Mode Counting in Acoustic Signals

Steven Bradley, Mateo Ibarguen, Nathan Phillibert

The Agenda

- The Problem
- The Data
- Model Architectures
- Results
  - Comparing Model Performance
  - Effect of Channels on Performance
  - Channel Padding
- Future Work

The Problem

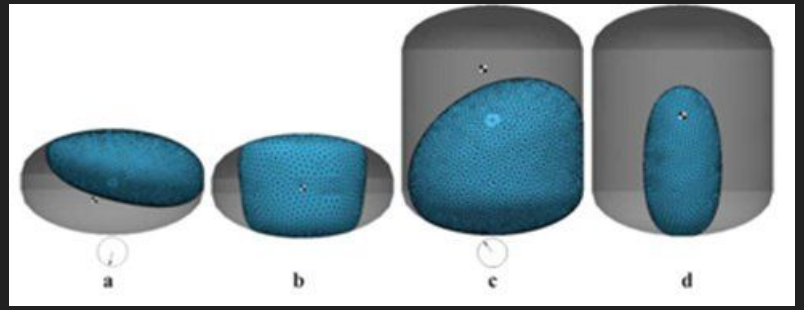
Measure the volume of fluid in a tank

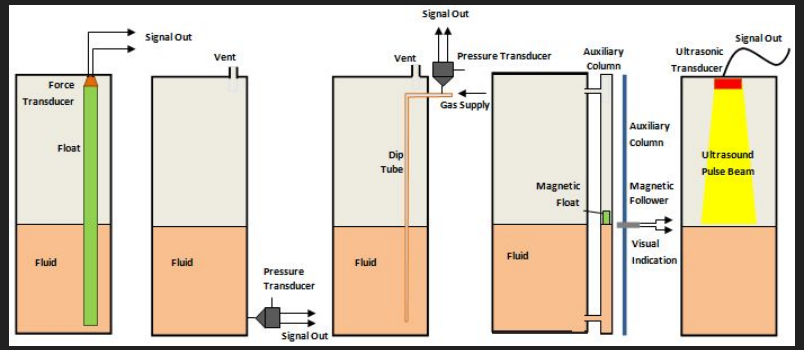
… in space

Measuring Tank Volume: Zero Gravity

- Liquids behave differently in low-gravity environments
- Traditional sensors will not work

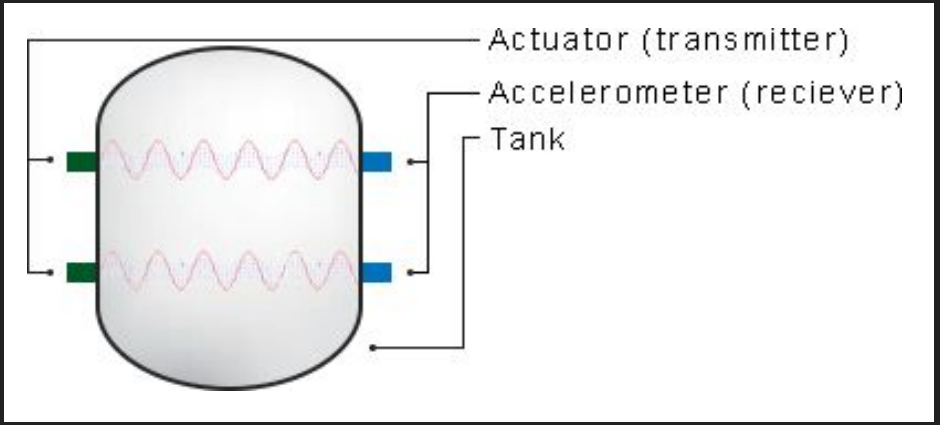
Liquid doesn't settle to the bottom of tank in low-gravity



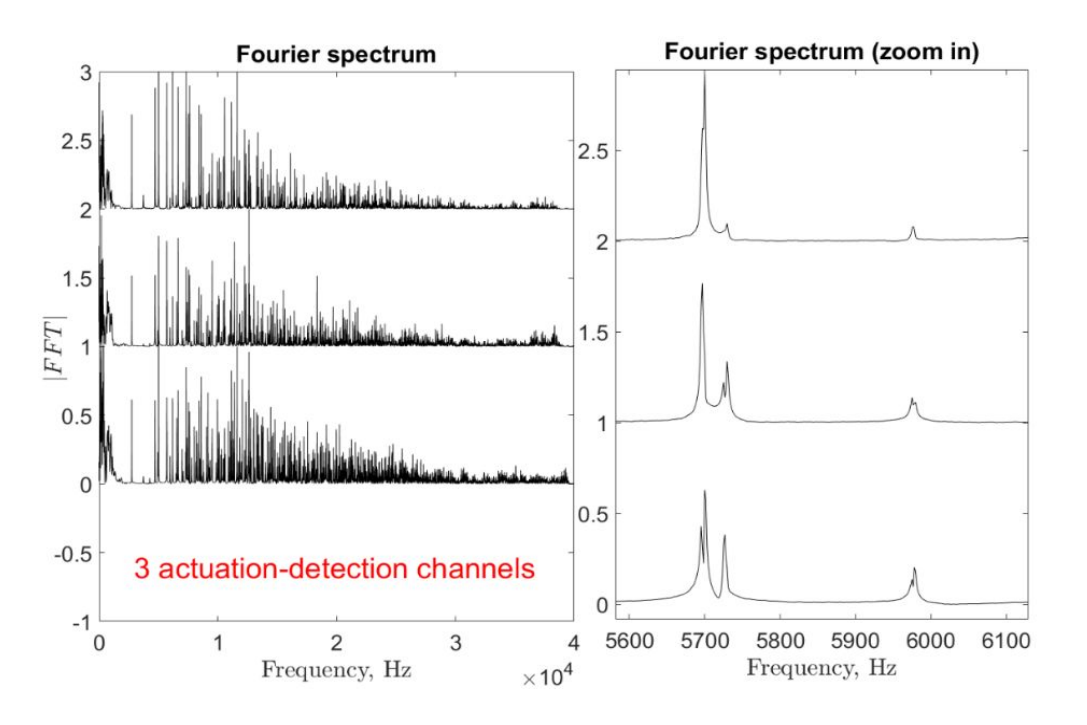


Spectral Mass Gauging in Unsettled Liquids

- Estimate the **volume** of unsettled liquids in **low-gravity**
- Acoustic resonances



