Practical Work Report

Topics in Data Science

<AI model to recognize customized handwritten>

Student: Muhammet Ali YILDIZ , 1709981,

https://github.com/mtech00/dst\_17099 81\_2024.git

,mtech00, [yldzmuhammedali@gmail.com](mailto:yldzmuhammedali@gmail.com)  
Muhammet-Ali-YILDIZ-1709981

## 

Course:Datascience and Artificial intelligence

[**1. Job Description**](#_heading=h.4d34og8) [**3**](#_heading=h.4d34og8)

[**2. Implementation of Work**](#_heading=h.2s8eyo1) [**3**](#_heading=h.2s8eyo1)

[**3. Operation of work**](#_heading=h.17dp8vu) [**3**](#_heading=h.17dp8vu)

[**4. Conclusion**](#_heading=h.3rdcrjn) [**3**](#_heading=h.3rdcrjn)

[**Bibliography**](#_heading=h.lnxbz9) [**3**](#_heading=h.lnxbz9)

## 

## 

## Job description

1. **Image Processing**: The project begins with loading an image file using the Python Imaging Library (PIL). The image is then resized to a specified dimension using the **resize()** function. Additionally, the image is split into smaller segments using a custom function (**split\_image**) to facilitate further processing.

1. **Data Conversion**: After splitting the image, each segment is converted into numerical format representing RGB values. This conversion is crucial for feeding image data into machine learning models. The numerical data is saved as CSV files for easy access and manipulation.
2. **Data Preparation**: The converted CSV files are then prepared for model training and evaluation. The data is split into training and testing datasets using a predefined ratio. This ensures that the model is trained on a portion of the data and tested on unseen samples to assess its generalization ability.
3. **Model Training**: A Decision Tree Regressor model is chosen for its simplicity and interpretability. The model is trained using the training dataset, where features represent RGB values of image segments, and the target variable is derived from the image data.
4. **Model Evaluation**: The trained model's performance is evaluated using the testing dataset. Metrics such as accuracy are calculated to assess how well the model predicts the target variable. This step is crucial for understanding the model's effectiveness and potential areas for improvement.

The practical work aims to develop an AI model capable of recognizing customized handwritten text. The project involves the implementation of various data science tasks, including image processing, data conversion, model training, and evaluation. The architectural drawing depicts the workflow of the project, detailing the sequence of tasks and scripts used.

## Work Implementation

The implementation of the work involves leveraging Python libraries for image processing and machine learning tasks. The project begins with loading and resizing the input image using the Python Imaging Library (PIL). The resized image is then split into smaller segments to facilitate further processing. Each segment is converted into numerical format representing RGB values and saved as CSV files for model training. The data is then prepared, and a Decision Tree Regressor model is trained using the RGB values of image segments as features. Model evaluation metrics such as accuracy are calculated to assess the model's performance.

1.Image Processing: The image processing stage begins with resizing the original image to a standard dimension to ensure consistency in input data. Subsequently, the resized image is split into smaller segments using a custom function that divides the image into a grid of specified rows and columns.

2. Data Conversion: Each segmented image is converted into a numerical format representing RGB values. This conversion transforms image data into a structured format suitable for machine learning algorithms. The converted data is then saved as CSV files for further analysis and model training.

3. Data Preparation: The converted CSV files are organized into training and testing datasets. The training dataset is used to train the model, while the testing dataset is reserved for evaluating the model's performance. This step ensures unbiased assessment and generalization of the model.

4. Model Training: The Decision Tree Regressor model is trained using the training dataset. The model learns patterns from the input features (RGB values) and predicts the target

variable. Training involves optimizing model parameters to minimize prediction errors

and improve overall performance.

5. Model Evaluation: The trained model is evaluated using the testing dataset to assess its

predictive accuracy and generalization ability. Evaluation metrics such as mean squared

Architectural Design Diagram

A diagram of a computer program

Description automatically generated with medium confidence

## Work operation

The work operates by executing the implemented scripts on sample handwritten images. The image processing scripts resize and split the input image, while the conversion script converts the segmented images into numerical format. The dataset building script prepares the data for model training, followed by the training script, which trains the Decision Tree Regressor model. Finally, the evaluation script assesses the model's performance using testing data.

Original image :A grid of numbers with black ink

Description automatically generated

**1:**

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

* This code section uses the Python Imaging Library (PIL) to resize an image.
* It opens an image file named "realdata\_handwriting.png" from the './datasets/' directory.
* The original size of the image is printed.
* The image is resized to 512x512 pixels using the resize() function.
* The resized image is saved as "realdata\_handwriting\_resized.png" in the './output/' directory.
* This code section splits the original image into smaller segments.
* It utilizes a custom function split\_image imported from the 'split\_image' module.
* The function parameters specify the input image path, number of rows and columns for segmentation, and output directory for saving the segmented images.

**2:** A screen shot of a computer program

Description automatically generated

* This code section converts the segmented images into numerical format (RGB values).
* It iterates through the directory containing segmented images.
* Each segmented image file is processed using a Python script 3.2conversion\_picture\_to\_numbers.py for conversion.
* The converted data is saved in CSV format for further analysis and model training.

