VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
INSTITUTE OF COMPUTER SCIENCE
INFORMATION TECHNOLOGIES STUDY PROGRAM

Problem-Based Project

Digitalization of a business cards

Done by:
Oleksii Morozov

Supervisor:
Dr. Linas Bukauskas

Vilnius
2025

Preface

This project was done during the 3rd semester of the study programme Information Technologies in the specialization of Innovative studies. We chose the topic Digitalization of a business cards suggested during the project market on the first lecture of the subject.

June 23, 2025

Oleksii Morozov

Contents

Abstract

This project is a sophisticated software designed to convert English business card images into a standardized digital format, with the possibility of further manipulations of already digitized cards. The goal of the project is to replace a large stack of physical business cards by storing them digitally on a remote server so that the user could view them from any computer. The user has the opportunity to choose a card or several cards for further conversion. The card is converted in several steps. The first step is to prepare the card's image by applying filters and extracting text using Tesseract OCR. The next step is to classify the text using regular expressions and natural language processing (NLP) SpaCy. Data sets are used to improve recognition of names and job titles. Finally, we receive a digitized version of the card, which is automatically saved in a remote PostgreSQL-based database in the table format. After saving, the user can view and delete saved cards. Before using the program, the user must install all the necessary modules and be on the MIF local network.

Keywords: Image processing, Tesseract-OCR, SpaCy, Image to text, PostgreSQL, regular expressions

Santrauka

Vizitinių kortelių skaitmeninimas

Šis projektas yra sudėtinga programinė įranga, skirta angliškų vizitinių kortelių atvaizdams konvertuoti į standartizuotą skaitmeninį formatą, su galimybe toliau manipuliuoti jau suskaitmenintas korteles. Projekto tikslas - pakeisti didelę fizinių vizitinių kortelių kiekį išsaugant jas skaitmeniniu būdu nuotoliniame serveryje, kad būtų įmanoma jas peržiūrėti iš bet kurio kompiuterio. Vartotojas turi galimybę pasirinkti kortelę ar kelias korteles tolimesniam konvertavimui. Konvertavimas vyksta keliais žingsniais. Pirmas žingsnis yra paruošti kortelės vaizdą, pritaikant filtrus ir ištraukiant tekstą naudojant Tesseract OCR. Sekantis žingsnis yra teksto klasifikacija naudojant įprastas išraiškas ir natūralios kalbos apdorojimą (angl. *natural language processing*, NLP) SpaCy. Duomenų rinkiniai pagerina vardų ir pareigų pavadinimų atpažinimą. Galiausiai, gaunama suskaitmenintą kortelės versiją, kuri lentelės pavidalu automatiškai išsaugoma nuotolinėje PostgreSQL-pagrįsta duomenų bazėje. Po išsaugojimo vartotojas gali peržiūrėti ir ištrinti išsaugotas korteles. Prieš naudodamasis programine įranga, vartotojas turi įdiegti visus reikiamus modulius ir būti prijungtas prie MIF vietinio tinklo.

Raktiniai žodžiai: vaizdo apdorojimas, Tesseract-OCR, SpaCy, vaizdas į tekstą, PostgreSQL, reguliarios išraiškos