Acoustic Resonances



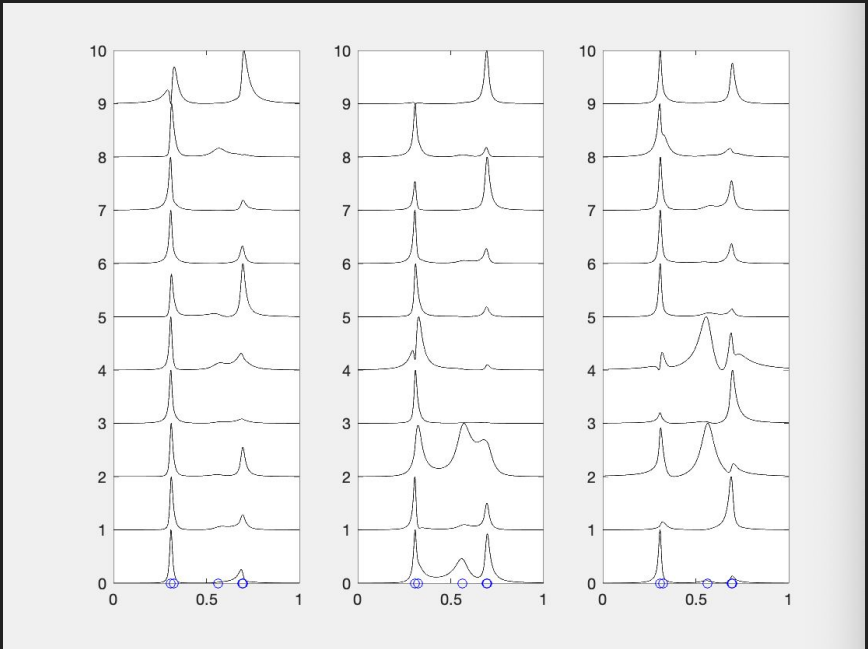
Global vs local

Multiple channels

Shell modes vs **liquid** modes

The Data

The Data



Original MATLAB spectrum visualization

- The original MATLAB script generates **simulated Fourier-transformed data**
- **Users manually estimate** where the modes are located by pressing 1
- The **correct mode locations** are later displayed with blue circles

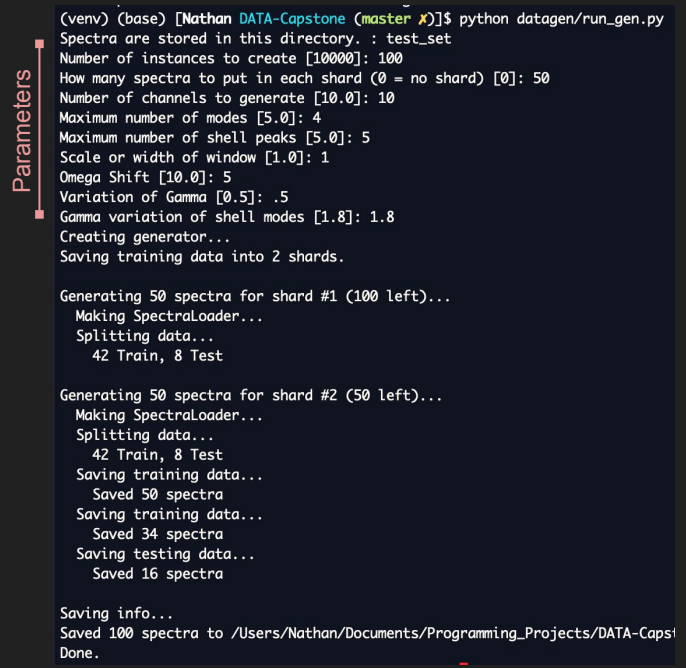
The Task: Use the MATLAB code to easily **generate a large amount of simulated data** with specified parameters

Dataset Parameters

NMax:	Maximum number of resonances
NMaxS:	Maximum number of shell modes in a spectral window
NC:	Number of channels
Gamma:	Damping of each resonance
dG:	Variation of gamma
dGS:	Variation of gamma for shell modes
Omega:	Frequency of each mode
Scale:	Width of a window

The Data

- Data is optionally sharded
- Data is saved as JSON format
- Stored on AWS server / S3 bucket
- Command line tool to generate new sets



Model Architecture

Model Development

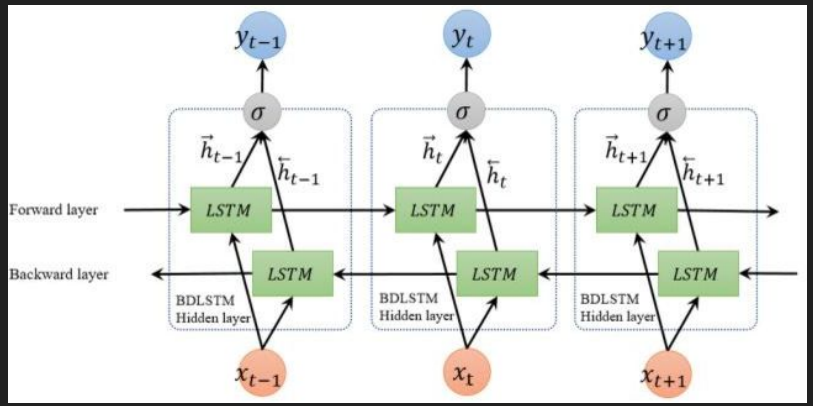
Neural Networks

- **Features extraction** would be non-trivial
- **Noise** and shell modes **filtering** would be non-trivial
- Existing Research

