OCR Conversion

By Ryan, Anne, Chris and Uros

What is OCR and what does it do?

-Optical Character Recognition

-Converts typed or written text to machine encoded text

-Uses include making electronic versions of written text, reading coupons, conversion between files, etc.

-Useful for extracting information and converting it into searchable and editable content

What was difficult in creating OCR software

-Most difficult aspect was image manipulation

-Image must be readable for program to work - difficult to read “bad” pictures

- Overcome most issues, also implemented a spellchecker

-Spellchecker converts small errors that occur often

-Common example was “Uros” read as “Ur-os”

-Spellchecker handles 1-letter errors (extra letter, letter missing, letter swapped)

What is the software built on?

-Our project uses two main libraries: tesseract and leptonica

-Tesseract is Google sponsored free software, did most of the work

 -It handles the conversion aspect

-Leptonica is an open source image processing software

 -It handles image manipulation and processing, and prepares images for conversion

What does Tesseract do?

-Tesseract uses multiple algorithms to locate blocks of text, narrow down on words, divide words to letters, identify letters, then combine them back

-Idea similar to merge sort, but no characters are rearranged

-Outputs all letters/words it recognizes, ignores those it can't

-Attempts to recognize all of image, but only picks up actual characters

How it works

8 major steps to convert

1. Line finding
2. Baseline fitting
3. Pitch detection
4. Proportional word finding
5. Word Recognition
6. Chopping joined characters -Optional
7. Associating broken characters -Optional
8. Output

1 - Line finding in Tesseract Alg.

-What: Allows algorithm to recognize words in skewed image WITHOUT deskewing

-How: Most text exists on a line, straight or sloped, so try and find that line under first batch of text. When found, measure height of text, and use that height as estimate for width between the lines, until no more space

-Why: To preserve image quality by not tampering with it

2 - Baseline fitting in Tesseract Alg.

-What: Enable “curved lines” on top of previous lines

-How: Area presumed to be text where lines did not work are tried again, using quadratic splines - basically curved lines formed from rounded edges. Lines remain parallel

-Why: To allow texts such as scanned books, etc.

3 - Pitch Detection in Tesseract Alg.

-What: Split letters “merged together” must be fixed

-How: Fixed pitch means font with letters of equal width. The algorithm checks if any words are fixed pitch, and if so, chops the word into letters and uses those letters as a tool for recognition. Recognizes fixed pitch by finding space in the words and cutting there, then checks if cuts are of equal width

-Why: To provide more accurate recognition and removed merged letters

Ex. of fixed pitch v.s. no fixed pitch: STEVE vs PIZZA

4 - Proportional Word Finding

-What: It is very difficult to deal with non fixed pitch words, so tesseract must figure out which letters are which

-How: By measuring spaces between letters, and using those values to determine where to cut between letters

-If no spaces exist, ex: “W/” - algorithm cannot detect space using vertical line test, do nothing.

5- Word Recognition and 6- Chopping

-Algorithm attempts to match to what it believes the letters are

-For letters the algorithm has low confidence or is unsure of, algorithm attempts to split letters and trying to recognize again

-Letters are chopped at any concave vertices of the image

-If no result, undo and try another point

Ex.: W/, will begin by chopping W multiple times

7- Associating and 8- Output

-If step 6 produced no better output, algorithm combines chopped fragments and attempts to find letters using multiple letters

-Very inefficient, but used as a rare last resort as >95%, algorithm will produce an output which could be of low certainty, instead of no output

-Last step is to display results, done using simple filestream

-Can be displayed on command line or saved to file

Benefits of using Tesseract

-Uros started coding our UI in C++

-Availability of information about Tesseract C++ implementation

-Fast image conversion as seen during our last presentation

-Easy to implement in existing code to fit our needs. I find it more compatible with our project goals compared to other OCR engines like OCRFeeder, OCRopus.

