National College of Ireland

Clothify: Personalized Clothing Recommendations and Identification Technology Configuration Manual

MSc Research Project
MSCAI

Havva Eda Karadeniz
x22219447

School of Computing
National College of Ireland

Supervisor: SHERESH ZAHOOR

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Havva Eda Karadeniz

Student ID: X22219447

Programme: Artificial Intelligence **Year:** 2024

Module: MSc Research Project

Lecturer: Sheresh Zahoor

Submission Due Date: 12/08/2024

Project Title: Clothify: Personalized Clothing Recommendations and Identification Technology

Word Count: 1250 **Page Count:** 9

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Havva Eda Karadeniz

Date: 10/08/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Clothify: Personalized Clothing Recommendations and Identification Technology Configuration Manual

Havva Eda Karadeniz
x22219447

1 Introduction

Complete setup and operation instructions for the Kaggle and Visual Studio Code (VS Code) apparel suggestion project are provided in this configuration handbook. The setup process is walked through in detail in the manual, which also includes instructions on how to install software, configure your workstation, and run the code to get the desired outcome.

The handbook also includes advice on how to show the final suggested article of clothing and visualize results using different charts.

2 Project Files Detail

Python was used in the Kaggle interactive environment along with Visual Studio Code to construct the project. An overview of the main files utilized in this project is provided below:

Preprocessing and Exploration of the Data: Done in the Kaggle notebook' data_preprocessing .ipynb'
Model Training and Evaluation: Completed in the Kaggle notebook 'model_training.ipynb.'
Recommendation System: Applied in the Kaggle notebook 'clothes_recommendation.ipynb.'
Every notebook is accessible within the Kaggle environment and can be used straight from Kaggle or downloaded.

3 System Specification

The recommended system specifications to run this project efficiently are:

• **Operating System:** Windows 10 or higher / macOS / Linux
• **RAM:** 16 GB or more (8 GB is the minimum requirement
• **Processor:** Intel i5 or higher
• **Disk Space:** At least 20 GB of free disk space
• **Python Version:** Python 3.10.2 or higher

3.1 Software Used

The following software and tools were used in this project:

• Kaggle Notebooks: For data preprocessing, model training, and evaluation.
• Visual Studio Code (VS Code): For coding, debugging, and additional development work.
• Python 3.10.2: The programming language used to develop the project.

• Pandas, NumPy, Matplotlib, Seaborn, TensorFlow: Python libraries for data manipulation, visualization, and machine learning.
• Scikit-learn: A Python library used for machine learning models.

4 Download and Install

4.1 Python Installation

If you are running the project locally in VS Code, ensure Python is installed:

https://www.python.org/downloads/

After installing Python, install VS Code:

https://code.visualstudio.com/Download

4.2 Installing Required Extensions and Packages

VS Code Extensions:

- Install the Python extension for VS Code from the Extensions Marketplace.

Required Python Packages: Open the terminal in VS Code and run the following command to install the required packages: Figure 1.

```
pip install pandas numpy matplotlib seaborn scikit-learn tensorflow
```

Figure 1: Required Python Packages

4.3 Setting Up Kaggle Environment

To work with Kaggle:

1. **Sign Up/Sign In:** Create an account or log in at Kaggle.
2. **Create a New Notebook:** Navigate to the Notebooks section and create a new notebook for running the project.
3. **Upload Data:** You can either upload your dataset or use Kaggle's inbuilt datasets for your project.

5 Project Development

5.1 Importing Libraries

In both VS Code and Kaggle, start by importing all necessary libraries. The following code snippet imports the essential libraries used in this project: Figure 2.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report
import tensorflow as tf
```

Figure 2: Importing Libraries

5.2 Data Preprocessing

The data preprocessing step involves handling missing values, normalizing data, and preparing it for model training.

Handling Missing Values:

```
df.fillna(df.mean(), inplace=True)
```

Figure 3

Data Normalization:

```
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)
```

Figure 4

Splitting Data:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 5

5.3 Model Training

The model is trained using TensorFlow's Keras API. The following code snippet shows how to define and train a neural network model: Figure 6.

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

history = model.fit(X_train, y_train, epochs=50, validation_data=(X_test, y_test))
```

Figure 6: Define and train a neural network model

5.4 Model Evaluation

After training the model, evaluate its performance using the test data: Figure 7.

```
test_loss, test_accuracy = model.evaluate(X_test, y_test)
print(f"Test Accuracy: {test_accuracy:.4f}")
```

Figure 7

5.5 Running the Project on Kaggle

To run the project on Kaggle:

1. **Upload the Notebook:** Upload your Jupyter Notebook files to Kaggle.
2. **Run All Cells:** Use the "Run All" option to execute all the cells in sequence.
3. **Check Outputs:** Ensure the outputs match the expected results described in this manual.

6 Visualization and Outputs

6.1 Accuracy and Loss Graphs

To visualize the model's performance during training, plot the accuracy and loss curves: Figure 8 and Figure 9.

```
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.title('Model Accuracy')
plt.legend()
plt.show()
```

Figure 8: Accuracy Graph

```
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Model Loss')
plt.legend()
plt.show()
```

Figure 9: Loss Graph

6.2 Category Distribution

Visualize the distribution of different categories in your dataset: Figure 10.

```
sns.countplot(x='category', data=df)
plt.title('Category Distribution')
plt.show()
```

Figure 10:categories

6.3 Color Distribution

Similarly, visualize the distribution of colors: Figure 11.

```
sns.countplot(x='color', data=df)
plt.title('Color Distribution')
plt.show()
```

Figure 11

6.4 Gender Distribution

Visualize the distribution of gender in the dataset: Figure 12.

```
sns.countplot(x='gender', data=df)
plt.title('Gender Distribution')
plt.show()
```