Loss Function

- Categorical Cross Entropy
- Poisson Loss
- Macro Averaged Mean Absolute Error

Model Architectures



CNN

Bidirectional LSTM

Ensemble CNN

CNN + LSTM

Attention

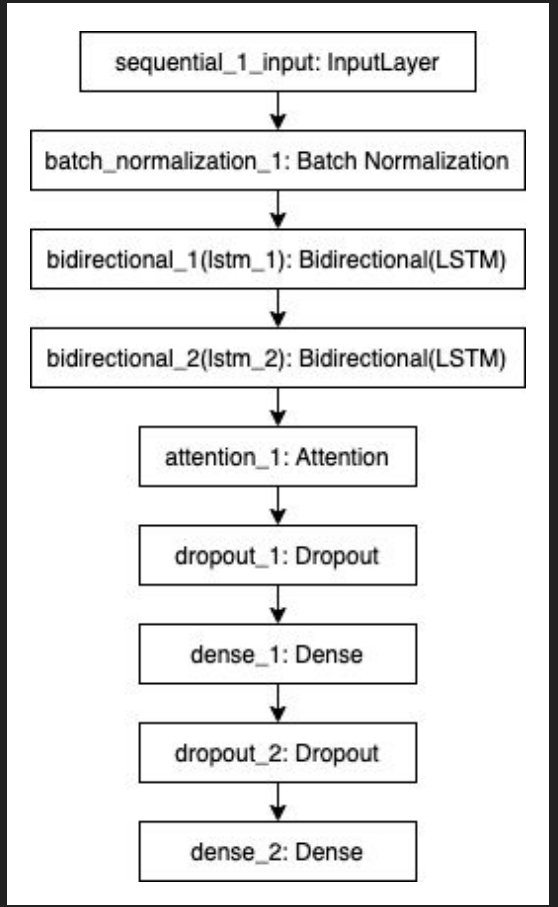
Results

Results Overview

- **Evaluating Performance** of 3 Model Architectures
- Effect of **Variable Channels**
- Channel Padding

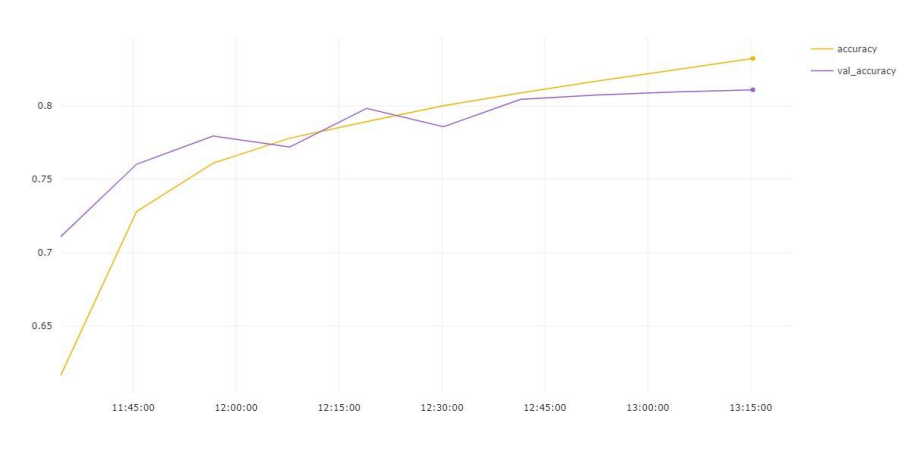
LSTM Model

- One of our initial models
- **Bidirectional LSTM**
- **Attention Layer**

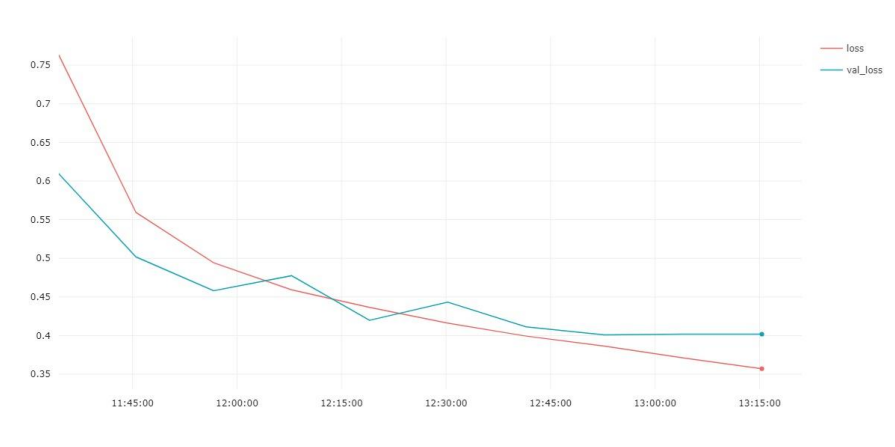


LSTM Model

Training Accuracy
Validation Accuracy



Training Loss
Validation Loss



LSTM Model Results

```
Evaluated with:
     50 channels
Omega Shift =  10
Delta Gamma = 0.5
Delta Gamma Shell = 0.5
Max Modes =   4
Max Shell Modes =   5
```

1v2v3v4 Classifier	Predict 1	Predict 2	Predict 3	Predict 4
Actual 1	250	0	0	0
Actual 2	1	245	4	0
Actual 3	0	38	180	32
Actual 4	0	8	98	144

Test Accuracy: 81.1%
Test Loss: 0.401
Test MAE: 0.118
Test MSE: 0.064

LSTM Model Results

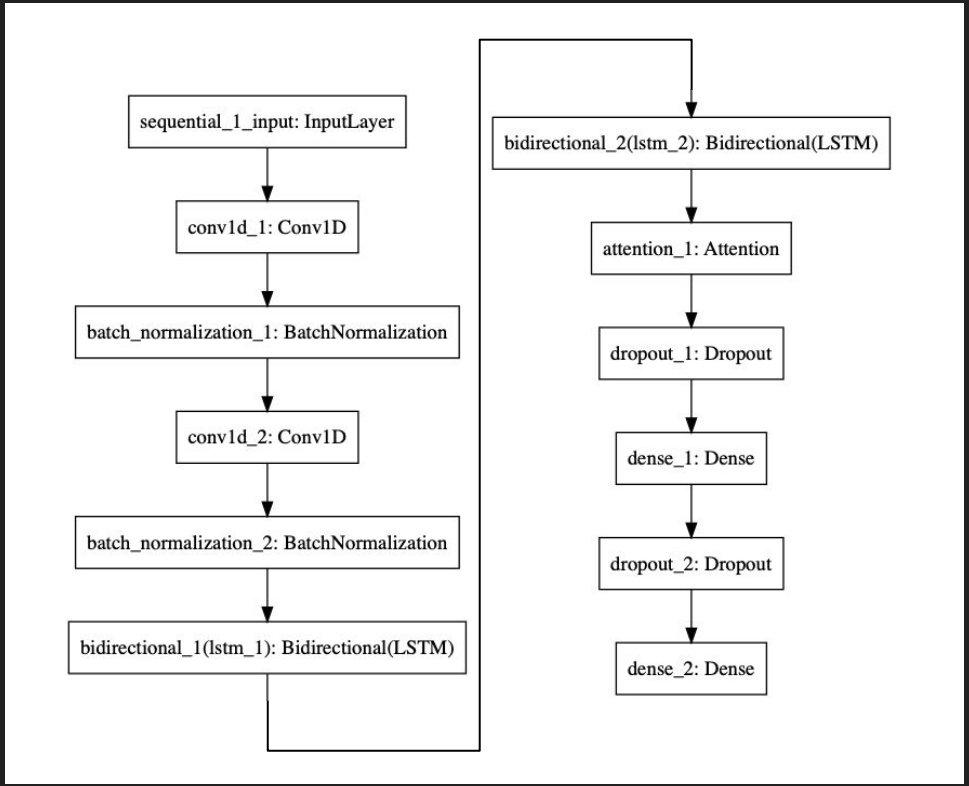
Precision / Recall
by Class (number of modes)

