ESE3600: Tiny Machine Learning Fall 2022

Project Report

Handed out: *Due:*

Name: *Xiang Chen*
PennKey: xianc
PennID: *73038738*

Name: *Rachael Tamakloe*
PennKey:*rachealt*
PennID: *51474029*

Introduction:

Our project is called RoboDog. This project is based on using either magic wand or keyword spotting to control a robot and make it act like a dog. We implement actions like turn, move forward, and spin just like commands you would give to a dog. We have the option to switch to either input using the keyword spotting or the magic wand to send commands to the robot. Currently they both have the same commands, but future improvements to the project might be that there could be an arm attached to a servo and you can control that with keyword spotting to fetch an object just like a dog would.

Motivation:

Our motivation is so that people that could not have pets could have a pet in the form of this robodog. Often when I travel from place to place, I often feel lonely and I am always jealous of my friends having pets that could comfort them. My apartment does not allow pets so even though I always wanted a pet like a dog or cat, I am unable to ever get one until I move out and buy my own house. Because of such limitations, this project is aimed at people that can get a robotic dog that can travel everywhere with them. By doing commands that a dog would such as doing hand gestures (magic wand) or voice commands (KWS), it opens up a lot of possibilities for people that want a dog but cannot get one. The RoboDog might also be appealing for first time pet owners that are on the fence of getting a dog. Before getting an actual dog, it is important to know that you are really committed and know what you are getting into. If someone adopts a dog and cannot handle the responsibility of caring for it anymore, the dog will go back to the shelter and that's a lot of wasted resources and time for everyone. By having a trial with

RoboDog, you can experience how a dog will feel when you own one, and you can really decide if you want a dog or not.

Dataset Details:

We have a total of 2 datasets, one for the magic wand and one for the keyword spotting. The magic wand dataset has a total of 6 magic wand symbols/patterns. These patterns were made keeping in mind how a normal dog would respond to a similar command and similar gesture. The pattern “>” corresponds to turning right, the pattern “<” corresponds to turning left, the pattern “^” corresponds to moving forward, the pattern “e” corresponds to stopping, the pattern “o” corresponds to doing a 360 spin, and lastly the pattern “v” corresponds to switching to KWS mode.There are 120 magic wand datasets for each symbol and we did an 80-10-10 split for training, validation, and test. We used Pete’s magic wand tool to gather and label our magic wand dataset.

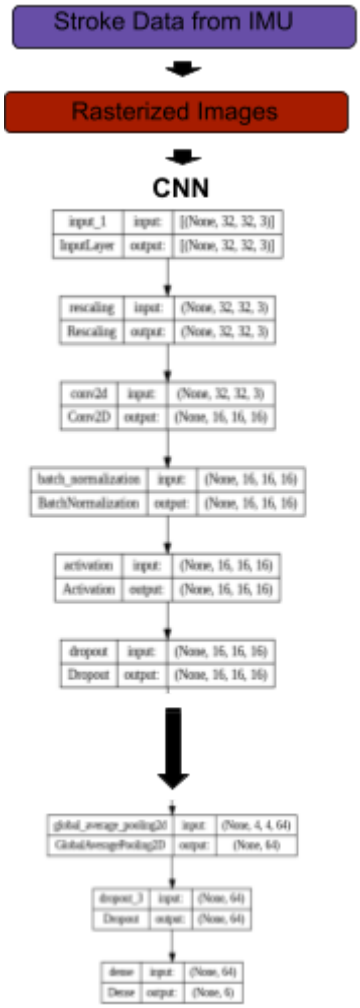
For the KWS we have a total of 6 keywords. They are named up, stop, left, right, spin, and fetch. In addition to these keywords we also have 2 additional labels for unknown words and silence. For the unknown words, we randomly subsampled 120 audio recordings from Pete’s dataset. For the silent label, we recorded 120 silent audio recordings. We have 120 datasets for each keyword and they are also split for 80-10-10 for training, validation, and test. Up, stop, left, right, and spin are intuitive. We initially watched the fetch keyword to serve another purpose, but at the last minute we re-purposed our dataset for fetch as the keyword for the multi-tenant model. When the model registers that the keyword fetch has been uttered, the robot switches from KWS mode to Magic wand mode.This way you can have 2 different types of inputs depending on preference.

Model Choice and Design:

KWS Model Architecture:

We designed the KWS model architecture using Edge impulse. Edge impulse splits the model building phase into 2 blocks that comprise data preprocessing and a learner block. For the KWS model, we used Mel-frequency cepstral coefficient(MFCC) for preprocessing and a 2- part Convolutional network for the learner block. Preprocessing audio data using MFCC is a commonly used technique in speech processing to better capture relevant features of audio data. By employing this technique in our model, we are able to make the KWS model more accurate and efficient in using relevant features for prediction. CNNs were used for the learner block because they are effective at processing audio data in addition to image data. By using a 2-part CNN model architecture, our KWS model is able to synthesize more complex patterns in the audio data, making its performance better in terms of accuracy and generalizability.