-”All in one package”, instead of having to use multiple software or algorithms (ex: KNN)

-After our last alpha demo presentation, we were able to implement conversion of pdf to text at the desktop level

What was learned using Tesseract

-Difficulty of using, understanding and modifying very large software

--Training Tesseract (improving quality of output): Involved (1) Image processing, Rescaling, Binarisation, Noise Removal, Rotation / Deskewing, Border Removal, Tools / Libraries, Page segmentation method, Dictionaries, word lists, and patterns

-Reading the giant 100+ page instructions was mandatory, still left room for questions about certain functions

-Even software worked on by dozens for years has scope and limitations

-Does not support MS word as an output file format (.docx)

-Different/Strange fonts or bad lighting/background yield no results

Google Cloud API

-Cloud SQL v1beta4 API

-REST API (Representational state transfer)

-REST systems aim for fast performance, reliability, and the ability to grow

-Works by reusing components that can be managed and updated without affecting the system as a whole, even while it is running.

Rest API

Rest API constraints:

- Client-server architecture
- Statelessness
- Cacheability
- Layered System
- Code on demand
- Uniform interface

Cloud API

Cloud API is Defined by:

- BackupRuns
- Databases
- Instances
- Flags

Cloud API Continued

- Operations
- SslCerts
- Tiers
- Users resources

Cloud API

-Can be used indirectly with:

a) Google Cloud SDK

b) One of the Client Libraries

c) Google Cloud Platform Console, a graphical web interface.

How Database API works

1) Choose type of database
   - MySQL
   - PostgreSQL
2) Create an instance
3) Connect via google shell to server
4) Create database and subsequent table

ACID

- Atomicity
- Consistency
- Isolation
- Durability

Applications of the API

- Export database
- Link
- Import offline created databases, data
  - import directly to another database
  - SQL or CSV files

Pros/Cons

-Cost

-Storage capacity

-Ease of use

Using OCR online

-Website: OCR Conversion

-Accessible on both desktop and mobile devices

-Quick login using Google Login

-Will show you converted text on screen

-Copy + Paste / Download / Email / Save to Drive

What was Difficult in Making the Website?

-Applying the C++ code of the OCR to the website

Website mainly focuses on HTML, CSS, and JavaScript

-Attempts to get the C++ to work on the website:

✘ Putting the C++ code into a Node.js file or Java file - Complicated

process

✘ Ocrad.js - It is a different OCR than the one we used for the application

✔ Using Tesseract in JavaScript - requires some time to convert

Using Tesseract.js

-Pure Javascript

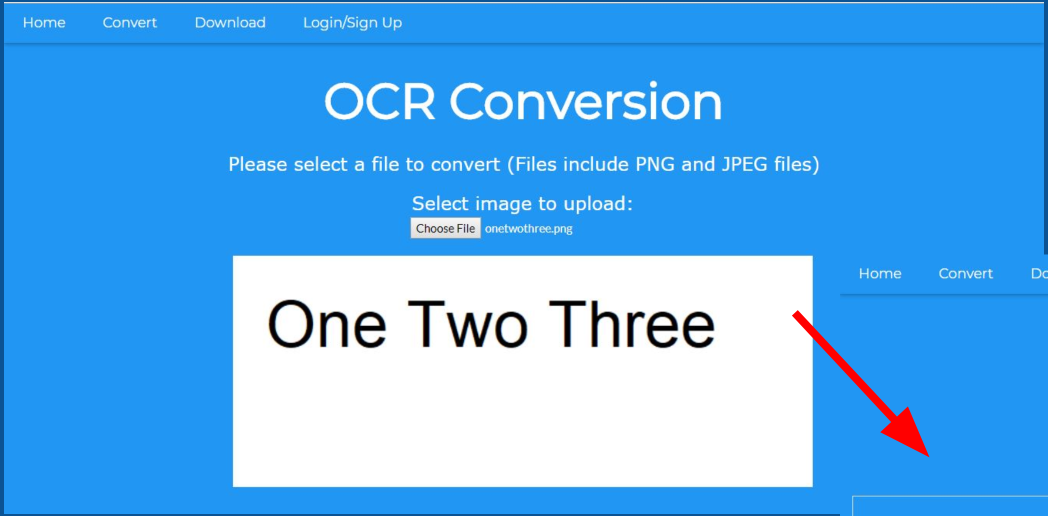
-Works with browser and Node.js

-<script

src='https://cdn.rawgit.com/naptha/tesseract.js/1.0.10/dist/tesseract.js'></script>