Recall

Precision

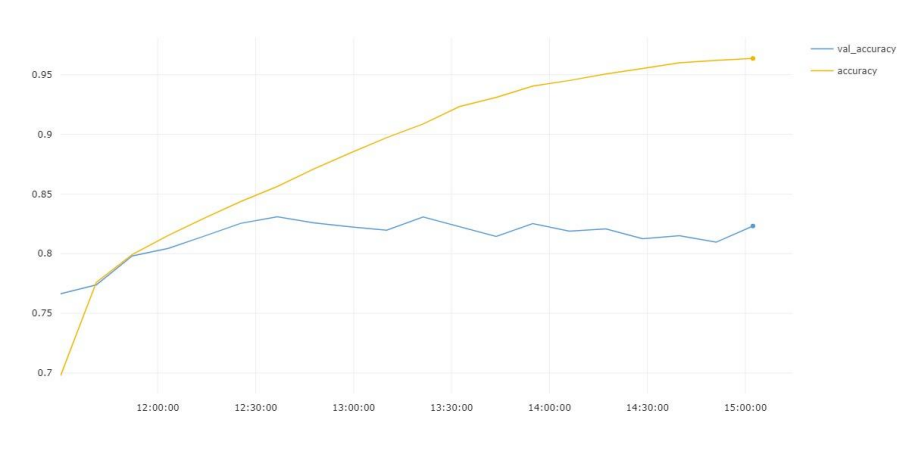
1D CNN + LSTM Model

- Additional two **convolutional layers**
- **1D** Convolutions

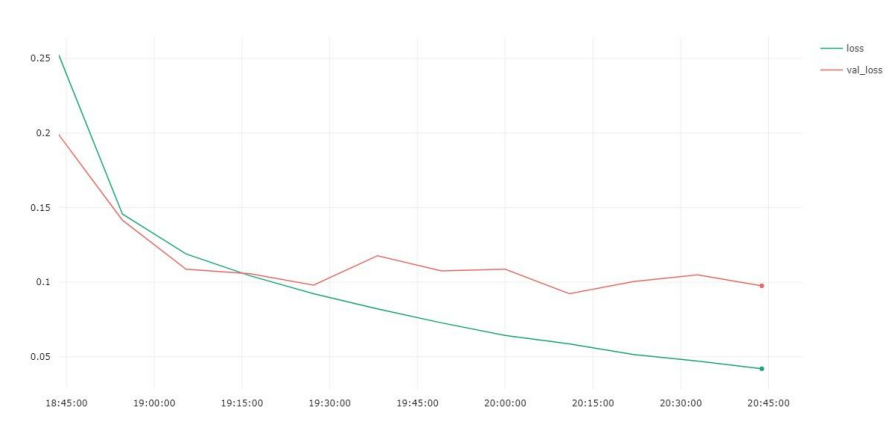


1D CNN + LSTM Model

Training Accuracy
Validation Accuracy



Training Loss
Validation Loss



1D CNN + LSTM Model Results

```
Evaluated with:
50 channels
Omega Shift =  10
Delta Gamma = 0.5
Delta Gamma Shell = 0.5
Max Modes =   4
Max Shell Modes =   5
```

1v2v3v4 Classifier	Predict 1	Predict 2	Predict 3	Predict 4
Actual 1	250	0	0	0
Actual 2	1	241	8	0
Actual 3	0	21	169	60
Actual 4	0	3	79	168

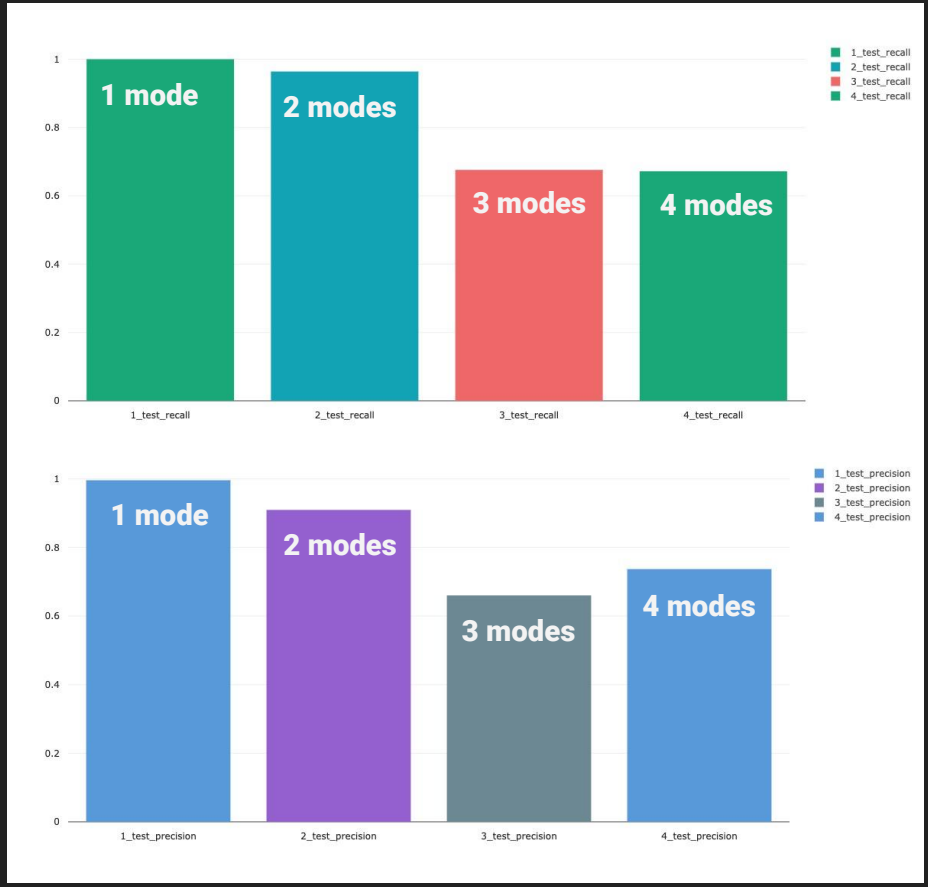
Test Accuracy: 82.3%
Test Loss: 0.659
Test MAE: 0.092
Test MSE: 0.073