Introduction

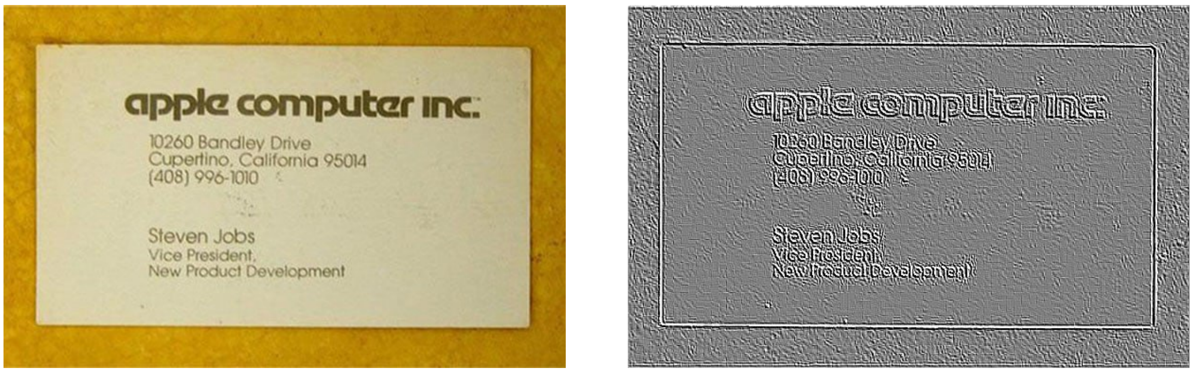
Business card scanning software systems make the process of capturing, organizing, and managing contact information from physical to virtual business cards. To use the program the user needs to provide an image of a card they have. It provides image preprocessing, text extraction, and text classification, and returns the results. Also, users can save their cards in the remote database. The software is written in Python.

1 Problem Analysis

1.1 Image processing and text extraction

Raw image data direct from a camera may have various problems, and therefore it is not likely to produce the best computer vision results. This is why careful consideration of image preprocessing is fundamental, additionally, some exploratory work is required to fine-tune the image preprocessing stage for best results [7]. Tesseract does various image processing operations internally, such as rescaling, binarisation, noise removal, dilation, erosion, and deskewing (using the Leptonica library) before doing the actual OCR. It generally does a very good job of this, but there will inevitably be cases where it isn't good enough, which can result in a significant reduction in accuracy [1]. Also, considering the importance of the accuracy of the data received from the business card it's necessary to improve the quality of text extraction by adding conditions for loaded images, such as sufficient lighting and the image should be taken at almost a right angle. From here we can say that for a business card scanner, there is no need to add many additional image preparation tools.

Filters are selected situationally depending on the image. Since developing an algorithm to determine which filters are needed is a time-consuming task, we will choose 2 main filters that are suitable for almost any situation. In Figure 1 an example of the use of the EMBOSS filter [4] is presented, which is used for images with poor lighting. In this case, the filter will make it difficult to recognize the text due to the large amount of noise and deteriorated text quality.



(a) Original image

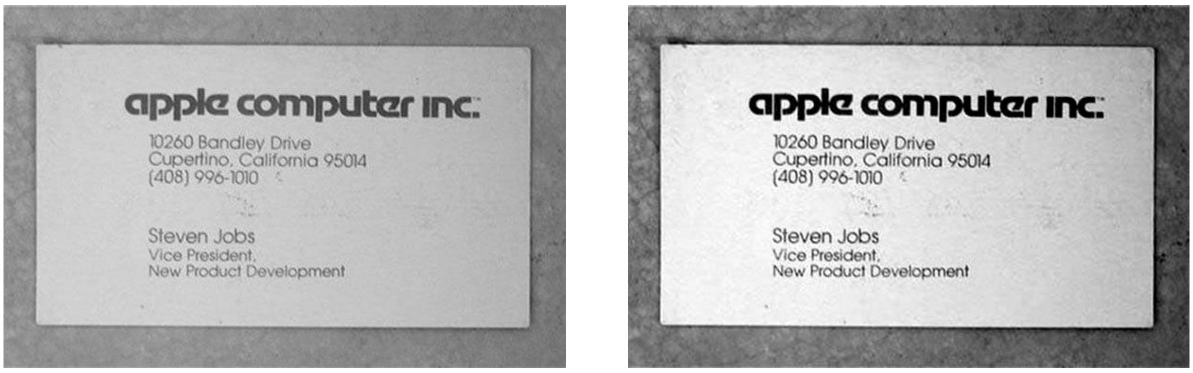
(b) Image after EMBOSS filter

Figure 1. Bad example of using filters

The most suitable filters for the vast majority of cases will be the grayscale (mode "L") and contrast method. Using the L-filter for the image that changes the saturation of the three primary colors (Red, Green, and Blue) or grayscaling can reduce dimensionality, improve robustness to lighting changes, and potentially boost specific computer vision tasks, all while saving computational resources [7]. The formula 1.1 demonstrates the dimming of red, green, and blue colors.