-Shows progress of converting the image

-Able to extract other languages!



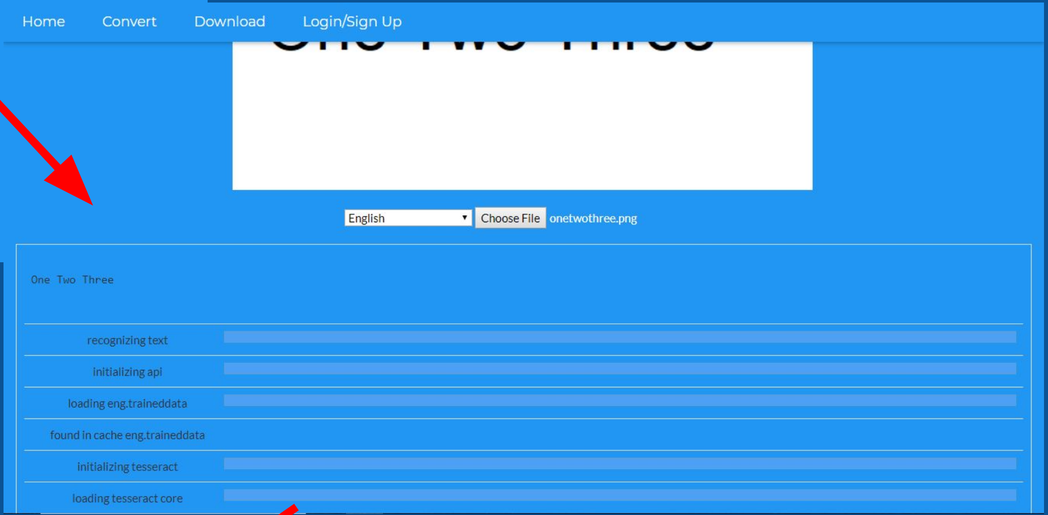
OCR Conversion

Please select a file to convert (Files include PNG and JPEG files)

Select image to upload:

onetwothree.png

One Two Three

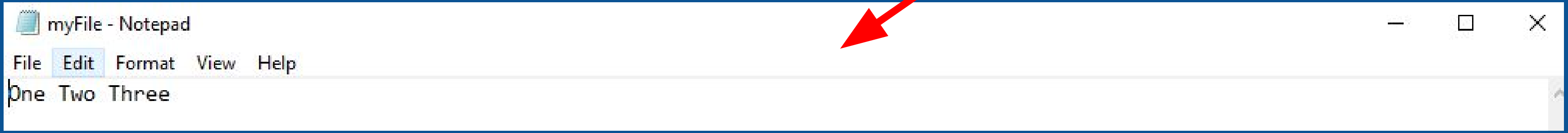


English

onetwothree.png

```
One Two Three
```

```
recognizing text
initializing api
loading eng.traineddata
found in cache eng.traineddata
initializing tesseract
loading tesseract core
```



```
One Two Three
```

Thank you for your attention

OCR component written by Christopher Eidangbe and Uros Milanovic,

using the Tesseract and Leptonica open source libraries

Website component primarily written by Anne Liang

Database component written by Ryan McCormick

How Database API works

- Create an instance
- Connect via google shell to server
- Create database and subsequent table
- Using the Cloud SDK
  - Uses gcloud
  - Manages SQL instances
- phpMyAdmin
  - Used to administer MySQL over the web