1D CNN + LSTM Model Results

Precision / Recall
by Class (number of modes)

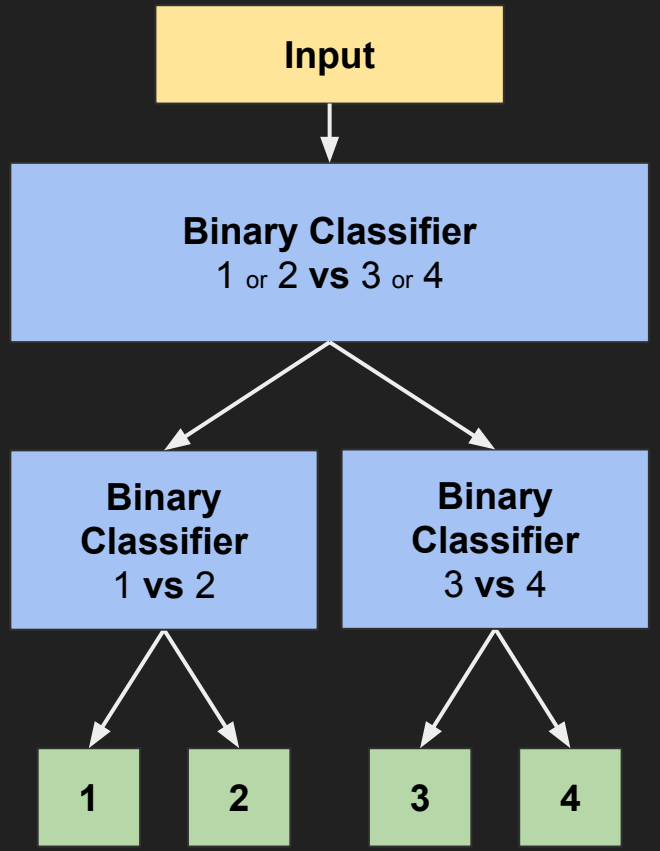
Recall

Precision



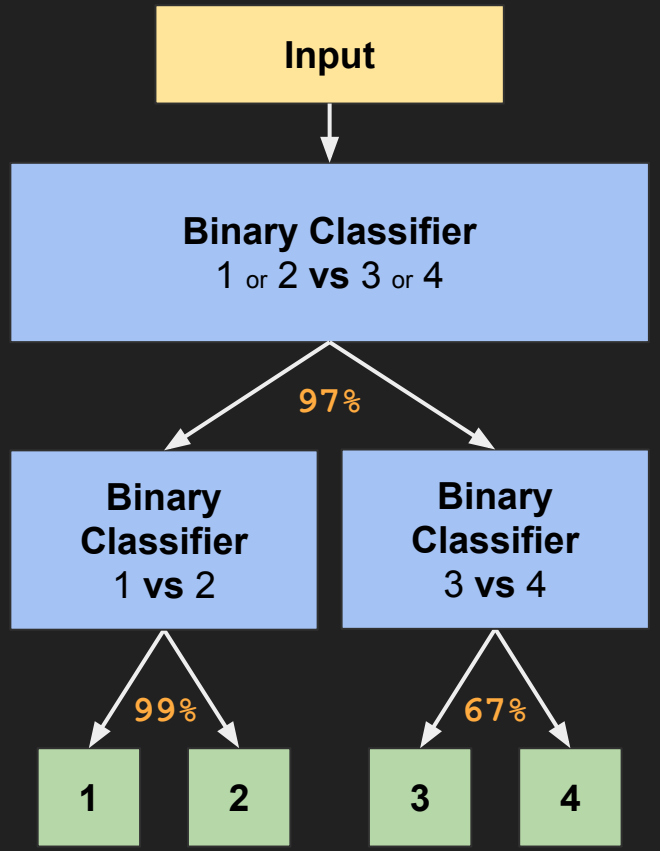
1D CNN + LSTM Binary Ensemble

- Three **binary** classifiers to classify **1-4 modes**
- Same architecture as 1D CNN + LSTM



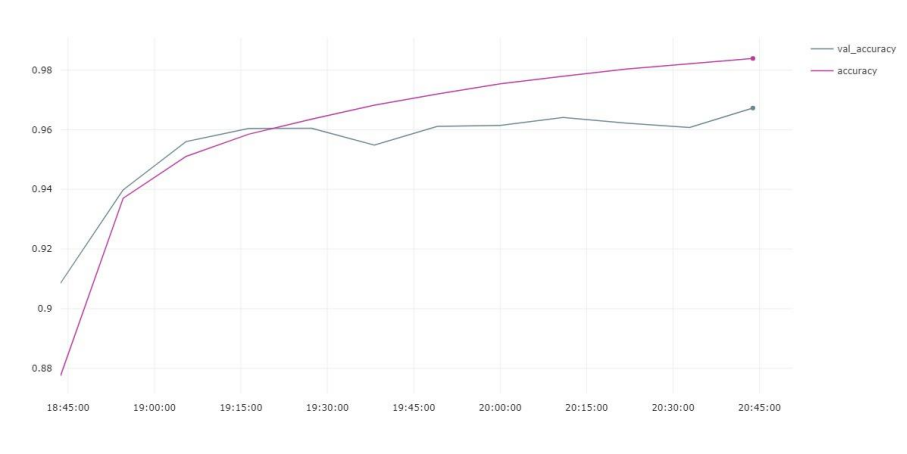
1D CNN + LSTM Binary Ensemble

- Three **binary** classifiers to classify **1-4 modes**
- Same architecture as 1D CNN + LSTM