$$L = \frac{R \cdot 299}{1000} + \frac{G \cdot 587}{1000} + \frac{B \cdot 114}{1000} \tag{1.1}$$

Contrast is the kind of distance between the colors. If you increase the contrast, the colors are more vivid [7]. By increasing the image contrast by exactly 2 times, we will get clearer and brighter text [4]. In Figure 2 final result is represented by photo b.


apple computer inc.
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010
Steven Jobs
Vice President,
New Product Development

apple computer inc.
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010
Steven Jobs
Vice President,
New Product Development

(a) Image after grayscaling (b) Image after contrast filter

Figure 2. Suitable filters

1.2 Classification of scanned text

Text analysis is a crucial task that aims to classify text from the business card into predefined categories to facilitate their management and analysis. Natural language processing (NLP) has emerged as an important tool to overcome this challenge by offering sophisticated techniques for studying and understanding human language. These algorithms are very good at retrieving texts (IR), splitting them into parts, checking the spellings, and word-level analysis, but not successful for analysis at sentence and paragraph levels. Hence, when it comes to the question of interpreting sentences and extracting meaningful information, however, the capabilities of these algorithms are still very limited [3].

Today there are many tools for linguistic analysis, but the best 2 for this task are NLTK [8] and SpaCy [6] due to their simplicity, accessibility, and compactness. After conducting a study, it was revealed that NLTK cannot cope with text recognition without any context, even after training. SpaCy showed much better results, but there are still problems with recognizing people's names, hence the development of additional algorithms is necessary to simplify classification and increase the accuracy of the result.

One of the tasks of improving the quality of classification results is to recognize simpler entities such as telephone, email, and website and remove them from the original text. Since all of the listed entities have a strict structure, regular expressions will be used. In the regular expression recognition technique, each character in the text to be searched is examined in sequence against a list of all possible current characters [10]. This method is not suitable for recognizing physical addresses, because different countries have different writing standards, or there may be no standardization at all.

The main part of text classifying is the names of people and companies. The most complex task of the name extraction process is the splitting of one candidate sequence into two or more names. When a sequence contains more than one name, it is a linguistic structure that needs to be parsed. It poses, unfortunately, a subset of the most thorny problems known in natural language parsing [9]. To solve this problem, we consider taking additional steps using additional data to process the results obtained after NLP.

1.3 Data storing

Business card information includes a range of data such as name, title, affiliation, contact details, and links to web resources. There are 2 ways to save this type of data: locally (on the user's