Figure 12

6.5 Recommended Clothing Image

```python
from PIL import Image
import matplotlib.pyplot as plt

image_path = 'recommended_clothing.jpg'
img = Image.open(image_path)
plt.imshow(img)
plt.title('Recommended Clothing')
plt.axis('off')
plt.show()
```

Figure 13

7 Expected Outputs

After successfully running the code, you should obtain the following outputs:

1. **Accuracy and Loss Graphs:** Visual representations of how well the model performed during training.
   Accuracy Graph: Displays how the model's accuracy evolves over the epochs.
   Loss Graph: Shows the changes in the loss function during training and validation.
2. **Category, Color, Gender, and Sleeve Length Distributions:** Distribution plots of different attributes in your dataset (these examples were provided in the earlier sections).
3. **Recommended Clothing Image:** The final output of the recommendation system, displaying the recommended clothing item.

When you run the complete code, you will be able to access all outputs generated by the project. Above, examples are provided for a few of these outputs, such as the accuracy and loss graphs. To familiarize yourself with the overall results, it is recommended to review the outputs generated by the full code execution.

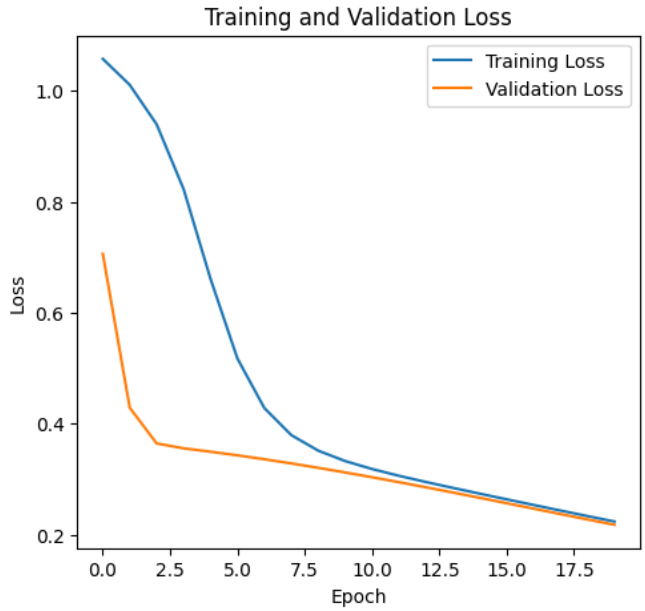

Figure 14: Loss Graph

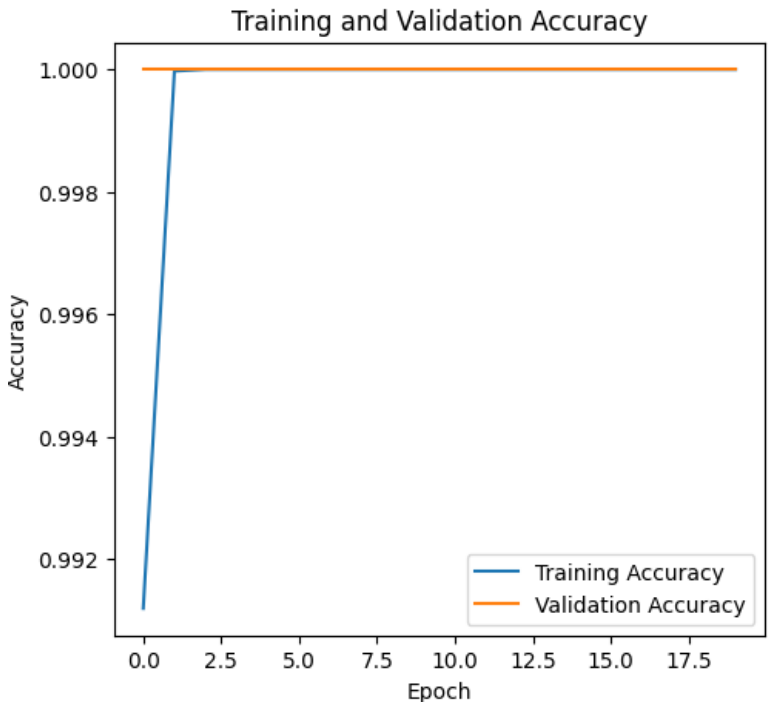

Figure 15: Accuracy Graph

8 Conclusion

This configuration manual provides all the necessary steps to set up and run the clothing recommendation project using Kaggle and VS Code. By following these instructions, you should be able to replicate the results and understand the performance of the model. The visualizations and outputs included in the manual help to present the findings clearly and effectively.

References

Kaggle. (n.d.). *Kaggle Notebooks*. Retrieved from https://www.kaggle.com/notebooks
Visual Studio Code. (n.d.). *Visual Studio Code Downloads*. Retrieved from https://code.visualstudio.com/Download
Python Software Foundation. (n.d.). *Python Downloads*. Retrieved from https://www.python.org/downloads/
TensorFlow. (n.d.). *TensorFlow Documentation*. Retrieved from https://www.tensorflow.org/
Pandas Documentation. (n.d.). *Pandas Library*. Retrieved from https://pandas.pydata.org/
NumPy Documentation. (n.d.). *NumPy Library*. Retrieved from https://numpy.org/
Matplotlib Documentation. (n.d.). *Matplotlib Library*. Retrieved from https://matplotlib.org/
Seaborn Documentation. (n.d.). *Seaborn Library*. Retrieved from https://seaborn.pydata.org/
Scikit-learn Documentation. (n.d.). *Scikit-learn Library*. Retrieved from https://scikit-learn.org/