailed below.

Methods

Computing Workflows

We start with an open source computer vision library such as Leptonica http://www.leptonica.com/ to pre-process the images from the camera. Then we utilize OCR libraries to extract text. Tesseract is freely available from Google code (http://code.google.com/p/tesseract-ocr/) and is considered one of the most accurate free OCR engines currently available. We anticipate research and development with alternative pre-processing and artificial intelligence techniques as well. Other computer vision software that may prove useful for identifying geometry or image reconstruction includes the OpenCV library, now available for the Android mobile operating system: http://code.opencv.org/projects/opencv/wiki/OpenCV4Android

Our research and development approach will initially use aspects of Leptonica combined with OCR software, like Tesseract. We hope to explore using OpenGL for displaying interface elements on the device and, also, experiment with posting or sending information to the student's social networks (e.g. twitter and other online commenting pages) or school email account.

Applied studies in app use by library patrons.

Rapid prototyping to study apps is a technique used in previous mobile app design (Hahn & Morales, 2011). This methodology includes collecting detailed qualitative use data of the app through interview and observation by a small test group. By collecting use preferences early in the design phase, we are able to make available to students an initial release that has been vetted by use preferences and desired functionality.

With rapid use studies we can identify user problems within the first five user tests. We anticipate using a total of 10 test users to get at usable software for an initial release. In addition to studying the app and how users interact with scanned text in an app, we expect to uncover in our rapid prototyping studies additional uses for scanned textual images in library settings.

Future Work

Future research and development includes a number of open problem areas. Firstly, we anticipate exploring a variety of algorithms for textual analysis. While our lab employs researchers in artificial intelligence, there may be implications from biological systems, particularly from neural networks and the statistical techniques used in analyzing DNA sequences.

Algorithms

The mobile phone camera will capture images formatted as RGB8888. Leptonica will be used to convert the RGB8888 formatted images into 8-bit luma formatted images and store them into C data structures. This pre-processing step will help prepare the captured images from the mobile device for text analysis by Tesseract.

Tesseract uses a combination of many different algorithms to produce digital text from the images provided by Leptonica. The following summarizes the text extraction process described by Ray Smith in “An Overview of the Tesseract OCR Engine” (2007). First, after thresholding and edge detection algorithms are applied, the image provided to Tesseract undergoes connected-component analysis, thus determining connected regions of interest (aka blobs). Least median of squares fitting and quadratic spline fitting are then applied to the detected blobs in order to find the baselines of the text along with the skew.

Based on the baselines, Tesseract determines if the text has a fixed-pitch or not. If the text has a fixed pitch, the pitch is used to chop the text into characters. If the text does not have a fixed pitch, fuzzy spaces are used in place of ambiguous text spacing, blobs are chopped on candidate concave vertices, and then, if necessary, an A* search of a segmentation graph of possible combinations of maximally chopped blobs into candidate characters is performed. X-position, y-position, and angle are features that are extracted from the blobs.

These features are used to classify the blobs as characters by comparing the features to learned prototype characters. Once blobs are classified as characters, the best word candidate is chosen based on normalized distance calculations. Finally, the process is repeated using the results as training data for the second pass.

Modules

Our future work will include connecting the OCR module to the libraries other modules. App development in our lab includes designing modularly such that additional components of apps that we design can be integrated with other mobile initiatives. For example, our wayfinding module, which directs users to items in the library, will be connected with the OCR modular such that students can be guided to recommendations based on scanned text.

We invite international collaborations from researchers and hope to make available to optical character recognition app to broader audiences beyond the University of Illinois. Our modular application framework is available here from our github space: https://github.com/minrva.

Works Cited

Hahn & Morales (2011). Rapid prototyping a collection-based mobile wayfinding application, *Journal of Library Administration,* Volume 37, Issue 5, September 2011, Pages 416–422.

Smith, Ray (2007). “An Overview of the Tesseract OCR Engine”, Ray Smith, Proc. Ninth Int. Conference on Document Analysis and Recognition (ICDAR), 2007, pp. 629-633. http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en/us/pubs/archive/33418.pdf

Consulted

Overview of the Leptonica Library: http://www.leptonica.com/library-overview.html

Smith, R. (1994). A Simple and Efficient Skew Detection Algorithm via Text Row Algorithm
http://www.hpl.hp.com/techreports/94/HPL-94-113.pdf