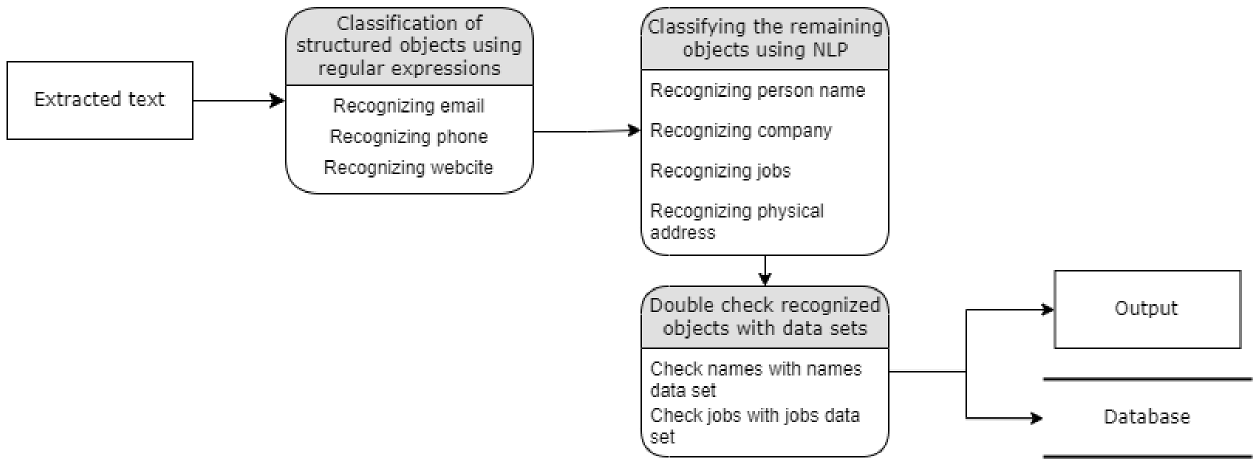

Extracted text
Classification of structured objects using regular expressions
Recognizing email
Recognizing phone
Recognizing webcite
Classifying the remaining objects using NLP
Recognizing person name
Recognizing company
Recognizing jobs
Recognizing physical address
Double check recognized objects with data sets
Check names with names data set
Check jobs with jobs data set
Output
Database

Figure 3. Data flow during text classification

computer) and on the server. The second option today is much more in demand and convenient from the user's perspective because there is no connection to a specific machine. The best solution would be a relational database, such as PostgreSQL, which allows the creation of clearly defined tables and schemas. This structured approach makes it easy to organize various data points into meaningful categories, making data access and analysis easier.

1.4 Analysis result

After analyzing the project, filters for image preprocessing, a tool for NLP and data sets to improve the results, and a method for storing data were determined. Also, during the research, it was found that it is currently impossible to achieve the ideal operation of the program due to the human factor, the imperfection of computer vision, and the lack of a tool for classifying text without context. Thus, the expected accuracy of the final product will vary between 66.52% and 81.92% (based on data from the AETHER BCR and ABBY BCR projects).

2 Design

2.1 Functional Requirements

1. Program should be able to accept an image of a business card as input;
2. Program must support for common image formats: JPEG, PNG, and WEBP;
3. Scan image and recognize data fields: full name, phone number, company name, address, email, job title, URL;
4. Scanned entities should be stored in table format;
5. Users able to view contacts via command line interface (CLI);
6. Display the scanned cards in an easy-to-understand format;
7. Program must work in the VU MIF network.

2.2 Non-Functional Requirements

1. OCR accuracy at least 80% on English business cards with good quality pictures;
2. Actions taken by the user, such as picture submission, should provide textual feedback to indicate success or any issues.
3. The image processing server should be able to handle concurrent user input;
4. Clear documentation, detailing program functions and user capabilities, should be provided to help end-users utilize the system effectively.

3 Architecture

3.1 Client

The client function is executed exclusively on the user's machine, utilizing a pre-installed program accessible through the command line interface. This command-based interface allows direct interaction with the program, enabling users to control its operations and analyze their outputs. Several essential modules must be installed to ensure full functionality, including Tesseract, an optical character recognition (OCR) tool, and SpaCy, a natural language processing (NLP) library, to extract and process text information. Also, the machine must be connected to the MIF network.

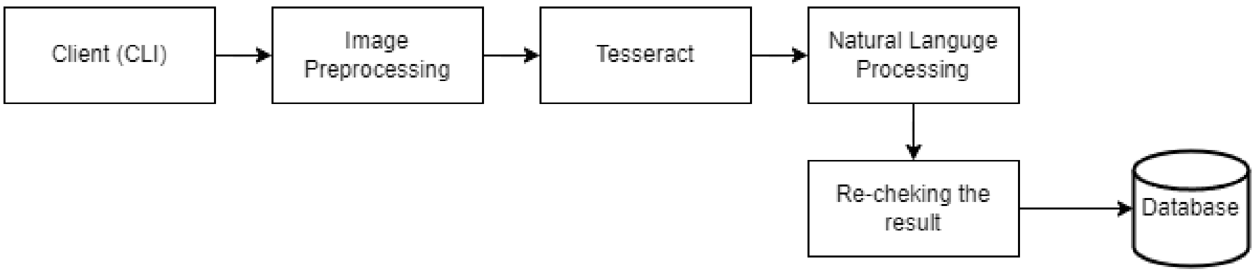


Figure 4. System Architecture (Simplified)

