温州肯恩大学

WENZHOU-KEAN UNIVERSITY

PART-1: Pen Grip Pose Detection based on MideaPipe

PART-2: AI-Aided Crop Recommendation Software Requirement Specification

PART-3: Pen Grip Pose Detection based on MideaPipe

In Partial Fulfillment of the Requirements

for the Bachelor of Science in Computer Science

by

Keming Xing

1162866

June, 2024

Listing

1. Abstract

My graduation project primarily encompasses two major sections: Pen Grip Detection and Agricultural Data Analysis. In the first section, I introduce the conception, discussion, and requirements documentation for the Pen Grip Detection project. The second section focuses on the Agricultural Data Analysis project, including the review of requirement documents, data set acquisition, and optimization. In this project, we employed one-hot encoding to transform qualitative data into quantitative data and successfully built a high-accuracy random forest multi-classification model for data analysis. In the third section, I revisited the Pen Grip Detection project, analyzed the previous group's code, and made optimizations. Through data preprocessing and the construction of a binary classification model, we not only improved the code structure but also assisted team members in integrating the code modules. Ultimately, by demonstrating the code execution results, we enhanced the project's presentation quality. Our work not only improved the project's accuracy and practicality but also fostered collaborative growth among team members.

2. Keywords

Data analysis, random forest model, one-hot vector encoding, data classification, Python, software requirements document, MediaPipe, OpenCV, computer vision, Camera reading

3. Contents

First of all, since this graduation project is divided into three parts, the following parts will also discuss the contents of the three parts separately in chronological order.

For the first part, the main tasks of our group are to investigate project usability, write proposals, and write project requirements documents. Among them, the proposal mainly includes project goals and effect estimates, background and literature surveys, case studies, and sketch introductions.

In proposal, we mainly give the following content:

The main goal of the project is to develop a system for detecting and analyzing handwriting gestures using MediaPipe, an open source framework for building machine learning solutions. The scope of the project includes:

- Implementing hand detection and tracking using MediaPipe's Hand Tracking solution to locate and track the position and movement of the hand.
- Developing a model for recognizing different handwriting postures, such as pen grip variations and hand orientations
- Integrating real-time feedback to provide users with guidance on improving their handwriting posture
- Creating a user-friendly interface for visualizing and analyzing handwriting posture data.

Handwriting posture plays a crucial role in the legibility, speed, and comfort of writing. Improper posture can lead to discomfort, fatigue, and even long-term health issues such as repetitive strain injuries(Blackburn, 2001). Various studies have been conducted in the field of handwriting analysis and ergonomics to understand the impact of posture on writing performance and to develop techniques for improving posture.

After evaluating the prediction results and feasibility assessment, we looked for

similar cases, such as MediaPipe's related gesture detection model and the application of other visual algorithms, and completed the Case Study:

- Nagaraju (2022) used the MediaPipe framework to develop a digital handwriting recognition system that can detect hand gestures and analyze writing gestures. This provides the technical basis for this study.
- Zhang et al. (2020) developed a real-time hand tracking system using the MediaPipe hand tracking module. The system can locate and track hand position and movement. This provides a methodological basis for this study.
- Tao et al. (2018) developed an American Sign Language alphabet recognition system using convolutional neural networks and multiview data augmentation. Their research demonstrated hand gesture tracking and recognition capabilities using MediaPipe and machine learning algorithms. This provides an algorithmic basis for this study.

After predicting and analyzing all possible outcomes, we wrote a simple deliverable presentation for subsequent judgment to other groups. The content is as follows:

- A fully functional handwriting posture detection system implemented using MediaPipe.
- Documentation and user guides explaining the system's functionality, usage instructions, and integration guidelines.
- A research report summarizing the methodology, results, and findings of the project, including insights from the literature review and case study.

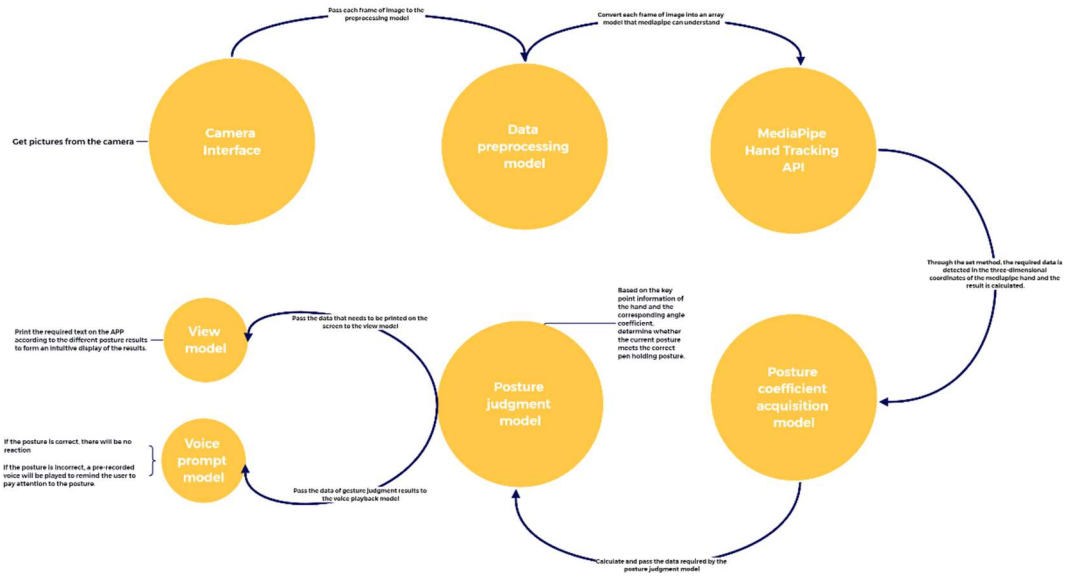
- Source code and model weights for the developed system, released under an open-source license to encourage further research and development in the field.
- The take-home deliverables will enable stakeholders, including researchers, educators, and developers, to understand, utilize, and build upon the outcomes of the project.

The above describes the Proposal section of the first part of my project. In this phase, I actively participated in the conception and discussion of ideas. I utilized my knowledge of MediaPipe and data analysis to explain the project's principles and feasibility to my teammates, thereby helping them gain a deeper understanding of the project.

The remaining section of the first part covers the preparation of the Software Requirements Document (SRS). Although the document was divided into sections and assigned to different team members, I was responsible for some of the crucial parts. Specifically, I handled the Product Perspective, including the System Interfaces, User Interfaces, Hardware Interfaces, Software Interfaces, Communications Interfaces, Memory Constraints, Operations, and Site Adaptation Requirements. In this role, I ensured that each of these sections was meticulously detailed, accurately reflecting the project's requirements and specifications. These sections can be primarily categorized into three main types: the minimum support parameters required by the software, the key interfaces of the software and their interconnections, and the network security issues involved.

