Higher Education Team: Final Submission Documentation

I. About and Authors

This report was written by Kailey Cozart Quesada and is for the higher education team for the Summer, 2024 semester of CS 8903 in the Human Augmented Analytics lab. The team members whose work is represented here are the following: Ayush Parikh, Terry Junsoo Park, Kailey Cozart Quesada. The advisors for this project were Breanna Shi and Nicholas Lytle.

The goal of the higher education project is to create and maintain a structure for large research groups in higher education. Code base solutions, contribution tracking, resource management, researcher support, and program development are included in this project. Additionally, members of this team often participate in additional projects individually, as directed. This semester, the sub-projects within the higher education team were following: code management, resource management, and program development.

II. Methods: Codebase Management

When it comes to designing a system for organizing the code base of a large online research program, there are a few things to consider. First of all, the priorities of the organization must be considered. For Georgia Institute of Technology's Human-Augmented Analytics Group, priorities include the following:
(1) Code needs to be open source and easily accessible. (No "lost" code.)
(2) Code contributions need to be tracked for publication attributions.
(3) Code needs to be marked as either: in-progress, tested and documented, or abandoned.
(4) The instructor has to clearly be able to see the code you wrote each week.
(5) Codebase management procedures should be documented for future use.

Automated or low-human-cost solutions are preferred. In the following sections, the chosen solutions, as well as less favorable candidates will be discussed. Future codebase managers should read section **(5)** for the list of the procedures required each semester.

A. *Code needs to be open source and easily accessible.*

For this priority, two solutions were considered, a single repository solution and a multi-repository solution.

Single Repository: The original single repository solution was suggested by Breanna Shi. In this scenario, a single database would house all scripts written by all teams. This approach was discussed with the student researchers, but merge conflicts, repository size limits, and usability concerns for researchers were the primary reasons that this solution was not selected. Specifically, GitHub has a soft limit of 5 GB for the size of a repository. Depending on the number of projects that are pursued over the next few years, this could be a problem when considering scalability. Additionally, if everything was on one repository, a researcher could unintentionally destroy other team's projects with a careless commit.

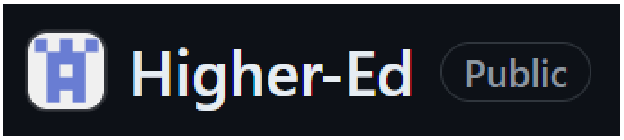
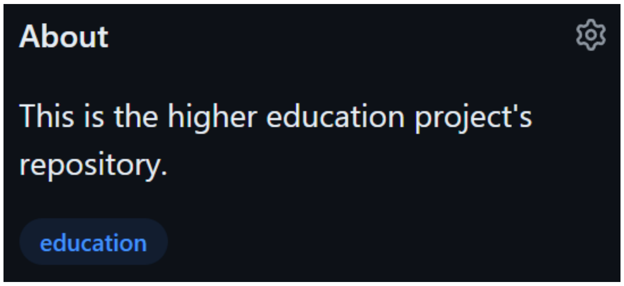


Fig. 1: Ensure each repository is made public.



This is the higher education project's repository.

Fig. 2: Add topics by selecting the settings button.

Multiple Repositories: Because of these things, it was determined that a public repository should exist inside of Human-Augmented-Analytics for each project. For each project repository, topics can be added, and then those topics can be searched for on the organization home page. Note that some projects, such as the NREL competition project, should be set to private during development and then made public after the competition is complete. Additionally, some larger projects with multiple publications might have more than one repository.

