Introduction - The goal of this program was to simulate a neural network to find the effect of different hyper parameters on accuracy and runtime. The program takes user inputs from a graphical user interface and runs a neural net based on those inputs. The neural net used was from an imported package: deeplearning4j. These imports allow us to create a neural network by easily adding certain adjustments to each layer of the network.

User Stories -
Current Features

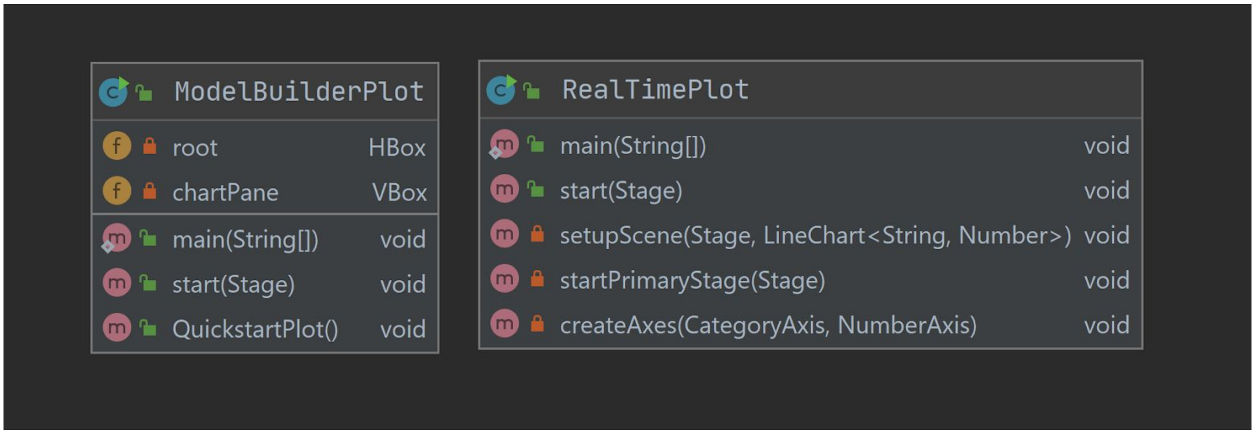
- Should have a GUI
    - Should accept user input via text fields and drop down boxes
    - Should show output as text and graphs
    - Should have buttons to start the model
    - Should have buttons to reset the graph
- Should be able to train models
- Should allow the user to change variables such as epochs, batch size, etc.
- Should report the accuracy and time to train of the most recent run on the GUI
- Should graph the accuracy and time to train of all runs to be shown on the GUI

Features that didn't make it into the final project

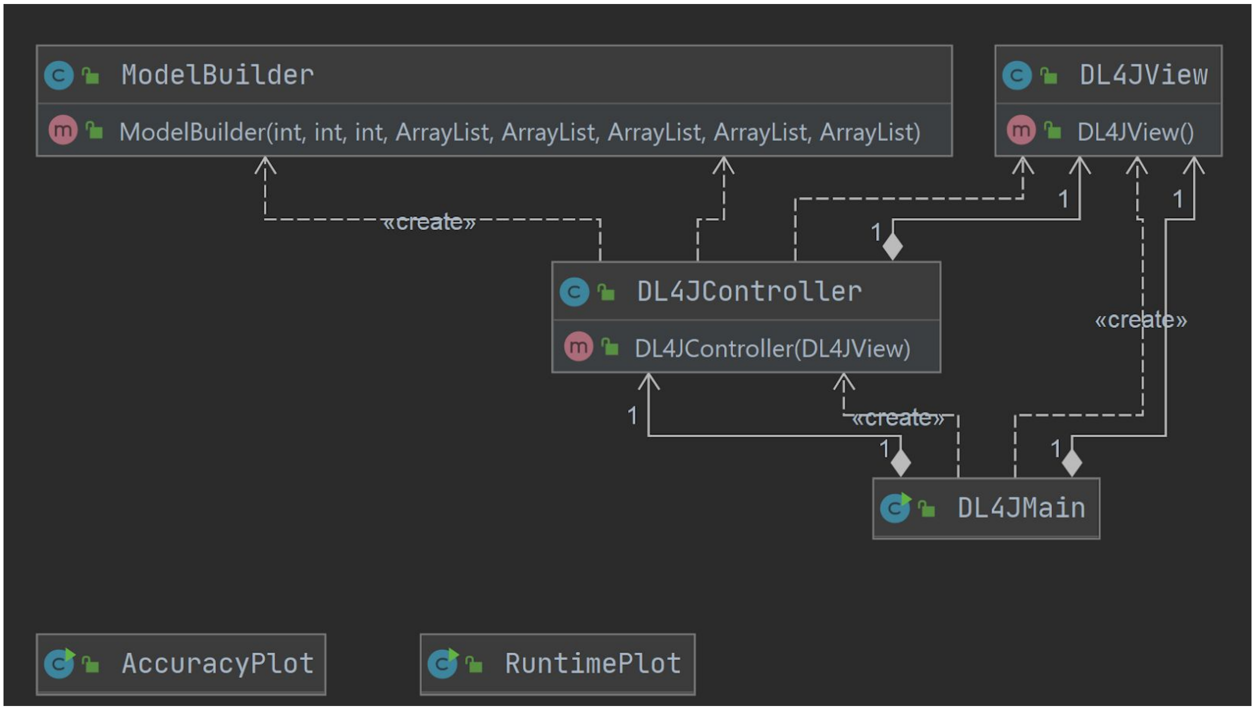
- Ability to upload your own dataset
- Ability to switch between CPU and GPU for training
-