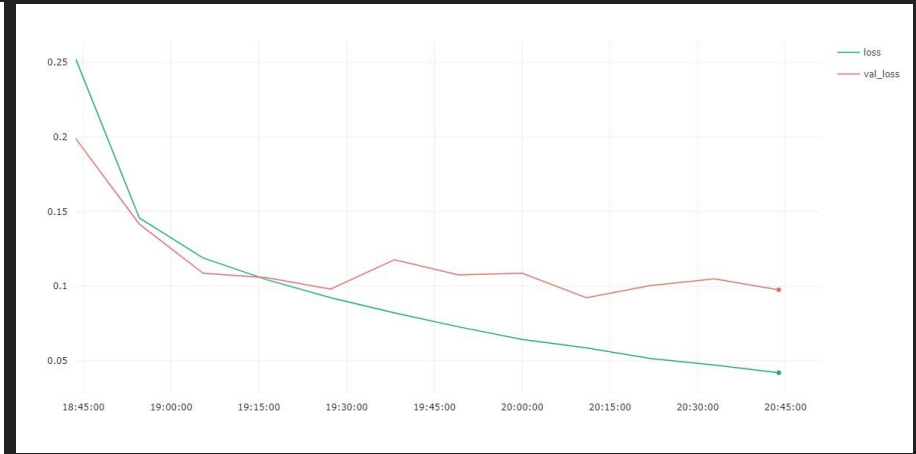


1D CNN + LSTM Binary Ensemble

Training Accuracy 1|2 v 3|4
Validation Accuracy 1|2 v 3|4

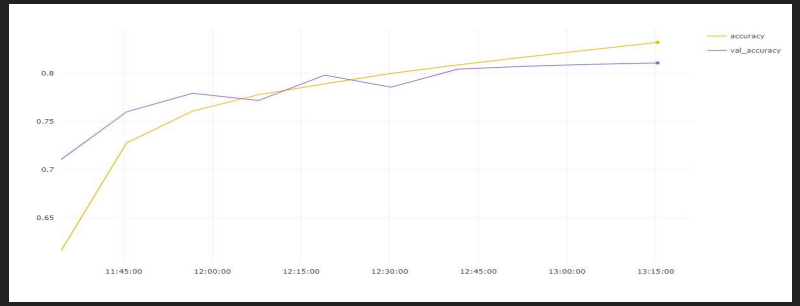


Training Loss 1|2 v 3|4
Validation Loss 1|2 v 3|4

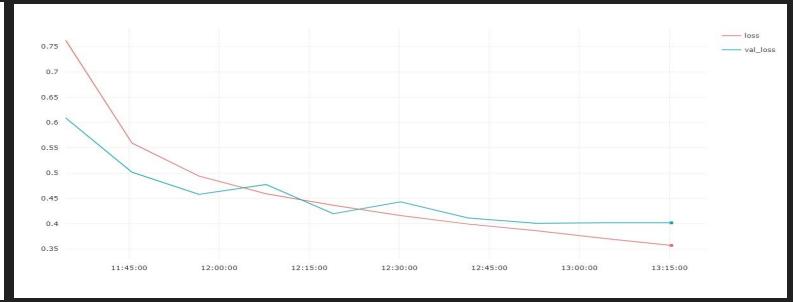


1D CNN + LSTM Binary Ensemble

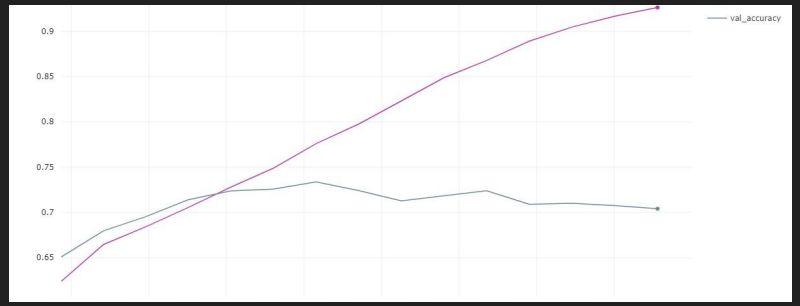
Training Accuracy 1 v 2
Validation Accuracy 1 v 2



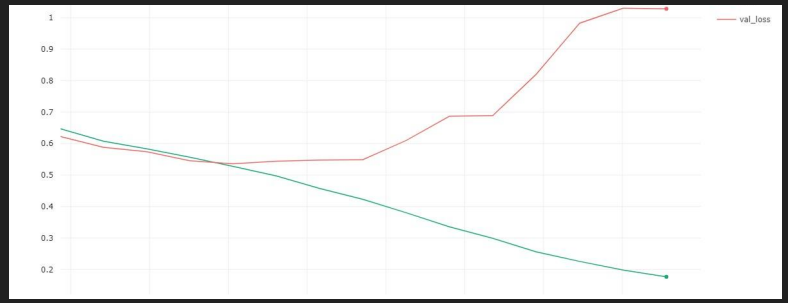
Training Loss 1 v 2
Validation Loss 1 v 2