First, the minimum support parameters for the software include basic requirements regarding computer hardware. For example, one requirement states: "In 95% of cases, the response time during regular periods should not exceed 1.5 seconds, and during peak periods, it should not exceed 4 seconds."

In the main interface section of the software, the content becomes more crucial as it encompasses all the functionalities of the final product and the relationships between each module. These interfaces include the MediaPipe Hand Tracking API, Camera Interface API, Voice module interface, Data preprocessing model, Posture coefficient acquisition model, Posture judgment model, and View model. I also created a diagram illustrating their relationships, which can be seen in the diagram below. These interfaces are interconnected and mutually influential, forming the optimal components of the software. For instance, in the Posture judgment model, I wrote: "Based on the key point information of the hand and the corresponding angle coefficient, determine whether the current posture meets the correct pen holding posture."



Picture1: Schematic diagram of the relationship between various interfaces

In addressing network security issues, my primary focus was on safeguarding user data and ensuring the security of user logins. In the context of software usage, protecting user privacy is paramount. Through analysis in these areas, the software can be made more robust. For instance, in the Encrypted Data Transmission section, I mentioned: "Network-transmitted data should undergo encryption to ensure data confidentiality, preventing eavesdropping, theft, or tampering during data collection, transmission, and processing. Business data should be encrypted during storage to ensure its inviolability."

In the preparation of the Software Requirements Document, I applied the knowledge I acquired in software engineering. Both functional requirements (interface documents) and non-functional requirements (hardware requirements and network security requirements) were carefully considered and documented based on practical considerations. Additionally, I was responsible for the integration of the final requirements document. By crafting the table of contents, introduction, and summary sections, I enhanced the readability of the requirements document.

In the first part of the project, we encountered several challenges. Since this stage only involves planning and conceptualizing the target software program, there may be ideas that are difficult to implement. To mitigate this situation, we needed to lower certain standards and conduct early investigations into the feasibility of these ideas. If some ideas prove unattainable, we needed to determine to what extent we could lower the standards to align with our vision for the software. Overall, as part of software design, the work in this stage is relatively straightforward, but it requires ensuring that

the ideas are reasonable and objective, conducting practical investigations, and analyzing and evaluating the expected outcomes.

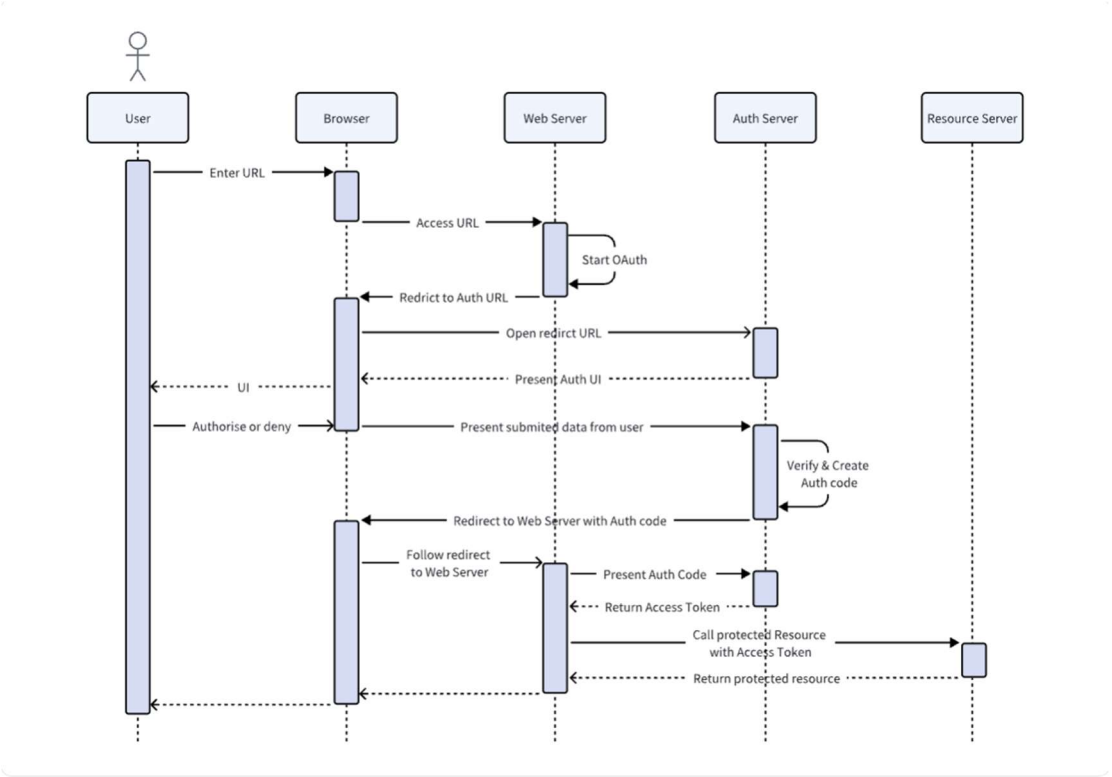
The above represents all the work content of the first part, with the summary to be included in the Results section. Next, we will move on to the work content of the second part.

In the second part, we were responsible for transitioning from an idea to a basic program. During this transition, we not only needed to thoroughly understand the requirement document provided by the previous group but also had to write relevant code based on the requirements to improve feasibility. In this part, we conducted evaluations of the results from the previous group, wrote the SDD (Software Design Description), analyzed user stories, and wrote some code.

Firstly, in understanding the requirement document from the previous group, we thoroughly examined each interface provided by the previous group. We learned that this was primarily a software for recommending crops for cultivation. Users can input parameters of their land, such as pH value and soil humidity, to receive recommendations from the system regarding suitable crops. However, we encountered an issue: the requirement document did not include a dataset. As a data classification project, having a dataset is essential. Therefore, we began searching for available datasets on platforms like Kaggle. Eventually, we found a dataset containing various soil data and recommended crop types.

After finding the dataset, we began the process of writing the SDD (Software Design Description). During this process, we created use-case diagrams, sequence

diagrams, and class diagrams. Below is an example of one of the diagrams. Through these diagrams, we understood the relationships between various interfaces, provided some usage scenarios, and presented preliminary UI design sketches for the user interface.