3.2 Database

A PostgreSQL database is used to store and manage the scanned images. This database is hosted on a remote virtual machine that is connected to the local MIF network. Each scanned image is assigned a unique identifier to ensure efficient retrieval and organization. This approach provides data integrity and scalability, making it suitable for managing large volumes of images. In addition, remote deployment of the database provides enhanced security and availability. The database has only 1 table holding business card info, as seen in Figure 5.

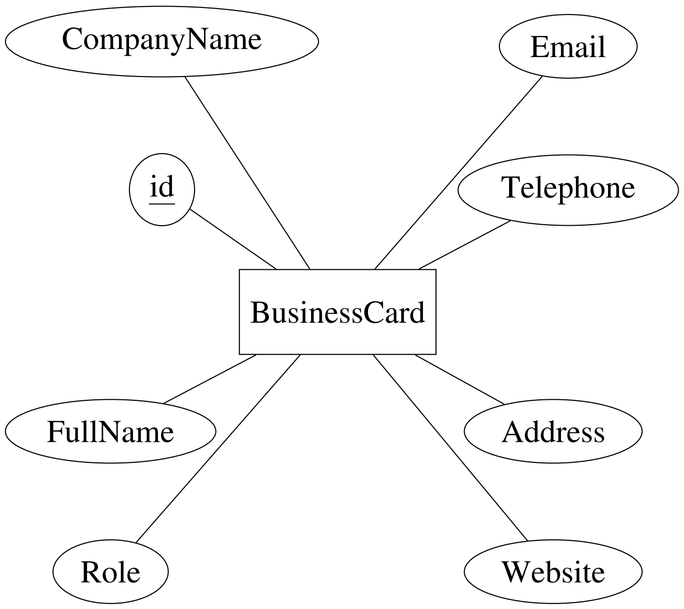


Figure 5. E-R diagram

3.3 Image preprocesing

Image processing [2] has one major step --- filtering the image. Filters are necessary because they improve Tesseract accuracy and remove any unnecessary information contained in the photo. Image filtering steps:

1. Built-in filters of Tesseract [1] (Leptonica filters);
2. Make image grayscaled;
3. Contrast the image.

More about filters was written in the 1 chapter and the results were presented in Figure 2.

3.4 Natural Language Processing

The main tool for text classification is SpaCy [6], whose main task is the initial recognition of names of people, companies, and positions.

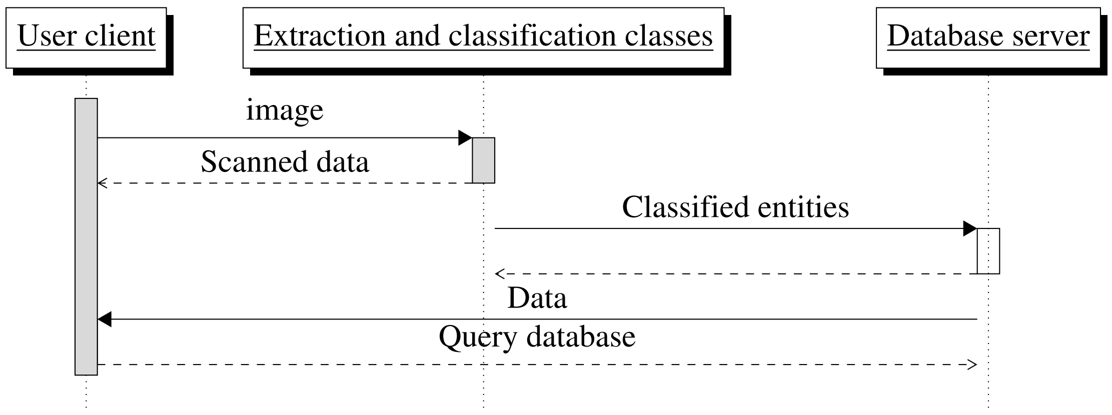


Figure 6. Sequence diagram of server-client system

Also, the program uses regular expressions because they have hard rules that will either match or not, there is no ambiguity like with statistical language models. Although great, they cannot identify differences between "Steven Jobs" and "Apple Company". Syntax wise they are the same, just different letters that we attach meaning to. We use it for emails, phones, and websites because it has a well-defined structure. Telephones could also be added, but there are too many formats of telephones which is why we left it to the language model.

