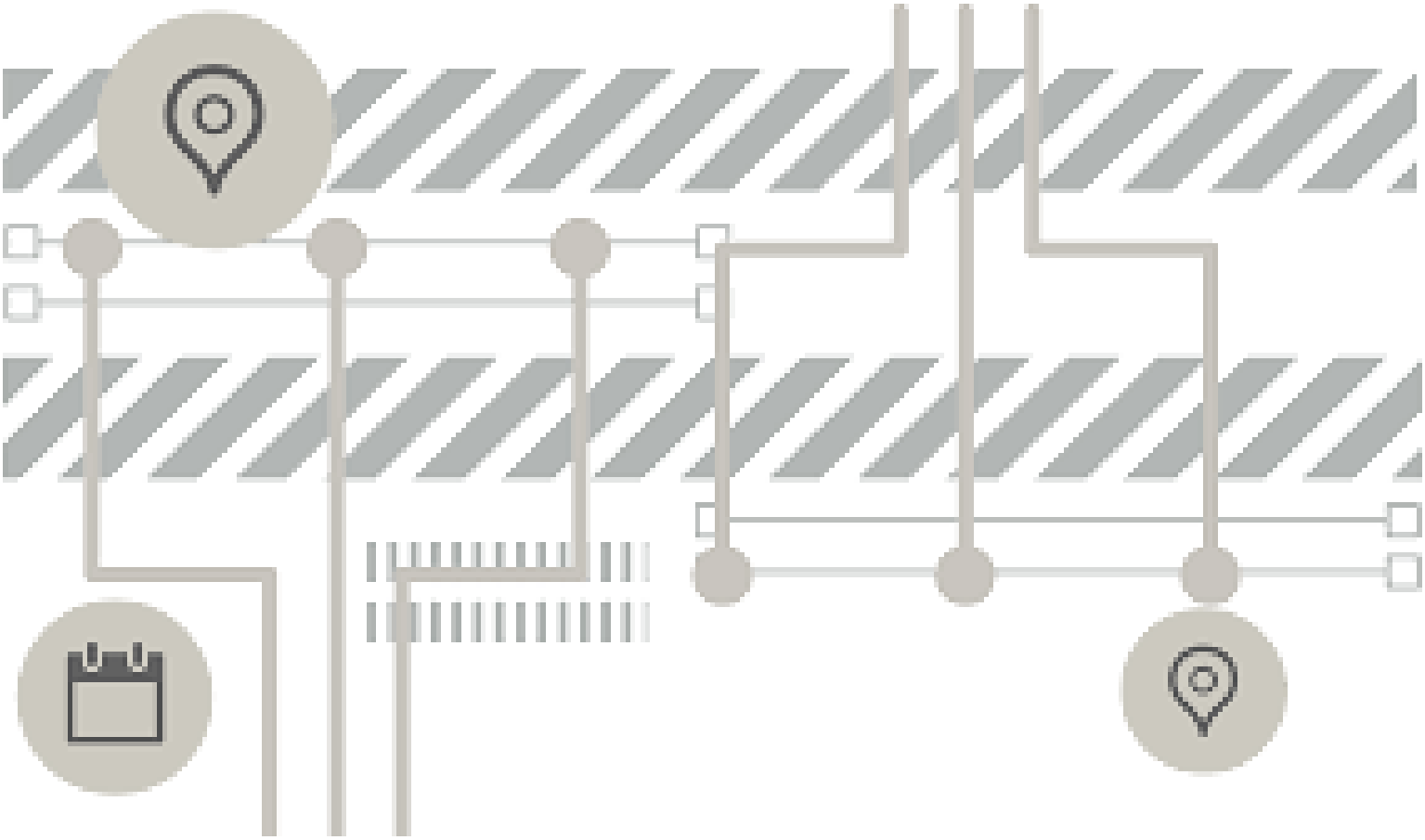


NAMED ENTITY RECOGNITION FOR SCANNED IMAGES

PROJECT REPORT
2021

Anuva Goyal , Kritika

ABSTRACT

In this project, we built the model using machine learning libraries and algorithms, to categorize and provide important information from the extracted text from the images. The overall project consists of two models:

- OPTICAL CHARACTER RECOGNITION (OCR): This technology is a business solution for automating data extraction from printed or written text from a scanned document or image file and then converting the text into a machine-readable form to be used for data processing like editing or searching.

