CS11 - Gesture Recognition Keyboard

Feedback #1:

Category	Description	Reviewer's comment	Action taken by reviewed group
Build	Could you clone from Git and build using the Readme?	I was not able to build the project, mostly because I do not have an IOS device, and have no experience with swift.	We recognize and understand that some users cannot build this project. We clearly communicated this during the demo.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Variable names make sense, and code style is alright. The code could be made far more readable by splitting up the modules into individual files based on the functionality, rather than having one large UI doc with functionality in it.	We understand this criticism but feel that the functions we used are the best way to accomplish this, as choosing to abstract away portions of the code actually makes it less readable and harder to understand as each function relates to a specific key on the keyboard, and they all do different things.
Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The code has been abstracted well and makes good use of toolkits.	Thank you.

Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	There are no unit tests, though I do see spaces where they could be implemented, checking for internal function responses, etc.	We communicated that our project doesn't warrant unit tests unfortunately. Our algorithm is largely affected but our training data, which we have added. The model has testing data associated with it, which we used to test the validity of our model. The UI is the only part that doesn't have unit tests, which we feel is acceptable.
Requirements	Does the code fulfill the requirements?	In many ways yes, the algorithm is the only thing that needs to be tuned. It is not quite up to the 85% mark yet.	We have performed more training for our data so that our algorithm increases in speed and accuracy. We haven't been able to hit 85% yet, but we have addressed this with our client, and will include the reasons why in our report.
Other	Are there other things that stand out that can be improved?	Nope, good job everyone, If I was a TA, I would give your progress a 3 or a 4.	Thank you. We believe we have addressed the issues with our project.

Feedback #2:

Category	Description	Reviewer's comment	Action taken by reviewed group
Build	Could you clone from Git and build using the Readme?	I could clone, but I failed to run it with the instruction. I don't know why.	This comment is not helpful because we don't know what error occurred. We found that one major issue with building our project is adding an ssh key to the user's github so that the user has proper permissions for SnapKit.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Not very easy to follow, too many packages stuff needs a better explanation. The lighting needs to be improved, such as some lines are extreme long.	We have added more comments so the code is easier to follow. We understand that some experience with Swift4 is required to further understanding.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	Not for all the parts, there are some duplicate uses for the swift stuff. I'm little confused about how they used the swift package to recognize the character.	We went over our code and cannot eliminate duplicates. We are not sure which duplicates the reviewer is referring to. We may not have fully explained how the createML toolkit functions, but we have added some additional comments to help with understanding.

Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	Yes, there should be, because they have many datasets, the accuracy of classifying a character is worth to test with the functionality.	We have unit tests for our training data. Any other unit tests for our project is not necessary because unit tests on the UI are purely to test superficial things, which can be done manually, and doesn’t have enough variability to warrant unit tests.
Requirements	Does the code fulfill the requirements?	Not fully meet the requirements yet, the demo is failed due to some dependencies stuff.	We have fixed the minor dependency issues.
Other	Are there other things that stand out that can be improved?	Make the instructions for set up better.	We have clarified how to build the system and the requirements.

Feedback #3:

Category	Description	Reviewer’s comment	Action taken by reviewed group
Build	Could you clone from Git and build using the Readme?	Team was able to build during demo.	Yes.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	- I would suggest renaming 'First/SecondViewController' to something clearer - Mostly follows good coding practices, however I would suggest using more comments to clarify what you are doing	We feel like the name ‘firstviewController’ is descriptive, as it is the controller for the first view in our containing app. We have also added more comments to clarify what we are doing.
Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	- Would it be possible for you to turn the code blocks in keyboardView() into a function that is repeatedly called with different parameters? Much of the code looks identical aside from the input values.	We understand that these functions do look very similar. However, they are in fact different and necessary. We have addressed this concern but adding comments. More specifically, each one is connected to a button on the keyboard, and they all have primarily unique code, which makes them an impractical candidate for functionalization.
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	Unit tests do not appear to be present. They could be useful for testing the app side of the project. I	Since the app is for demoing purposes, we do not have unit tests for it. We have some unit tests for