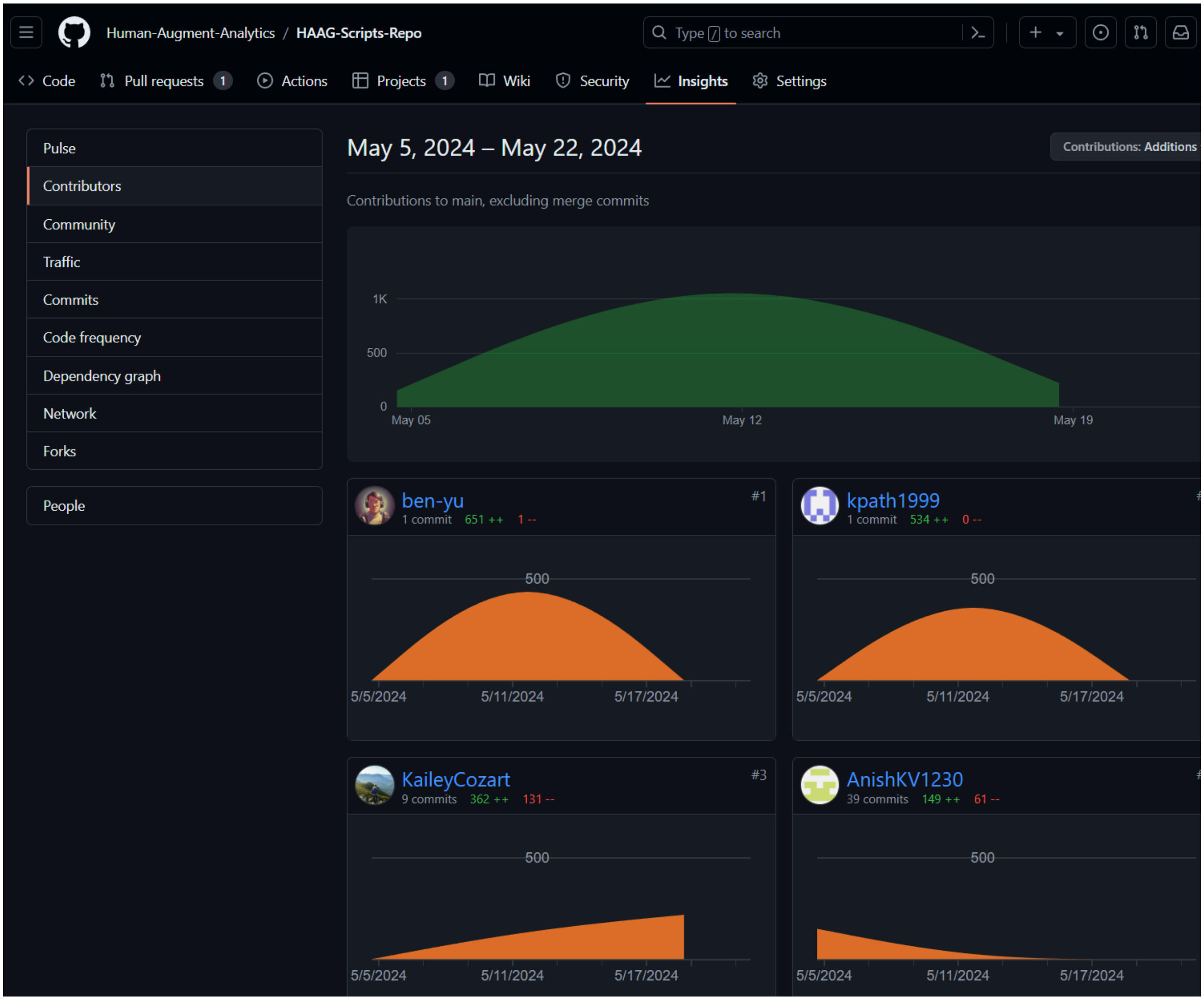
Solution: Public repository in Human-Augmented-Analytics for each project, with topics added to each project for searchability.

B. Code contributions need to be tracked for publication attributions.

For this, two solutions were considered. The first was a code solution to read git logs and calculate the contributions of each contributor. The second solution was to use GitHub's insights tab, which is attached to each repository.

Write Code to Read Git Logs: While the code written could be improved to sort users by the number of changes and could be set up to run with every merge to main, it seems unnecessary, since most repositories are not too large, and the majority of this information is available in the GitHub insights tab. Because it didn't seem worthwhile to pursue this solution, it was abandoned for GitHub insights.

Use GitHub Insights: The GitHub insights tab very easily allows researchers to assess who has significantly contributed to a project. One can very quickly see which contributors should be cited and you can filter by commits, additions, or deletions. See the screenshot in Figure 3.



May 5, 2024 – May 22, 2024

Contributions to main, excluding merge commits

ben-yu
1 commit 651 ++ 1 --

kpath1999
1 commit 534 ++ 0 --

KaileyCozart
9 commits 362 ++ 131 --

AnishKV1230
39 commits 149 ++ 61 --

Fig. 3: Screenshot of GitHub insights.

Solution: Use GitHub insights. Authors who have contributed substantially to a project should be cited.

C. Code needs to be marked as either: in-progress, tested and documented, or abandoned.

The first proposed solution for this item was to use a workflow for tagging within the code files or folders themselves. The second proposed solution was to use GitHub projects and a code tracking tag.