Picture2: Sequence Diagram of our Program

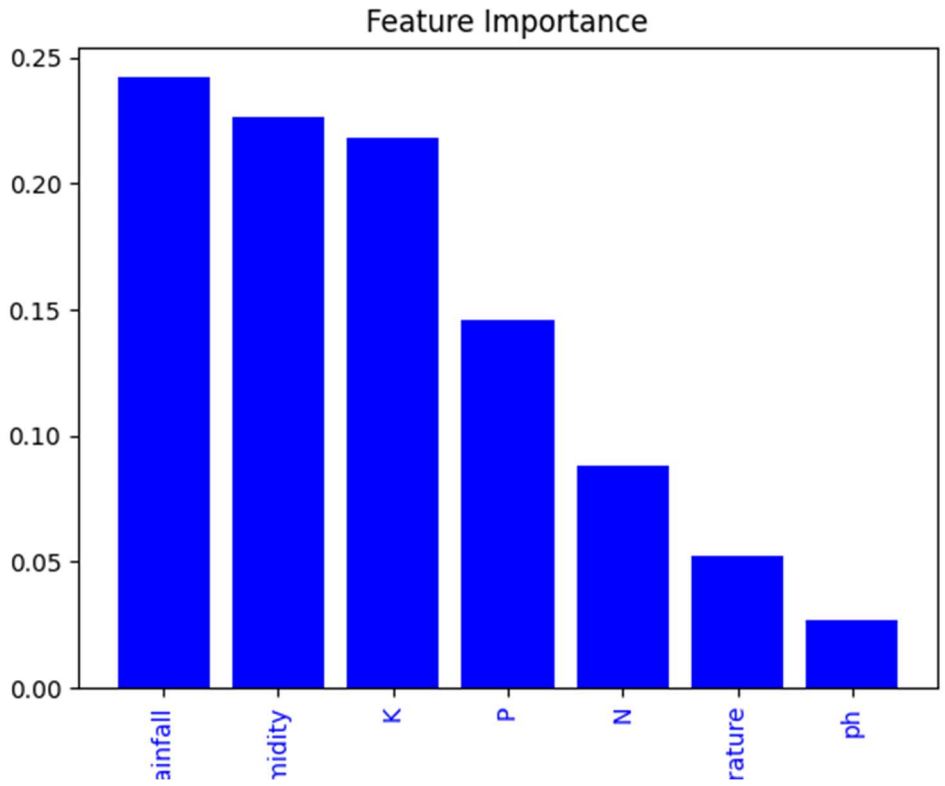
After completing these preparatory tasks, we began the coding phase, which was divided into three main parts: the interactive UI design part, the button interaction processing part, the server setup part, the data preprocessing part, the model construction part, the model evaluation part, and the organization and integration part. My responsibilities included the data preprocessing part, the model construction part, and the model evaluation part. In fact, prior to this, I had never worked with data analysis. I had only studied some knowledge about convolutional neural networks and was familiar with definitions related to accuracy, recall rate, and F1 score in data

statistics.

Firstly, in the data preprocessing part, I learned and utilized one-hot encoding to support dataset preprocessing. One-hot encoding enables each state to have its own independent register bit. Based on one-hot encoding, qualitative data in the dataset can be converted into quantitative data. For example, for data indicating whether it belongs to region A or region B, it can be transformed into A-True: 1, B-True: 0, or A-True: 0, B-True: 1, converting A or B into computer-understandable 1 and 0, thereby improving the accuracy of the dataset. In addition to these preprocessing techniques, I also used SHAP to generate some analysis charts for dataset features. For example, the following chart illustrates an analysis of feature importance.

N	P	K	temperatu	humidity	ph	rainfall	label_appl	label_bana	label_black	label_chick	label_coco	label_coffe	label_cottc	label_grap	label_jute	label_kidne	label_lentil
90	42	43	20.87974	82.00274	6.502985	202.9355	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
85	58	41	21.77046	80.31964	7.038096	226.6555	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
60	55	44	23.00446	82.32076	7.840207	263.9642	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
74	35	40	26.4911	80.15836	6.980401	242.864	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
78	42	42	20.13017	81.60487	7.628473	262.7173	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
69	37	42	23.05805	83.37012	7.073454	251.055	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
69	55	38	22.70884	82.63941	5.700806	271.3249	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
94	53	40	20.27774	82.89409	5.718627	241.9742	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
89	54	38	24.51588	83.53522	6.685346	230.4462	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
68	58	38	23.22397	83.03323	6.336254	221.2092	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
91	53	40	26.52724	81.41754	5.386168	264.6149	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
90	46	42	23.97898	81.45062	7.502834	250.0832	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
78	58	44	26.8008	80.88685	5.108682	284.4365	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
93	56	36	24.01498	82.05687	6.984354	185.2773	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
94	50	37	25.66585	80.66385	6.94802	209.587	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
60	48	39	24.28209	80.30026	7.042299	231.0863	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
85	38	41	21.58712	82.78837	6.249051	276.6552	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
91	35	39	23.79392	80.41818	6.97086	206.2612	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
77	38	36	21.86525	80.1923	5.953933	224.555	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
88	35	40	23.57944	83.5876	5.853932	291.2987	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
89	45	36	21.32504	80.47476	6.442475	185.4975	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
76	40	43	25.15746	83.11713	5.070176	231.3843	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
67	59	41	21.94767	80.97384	6.012633	213.3561	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
83	41	43	21.05254	82.6784	6.254028	233.1076	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
98	47	37	23.48381	81.33265	7.375483	224.0581	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
66	53	41	25.07564	80.52389	7.778915	257.0039	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
97	59	43	26.35927	84.04404	6.2865	271.3586	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
97	50	41	24.52923	80.54499	7.07096	260.2634	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
60	49	44	20.77576	84.49774	6.244841	240.0811	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
84	51	35	22.30157	80.64416	6.043305	197.9791	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
73	57	41	21.44654	84.94376	5.824709	272.2017	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
92	35	40	22.17932	80.33127	6.357389	200.0883	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Picture3: Processed data chart example