- NAMED ENTITY RECOGNITION (NER): It is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into pre-defined categories such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc.

CONTENTS

PROBLEM STATEMENT

To create a text recognition and classification model using machine learning libraries and algorithms. The tasks involved are following:

1.Capturing the live image
2.Cleaning the image
3.Extracting text using pytesseract
4.Classifying text using regex(REs)
5.Classifying text using SpaCy
6.Saving the output

LIBRARIES AND ALGORITHMS

OpenCV :
OpenCV is a huge open-source library for computer vision, machine learning, and image processing. It can process images and videos to identify objects, faces, or even the handwriting of a human.

Pytesseract :
Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and "read" the text embedded in images.
Python-tesseract is a wrapper for Google's Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, BMP, tiff, and others.

REs :
Regular expressions (called REs, or regexes, or regex patterns) are essentially a tiny, highly specialized programming language embedded inside Python and made available through the re module. Using this little language, you specify the rules for the set of possible strings that you want to match; this set might contain English sentences, or e-mail addresses, or TeX commands, or anything you like.

Convutional Neural Network (CNN) :
A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms.

spaCy :
This is an open-source software library for advanced natural language processing, written in the programming languages Python and Cython.
spaCy is designed specifically for production use and helps you build applications that process and "understand" large volumes of text. It can be used to build information extraction or natural language understanding systems or to pre-process text for deep learning.

NLTK :
It is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

BERT :
This is an open source machine learning framework for natural language processing (NLP). It is designed to help computers understand the meaning of ambiguous language in text by using surrounding text to establish context. This framework was pre-trained using text from Wikipedia and can be fine-tuned with question and answer datasets.
BERT, which stands for Bidirectional Encoder Representations from Transformers, is based on Transformers, a deep learning model in which every output element is connected to every input element, and the weightings between them are dynamically calculated based upon their connection.

PyTorch :
It is an open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab. It is easier to learn and lighter to work with, and hence, is better for passion projects and building rapid prototypes.

TECHNIQUES

1. FOR OCR:

OCR has two parts to it. The first part is text detection where the textual part within the image is determined. This localization of text within the image is important for the second part of OCR, text recognition, where the text is extracted from the image. Using these techniques together is how you can extract text from any image.

(i) Using CNN:

Step1 : Built a digit(0-9) + A-Z characters classifier using a CNN architecture.
Step2 : Applied character segmentation for the handwritten word image.
Step3 : Classified each segmented letter and then get the final word in the image.

(ii) Using Pytesseract:

Since it is a pretrained model we can directly use it to extracttext from the images. It can be downloaded using pip install command.

2. FOR NER:

(i) Using NLTK:

Step1 : Tokenized the extracted text
Step2 : Created chunks (phrases from unstructured text) using POS tagging
Step3 : Added labels to entities

(ii) Using BERT:

NER model can be trained by feeding the output vector of each token into a classification layer that predicts the NER label.

Step1 : Loaded the dataset
Step2 : Preprocessed the text
Step3 : Pretrained the BERT model
Srep4 : Predicted the entities

(iii) Using spaCy:

After tokenization, spaCy can parse and tag a given Doc. This is where the trained pipeline and its statistical models come in, which enable spaCy to make predictions of which tag or label most likely applies in this context. A trained component includes binary data that is produced by showing a system enough examples for it to make predictions that generalize across the language .
Like many NLP libraries, spaCy encodes all strings to hash values to reduce memory usage and improve efficiency.

