REPORT - 1

BRIEF STATEMENT:

The presented algorithm is designed for multi-label classification, specifically for predicting the subject areas of arXiv papers based on their abstract bodies. The model is built using shallow neural networks and is implemented using TensorFlow and Keras. Actually this work is inspired from "Large-scale multi-label text classification" at Keras NLP documentation. The workflow includes importing necessary libraries, extracting text from PDF files, isolating abstracts, text vectorization, loading a pre-trained model, and performing inference on new abstracts. The algorithm demonstrates an application of natural language processing (NLP) techniques to categorize academic papers efficiently.

Usage in Finance:

While the primary focus of this algorithm is on classifying research papers, it can be adapted for use in finance with some modifications. Here are potential applications:

1. Research Paper Classification in Finance:

- ❖ Instead of arXiv papers, the algorithm could be trained on a dataset of finance-related research papers.
- ❖ The model could then predict the subject areas or topics of financial research papers, aiding researchers and analysts in organizing and categorizing vast amounts of literature.

2. **News and Document Classification:**

- ❖ The algorithm can be extended to classify financial news articles, reports, or documents based on their content.
- ❖ This could assist financial professionals in quickly identifying relevant information, market trends, or regulatory updates by automating the categorization process.

3. **Sentiment Analysis:**

- ❖ With slight modifications, the model could be trained to perform sentiment analysis on financial texts.
- ❖ This could be valuable for gauging market sentiment from news articles or social media, providing insights for investment decision-making.

4. **Risk Assessment:**

- ❖ By incorporating additional features and training data related to financial risk, the algorithm could assist in automated risk assessment for financial documents.

❖ This application could be useful for compliance teams in financial institutions.

It's important to note that adapting the algorithm for finance would require a domain-specific dataset for training and fine-tuning, as financial language and terminology differ from general academic literature. Additionally, careful consideration and validation would be needed to ensure the model's effectiveness and reliability in a financial context.

WORKFLOW:

1. **Imports Libraries:**

Imports necessary libraries such as TensorFlow, Keras, NumPy, pandas, Matplotlib, fitz (PyMuPDF), and others.

```
from tensorflow.keras.models import load_model
from tensorflow.keras import layers
from tensorflow import keras
import tensorflow as tf
from sklearn.model_selection import train_test_split
from ast import literal_eval
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import fitz
from docx import Document
```

2. **PDF Text Extraction:**

Defines functions (`extract_text_from_pdf` and `convert_to_docx`) to extract text from a PDF file using PyMuPDF and convert the extracted text to a Word document.

```
def extract_text_from_pdf(pdf_path):
    doc = fitz.open(pdf_path)
    text = ""
    for page_num in range(doc.page_count):
        page = doc[page_num]
        text += page.get_text()
    return text
```

3. **Abstract Extraction:**

Defines a function (`extract_abstract_from_pdf`) to extract the abstract from a PDF file. It looks like it tries to locate the abstract based on the presence of the keyword "Keywords" and extracts the text before that.

```
def extract_abstract_from_pdf(pdf_path):
    doc = fitz.open(pdf_path)
    abstract = ""
```

4. **Text Vectorization:**

Utilizes TensorFlow's TextVectorization layer to preprocess text data for training and inference. It adapts the vectorizer to the training data.

```
def make_inference_dataset(texts):
    texts = tf.convert_to_tensor(texts)
    texts = text_vectorizer(texts)
    return texts
```

5. **Load Trained Model:**

Loads a pre-trained model using Keras' `load_model` function. The model is loaded from a file path specified in `saved_model_path`.

```
saved_model_path = "C:/Users/lenovo/Desktop/Research/TASK 1/my_model"
loaded_model = load_model(saved_model_path)
```

6. **Inference Preparation:**

Defines a function (`make_inference_dataset`) to prepare text data for inference using the loaded model.

```
def make_inference_dataset(texts):
    texts = tf.convert_to_tensor(texts)
    texts = text_vectorizer(texts)
    return texts
```

7. **Perform Inference:**

Calls `perform_inference` function to make predictions on a given abstract using the loaded model.

```

def perform_inference(model, abstracts):
    inference_dataset = make_inference_dataset(abstracts)
    predicted_probabilities = model.predict(inference_dataset)
```

8. **Data Loading:**

Loads data from a CSV file using pandas (`arxiv_data`). It seems to be related to arXiv data.

```
arxiv_data = pd.read_csv("https://github.com/soumik12345/multi-label-text-
classification/releases/download/v0.2/arxiv_data.csv")
```

9. **Vocabulary and Text Vectorization Setup:**

Sets up a StringLookup layer for multi-hot encoding, determines the vocabulary size, and initializes a TextVectorization layer.

```
terms = tf.ragged.constant(arxiv_data["terms"].apply(literal_eval).values)
lookup = tf.keras.layers.StringLookup(output_mode="multi_hot")
lookup.adapt(terms)
vocab = lookup.get_vocabulary()

vocabulary = set()
arxiv_data["summaries"].str.lower().str.split().apply(vocabulary.update)
vocabulary_size = len(vocabulary)

text_vectorizer = layers.TextVectorization(max_tokens=153375, ngrams=2,
output_mode="tf_idf")
text_vectorizer.adapt(arxiv_data["summaries"])
```