OOD -

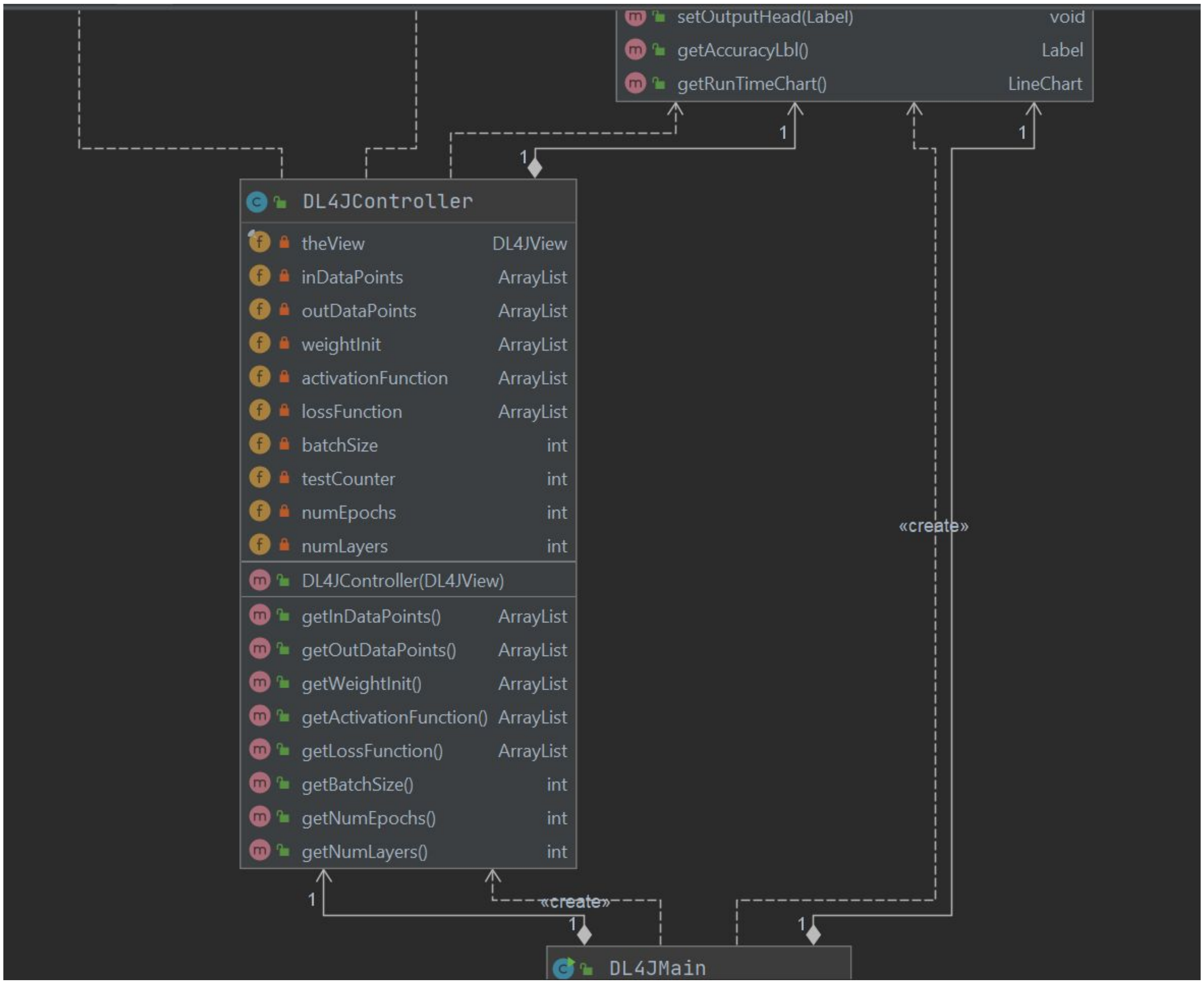
The neural network project consists of a front end that includes a user interface with visualizations of network runtime and accuracy. The backend consists of the neural network itself. We used the deeplearning4J library for our neural network.



