YAKE Keyword Extraction Report

Filename: Key word extraction

Tool Used: YAKE

Objective:

To extract meaningful keywords from .txt, .pdf, or .csv files using the YAKE (Yet Another Keyword Extractor) library, while filtering out irrelevant or unwanted terms. The extracted keywords are saved in both .json and .txt formats.

Top Keywords Extracted:

Rank	Keyword
1	climate
2	services
3	nacional
4	forestales
5	agrcolas
6	desert
7	instituto
8	investigation
9	methodology
10	deinvestigaciones
11	etal
12	mexico
13	ypecuarias
14	conceptualization
15	ecosystems
16	management
17	computing
18	datavisualization
19	ofthechihuahuan
20	description

My Code:

```
import os
import json
import pandas as pd
import PyPDF2
import yake
```

```
import re

blacklist = {
    "one", "two", "three", "four", "licensed", "umcg",
"ncs", "ncsa",
    "mednerdette", "image", "cc", "av",
"modifiedtricuspid", "stethescope",
    "separates", "and", "the", "of", "in", "to", "for",
"a"
}

def clean_text(text):
    text = re.sub(r'\n+', ' ', text)
    text = re.sub(r'[•–“”"\'-]', '', text)
    text = re.sub(r'\s{2,}', ' ', text)
    text = re.sub(r'\b\d+\b', '', text)
    text = re.sub(r'[^A-Za-z0-9\s]', '', text)
    return text.strip()

def read_text_file(path):
    with open(path, 'r', encoding='utf-8') as f:
        return f.read()

def read_pdf_file(path):
    text = ""
    with open(path, 'rb') as f:
        reader = PyPDF2.PdfReader(f)
        for page in reader.pages:
            page_text = page.extract_text()
            if page_text:
                text += page_text + "\n"
    return text

def read_csv_file(path):
    df = pd.read_csv(path)
```

```
    return "\n".join(df.astype(str).apply(lambda x: "
".join(x), axis=1))

def yake_keywords(text, top_n=20):
    kw_extractor = yake.KeywordExtractor(lan="en", n=1,
top=top_n)
    keywords = kw_extractor.extract_keywords(text)
    return [kw[0].lower() for kw in keywords if
kw[0].isalpha() and len(kw[0]) > 2 and kw[0].lower() not
in blacklist]

def extract_yake_keywords(file_path):
    ext = os.path.splitext(file_path)[1].lower()
    if ext == ".txt":
        text = read_text_file(file_path)
    elif ext == ".pdf":
        text = read_pdf_file(file_path)
    elif ext == ".csv":
        text = read_csv_file(file_path)
    else:
        raise ValueError("Unsupported file type: " + ext)

    clean = clean_text(text)
    keywords = yake_keywords(clean)

    print("\nTop YAKE keywords:")
    print(keywords)

    result = {
        "filename": os.path.basename(file_path),
        "yake_keywords": keywords
    }

    base =
os.path.splitext(os.path.basename(file_path))[0]
    json_file = f"yake_keywords_{base}.json"
```

```
    txt_file = f"yake_keywords_{base}.txt"

    with open(json_file, 'w', encoding='utf-8') as f:
        json.dump(result, f, indent=2, ensure_ascii=False)

    with open(txt_file, 'w', encoding='utf-8') as f:
        f.write("Top YAKE keywords:\n" +
"\n".join(keywords))

    print(f"\nKeywords exported to:\n→ {json_file}\n→
{txt_file}")

# Example Usage
file_path = r"C:\Users\kotii\Desktop\Heart file\Article 1-
main.pdf"
extract_yake_keywords(file_path)
```

Output:

```
Top YAKE keywords:
['climate', 'services', 'nacional', 'forestales', 'agrcolas', 'desert', 'instituto', 'investigation', 'methodolog
y', 'deinvestigaciones', 'etal', 'mexico', 'ypecuarias', 'conceptualization', 'ecosystems', 'management', 'comput
ing', 'datavisualization', 'ofthechihuahuan', 'description']

Keywords exported to:
→ yake_keywords_Article 1-main.json
→ yake_keywords_Article 1-main.txt
```

Explanation about Code:

Importing the modules:

import os, json, pandas as pd, PyPDF2, yake, re

os: File path handling and extension extraction

json: Saving keyword output in a structured JSON format

pandas: Reading and processing .csv files

PyPDF2: Reading and extracting text from PDF documents

yake: Keyword extraction based on statistical features

re: In text analysis regular expressions is commonly used for pattern matching, text extraction, text cleaning, tokenization and validation.

Blacklist:

A predefined set of stop words or irrelevant terms that are unnecessary in the final keyword results. This includes common English stop words and domain-specific noise terms.

Text Cleaning Function:

Regex is used for text cleaning tasks such as removing unwanted characters, white spaces or punctuation marks from text data. This ensures that the text is standardized and ready for further analysis or processing.

Tokenization: Regex is used for splitting text into tokens or smaller units such as words or sentences. It is a fundamental step in many text analysis tasks including natural language processing.

File Reading Functions:

def read_text_file(path): ... Opens and reads a text file

def read_pdf_file(path): ... Extracts the pdf file

def read_csv_file(path): ... Opens and reads a csv file

Main Function:

extract_yake_keywords (file_path)

Detects the file type (.txt, .pdf, .csv) by extension and loads the content accordingly.

Cleans the extracted text.

Extracts top keywords with YAKE from the cleaned text.

Output:

Save as JSON and TXT file.

Example:

- file_path = r"C:\path\to\file.pdf"
- extract_yake_keywords(file_path)

Observations:

The document includes a mix of English and Spanish terms.
Keywords like 'data visualization', 'computing', and 'methodology' suggest the paper discusses technical approaches.
Terms like 'desert', 'climate', and 'ecosystems' point toward an environmental science focus.