10. **PDF Processing and Inference:**

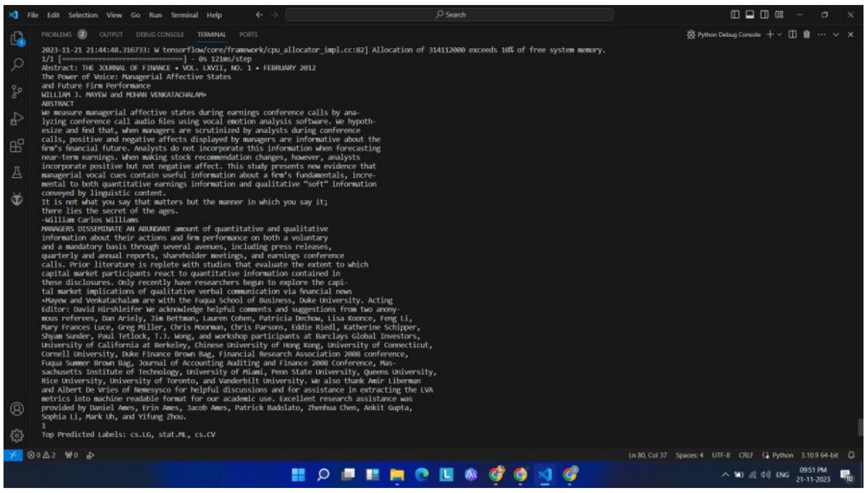
Specifies a PDF file path, extracts the abstract from the PDF, performs inference using the loaded model on the extracted abstract, and converts the entire PDF text to a Word document.

```
def perform_inference(model, abstracts):
    inference_dataset = make_inference_dataset(abstracts)
    predicted_probabilities = model.predict(inference_dataset)

    for i, text in enumerate(abstracts):
        predicted_labels = [label for _, label in
sorted(zip(predicted_probabilities[i], vocab), reverse=True)][:3]
        predicted_labels_string = ", ".join(predicted_labels)

        print(f"Abstract: {text}")
        print(f"Top Predicted Labels: {predicted_labels_string}")
        print()
```

RESULTS:



```
2023-11-21 21:44:48.316733: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 314112000 exceeds 10% of free system memory.
1/1 [==============================] - 0s 121ms/step
Abstract: THE JOURNAL OF FINANCE • VOL. LXVII, NO. 1 • FEBRUARY 2012
The Power of Voice: Managerial Affective States
and Future Firm Performance
WILLIAM J. MAYEW and MOHAN VENKATACHALAM∗
ABSTRACT
We measure managerial affective states during earnings conference calls by ana-
lyzing conference call audio files using vocal emotion analysis software. We hypoth-
esize and find that, when managers are scrutinized by analysts during conference
calls, positive and negative affects displayed by managers are informative about the
firm's financial future. Analysts do not incorporate this information when forecasting
near-term earnings. When making stock recommendation changes, however, analysts
incorporate positive but not negative affect. This study presents new evidence that
managerial vocal cues contain useful information about a firm's fundamentals, incre-
mental to both quantitative earnings information and qualitative "soft" information
conveyed by linguistic content.
It is not what you say that matters but the manner in which you say it;
there lies the secret of the ages.
-William Carlos Williams
MANAGERS DISSEMINATE AN ABUNDANT amount of quantitative and qualitative
information about their actions and firm performance on both a voluntary
and a mandatory basis through several avenues, including press releases,
quarterly and annual reports, shareholder meetings, and earnings conference
calls. Prior literature is replete with studies that evaluate the extent to which
capital market participants react to quantitative information contained in
these disclosures. Only recently have researchers begun to explore the capi-
tal market implications of qualitative verbal communication via financial news
∗Mayew and Venkatachalam are with the Fuqua School of Business, Duke University. Acting
Editor: David Hirshleifer We acknowledge helpful comments and suggestions from two anony-
mous referees, Dan Ariely, Jim Bettman, Lauren Cohen, Patricia Dechow, Lisa Koonce, Feng Li,
Mary Frances Luce, Greg Miller, Chris Moorman, Chris Parsons, Eddie Riedl, Katherine Schipper,
Shyam Sunder, Paul Tetlock, T.J. Wong, and workshop participants at Barclays Global Investors,
University of California at Berkeley, Chinese University of Hong Kong, University of Connecticut,
Cornell University, Duke Finance Brown Bag, Financial Research Association 2008 conference,
Fuqua Summer Brown Bag, Journal of Accounting Auditing and Finance 2008 Conference, Mas-
sachusetts Institute of Technology, University of Miami, Penn State University, Queens University,
Rice University, University of Toronto, and Vanderbilt University. We also thank Amir Liberman
and Albert De Vries of Nemesysco for helpful discussions and for assistance in extracting the LVA
metrics into machine readable format for our academic use. Excellent research assistance was
provided by Daniel Ames, Erin Ames, Jacob Ames, Patrick Badolato, Zhenhua Chen, Ankit Gupta,
Sophia Li, Mark Uh, and Yifung Zhou.
1
Top Predicted Labels: cs.LG, stat.ML, cs.CV
```

The developed algorithm showcases the effectiveness of NLP techniques in categorizing academic papers. Its adaptability to finance underscores its potential utility in organizing and analyzing financial documents, offering a versatile tool for professionals in the field. Further fine-tuning and validation in the financial domain are recommended for optimal performance.