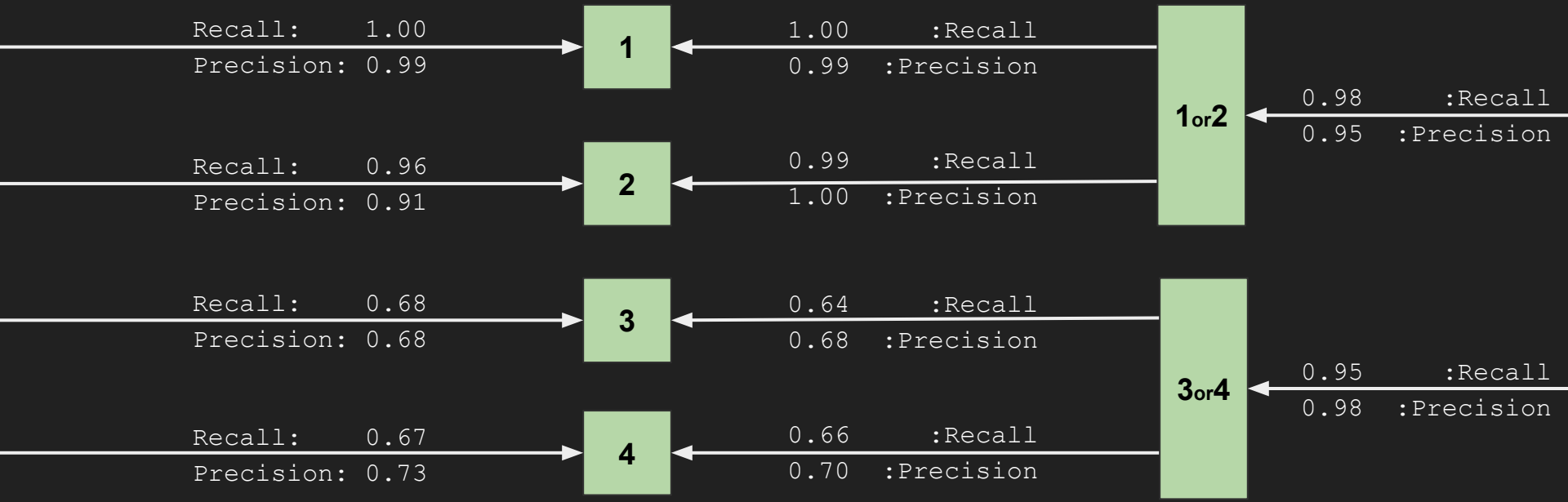
Training Accuracy 3 v 4
Validation Accuracy 3 v 4



Training Loss 3 v 4
Validation Loss 3 v 4



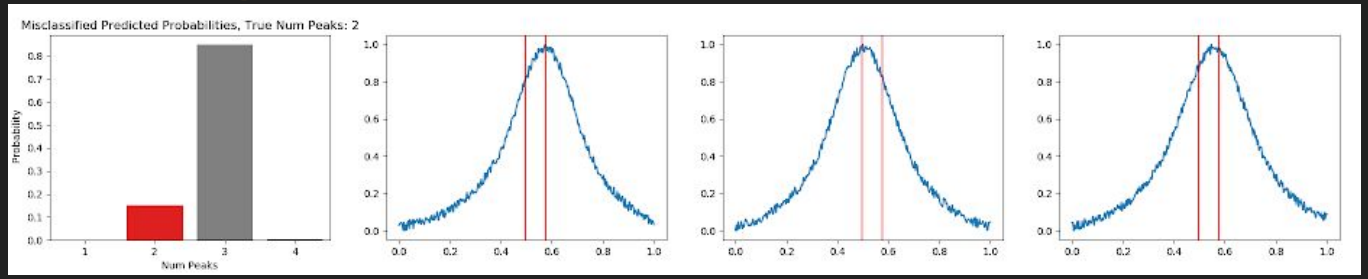
‘All-in-One’ Categorical vs Binary Ensemble Classifier



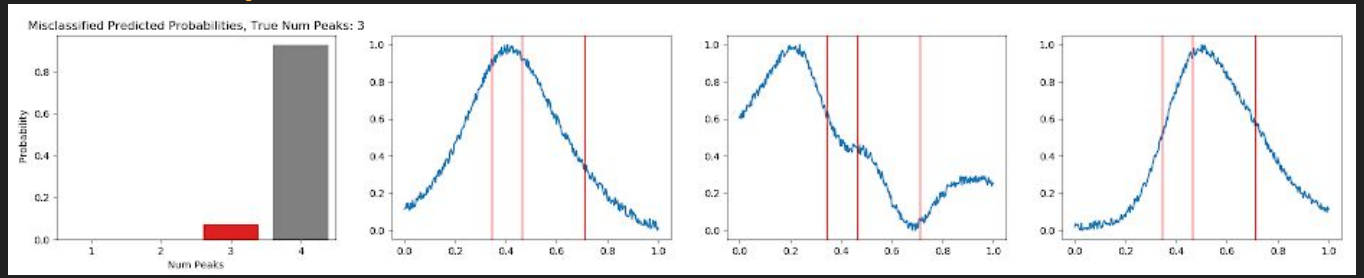
** Scores are calculated independently of parent classifier*