EMAIL entity regular expression

```
[a-z0-9\.\-+_]+@[a-z0-9\.\-+_]+\.[a-z]+
```

Explanation

1. `"[a-z0-9\.\-+_]+"` - This part matches the username part of the email address. The "+" symbol indicates that this group can be repeated one or more times. The parentheses around the group are optional and are used for grouping and capturing purposes.
2. `"@"` - This symbol matches the literal "@" character that separates the username from the domain name.

3. `"[a-z0-9\.\-+_]+"` - This part matches the domain name part of the email address. The requirements are the same as for the username part.

4. `"\."` - This symbol matches the literal period character that separates the top-level domain from the subdomains.

5. `"[a-z]+"` - This part matches the top-level domain of the email address. The top-level domain is the last part of the domain name, such as "com", "edu", or "org". The "+" symbol indicates that this group can be repeated one or more times.

WEBSITE entity regular expression

For sites, the longest pattern was selected to achieve 100% accuracy in site classification. The regex patterns of websites are intended to match any URLs [5].

Main parts

1. `((http|https):\/\/)?` - Can begin with "http://" or "https://"

2. `([\w_-]+(?:(?:\.[\w_-]+)+))` - Match word one or more times if it has dot and word one or more times later in the structure

PHONE entity regular expression

```
[\+\(]?[1-9][0-9 .\-\(\)]{8,}[0-9]
```

Explanation

1. `[\+\(]`: This optional group allows either a plus sign (+) or a left parenthesis ((), which are common prefixes for international phone numbers.

2. `[1-9]`: This character class matches a single digit from 1 to 9, indicating the country code.

3. `[0-9 .\-\(\)]{8,}`: This group matches a sequence of 8 or more digits, optionally separated by spaces, periods, or parentheses. This represents the national phone number.

4. `[0-9]`: This final character class matches a single digit, indicating the country code extension (if applicable).

4 Results of scanning

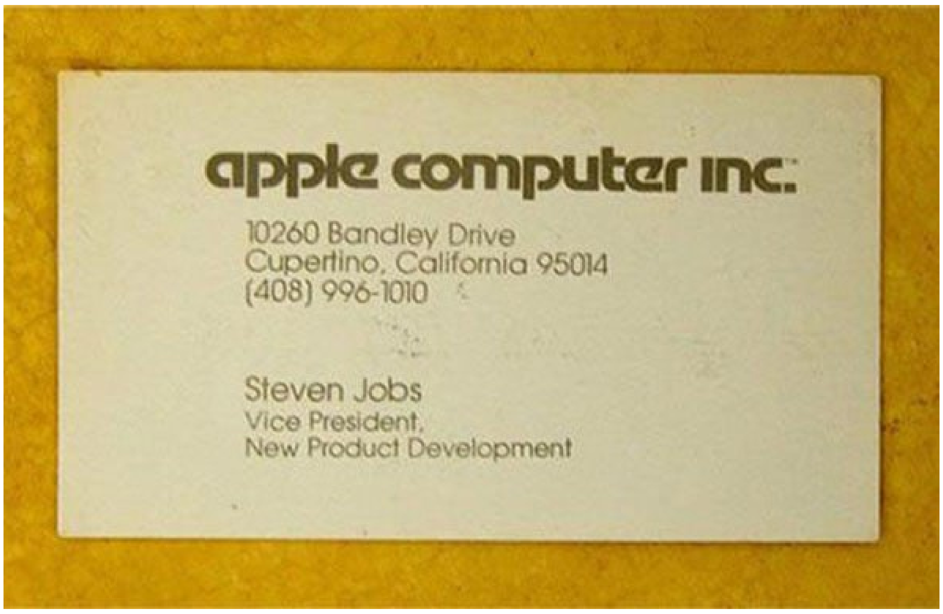

apple computer inc.
10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010
Steven Jobs
Vice President,
New Product Development