Tagging within Code Files: At first, a workflow for tagging individual files or folders was proposed. With this workflow, there would be a readme in each project folder or at the top of each file indicating whether that code is in-progress, ready for testing, tested and documented, or abandoned. However, when talking to student researchers, it was determined that tagging each file would probably be too tedious, especially for certain projects.

Tagging with a GitHub Projects Template and Code Tracking Tag: Instead of tracking code within a file or folder, each project can be broken down into sections or scripts, and we can track the progress of the code through GitHub projects. The "Ready (Junkyard 1)" box is for items that are ready to be started. The "In-progress (Junkyard 2)" is for items that are actively being worked on. The "Ready for Testing and Documenting (Junkyard 3)" is for items that are completed and awaiting testing and documenting. The "In Testing and Documenting (Thrift Shop)" is for indicating that someone is currently testing and documenting the given code. The "Tested and Documented" section is for items that are ready to use. The "Abandoned (Graveyard)" section is for code that was determined to be unnecessary.

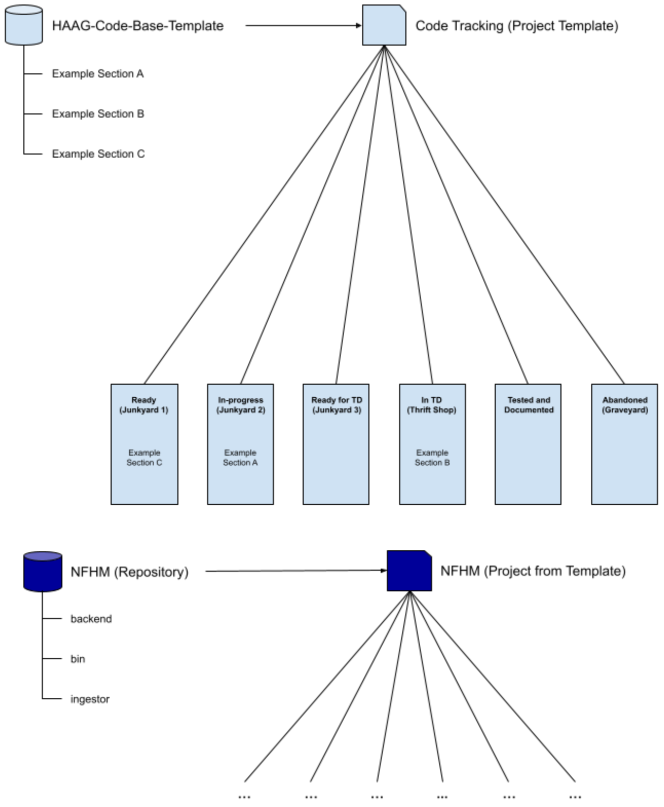
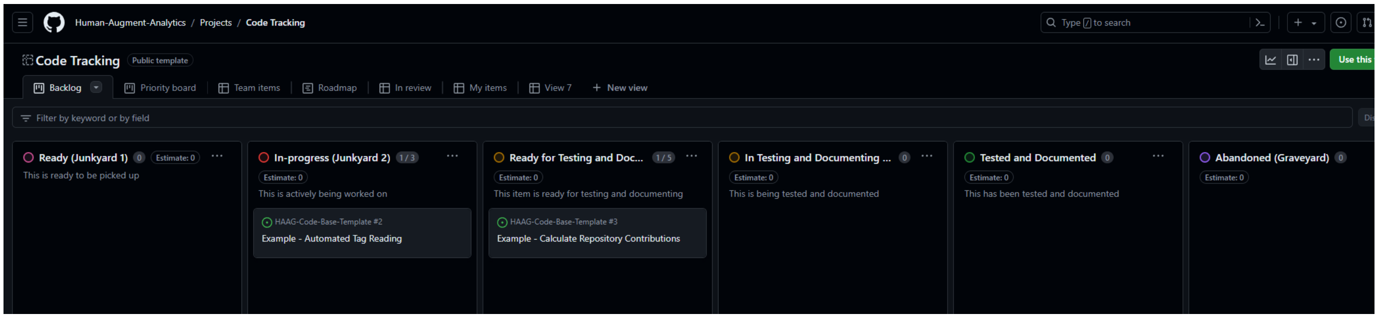


Fig. 4: Each repository has its own GitHub project from a HAAG template to track code.