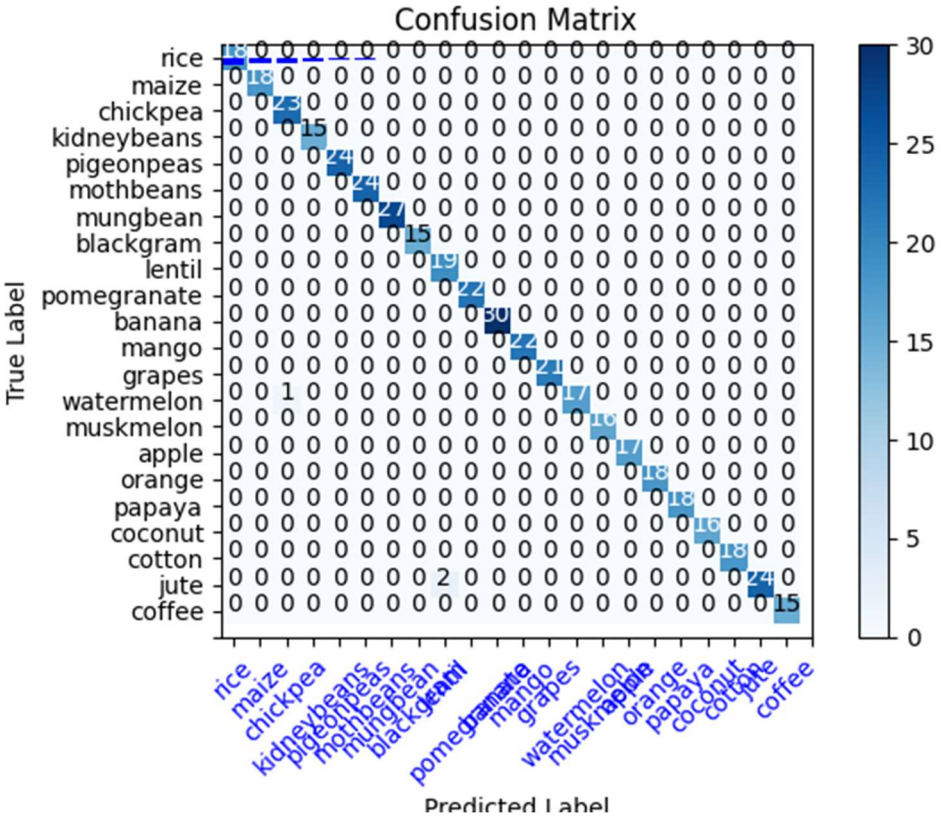


Picture4: Feature importance histogram of the dataset

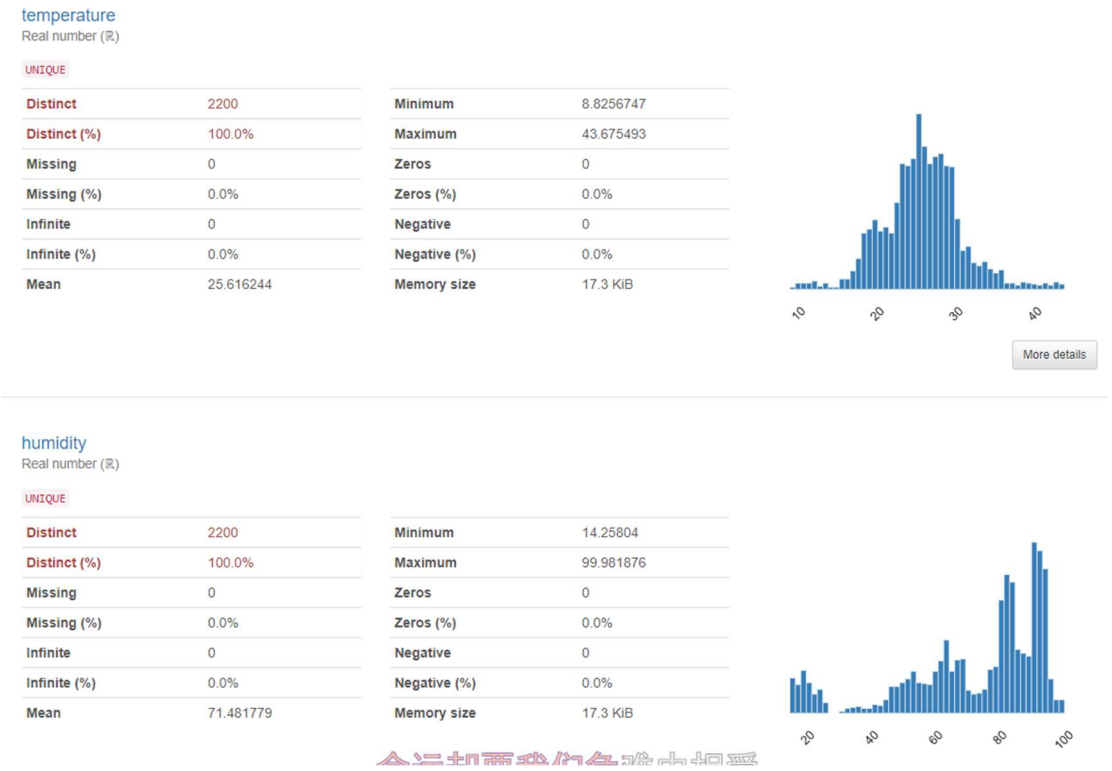
In establishing the data classification model, I utilized the Random Forest model. I studied relevant knowledge about the Random Forest model online. It is a classifier consisting of multiple decision trees, and its output category is determined by the mode of the categories output by individual trees. With the Random Forest model, we can implement binary classification models or multi-classification models with multiple features. I used the scikit-learn API as the construction API for the Random Forest model. I divided the dataset into 80% for training and the remaining 20% for testing. Finally, through training on the training set, we obtained a Random Forest model that can be used to determine the optimal crop.

During the model analysis stage, I not only used the test set for model testing to obtain accuracy, recall rate, and F1 score but also created a confusion matrix heatmap

to visually present the accuracy of the model. Additionally, I analyzed the impact of various data on the final model evaluation and organized the results into an HTML file, which includes the heatmap and detailed analysis. The specific content is shown in the figure below:



Picture5: Confusion matrix heat map of the model



Picture6: Example diagram of the impact of each data on the final evaluation of the model

Overall, my contributions in the second part are as described above. During the task, I planned the following workflow for myself: In the first week, I needed to find similar datasets, participate in project discussions, and learn about dataset analysis. In the second week, my tasks included learning about dataset analysis, finding a model that fits the current dataset, and writing the code to analyze the dataset. In the third week, I needed to continue learning about dataset analysis, write the code to analyze the dataset, and learn methods of data visualization. In the final week, my tasks involved testing the trained model using the test set, performing data visualization of the test results, and importing the model into the application.

This part may have been the most challenging for me because all the knowledge involved here was completely new to me. It was my first time delving into data analysis,

and I successfully applied it to the project's data. Additionally, I needed to understand the underlying principles of data preprocessing and analysis to avoid errors. In the final part of model evaluation, I also spent a lot of time learning about data visualization. I presented the impact of different data on the results in the most visible form through prominent charts.

At the same time, I am still encouraging other members of the same group to complete their corresponding work. At work, I often supervise and supervise their work progress and whether the code they write can be integrated, because for a project, only part of the code is useless, and all the codes must be integrated., only when the whole can be put together and used can it be considered complete. During the subsequent supervision, I noticed that the servers responsible for other students had been successfully built and running normally, and the UI part was also going smoothly. I designed a good-looking UI and question system for the interface, which improved the quality of the entire project.

In the third part, I returned to the project I worked on in the first part, which is about gesture detection. In this section, we needed to complete the entire project code, add additional features, conduct A/B testing for the software, and prepare for project presentation and demonstration.

Firstly, upon regaining access to this project, it's essential to conduct an assessment of its current progress and functionalities. We need to evaluate and score each feature based on its completion status and quality. While the project has already implemented most of the functionalities by the time I regained access, a closer inspection reveals

numerous existing issues that need to be addressed.

The first problem is there's a lack of a unified interactive interface, which appears quite rudimentary and can only be accessed from the code interface. Secondly, frequent stuttering occurs during the reading of video frames, with the FPS dropping below 3 during analysis. After a thorough analysis of the code, it's evident that this is due to the previous team's failure to utilize the continuous video frame encoding method. Within MediaPipe, there are two modes of frame retrieval: static frame retrieval and continuous frame retrieval. In the continuous frame analysis mode, the model predicts the next frame based on the previous frame, reducing the CPU usage of the program. Additionally, the model is restarted in every loop iteration, significantly reducing the analysis speed. To address this issue, we need to not only rewrite the code section related to OpenCV frame retrieval but also employ multithreading techniques to optimize the model's runtime speed.

