MNIT Jaipur

Final Year Project

Automated Library Book Location Management

Smart Library

Mobile Book Spine Recognition

Book Title Recognition for Smart Library

The Smart Bookshelf

Ideas –

* Use online amazon database to store recognized data and fetch data.
* Can use string pattern matching for correcting book names. For it, we have to make a book database containing all books name OR use spelling correction or nearest Levenshtein distance matching against a dictionary to correct spellings.
* After spine recognition, the user can select individual spines in the phone’s viewﬁnder to obtain more information about a particular book, without ever having to take that book off the bookshelf.
* We can also have a database of images of book spine for every book and then use some image similarity measures between images, already stored in database and taken from a camera.
* Use the digital compass and accelerometer on the smartphone to estimate location of the identified books. The digital compass tells us which direction the phone is facing when a book is photographed, while a temporal trace of the accelerometer informs us how far vertically and horizontally the phone has moved from the last anchor point.
* Once the book is identiﬁed, additional stored information could also be retrieved, e.g. a short description of the book.
* Indexing and searching from a book database.
* Find the location of books on the bookshelves.

Two Tasks –

* Library Inventory Building
* Retrieval System

Two Approaches –

* Text-based Retrieval
* Image-matching Retrieval

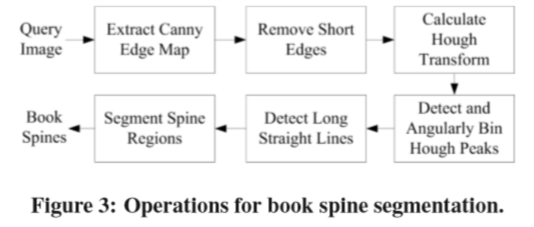
1. Building Book Inventories using Smartphones

* Generating an inventory of books by snapping photos of a bookshelf with a smartphone.
* The location of each book is inferred from the smartphone’s sensor readings, including accelerometer traces, digital compass measurements, and Wi-Fi signatures. This location information is combined with the image recognition results to construct a **location-aware book inventory**.
* More cost-effective and convenient method for building and updating book inventories.
* A book recognition algorithm is used to reliably segment and identify each book in a photo
* Feature-based book spine recognition

**Working** –

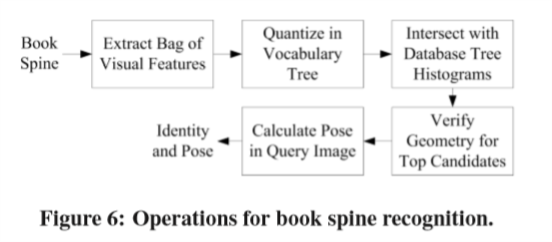
* Image analyser
  + Book spine segmentation
  + Book spine recognition - Match the segmented spine against an online database of book spines.
* Location Tracker - determine the locations of individual books, specifying which room, which bookshelf, and where in the bookshelf each recognized book can be found.

**Book Spine Segmentation –**



* Canny edge map
* Hough transform
  + Angular histogram
  + Dominant angle θ spines

**Book Spine Recognition –**



* bag-of-visual-features (BoVF) and SURF features
* tree histogram

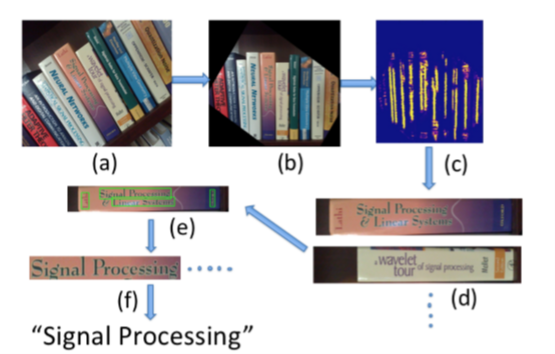
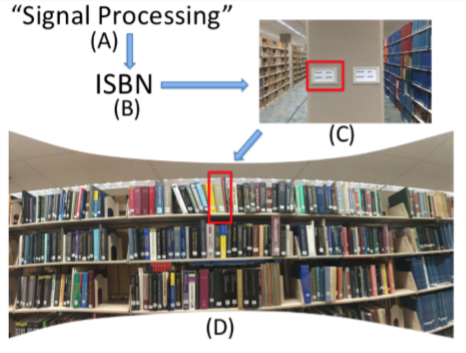
**Location Tracker –**

* Wi-Fi signatures
* Smartphone’s digital compass and accelerometer readings

Segmentation is a crucial step required for accurate spine recognition.

2. Smart Library: Identifying Books in a Library using Richly Supervised Deep Scene Text Reading

* Managing books in a large bookshelf and ﬁnding books on it often leads to tedious manual work, especially for large book collections where books might be missing or misplaced.
* Manually searching bookshelves is time-consuming and often not fruitful depending on how vague the search is.
* Recently, deep neural models, such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) have achieved great success for scene text detection and recognition.
* Build a deep neural network-based scene text reading system. For text recognition, adopt a deep sequential labeling model based on convolutional and recurrent neural architectures.
* For text localization in a library environment, design a book spine segmentation method based on Hough transform and scene text saliency.



Building a book inventory Locating a book in a library

Text Localization

First segment each book spine image and then localize text on these images.

Book Spine Segmentation

* Use Hough Transform as a pre-processing step to extract the dominant direction of books in the image.
* Dominant direction is then used to rotate the entire image.
* Then apply a text/non-text CNN model trained on 32×64 color image patches to generate saliency maps of the rotated image.
* Use a non-max suppression to ﬁnd the segmenting point of each book along the vertical axis.