		would suggest looking into XCode's unit testing and UI input automation. However, much of the project is based on ML models that are not as easily tested via unit testing	our model but the rest of the code does not make sense nor require unit tests because unit tests on the UI are purely to test superficial things, which can be done manually, and doesn't have enough variability to warrant unit tests.
Requirements	Does the code fulfill the requirements?	The code appears to satisfy the requirements pertaining to the app. Other requirements are based on the ML models, and thus require different evaluation.	Thank you.
Other	Are there other things that stand out that can be improved?	The code appears solid. I would mainly suggest adding more comments and look into condensing your code if possible.	Thank you, we have added more comments for better understanding and clarification.

Feedback #4:

Category	Description	Reviewer's comment	Action taken by reviewed group
Build	Could you clone from Git and build using the Readme?	As they said, this isn't currently possibly because they're utilizing a third party library and I don't have a developer account. However, I found the instructions very clear to follow, and I believe if I had an Apple developer account they would definitely be sufficient to get the project built.	Thank you.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The one thing that did throw me o was naming the view controllers "First/SecondViewController," which gives very little contextual information about what they're actually doing in the program. However, other than that Most of the naming conventions were consistent and the code was rather easy to follow. It definitely followed along the lines of standard Swift conventions I've seen.	We feel like the name 'firstviewController' is descriptive, as it is the controller for the first view in our containing app.

Implementation	is it shorter/easier/faster/cleaner/ safer to write functionally equivalent code? Do you see useful abstractions?	The switch to Apple ML model was a really smart move, and it definitely cut down the code base quite a bit from what it would be. That being said, it definitely may halt the long term extensibility of the project, and I imagine sometime going forward in the future this would have to be migrated to a custom algorithm. Also I saw a ton of repeated code for the "xxxButton" lines in your KeyboardViewController. I would imagine this could be shortened quite a bit if you managed to store the information in an array, but perhaps not. Altogether the code was great though, and besides those minor dierences I thought it was pretty succinct.	Thank you, we have discussed this with our client. Thank you, we do understand that some of our code looks really similar, but the functions actually serve their individual purpose and cannot be combined.
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	It's rather hard to write unit tests for this, however basic UI tests can be written that test out, for example, pivoting between the view-controllers. It wouldn't be super helpful, though.	We agree. Since the app is for demoing purposes, we do not have unit tests for it. We have some unit tests for our model but the rest of the code does not make sense nor require unit tests because

			unit tests on the UI are purely to test superficial things, which can be done manually, and doesn’t have enough variability to warrant unit tests.
Requirements	Does the code fulfill the requirements?	Yes, besides the minor dependency issues which should be easily xable the codebase as a whole definitely meets the requirements.	We have addressed the minor dependency issue.
Other	Are there other things that stand out that can be improved?	Perhaps more test cases to better t your machine learning model, along with the minor xes and usability/readability I suggested. Otherwise I think it looks great, and nothing major comes to mind from me.	We have added more training data to increase accuracy of the model.

Feedback #5:

Category	Description	Reviewer's comment	Action taken by reviewed group
Build	Could you clone from Git and build using the Readme?	Project video and demo shows that it builds properly. The readme has good documentation for building and includes screenshots.	Thank you.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The code is well written and adheres to a common style.	Thank you.
Implementation	is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The code is organized in a logical way. Code has been properly refactored.	Thank you.
Maintainability	Are there unit tests? Should there be? Are the test covering interesting cases? Are they readable?	There is a skeleton for some tests but no unit tests as of yet. It might be a good idea to add these eventually.	Since the app is for demoing purposes, we do not have unit tests for it. We have some unit tests for our model but the rest of the code does not make sense nor require unit tests because unit tests on the UI are purely to test superficial things, which can be done manually, and doesn't have enough variability to warrant unit tests.

Requirements	Does the code fulfill the requirements?	Yes.	Thank you.
Other	Are there other things that stand out that can be improved?		