The third issue is that the model's predictions are not based on data analysis but rather on simplistic size comparisons. This approach results in very low accuracy because any data falling outside the specified range for correct pen-holding posture is deemed incorrect. Consequently, most of the predicted gestures are erroneous. To rectify this situation, we need to employ data analysis to build a classification model. Subsequently, the data obtained from the classification model should be integrated into the existing program, ensuring smooth operation. The fourth issue pertains to the complexity of the code. Due to the code's disorganized format and multiple versions, it requires meticulous scrutiny to confirm the purpose of each line of code and the sources

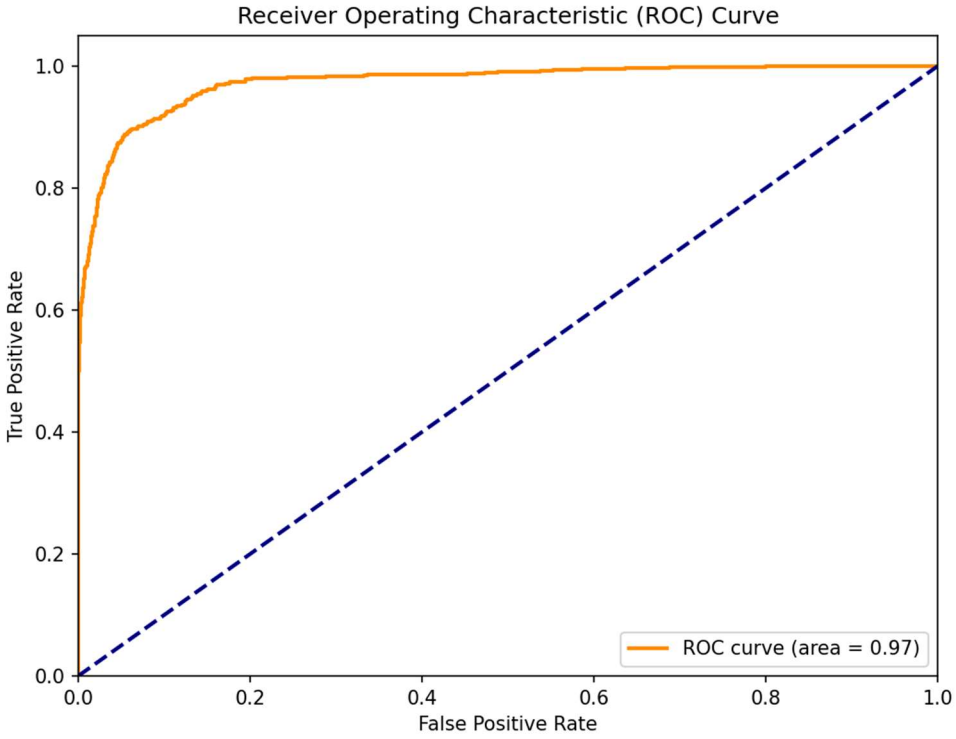
of their libraries.

I delegated the first, second and fourth issues to other team members and began tackling the third problem myself. I considered it the most challenging because it required not only preprocessing the dataset but also building a binary classification model. Furthermore, it entailed creating interfaces without disrupting the existing code structure to ensure smooth importation of feature data required for classification and exporting of result data post-classification. Despite the daunting nature of this task, its similarities to the work I completed in the second part prompted me to once again utilize one-hot encoding and the random forest model for this data analysis endeavor.

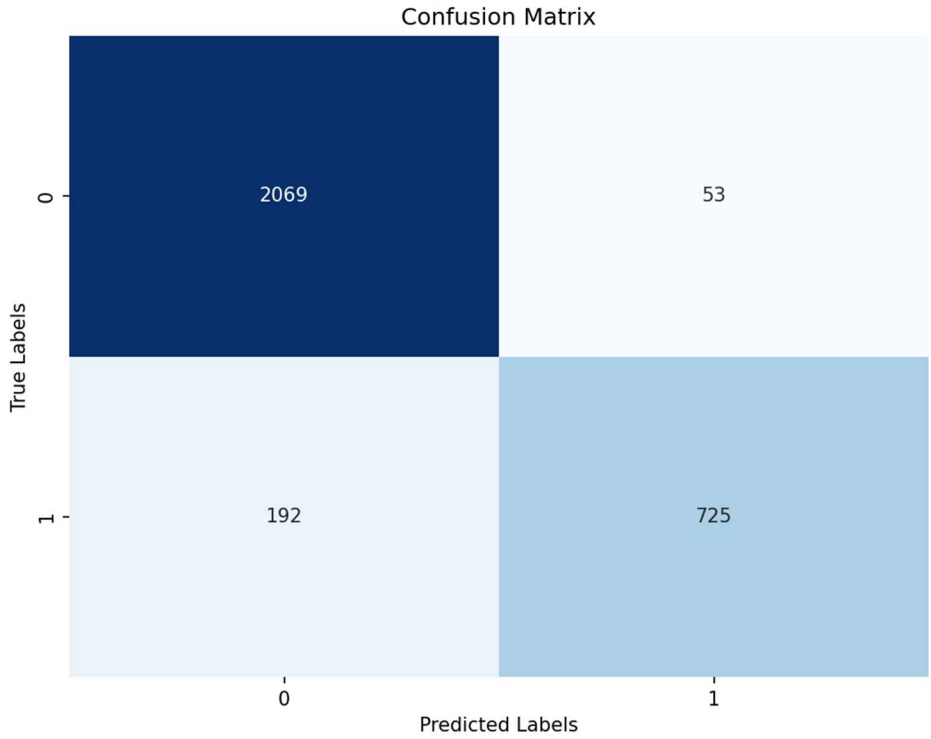
The previous group captured and converted the data into a usable format. To preprocess this dataset, I first needed to understand it thoroughly. After multiple comparisons and inquiries with the previous team members, I finally grasped the meaning of each column in the dataset. Based on this understanding, I separated the feature columns from the result columns and applied one-hot encoding to them. However, since I only needed the model this time, I did not analyze each feature individually. With the preprocessed dataset, I could then begin training the classification model.

With the experience from the previous task, I again utilized the random forest model for data analysis and classification. I learned from my previous mistakes and adjusted the thresholds for the trees in the random forest. I recorded the model's accuracy, recall, and F1 score, and generated a confusion matrix and ROC curve to evaluate the model's accuracy. After testing the model with the test set, we achieved an

accuracy of 93%, a recall of 79%, and an F1 score of 86%. Overall, the results from this classification model were consistent with expected patterns. By comparing the confusion matrix and the ROC curve, I found that the ROC curve was very close to the ideal standard. Therefore, I concluded that this classification model was correctly built. Below are the confusion matrix and ROC curve for this model.