Figure 7. Original business card

Extracted text after preprocessing:

```
apple computer inc.

10260 Bandley Drive
Cupertino, California 95014
(408) 996-1010 ©

Steven Jobs
Vice President,
New Product Development
```

Classified text:

```
[text_classifier.py:59 - classify()]
RESULT: {'person': 'Steven Jobs', 'org': ['Apple Computer Inc',
'Bandley Drive'], 'jobs': ['Vice President',
'Product Development'], 'phones': ['(408) 996-1010'], 'emails': [],
'websites': []}
```

5 Uniform Descriptions of Unit Tests

The purpose of unit tests is to ensure that individual components or "units" of a program work as intended. They help identify bugs early in the development process, making it easier to maintain and enhance the software over time. Unit tests also provide documentation for how each component should behave and facilitate smoother integration of these components into larger systems. In summary, unit tests contribute to the reliability, quality, and robustness of software. Testing Classifier: `test_removeSubstrings(self)`. This test verifies the method's ability to remove specified substrings from a given input string. It provides an input string and a list of substrings to remove and checks if the output string indeed lacks those substrings.

`test_enhance_text_for_spacy(self)`. This test examines the method's capability to convert an input string into a format suitable for processing by the spaCy library. It compares the enhanced output to a predefined expected string, ensuring the proper formatting.

`test_try_spacy(self, input_text, expected_result)`. This test case evaluates the method's ability to extract named entities from a given input text using spaCy. It provides input text and an expected dictionary of entities, checking if the method returns the expected entity labels and corresponding mentions.

`test_find_jobs(self, maybe_job, expected_result)`. This test verifies the method's effectiveness in identifying job-related terms from an input string. It presents potential job titles and their corresponding expected outcomes (True or False), ensuring the method correctly identifies job-related terms.

`test_find_phones(self, maybe_phone, expected_result)`. This test examines the method's ability to detect potentially valid phone numbers from an input string. It provides possible phone numbers and their corresponding expected outcomes (True or False), ensuring the method accurately identifies phone number formats.

`test_find_emails(self, maybe_email, expected_result)`. This test validates the emails method's ability to locate potentially valid email addresses from an input string. It presents possible email addresses and their corresponding expected outcomes (True or False), confirming the method correctly identifies email address patterns.

References

[1] Tesseract documentation improving the quality of the output. https://tesseract-ocr.github.io/tessdoc/ImproveQuality.html. Accessed: 2024-02-12.

[2] Shazia Akram, Mehraj-Ud-Din Dar, and Aasia Quyoum. Document image processing- a review. International Journal of Computer Applications, 10(5):35--40, 2010.

[3] KR1442 Chowdhary and KR Chowdhary. Natural language processing. Fundamentals of artificial intelligence, pages 603--649, 2020.

[4] Alex Clark. Pillow (pil fork) documentation, 2015.

[5] John Gruber. Markdown: Syntax. https://daringfireball.net/projects/markdown/syntax, 2010. Accessed: February 12, 2024.

[6] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[7] Scott Krig and Scott Krig. Image pre-processing. Computer Vision Metrics: Textbook Edition, pages 35--74, 2016.

[8] Edward Loper and Steven Bird. Nltk: The natural language toolkit. arXiv preprint cs/0205028, 2002.

[9] Yael Ravin and Nina Wacholder. Extracting names from natural-language text. Citeseer, 1997.

[10] Ken Thompson. Programming techniques: Regular expression search algorithm. Communications of the ACM, 11(6):419--422, 1968.