Ready (Junkyard 1) 0 Estimate: 0
This is ready to be picked up

In-progress (Junkyard 2) 1 / 3
Estimate: 0
This is actively being worked on
HAAG-Code-Base-Template #2
Example - Automated Tag Reading

Ready for Testing and Doc... 1 / 5
Estimate: 0
This item is ready for testing and documenting
HAAG-Code-Base-Template #3
Example - Calculate Repository Contributions

In Testing and Documenting ... 0
Estimate: 0
This is being tested and documented

Tested and Documented 0
Estimate: 0
This has been tested and documented

Abandoned (Graveyard) 0
Estimate: 0

Fig. 5: Screenshot of the GitHub project template for HAAG.

Solution: Use a GitHub projects template and code tracking tag to track sections of project code.

D. The instructor has to clearly be able to see the code you wrote each week.

For this, several solutions were proposed. Students could potentially link to code folders on GitHub, add code blocks directly into their report, etc. Based on feedback from Breanna Shi, it seems like linking to the exact code you wrote in your project's repository, as well as pasting key code blocks into your report with explanations is best.

Solution: In their report, students can give both links to the files that they worked on and screenshots of the code blocks from that week.

E. Codebase Management Procedures

This section will go over the procedures that the future codebase managers should either use or change in the future. As discussed in **(1)**, the codebase manager needs to create or add public repositories that will house all data for each project if it does not already exist. Additionally, the codebase manager or members of each team need to add relevant topics to their repositories. These topics might include things like computer-vision, biology, or machine-learning. The codebase manager needs to make sure that projects that were set to private during development (such as competition projects) are made public at the appropriate time. Larger projects with multiple repositories must also be managed by the codebase manager as needed.

For **(2)**, no regular actions are required by the codebase manager. When a publication needs to cite researchers for their contributions, simply go to the "Insights" tab of a repository, as shown in Figure 3.

As discussed in **(3)**, the codebase manager will need to inform researchers that their code must be tracked for future reference. At the end of the semester each team's GitHub project should be checked to ensure that the team has tracked their code. Additionally, ensure that all code for each team has been uploaded to the appropriate repository. Once again, the goal is to not lose any needed code. There is some important existing documentation for this step. First, there is both a YouTube video and pdf for explaining how teams can track their code using a "code tracking" issues tag and GitHub projects. The links are the following: YouTube (https://youtu.be/Bl0AcWL1m1k), pdf (https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Code%20Management/GitHub%20Projects%20Documentation_%20Creating%20a%20Project%20Item.pdf). Secondly, there is documentation for creating a project from the HAAG GitHub project template: https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Code%20Management/GitHub%20Projects%20Documentation_%20Creating%20a%20Project%20from%20HAAG%20Template.pdf. This will need to be done by the codebase manager whenever a new repository is created.

For **(4)**, the codebase manager simply needs to ensure that students follow the instructor's procedure for making weekly code contributions clear. This is to insure that the instructor knows that the students are contributing consistently every week.

Codebase management was done by Kailey Cozart Quesada. Please check the contact information section if you have additional questions.

III. Methods: Resource Management

The following items are the objectives for the resource management portion of the higher education team:
(1) The group members must have the minimum computational and technical resources to be able to complete their projects.
(2) Volunteers must be recruited and provided to the requesting teams in a timely manner such that projects can be completed on time.
(3) A resource management system must be developed and documented.

A. Resource Management Procedures

The resource manager has several responsibilities. First of all, they must address computational and knowledge gap requests from all of the researchers on the team, as well as the program director. Resource responsibilities are listed below.

PACE: The group members in most of the teams will require GPU access to run their deep learning models. To respond to computational resources, the resource officer must take on a "TA" role for PACE ICE (Georgia Tech's HPC clusters), familiarize themselves with the process of making requests to PACE support for group members, and communicate with PACE support to provide GPU resources. Preferably, they should be familiar with running Slurm jobs on PACE, as well as running the virtual remote desktop on PACE ICE.

Dropbox: Additionally, the resource officer must be familiar with the Dropbox setup and communicate with Georgia Tech's OIT support to provide group members with McGrath lab's Dropbox access.

McGrath GPU: The resource manager must also become familiar with McGrath lab's GPU and SSH to provide SSH access to group members.

SRG Computers: The resource manager should be able to assist with providing access to SRG computers for group members if needed.

SharePoint Site: The resource officer should also maintain the team's SharePoint site, which hosts all of the team's high-level documentation. The resource officer should aim to develop the internal SharePoint site and add functionalities for data collection and automating recruitment functions using Microsoft Power Automate.