Picture7: ROC curve of two-classification model of pen holding posture



Picture8: Confusion matrix of two-classification model of pen holding posture

However, after I completed my part, the teammate responsible for the fourth part encountered issues. He was unable to integrate the code because he did not understand how to import these parameters or use the main function to unify the data. Admittedly, I wasn't very skilled in this area either, so I decided to learn about it. After some time studying, I understood the usage of the main function and initialization methods. I modified parts of the code, incorporated the multithreading and continuous video frame processing code from the first part, and integrated the model into the analysis of the camera feed data. I also ensured the code was well-visualized, enabling the execution of all necessary preparations by simply running the main function.

Lastly, for the overall project testing phase, we evaluated each component of the program against the requirements outlined in the documentation. For example, we

verified that the frame rate for real-time video display met the requirement of 30 frames per second. Once we confirmed that each part met the necessary standards, we began preparing for the final thesis presentation.

During the poster creation phase, I provided the main images and key technical descriptions for my teammates. Prior to the presentation, I listed the important sections of the code and explained the corresponding concepts to my team members. This included the significance and use of the ROC curve, the principles of the random forest model, and how to call its API. This ensured that each team member could discuss a specific part of the program, thereby ensuring a smooth project presentation.

In this phase, I consolidated the new knowledge I gained in data preprocessing and analysis, and I learned more about constructing binary classification models. I successfully completed the tasks of data preprocessing and analysis and assisted a teammate with code integration. Compared to the previous group's program, our classification model's accuracy significantly improved to 93%, and the frame rate for analysis increased to meet real-time standards. I also acquired a deeper understanding of how to connect interfaces and applied this knowledge in practice.

Although the project's completion level is very high, we identified a few issues. For example, Professor Pinata noted that the font size of the results display was small and hard to read for many people. This feedback was valuable, and we immediately adjusted the font size and added different colors for different results—green for correct results and red for incorrect ones. Additionally, we discussed how to implement this software in real-world applications. One challenge is the shakiness of the hand skeleton

detected by MediaPipe, which can cause false readings. After internal discussions, we decided that if all prediction results over a short period are incorrect, the hand posture is likely incorrect. Conversely, if any frame in that period shows a correct prediction, the posture is probably correct, despite some detection shakiness.

Furthermore, we analyzed potential issues and corresponding solutions. Since the dataset consists of photos of male hands aged 20-23, there may be analysis errors due to gender and age differences. To mitigate this, we should convert distances to ratios or expand the dataset to include samples from various ages and genders. In summary, while the project is highly complete, there are still areas for improvement.

4. Literature Review

PART-1: Handwriting posture plays a crucial role in the legibility, speed, and comfort of writing. Improper posture can lead to discomfort, fatigue, and even long-term health issues such as repetitive strain injuries (Blackburn, 2001). Various studies have been conducted in the field of handwriting analysis and ergonomics to understand the impact of posture on writing performance and to develop techniques for improving posture.

Part-2:

In data analysis and machine learning projects, preprocessing the dataset is crucial. Common preprocessing methods include handling missing values, data standardization, feature selection, and feature engineering. Among these, one-hot encoding is a commonly used technique, especially for handling categorical variables. By encoding categorical variables into binary form, models can better understand and process these

variables. When preparing data for model training, one-hot encoding can effectively improve the performance and accuracy of the model.

In research on machine learning models for crop recommendation systems, common algorithms include random forest, support vector machine (SVM), and neural networks. These algorithms demonstrate wide applicability and effectiveness in agricultural decision-making. Random forest is an ensemble learning method that can handle large amounts of features and samples, showing excellent performance in dealing with nonlinear relationships and high-dimensional data. SVM performs well in classification and regression tasks, especially for nonlinear classification problems. Neural networks are powerful models capable of learning complex patterns and relationships, although they may require more computational resources and time for training and tuning.

Data visualization plays a crucial role in data analysis, aiding in a more intuitive understanding and interpretation of data. Common data visualization techniques include confusion matrix analysis, heatmap visualization, and feature importance plots. Confusion matrices provide a clear representation of the model's performance across different categories, aiding in the assessment of model performance. Heatmap analysis visually presents the relationships and patterns among data, facilitating the discovery of hidden trends and patterns. Feature importance plots help determine which features have the greatest impact on the model's prediction results, guiding feature selection and model optimization.

Part-3:

Achieving real-time processing is a primary challenge in gesture detection. The literature emphasizes the importance of optimizing video frame analysis to maintain high frame rates, essential for smooth user experiences. Techniques such as multithreading and efficient frame encoding are crucial. Studies have shown that leveraging continuous frame analysis can significantly enhance performance by predicting subsequent frames based on previous ones, thereby reducing the computational burden (Zhang et al., 2021).

Another critical challenge is ensuring high accuracy and robustness of the detection models. Inadequate data preprocessing and simplistic feature extraction methods can lead to poor performance. Advanced data preprocessing techniques, including one-hot encoding and comprehensive feature selection, are vital for improving model accuracy (Han et al., 2020). Additionally, employing sophisticated classification models like Random Forests can enhance prediction accuracy and interpretability (Breiman, 2001).

Gesture detection systems have significant applications in educational tools and interactive user interfaces. These systems can assist in teaching and learning by providing real-time feedback on hand postures and movements. The implementation discussed in the project highlights the integration of a gesture detection model with an educational tool for evaluating pen-holding postures. The use of Random Forests for classification and the optimization of frame retrieval processes were key to achieving high accuracy and real-time performance.

5. Results& Conclusions

Part-1: Pen Grip Pose Detection based on MideaPipe

During the initial phase of the project, my responsibilities mainly focused on proposing and explaining various ideas, writing the interface part of the requirement document, and processing file formats. In this stage, I actively applied the knowledge I had acquired and drafted a comprehensive project proposal. Through in-depth discussions with team members, I proposed multiple solutions. I also continuously refined the requirement document using the skills I had acquired from previous courses. Throughout this process, I continued to learn new knowledge, including delving into the detailed processes of case analysis.

PART-2: AI-Aided Crop Recommendation Software Requirement Specification

In the second phase of our project, we transitioned from conceptualization to implementation, meticulously analyzing the previous group's requirements and crafting code to realize these objectives. Despite encountering challenges such as the absence of a dataset initially, we persevered, sourcing pertinent data and delineating our approach through the Software Design Description (SDD). Subsequent to preparatory tasks, we delved into coding, with my responsibilities spanning data preprocessing, model construction, and evaluation, realms entirely new to me. Through diligent learning and application, I contributed substantively to the project's progression, culminating in the establishment of a robust model for crop recommendation.

Navigating through this phase was undoubtedly challenging, given the novelty of the terrain. However, by diligently acquainting myself with data analysis principles and visualization techniques, I successfully navigated uncharted waters, contributing

substantively to the project's advancement. Furthermore, my role extended beyond individual contributions; I assumed a supervisory role, ensuring cohesion amongst team members and overseeing code integration. Through diligent oversight, I ensured the smooth operation of servers and the seamless progression of UI development, enhancing the project's overall quality.