(iv) Using REs:

Created patterns that matched with the following sub-strings:
Email
Name
Phone Number

Used datefinder module to extract dates from the text.

OBSERVATIONS AND RESULT

1. FOR OCR:

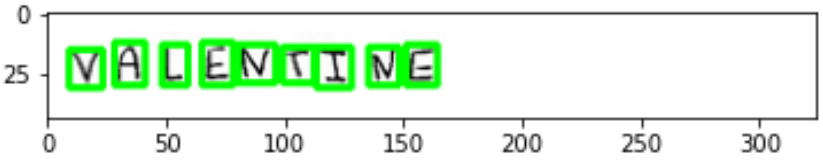


Fig 1. Result of cnn model

The recognition part is dependent on the contour detection code, so if the opencv library is not able to find the character contour, then this method will fail.There could be a lot of variation in a single handwritten letter in terms of writing style, therefore a lot more examples are needed for training this model.
This model will not work for connected texts like a cursive handwritten word.

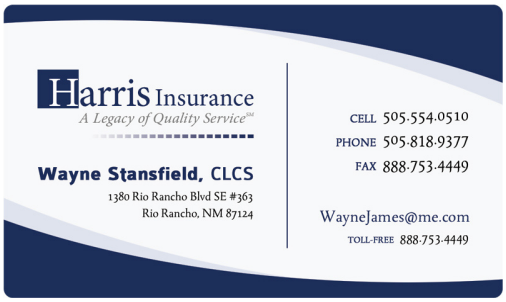


Fig 2. Sample business card

```
# image_to_string method reads all the characters!
text = pytesseract.image_to_string(result)
print(text)
```

```
Bilar ris Insurance

A Legacy of Quality §

CELL 505.554.0510
PHONE 595-818-9377
FAX 888-753-4449


Wayne Stansfield, CLCS
1380 Rio Rancho Blvd SE #363

Rio Rancho, NM 87124 WayneJames@me.com

TOLL-FREE 888-753-4449
```

Fig 3. Text extracted using pytesseract

CONCLUSION : As we can observe from the above observations that pytesseract is able to extract most of the text without any major drawback. Hence, we opted this for our final project.

2. FOR NER:

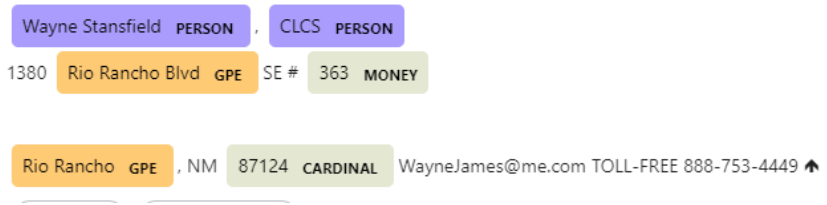


Fig 4. Result of spaCy

```
    # First name and Last name are always Proper Nouns
    pattern = [{'POS': 'PROPN'}, {'POS': 'PROPN'}]

    matcher.add('NAME', None, pattern)

    matches = matcher(nlp_text)

    for match_id, start, end in matches:
        span = nlp_text[start:end]
        return span.text

name = extract_name(text)
print(name)
extracted_text['Name'] =  name
```

```
Wayne Stansfield
```

```
#DATE
import datefinder
matches = datefinder.find_dates(text)
for match in matches:
    print(match)

extracted_text['Date'] =  match
```

```
1380-08-13 00:00:00
```

```
#PHONE NUMBER
import re
def get_phone_numbers(string):
    r = re.compile(r'(\d{3}[-\.\s]??\d{3}[-\.\s]??\d{4}|\(\d{3
    phone_numbers = r.findall(string)
    return [re.sub(r'\D', '', num) for num in phone_numbers]

phone_number= get_phone_numbers(text)
if len(phone_number) <= 10:
    print(phone_number)
    extracted_text['Phone number'] = phone_number
```

```
['5055540510', '5958189377', '8887534449', '8887534449']
```

```
#E-MAIL
import re
def get_email_addresses(string):
    r = re.compile(r'[\w\.-]+@[\w\.-]+')
    return r.findall(string)

email = get_email_addresses(text)
print(email)
extracted_text['E-mail'] = email
```

