Brief Report on ICDAR Paper Classification Assessment

Introduction

This project focuses on automating the classification of academic papers submitted to the ICDAR 2024 conference using natural language processing (NLP) techniques. The task involves extracting metadata from each paper (title, authors) and categorizing the papers into predefined groups such as Optical Character Recognition (OCR), Document Layout Understanding, Datasets, and others.

Methodology

1. **Data Retrieval Visualization and Extraction:**

   - **Data Retrieval** : ***Requests*** library is imported, in order to download the zip file containing the pdf papers, which will be used to extract the necessary information (title and author(s) of the file), from the GitHub repository, where the file is located at. It is important to note that the raw version of the file is needed, for the retrieval process to be successful. A new directory is created, in which the zip file is stored. Subsequently, using ***pathlib*** and ***zipfile*** libraries, all pdf files that represent the papers are extracted from the initial zip file. For each pdf file, its path is stored in a list.

   - **Data Visualization** : Before the extraction process begins, it is crucial that a basic knowledge on the format and contents of the data has been established. To do that, a custom function is defined that iterates a random pdf file using ***PyMuPDF*** library. The pdf file is accessed by providing its path from the list that was created in the previous step. The general assumption that is being made during the visualization and extraction process, based on the analysis of each paper in the dataset, is that the majority of titles are located between the first and third line of the page while all authors are mentioned below that point up until around the sixth line. This assumption is being made because the initial solution, i.e. the use of the built-in function **metadata,** which is commonly used in PyMuPDF library for automatically storing data such as the file's title, author and other relative information, was found ineffective when it came to this particular set of files**.**

   - **Text Extraction:** Using ***PyMuPDF***, another function reads each PDF file and extracts its first, and only, page. The metadata (title and authors) is then extracted and stored in a dictionary. By implementing ***Pandas*** library, the dictionary is then transformed into a dataframe that contains 3 columns

     1. Id : Contains the name of the pdf file.
     2. Title : The title of that file
     3. Authors : The names of all the authors that contributed on that file

- **Text Normalization:** Punctuation is removed (except for commas in author names), and authors are split into lists.

2. **Text Pre-processing and classification:**

   - **Pre-Processing :** Text pre-procesing techniques are applied on the text data to make it easier for the model to understand the patterns and classify it more accurately. Specifically the following techniques are being applied to the dataset :

     1. **Lower Casing**
     2. **Special Characters Removal**
     3. **Removal of all numbers in text**
     4. **Removal of all stopwords**
     5. **Noise Removal (dropping duplicates, missing values, etc...)**

     It is important to note that the original columns are kept mostly intact (only number and special character removal is applied) and 2 new columns are created that store the pre-processed data for each pdf file. This approach is taken since the original title and authors of each file are needed for later, when the extracted data will have to be stored in the final json file, that will contain the results of the classification.

   - **Classification:** The project uses a pre-trained zero-shot classification model from Hugging Face's transformers library to make classification on the unlabeled dataset, that was created using the extracted metadata from the pdf files. After the classification process a new **label** column is being created for each record in the dataset, in order to provide, for each entry, its most likely label. This will be useful when combining everything in a json file that will include the outputs of the code. A few words on zero-shot classification in order to provide further context. Zero-shot classification involves the process of using a model that has-pretrained on a different version of the problem it is being applied to. In this case the zero-shot classifier has been trained on multi-class text classification, but with different labels than the ones that are provided in this project.

3. **Output:** The classified papers are exported into a structured JSON file, where each paper is mapped to its category along with its title and authors.

Challenges

- **PDF Text Extraction Issues:** The initial extraction of text from the PDFs faced issues due to different formats and encoding inconsistencies, making some data extraction noisy. The solution involved rigorous text preprocessing and cleaning to normalize the text before passing it into the model.
- **Imbalanced Categories:** Some categories were underrepresented, leading to potential bias in classification. Addressing this imbalance is crucial for improving the model's accuracy on minority classes.