Running the Deep Learning Poker Playing Demo

This is a 4-part demo which includes generating card data from a webcam, training a model using transfer learning, testing the model trained and then showing how a deep learning model can be used in an app.

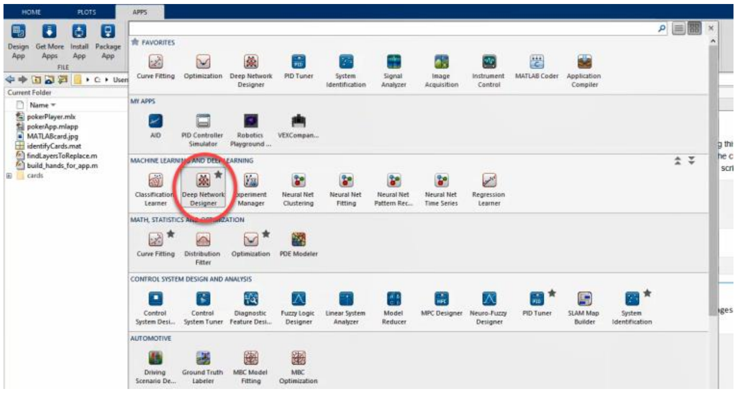
Generating card data

1. Open polerPlayer.mlx
2. Select only the checkbox for “generateData”
3. Run the script
4. A window will appear showing the camera view, make sure this is visible
5. Select a card at random from the deck of cards
6. Position the card in front of the camera
7. Enter the card’s number and suit into the command window and press enter
    a. This will begin taking pictures and storing them into a folder titled ‘cards’
    b. Move the card from side to side in order to get a good range of data for the card
8. Repeat steps 5-7 with 2 more cards from the deck
9. Once done taking pictures for both cards, enter ‘N’ into the command window to clear the camera
10. Open the cards folder to display the camera picture data taken
    a. You will notice that every other picture is flipped upside down, every 5th image is darkened, and every 6th image is lightened
    b. Lines 30-38 in the code is responsible for doing this
    c. This is done to account for the cards being held upside down and different lighting in the room

Training a deep learning model

In the pokerPlayer.mlx live script is code to use transfer learning to train a model to identify these cards. In this demo we will show how to do this process through the Deep Network Design app primarily.

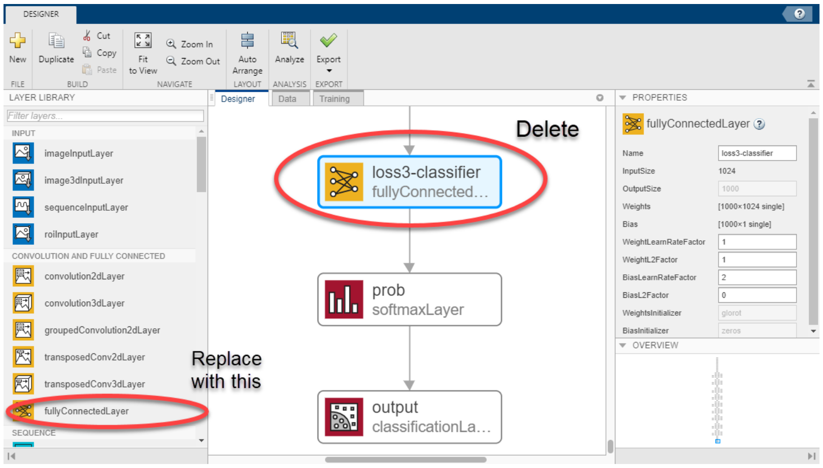
1. Go to the ‘APPS’ tab in MATLAB
2. Select the app dropdown and navigate to the “Deep Nerwork Designer” app



3. Under the Pretrained Networks section, select “GoogLeNet”

   a. This may require you to download a support package

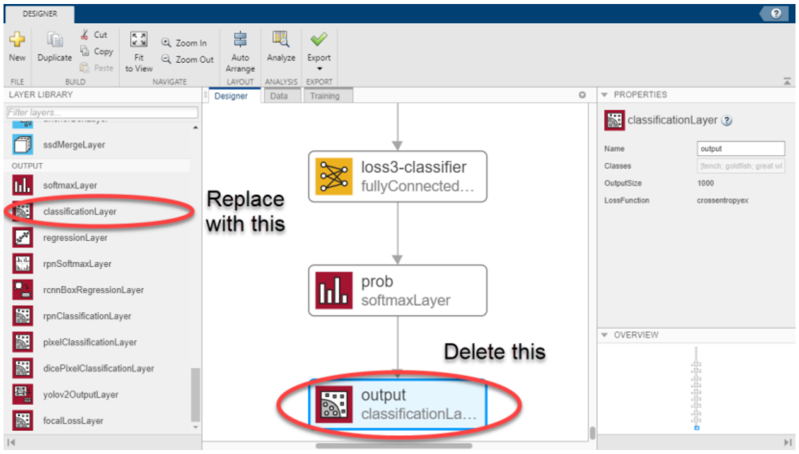
4. Scroll down to the bottom of the network and find the 3rd to last layer, the fullyConnectedLayer
5. Delete this and replace it with the same fullyConnectedLayer