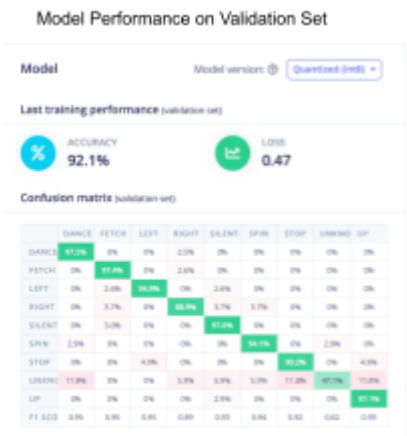
Magic Wand Model Architecture:



The Magic Wand model architecture can also be split into two parts. In the pre-processing step, the stroke data recorded using the IMU are converted into rasterized images. In the second step, the dataset of rasterized images are then fed into a Convolutional neural network to process and train on. In detail, our CNN architecture mainly comprises 3 convolutional blocks that each include a convolutional layer, a batch normalization layer, a reLu activation layer, and a dropout layer. The general use of a convolutional network allows for good processing of the image data of rasterized strokes. In addition, the use of batch normalization and dropout layers in the convolutional blocks help with the reduction of overfitting, further helping the performance of the model after deployment.

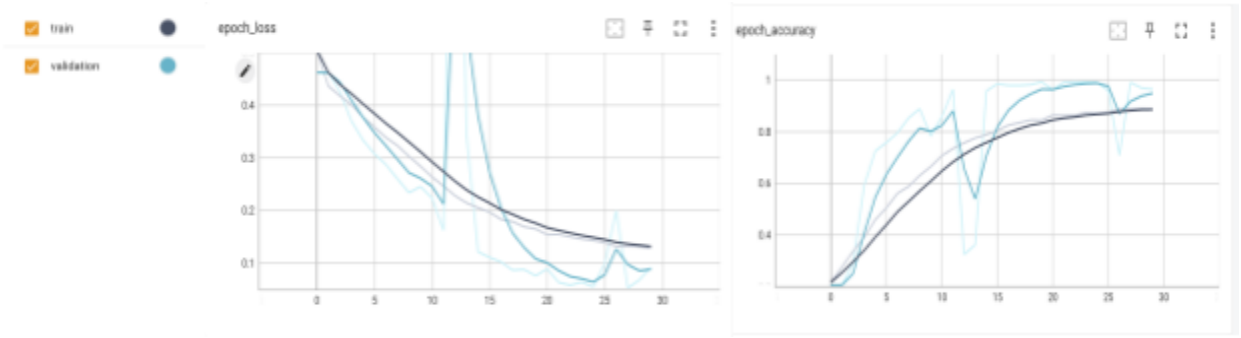
Model Training and Evaluation Results:

KWS Model Performance:



As shown in the diagrams above, the KWS model has a validation accuracy of 92% and a test accuracy of 88.19%. Taking a closer look at the prediction accuracy for each class, we note that the unknown class has the lowest accuracy. This however is expected, given that the dataset of unknown words comprises a mix of different words that the model has to predict as one class of words.

Magic Wand Model Performance:



By the end of the 30 epochs, our magic wand model achieved a training accuracy of 89% and a validation accuracy of 97%. As you can see, it is a very accurate model with no overfitting as the validation accuracy is always increasing throughout the training. When testing our model on the test dataset split, we got 99.9% accuracy on our quantized int8 model, meaning the model will be accurate when recognizing gestures when it is deployed on the Arduino.

Deployment/Hardware Details:

In terms of hardware/deployment details, our project makes use of three sets of arduinos and a two wheeled mobile robot. The first arduino is used for the Magic wand model, the second arduino for the KWS model, and the third arduino is used for control of the robot. The Magic wand arduino and the KWS arduino are connected to each other through a wired communication protocol called UART. Our Multi-tenant model has two modes. In the first mode, the robot is controlled by KWS. In the second mode, the robot is controlled by the Magic Wand Model. Depending on which mode the multi-tenant model is in, the magic wand arduino sends model outputs over to the arduino connected to the robot through a bluetooth connection. In the case that we are in the first mode(robot controlled by KWS), the magic wand arduino receives model outputs from the KWS arduino through the wired connection, and then sends those outputs to the arduino connected to the robot over bluetooth. In the case that we are in the second mode(robot control by Magic Wand), the magic wand arduino stops receiving model outputs from the KWS arduino and sends its own model outputs from the magic wand model over to the arduino connected to the robot through the established bluetooth connection.

Challenges/ Future Work:

We faced a lot of challenges when designing our multi-tenant model. We initially wanted our multi-tenant model to employ a visual processing component that would allow the robot to detect objects(specifically balls) as it moved around. However we ran into the issue of our deployed model being too big to be deployed on the arduino nano. Using a smaller sized architecture resulted in very bad performance, so we decided to forgo the use of the visual component all together.

In terms of training and deploying our KWS model and the Magic Wand, we had no issues with the magic wand, but the KWS model caused us a lot of trouble. Using our initial model structure trained on our Collab notebook, the model had very bad performance after deployment. We then pivoted to use Edge Impulse in hopes of building a better performing model. Using Edge impulse, the KWS model that we built had a very good performance. We decided to use this model for KWS.

In terms of future work, implementing more actions for the RoboDog and redesigning the hardware details for a more compact product would be good directions to head in. Currently, the Robodog only has two modes, and only responds to 5 movement commands. Adding more commands and more interesting behavior (such as barking, and fetching a ball) would make the functionality of the RoboDog more like a dog.

In terms of making our product more compact, we could possibly consider the use of another arduino board that has more space capacity to query multiple models on one board. Currently, the robo dog employs two separate arduino boards for the KWS and magic wand

model. Using a board with more space capacity would allow the deployment of both the KWS model and the Magic wand model on one board.