Smaller view to show how all of the classes interact with each other.



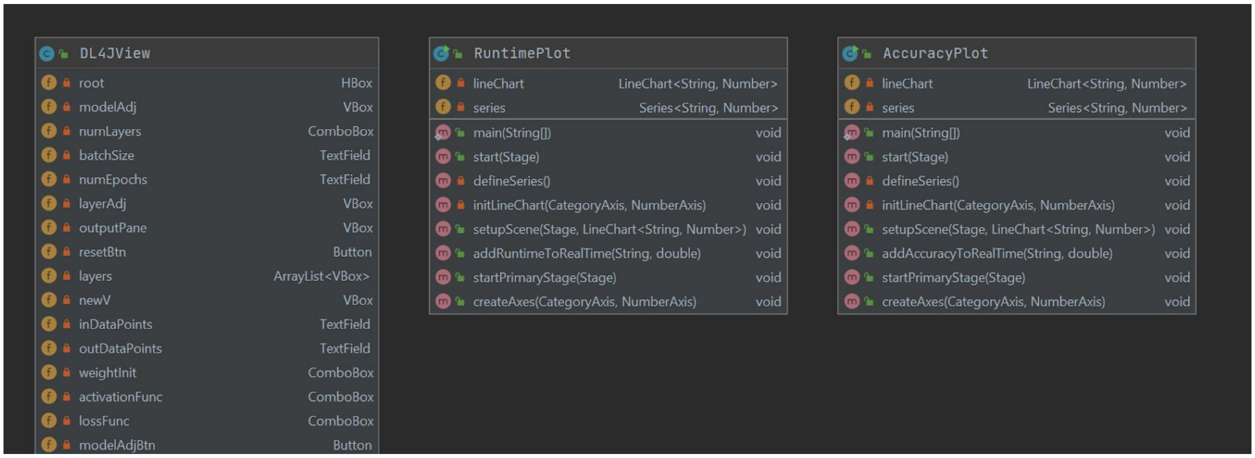
Bigger views of some of the methods inside the classes.



setOutputHead(Label)	void
getAccuracyLbl()	Label
getRunTimeChart()	LineChart

DL4JController	
theView	DL4JView
inDataPoints	ArrayList
outDataPoints	ArrayList
weightInit	ArrayList
activationFunction	ArrayList
lossFunction	ArrayList
batchSize	int
testCounter	int
numEpochs	int
numLayers	int
DL4JController(DL4JView)	
getInDataPoints()	ArrayList
getOutDataPoints()	ArrayList
getWeightInit()	ArrayList
getActivationFunction()	ArrayList
getLossFunction()	ArrayList
getBatchSize()	int
getNumEpochs()	int
getNumLayers()	int