fullyConnectedLayer

Name	loss3-classifier
InputSize	1024
OutputSize	1000
Weights	[1000×1024 single]
Bias	[1000×1 single]
WeightLearnRateFactor	1
WeightL2Factor	1
BiasLearnRateFactor	2
BiasL2Factor	0
WeightsInitializer	glorot
BiasInitializer	zeros

6. In the new fullyConnectedLayer, change the OutputSize option to 3
7. Delete the very last layer labeled output and replace it with the same classificationLayer

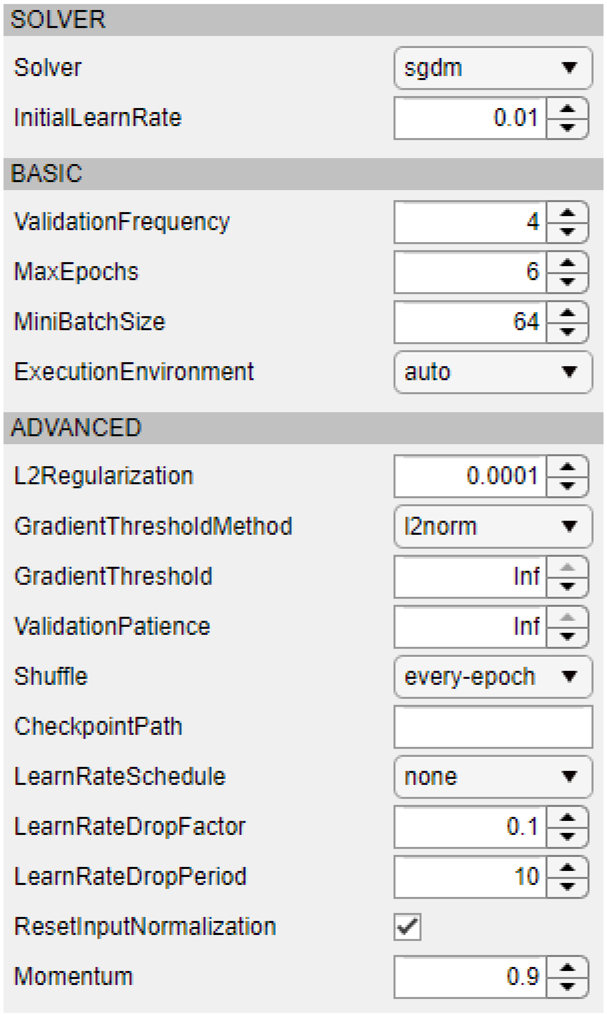


classificationLayer

Name	output
Classes	[tench; goldfish; great wl
OutputSize	1000
LossFunction	crossentropyex

8. Click on the “Data” tab
9. Click “Import Data”
10. Click the “Browse” button in the window that appears
11. Navigate to the ‘Cards’ folder in which the card picture data you just took is located and click “Select Folder”
12. Press okay
13. Click on the “Training” tab
14. Click on “Training Options”

15. Change the options to the options below but it's encouraged that you read through the [training options documentation]() and experiment with the various training options



SOLVER

Solver	sgdm
InitialLearnRate	0.01

BASIC

ValidationFrequency	4
MaxEpochs	6
MiniBatchSize	64
ExecutionEnvironment	auto

ADVANCED

L2Regularization	0.0001
GradientThresholdMethod	l2norm
GradientThreshold	Inf
ValidationPatience	Inf
Shuffle	every-epoch
CheckpointPath	
LearnRateSchedule	none
LearnRateDropFactor	0.1
LearnRateDropPeriod	10
ResetInputNormalization	☑
Momentum	0.9

16. Close this window and then press the "Train" button to start training the model
    a. This training will take anywhere from 3-8 minutes depending on your computer
    b. During this time switch the live script and explain doing this same process through code
17. Once the training is done click the "Export" drop down and select "Export Trained Network and Results"

Testing the model

1. Check to make sure the name of your model saved is the same as the model in line 94 of the code
2. Click the "Test Model" button
    a. This will run that section and open a window showing the camera, the predicted card, and the score

3. Show each of the cards that you took pictures in front of the camera to ensure that the model is working properly

Using the deep learning model in an app

1. Run the pokerApp.mlapp file
2. Press the “Start Game” button

Follow the process of playing the game – it’s pretty straight forward. Just make sure you hold up a card in front of the camera before you click the Add to hand, or Add to table button.