PART-3: Pen Grip Pose Detection based on MideaPipe

In the third phase of the project, I returned to the gesture detection project from the first phase, completing the entire code implementation, adding extra features, conducting A/B testing, and preparing for the project presentation and demonstration. Initially, we assessed the project's current progress and functionality, rating the quality and completion status of each feature. While most functionalities had been implemented, there were several issues that needed to be addressed, such as the lack of a unified user interface, frequent lag during video frame reading, and low accuracy in model predictions. By redesigning the code, employing multithreading techniques, and improving prediction methods, we significantly enhanced the project's performance and accuracy.

In addressing these issues, I was responsible for data preprocessing and building the classification model. Using one-hot encoding and the random forest model, I successfully constructed a classification model with an accuracy of 93%. Additionally, I assisted my teammates in integrating the code, ensuring the implementation of multithreading and continuous video frame processing, which improved the real-time analysis capabilities. During the project testing phase, we verified the performance of

each component and prepared for the project presentation. Although the overall project completion was high, we still identified and resolved some issues, such as adjusting the font size and color of the result display. We also discussed the practical application of gesture detection and potential improvement measures.

Overall, this phase not only reinforced my new knowledge in data preprocessing and analysis but also enhanced my understanding and application of binary classification model construction and interface integration.

6. References

[1] Blackburn-Brockman, E. (2001). Prewriting, planning, and professional communication. The English Journal, 91(2), 51. https://doi.org/10.2307/822345 .

[2] Nagaraju, Mr. M. (2022). Digital handwriting recognition using hand tracking by using media pipe and OPENCV libraries. International Journal for Research in Applied Science and Engineering Technology, 10(7), 659–666. https://doi.org/10.22214/ijraset.2022.44647

[3] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020, June 18). MediaPipe hands: On-device real-time hand tracking. arXiv.org. https://arxiv.org/abs/2006.10214

[4] Tao, W., Leu, M. C., & Yin, Z. (2018). American sign language alphabet recognition using convolutional neural networks with Multiview Augmentation and inference fusion. Engineering Applications of Artificial Intelligence, 76, 202–213. https://doi.org/10.1016/j.engappai.2018.09.006.

[5] DHAMODHARAN R. (2022). Machine learning in agriculture.

https://www.kaggle.com/code/dhamur/machine-learning-in-agriculture.

[6] TARIQ MAHMOOD. (2024). Crop_Recommedation_Agri_Project. https://www.kaggle.com/code/tariqbashir/crop-recommedation-agri-project/notebook.

[7] Sphinx. (2024). SHAP API. https://github.com/shap/shap

[8] scikit-learn. (2024). RandomForestClassifier API https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[9] Kuo, Yang & Zhang, Zhen. (2019). Real-time Pattern Recognition for Hand Gesture Based on ANN and Surface EMG. 799-802. 10.1109/ITAIC.2019.8785894.

[10] Jiawei, H., & Micheline, K. (2006). Data mining: concepts and techniques. Morgan kaufmann.

[11] Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001). https://doi.org/10.1023/A:1010933404324

7. Appendix

Program link:

- Part-1: https://github.com/kermit0125/CPS4951_Part1
- Part-2: https://github.com/CapstoneGroup09-1/DataSet_Analysis
- Part-3: https://github.com/kermit0125/Pen-Grip-Pose-Detection-based-on-MideaPipe

Introduction of API:

- Pandas: Pandas is one of the primary tools in Python for data manipulation and analysis. It provides efficient data structures and functions, especially suitable for dealing with structured data. The core data structure in Pandas is

DataFrame, which is similar to an Excel spreadsheet but more flexible and powerful. With Pandas, users can perform operations such as data cleaning, transformation, analysis, and visualization. This project is mainly used for the data cleaning part of data preprocessing.

- Scikit-learn:

  Scikit-learn is a Python library for machine learning, providing a wide range of machine learning algorithms and tools, including classification, regression, clustering, and dimensionality reduction. It is designed to be simple and easy to use, suitable for both beginners and experts in machine learning. Scikit-learn comes with many classical machine learning algorithms built-in and provides rich documentation and examples for users to learn and utilize. This project is mainly used for the scatter plot generation part of data analysis.

- OpenCV:

  OpenCV is an open-source computer vision library that provides a rich set of image processing and computer vision algorithms for tasks such as image processing, object detection, feature extraction, and image recognition. It supports multiple programming languages, including C++, Python, and Java, and is cross-platform, running on Windows, Linux, macOS, and other operating systems. OpenCV offers a wide range of image processing and computer vision algorithms, along with support for hardware acceleration and parallel computing. This project is mainly used to read the camera.

- MediaPipe:

MediaPipe is an open-source framework for building machine learning-based media processing pipelines, aimed at simplifying the development and deployment of media processing tasks. It provides a collection of pre-trained machine learning models and components for processing video, audio, and image data. MediaPipe supports real-time processing and offers an easy-to-use Python API for users to build custom media processing applications. This project is mainly used for picture processing and skeleton prediction.

Data collection mechanism

1. First step - Determine data needs:

   Determine the types, quantities, and formats of data needed for the project. This will help you identify which datasets to download from Kaggle and what kind of data to capture yourself.

2. Second step – Photograph or find data:

   In the second part, the data set is a relevant data set retrieved on kaggle.

   In the third part, the data set is shot by the project team from multiple angles and handed over to MediaPipe to read and analyze the data table.

3. Third Step - Create the data directory structure:

   Create a clear data directory structure, including folders for storing Kaggle data and your own captured data. Add documentation or descriptions to the directory structure describing the contents and purpose of each dataset.

Code written by myself

Part-2:

Code of main code

```
import numpy as np
import pandas as pd
import joblib

def predict_crop(N, P, K, temperature, humidity, ph, rainfall):
    # 加载模型
    model = joblib.load('random_forest_model.pkl')
    # 加载标签编码器
    encoder = joblib.load('label_encoder.pkl')

    # 构建输入数据
    data = np.array([[N, P, K, temperature, humidity, ph, rainfall]])
    # 进行预测
    prediction = model.predict(data)
    # 获取预测结果的标签
    predicted_label = encoder.inverse_transform(prediction)[0]

    return predicted_label

def main():
    print("请输入以下参数以预测作物类型:")
    N = float(input("土壤中的氮含量: "))
    P = float(input("土壤中的磷含量: "))
    K = float(input("土壤中的钾含量: "))
    temperature = float(input("气温（摄氏度）: "))
    humidity = float(input("湿度（百分比）: "))
    ph = float(input("土壤的 pH 值: "))
    rainfall = float(input("降雨量（毫米）: "))

    predicted_label = predict_crop(N, P, K, temperature, humidity, ph,
rainfall)
    print(f"预测的作物类型为: {predicted_label}")

if __name__ == "__main__":
    main()
```