DL4JMain



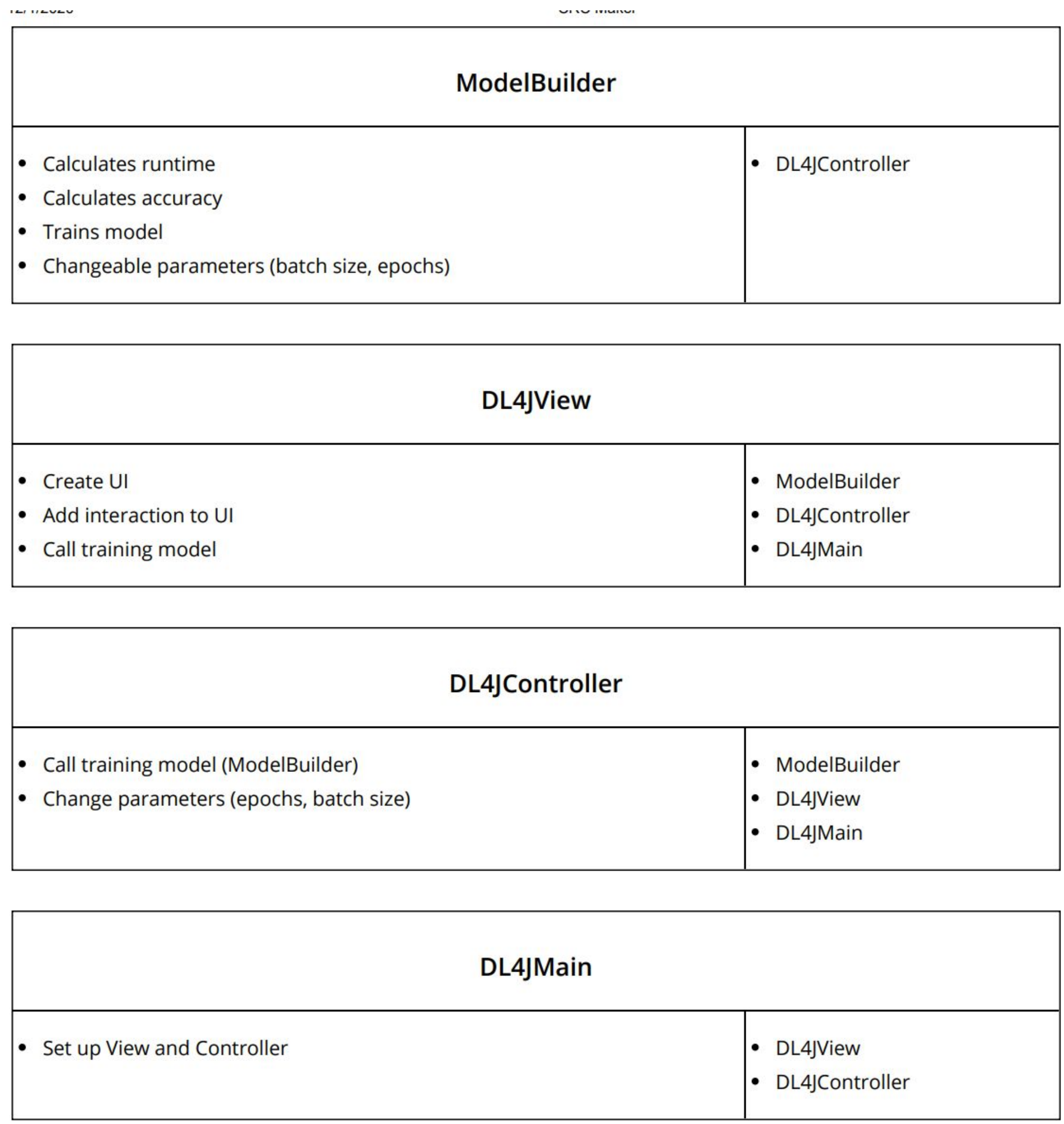
DL4JView	
root	HBox
modelAdj	VBox
numLayers	ComboBox
batchSize	TextField
numEpochs	TextField
layerAdj	VBox
outputPane	VBox
resetBtn	Button
layers	ArrayList<VBox>
newV	VBox
inDataPoints	TextField
outDataPoints	TextField
weightInit	ComboBox
activationFunc	ComboBox
lossFunc	ComboBox
modelAdjBtn	Button

RuntimePlot	
lineChart	LineChart<String, Number>
series	Series<String, Number>
main(String[])	void
start(Stage)	void
defineSeries()	void
initLineChart(CategoryAxis, NumberAxis)	void
setupScene(Stage, LineChart<String, Number>)	void
addRuntimeToRealTime(String, double)	void
startPrimaryStage(Stage)	void
createAxes(CategoryAxis, NumberAxis)	void

AccuracyPlot	
lineChart	LineChart<String, Number>
series	Series<String, Number>
main(String[])	void
start(Stage)	void
defineSeries()	void
initLineChart(CategoryAxis, NumberAxis)	void
setupScene(Stage, LineChart<String, Number>)	void
addAccuracyToRealTime(String, double)	void
startPrimaryStage(Stage)	void
createAxes(CategoryAxis, NumberAxis)	void

RuntimePlot and AccuracyPlot are not connected directly to the rest, but they are used to plot the results.

CRC Cards:



12/1/2020 CRC Maker

ModelBuilder

• Calculates runtime • Calculates accuracy • Trains model • Changeable parameters (batch size, epochs)	• DL4JController

DL4JView

• Create UI • Add interaction to UI • Call training model	• ModelBuilder • DL4JController • DL4JMain

DL4JController

• Call training model (ModelBuilder) • Change parameters (epochs, batch size)	• ModelBuilder • DL4JView • DL4JMain

DL4JMain

• Set up View and Controller	• DL4JView • DL4JController