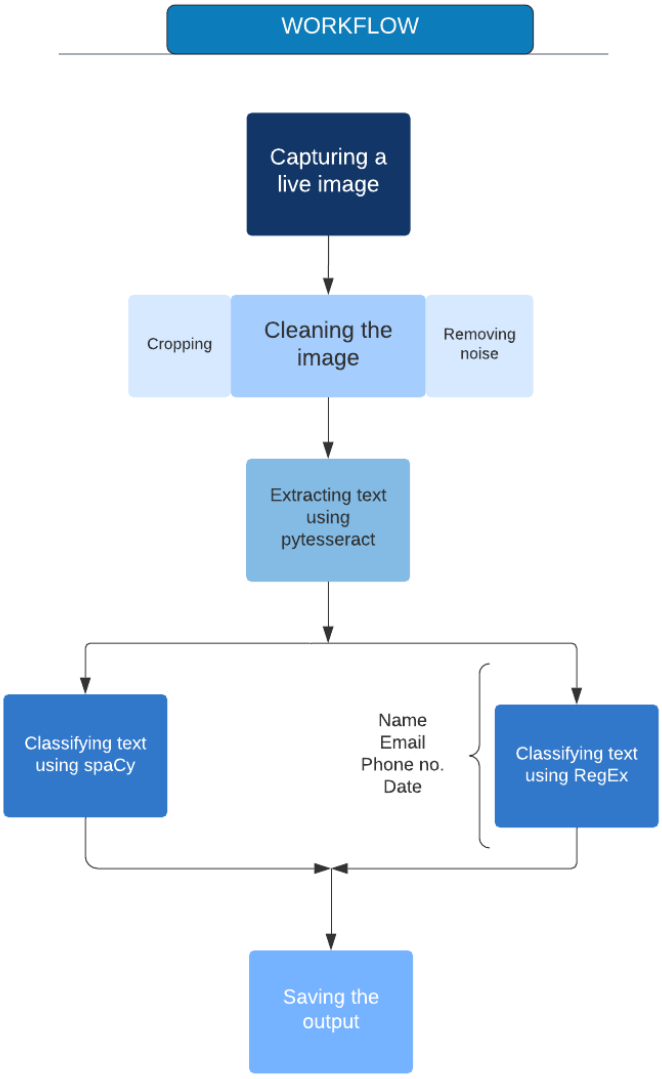
```
['WayneJames@me.com']
```

Fig 5. Result for REs

The main drawback of using bert model is the computational resources needed to train/fine tune and make instances. Moreover, the number of tokens are limited to 512: which is not suitable for our project.

The performance of NLTK was poor as the labels provided to the entity were not providing much relevant onformation.

CONCLUSION : spaCy and REs are giving satisfactory results and the model is able to classify the required fiels of information. Hence we used these in our final project



REFERENCES

Aman Kumar (2020) *Offline Handwritten Text OCR*, Available at: *https://www.kaggle.com/aman10kr/offline-handwritten-text-ocr* (Accessed: 10th June 2021).

Dipanjan Sarkar (2020) *Named Entity Recognition: A Practitioner's Guide to NLP*, Available at: *https://www.kdnuggets.com/2018/08/named-entity-recognition-practitioners-guide-nlp-4.html* (Accessed: 16th June 2021).

Filip Zelic, Anuj Sable (2020) *A comprehensive guide to OCR with Tesseract, OpenCV and Python*, Available at: *https://nanonets.com/blog/ocr-with-tesseract/* (Accessed: 10th June 2021).

Susan Li (2018) *Named Entity Recognition with NLTK and SpaCy*, Available at: *https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da* (Accessed: 16th June 2021).