Code of Data Processing

```
import warnings
warnings.filterwarnings("ignore")
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn import preprocessing
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
import itertools
import joblib

df = pd.read_csv('Crop_recommendation.csv')

# 分离标签与特征值
x = df.drop(['label'], axis=1) # 特征值
Y = df['label'] # 标签
labels = df['label'].tolist()
encode = preprocessing.LabelEncoder()
y = encode.fit_transform(Y) # 标签编码
print(y)

# 分离测试集以及数据集
x_train, x_test, y_train, y_test = model_selection.train_test_split(x,
y, test_size=0.2, random_state=10)

# 构建随机森林模型
model = RandomForestClassifier(max_depth=5, n_estimators=100,
random_state=5)
model.fit(x_train, y_train)

# 特征重要性分析
model.feature_importances_
feature_names = x_test.columns
feature_importances = model.feature_importances_
indices = np.argsort(feature_importances)[::-1]
plt.figure()
plt.title("Feature Importance")
plt.bar(range(len(feature_importances)), feature_importances[indices],
color='b')
plt.xticks(range(len(feature_importances)),
np.array(feature_names)[indices], color='b', rotation=90)
plt.savefig('my_plot.png')

# 预测测试集
```

```
y_pred = model.predict(x_test) # 定性数据
y_pred_quant = model.predict_proba(x_test) # 定量数据

# 混淆矩阵
confusion_matrix_model = confusion_matrix(y_test, y_pred)

# 混淆矩阵热力图绘制函数
def cnf_matrix_plotter(cm, classes):
    plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues) # 更换颜
色映射为蓝色
    plt.title('Confusion Matrix')
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    # 标准化数字格式，四舍五入到整数
    fmt = 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]),
range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True Label')
    plt.xlabel('Predicted Label')
    plt.show()
    plt.savefig('Confusion_matrix_heat_map.png')

# 调用方法绘制混淆矩阵热力图
cnf_matrix_plotter(confusion_matrix_model,
['rice','maize','chickpea','kidneybeans','pigeonpeas','mothbeans','mung
bean','blackgram','lentil',
'pomegranate','banana','mango','grapes','watermelon','muskmelon','apple
','orange','papaya','coconut','cotton','jute','coffee',''])


# 将模型保存为.pkl 文件
joblib.dump(model, 'random_forest_model.pkl')

# 创建并拟合标签编码器
encoder = preprocessing.LabelEncoder()
encoder.fit(labels)  # labels 是你的标签数据
```

```
# 将标签编码器保存为.pkl 文件
joblib.dump(encoder, 'label_encoder.pkl')
```

Part-3:

Code of Data Processing

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, precision_score,
recall_score, f1_score, roc_curve, auc
import seaborn as sns
import joblib


df=pd.read_csv('ProcessedData.csv')
X = df.drop('Gesture', axis=1)   # 特征列
y = df['Gesture']               # 标签列

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=10) # 划分测试集和训练集，测试集 20%，训练集 80%

model = RandomForestClassifier(max_depth=5, n_estimators=100,
random_state=5)  # 100 棵决策树，最大深度 5，随机数种子
model.fit(X_train, y_train)

# 计算预测结果和预测概率
y_predict = model.predict(X_test)
y_predict_proba = model.predict_proba(X_test)

# 计算混淆矩阵
cm = confusion_matrix(y_test, y_predict)
print("Confusion Matrix:")
print(cm)

# 计算 Precision、Recall、F1-Score
precision = precision_score(y_test, y_predict)
recall = recall_score(y_test, y_predict)
f1 = f1_score(y_test, y_predict)
print("\nPrecision:", precision)
print("Recall:", recall)
print("F1-Score:", f1)
```

```
# 绘制混淆矩阵的热图
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
# 保存混淆矩阵图像
plt.savefig('Confusion_Matrix.png')

# 计算 ROC 曲线
fpr, tpr, thresholds = roc_curve(y_test, y_predict_proba[:, 1])
roc_auc = auc(fpr, tpr)

# 绘制 ROC 曲线
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area
= %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
# 保存混淆矩阵图像
plt.savefig('ROC.png')


# 将模型保存到文件
joblib.dump(model, 'random_forest_model.pkl')
```

Code of Data Processing

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
 #pip install scikit-learn 导入 sklearn 包
from sklearn.model_selection import train_test_split        #导入划分训练
测试集功能的 APA
from sklearn.ensemble import RandomForestClassifier         #导入随机森林
模型的 APA
from sklearn.tree import export_graphviz                    #导入森林模型
Tree 可视化工具
from IPython.display import Image
```

```
import graphviz

df = pd.read_csv('ProcessedData.csv')

X = df.drop('Gesture',axis=1)   #特征列
y = df['Gesture']               #标签列

X_train,X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=10) #划分测试集和训练集，测试集 80%，训练集 20%

model = RandomForestClassifier(max_depth=5, n_estimators=100,
random_state=5)           #100 棵决策树，最大深度 5，随机数种子
model.fit(X_train, y_train)

estimator = model.estimators_[20]
print(estimator)

feature_names =X_train.columns
y_train_str =y_train.astype('str')
y_train_str[y_train_str=='0'] = 'Wrong'
y_train_str[y_train_str=='1'] = 'Correct'
y_train_str=y_train_str.values

import os
os.environ["PATH"] += os.pathsep + 'C:/Program Files/Graphviz/bin'

dot_file = 'tree.dot'
export_graphviz(estimator, out_file=dot_file,
                feature_names=feature_names,
                class_names=y_train_str,
                rounded=True, proportion=True,
                label='root',
                precision=2, filled=True)

graph = graphviz.Source.from_file(dot_file)
graph.render(filename='tree', format='png', view=True)
```

8. Acknowledgements

As I approach the completion of my studies and embark on a new journey, I would like to express my deepest gratitude to all who have supported and encouraged me.

First and foremost, I want to thank Dr. Pinata Winoto and Dr. Tiffany Tang for their diligent guidance and patient responses throughout the entire research process. Your expertise and profound insights have provided invaluable guidance for my thesis, and I have greatly benefited from them.

I also want to express gratitude to all the professors and scholars in the Computer Science Department of the College of Engineering. Through your teaching and research, you have imparted rich knowledge and inspiration to me, enabling me to continually progress and grow into a better scholar.

This school has provided me with numerous opportunities. Beyond examinations, the school has been dedicated to helping us develop in multiple directions. Each course comes with corresponding projects, which not only help us grasp knowledge but also allow us to truly apply it to real-life situations, which has taught me a lot.

Furthermore, I need to thank the student ambassadors. It is through them that I found a sense of belonging in this school and have grown in various aspects, whether in character, leadership, values, or through the diverse friendships I have made.

Lastly, I want to express my gratitude to my family and friends. Thank you for continuously supporting me on the path to pursuing my dreams, and for providing me with endless encouragement and understanding. Without your support, I would not have been able to successfully complete this journey.

To all those who have helped and supported me throughout my academic career, your contributions will always be cherished. Wishing everyone smooth sailing and a bright future!

With sincerest gratitude,

Keming Xing