**3:**

A screen shot of a computer program

Description automatically generated

* This code section reads the segmented images from the './output/' directory.
* It converts each image to a NumPy array and then to a 2D list of lists, where each inner list represents a row of RGB values.
* The RGB values are converted to strings and saved in CSV format, with each row representing a pixel in the image.

**4:**

A screen shot of a computer program

Description automatically generated

* This code section prepares the training and testing datasets from the converted image data.
* It reads the CSV files containing image data from the './output\_text/' directory.
* Rows are selected randomly for inclusion in the training dataset while ensuring a predefined ratio of training to testing data.
* The selected rows are used to generate the training and testing datasets, which are then saved as CSV files in the './datasets/' directory.

**5 :**

A screenshot of a computer program

Description automatically generated

* This code section defines functions to prepare the training and testing datasets for model training.
* The size\_x() function determines the size of the dataset.
* The num\_rows(x) function calculates the number of rows for training and testing based on a predefined ratio.
* The select\_rows(x) function randomly selects rows for inclusion in the training dataset while ensuring the correct ratio of training to testing data.
* The generate\_train() function generates the training and testing datasets.
* Finally, the datasets are saved as CSV files for further use.

**6:**

A screenshot of a computer program

Description automatically generated

1. **Importing Libraries**: The code imports necessary libraries such as matplotlib, pickle, numpy, pandas, LinearRegression, mean\_squared\_error, r2\_score, and DecisionTreeRegressor.
2. **Reading Training Data**: It reads the training data from the CSV file named "dataset\_finalrealdata\_handwriting\_train.csv" and stores it in the variable train\_data.
3. **Extracting Features and Target**: It extracts the input features (data\_X) and the target variable (data\_Y) from the training data.
4. **Decision Tree Regression**: The code initializes a DecisionTreeRegressor model (clf) and fits it to the training data using the fit method.
5. **Saving the Model**: It saves the trained model (clf\_svm\_model) using the pickle.dump() function to a file named "handwriting\_predictor\_model".
6. **Coefficient of Correlation**: Finally, it calculates the coefficient of correlation (rr) for the trained model using the score() method and prints it out.

**7:**

A screenshot of a computer program

Description automatically generated

* This code snippet evaluates the trained model using the evaluation dataset.
* It loads the evaluation data from a CSV file and separates the features (X) and the target variable (Y).
* The trained model is loaded from the saved file using pickle.
* Predictions are made using the loaded model on the evaluation dataset.
* Differences between predictions and actual values are calculated.
* The code then counts correct and wrong predictions and calculates the accuracy of the model.
* Finally, it prints the total number of correct and wrong predictions, along with the accuracy metrics.

## Conclusion

A graph of the same person

Description automatically generated with medium confidence

## In conclusion, the project successfully implements an AI model for recognizing customized handwritten text. However, there are areas for improvement, including model optimization, feature engineering, and data augmentation. Future work involves integrating the system into real-world applications, enhancing multilingual support, and adapting to different handwriting styles.

• Model Optimization: Experiment with different machine learning algorithms and model architectures to improve predictive performance.

• Feature Engineering: Explore additional features or transformations that may enhance the model's ability to capture relevant patterns in the data.

• Hyperparameter Tuning: Fine-tune model hyperparameters to optimize model performance and generalization.

• Data Augmentation: Consider techniques such as data augmentation to increase the diversity and size of the training dataset, potentially improving model robustness.

## Bibliography

**[1] Paulo Vieira Professor at IPG (thanks for the everything about the important informations )**

[2] Python Software Foundation. Python Language Reference, version 3.x. Available online at: https://docs.python.org/3/

[3] Pillow (PIL Fork) Documentation. Python Imaging Library (Fork). Available online at: https://pillow.readthedocs.io/en/stable/

[4] NumPy Documentation. NumPy User Guide. Available online at: https://numpy.org/doc/stable/

[5] Pandas Documentation. Pandas User Guide. Available online at: https://pandas.pydata.org/docs/

[6] scikit-learn Documentation. scikit-learn: Machine Learning in Python. Available

online at: https://scikit-learn.org/stable/documentation.html

[7] DecisionTreeRegressor Documentation. scikit-learn DecisionTreeRegressor API Reference. Available online at: https://scikit- learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

[8] Anaconda Documentation. Anaconda Distribution User Guide. Available online at: https://docs.anaconda.com/

[9] Jupyter Notebook Documentation. Project Jupyter. Available online at:

https://jupyter-notebook.readthedocs.io/en/stable/

[10] GitHub. Repository for split\_image. Available online at:

https://github.com/split\_image

[11] GitHub. Repository for conversion\_picture\_to\_numbers. Available online at:

https://github.com/mtech00/conversion\_picture\_to\_numbers

[12] Microsoft. Visual Studio Code Documentation. Visual Studio Code User Guide. Available online at: https://code.visualstudio.com/docs