Volunteer Recruitment: The resource officer should also recruit volunteers for the various volunteer tasks that the director requests based on the group's requirements. For example, volunteers might include annotators or technical writers. The volunteers will be recruited from a pool of the initial candidates, but the recruitment method is up to the resource officer. It is recommended that they use the existing Microsoft Automate Flow in the team's SharePoint to automate sending emails based on the volunteer pool excel document.

Volunteer Status Tracking: The resource officer should also maintain documents in the SharePoint to track the status of volunteers.

Extra Credit Tracking: The resource officer is also responsible for tracking extra credit earned by the group's members in the extra credit excel document in the team's SharePoint.

Resource Management System: The resource officer should also collect data, develop surveys, develop the resource management system to the best of their ability, and evaluate its performance for the research portion of the course.

The resource officer is responsible for responding to any other technical requests made by the team members, and independently finding the best solution. They should communicate with the school's various support resources, but keep the director in the loop. For knowledge gaps, the resource officer should familiarize to their best ability the various pipelines in the group, especially the computer vision project. They should be able to set up the environment for each component of the pipeline and aim to be able to run the entire pipeline on their own over the semester. They will assist with installation issues from the group members.

B. Unsuccessful Methods and Resources

PACE proved only to be useful for learning and for documentation, but it was not helpful to the Cichlid Computer Vision (CV) team. Furthermore, the McGrath GPU was not accessible via remote SSH, and IT did not respond to the issue. Therefore, this was also unsuccessful. The SRG computers were a backup plan and not used by the higher ed team. However, the CV team found them to be helpful. Finally, Slack data collection was explored to get analysis for the higher education manuscript. However, the chat data and other information was unavailable because a paid subscription was required to use those features.

C. Links to Important Documentation

The SharePoint can be found here, along with all of the documentation: https://gtvault.sharepoint.com/sites/HAAG/Shared%20Documents/Forms/AllItems.aspx?noAuthRedirect=1. Within the SharePoint, the survey data including the volunteers can be found here: https://gtvault.sharepoint.com/sites/HAAG/data/Forms/AllItems.aspx. Within the SharePoint, the mentorship program matches and the extra credit document can be found here: https://gtvault.sharepoint.com/sites/HAAG/Shared%20Documents/Forms/AllItems.aspx?id=%2Fsites%2FHAAG%2FShared%20Documents%2Fprograms&viewid=7436995c%2D5ea%2D44a4%2Da82a%2Dfe3a885500c9. The Automate Flow to automatically send the emails needs to be shared. The code was copied over here: https://github.com/Human-Augment-Analytics/Higher-Ed/tree/main/Higher%20Ed%20Files%20and%20Code/Resource%20Management/Automate%20Flow, but the file needs to be exported via Automate and imported to the new owner. This can be done at a later time.

Resource management was done by Terry Junsoo Park. Please check the contact information section if you have additional questions.

IV. Methods: Program Development

For this part of the higher education project, the mentorship program and seminar program were started. These two programs help our research group engage with the wider OMSCS community. The buddy program pairs HAAG researchers with OMSCS students who are not part of HAAG. The seminar program allows HAAG researchers and members outside the group to present a scholarly topic to the group. Each program will be discussed below.

A. Mentorship Program

The mentorship program has the following objectives: **(1)** To bolster the knowledge of academic research and computer science in students who are interested. **(2)** To allow students to make connections with their peers in OMSCS and to network. **(3)** To allow mentors and mentees to learn from eachothers' experiences. The mentorship program is covered at this location on the HAAG website: https://sites.gatech.edu/human-augmented-analytics-group/category/buddy-program/. The program starts with a training session for both mentors and mentees, and then mentors are asked to connect with mentees on their own. For some example materials for the mentorship program, see Kailey's mentorship slides here: https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Program%20Development/Mentorship%20Program%20Intro%20Meeting%20Example%20Slides%20-%20Kailey%20Cozart.pdf. These slides are just an example. If someone was to use these, they would need to be updated with relevant information, such as in the "my research projects" section.

B. Seminar Program

The seminar program was designed to allow students, professors, and others to share specific information about their area of expertise. This program allows both presenters and listeners to grow and learn. The seminar program has the following objectives: **(1)** To help students learn more about their peers in the Human Augmented Analytics Group. **(2)** To encourage learning from others about scholarly topics. **(3)** To facilitate discussions about scholarly topics. During its first semester, the seminar program hosted 5 seminars. For reference, previous seminars are available for viewing on the HAAG website: https://sites.gatech.edu/human-augmented-analytics-group/category/seminars/.