Text Localization within Book Spine

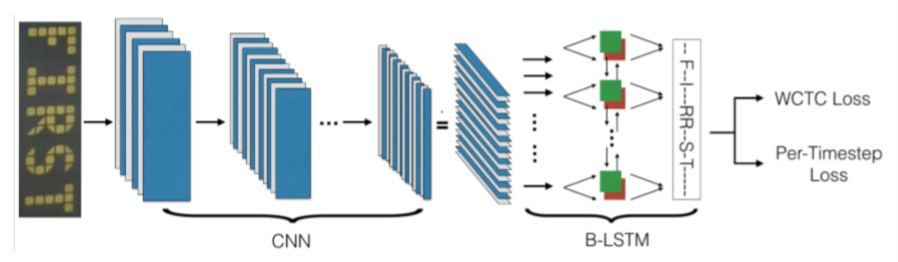
* Use region proposal based method for scene text detection.
* First generate extreme region - fast computation and high recall.
* Saliency maps generated by the CNN are then used to ﬁlter out non-text regions.
* A multi-orientation text line grouping algorithm is applied to ﬁnd different lines of text - by ﬁrst constructing a character graph and then aligning character components into text lines.
* We need to further decide whether a text line is upside down or not.

Text Recognition

* Text Recognition via Sequence Labeling, recognizing a sequence of characters simultaneously.
* Hybrid approach that combines a CNN with an RNN, casting scene text recognition problem as a sequential labeling task
* A bidirectional Long Short-Term Memory (B-LSTM) is applied on top of the learned sequential CNN features.

The CTC Loss

* Sequences X (prediction) and Y (target) have diﬀerent lengths, so we adopt CTC loss to allow an RNN to be trained.
* Adopt CTC loss to allow an RNN to be trained for sequence labeling task without exact alignment.
* Stochastic gradient descent (SGD) method is used for optimization.
* Use forward backward dynamic programming method for computation.
* Decoding (ﬁnding the most likely Y from the output sequence X) can be done by beam search.



Text Correction

* Train another RNN, employing a character-level sequence-to-sequence model

Experiments

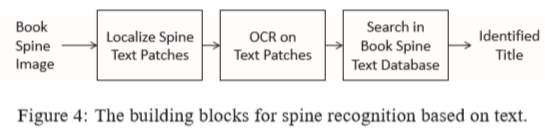
* Three widely-used benchmark datasets: IC03, SVT and III5K.
* OCR engine Tesseract performs poorly on all datasets.
* It outperforms methods with hand crafted features and several deep neural-based methods focusing individual characters, indicating the beneﬁts of learning sequential information.
* Book Spine Retrieval
* Reciprocal Rank (RR)
* RR = 1/K, where K is the rank position of the ﬁrst relevant document (the target book spine in our case)
* Average Reciprocal Rank (MRR) across multiple queries
* Use nearest neighbour matching
* tf-idf (term frequency-inverse document frequency) weights are used to rank returned results.
* Built search engine with Apache/Solr

3. Combining Image and Text Features: A Hybrid Approach to Mobile Book Spine Recognition

* Combines a text-based spine recognition pipeline with an image feature-based spine recognition pipeline.
* Text-based recognition pipeline - The text within the book spine image is recognized and used as keywords to search a book spine text database.
* Image feature-based recognition pipeline - The image features of the book spine image are searched through a book spine image database.
* Combine them to form the ﬁnal recognition result.

Book Spine Recognition

Recognizing Spines with Text



* Detect text in the extracted book spine image using a text detection algorithm based on Maximally Stable Extremal Regions (MSER) and Stroke Width Transform (SWT).
* Localized text is then extracted from the book spine image and denoised using an edge-preserving ﬁlter.
* Finally, the individual text patches are passed to an OCR engine for recognition. Limit the allowed recognized characters to only alphanumerical letters and a reduced set of punctuations and notations.

Text Search

* Make a book spine text database using inverted ﬁles, commonly used in text retrieval systems.
* Two approaches to ﬁnd matching words
  1. Exact word matching
  2. Nearest neighbor word matching
* Use tf-idf (term frequency-inverse document frequency). tf weights the word according to the number of occurrences within the spine text, and idf weights the score based on the how many different titles the word has occurred in.

Recognizing Spines with Image Features

* Construct a vocabulary tree using SURF features extracted from the database book spine images
* From the query spine, extract SURF features and use them to match the spines to a database of book spine images using a vocabulary tree with soft binning.
* Geometrically verify by estimating an afﬁne model between the two spine images using RANSAC

Combine the results of the text-based recognition pipeline with the image feature-based recognition pipeline to form the ﬁnal result.

Experiments

* Precision is the percentage of correctly identiﬁed titles out of the declared correct titles.
* Recall is the percentage of correctly identiﬁed titles out of all query spines.
* Spines with text that have generic fonts tend to be harder for the image feature-based system to recognize due to the similarity between visual features.
* Spines with graphical components and cursive text are rather challenging to OCR engines. spines with graphical components and cursive text, such as the spine shown in Fig. 7(b), are rather challenging to OCR engines
* Scoring scheme – NN matching + tf-idf weighting

Other Papers –

Nevetha et. al. used a line segment detector with several heuristic rules to extract book spines with Optical Character Recognition (OCR) then applied to read text.

Handcrafted featuresbased book spine segmentation suﬀers from image distortion and low contrast between books.