C. Unsuccessful Methods

While the mentorship program ran smoothly with existing methods, there were a few unsuccessful methods attempted for the seminar program. First of all, Slack is not useful for recruiting seminar speakers from outside the group. Instead, potential seminar presenters from outside Human Augmented Analytics should be contacted via other methods. Furthermore, inviting team members to seminars via Teams invites and Slack channel tagging for seminar reminders were not useful for boosting attendance. In most cases, attendance was still low. Some potential ways to increase attendance could include inviting a wider range of students to attend, being more mindful when scheduling seminars, and requiring student researchers to attend a certain number of seminars.

D. Program Development Procedures

The program manager oversees the buddy program and the seminar program. For the mentorship program, the program development manager will be responsible for recruiting mentees, leading the training session, potentially providing materials for mentors to use, and following up with mentors to assure that they have adequately engaged with their mentees. For the seminar program, the program development manager will be responsible for recruiting presenters for the program, hosting the seminars, and archiving them on the website for future viewing.

Program development was done by Ayush Parikh. Please check the contact information section if you have additional questions.

V. Resources

For the higher ed team, several resources were used. The usefulness of each will be evaluated below. Some of the resources below are considered “ready to use,” so descriptions of how to use them are not required.

A. PACE

For resource management, PACE was only useful for learning and documentation. PACE was not useful for the Cichlid Computer Vision Team or for the Higher Education Team. For Georgia Tech’s documentation of the PACE ICE cluster, read the following: https://gatech.service-now.com/home?id=kb_article_view&sysparm_article=KB0042102. Additionally, further documentation and steps written by Terry can be seen here: https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Resource%20Management/PACE%20ICE%20Setup.docx. For documentation for running slurm jobs on PACE, see the following documentation, which was written by a previous student: https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Resource%20Management/PACE%20tutorial.pdf.

B. McGrath GPU

For resource management, we were unable to set up SSH remotely and IT support did not respond to this issue. This was unsuccessful. For the McGrath GPU, see the following notes: https://github.com/Human-Augment-Analytics/Higher-Ed/blob/main/Higher%20Ed%20Files%20and%20Code/Resource%20Management/McGrath%20computer.docx. In short, the McGrath GPU is a computer in the McGrath Lab that we want to SSH into. The process to use it would be to SSH into it after being granted appropriate permissions (potentially using SecureCRT, a VPN tool from the school).

C. SRG Computers

For resource management this is a backup plan, and it was not used by the Higher Education Team this semester. However, the Cichlid Computer Vision Team were able to use it successfully. To use this resource, simply request access from support@arcs.gatech.edu.

D. Personal GPU

For resource management, a personal GPU was used. A GPU with Linux was set up to remotely access with AnyDesk. With this setup, Dropbox client worked well for moving files quickly from the collaborator to the GPU. Jupyter notebook web server could be used alternatively, and SyncThing was used for file sharing between the local machine and the GPU. The GPU served the CV pipeline purposes successfully.

E. Slack

For program development, Slack was useful for some things, like contacting members of the group, but was not successful for other things, like reminding students about seminar times or for helping with recruitment of outside students.

F. Microsoft 365 Platform (SharePoint, Microsoft Forms, Power Automate)

The Microsoft 365 Platform was foundational for the resource management system that was developed. Power Automate was essential to automate sending emails for recruitment. Microsoft Forms had built-in data analysis which was useful for the manuscript. For program development, Microsoft Forms was useful because it worked great for the application surveys and feedback surveys for both programs. Additionally, Microsoft Teams was useful for hosting the required meetings.

G. Dropbox

For both resource and program management, Dropbox was useful for file storage and sharing.

H. GitHub

For codebase management, GitHub was useful because it provided easy solutions to open source code sharing, code contributions, and code tracking. For using this resource, refer to the codebase section of this document.

VI. Contact Information

See the two tables below for information on who to contact and how to contact them. Non-GIT emails have been provided in case the students graduate.

Name	Contact For
Kailey Cozart Quesada	Codebase Management
Terry Junsoo Park	Resource Management
Ayush Parikh	Program Management

TABLE I: Who to Contact

Name	Emails
Kailey	kaileycozart@gmail.com, kcozart6@gatech.edu
Terry	jun_park@live.com, jpark3232@gatech.edu
Ayush	ayushnparikh@gmail.com, aparikh49@gatech.edu

TABLE II: Contact Information