Model Evaluation Visualizations

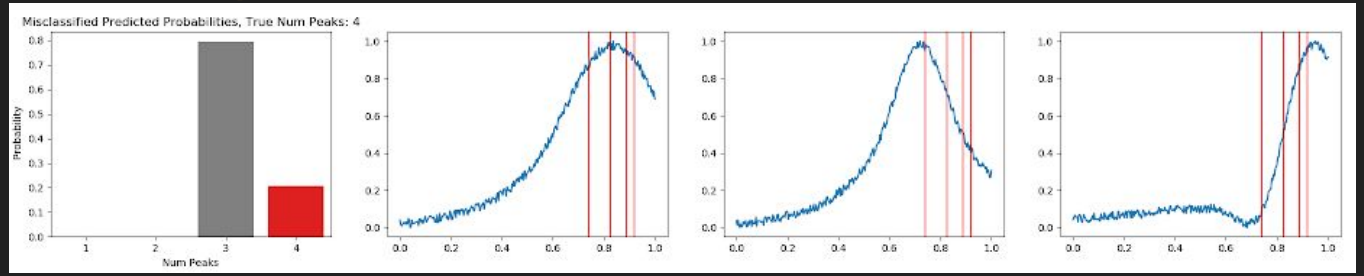
Misclassified spectrum with 2 modes

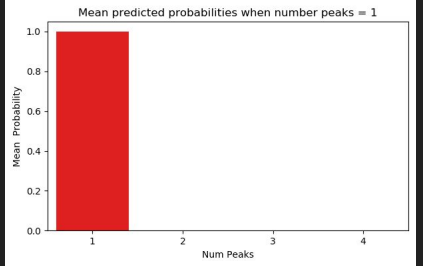


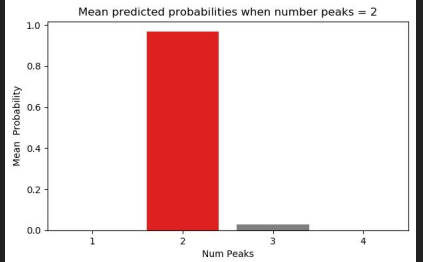
Misclassified spectrum with 3 modes

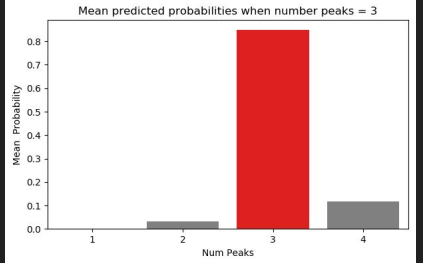


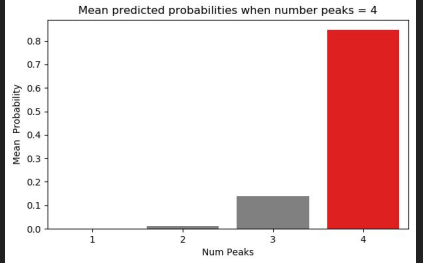
Misclassified spectrum with 4 modes





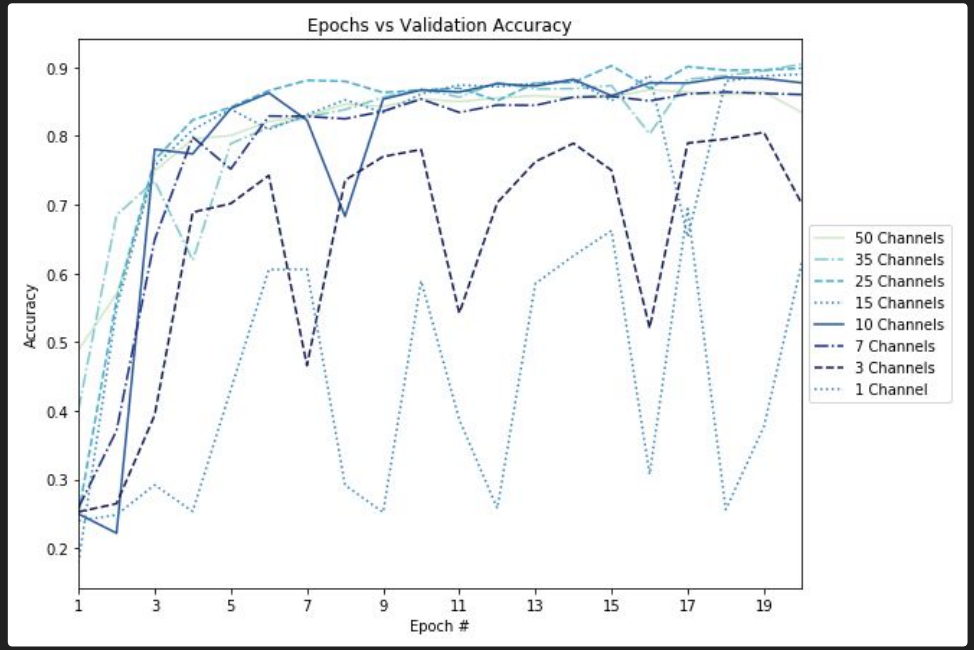




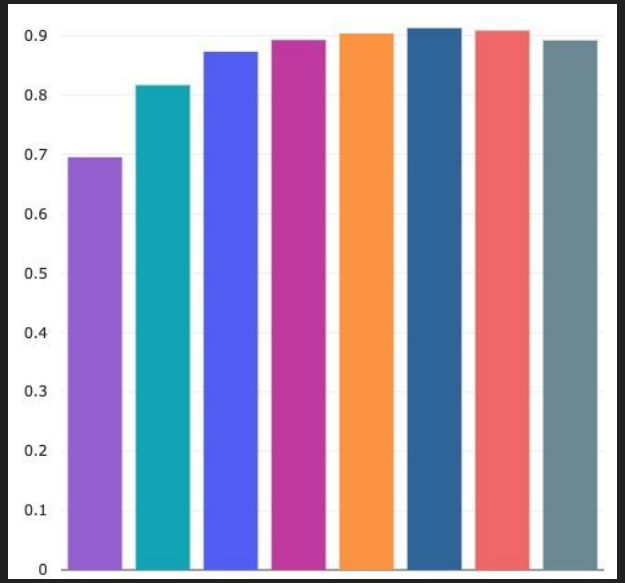


Channel Experiments

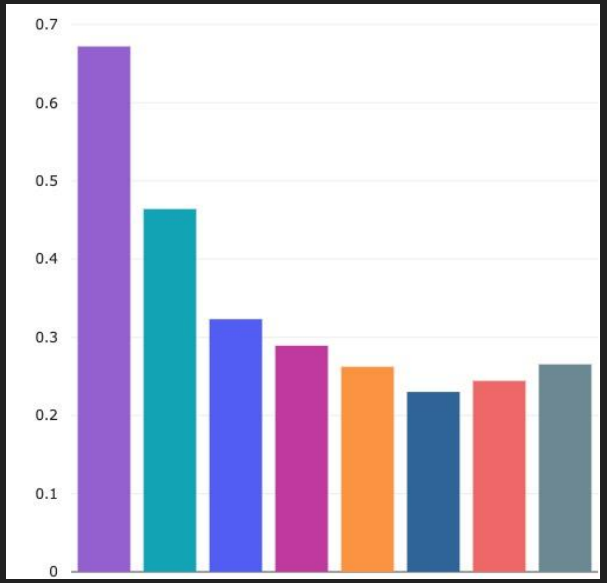
- Big improvement for 1, 3, and 7 channels
- Little difference for 10+ channels



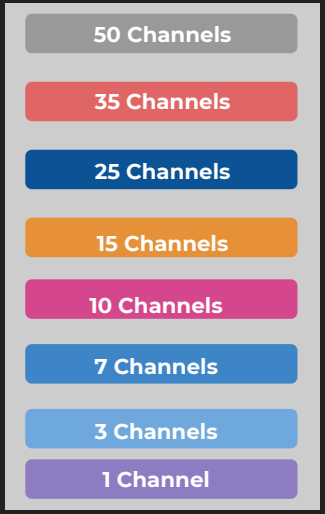
Channel Experiments



Max. Validation Accuracy



Min. Validation Loss



Channel Padding

- Want to be able to evaluate models with **different numbers of channels**
- If a **sample has less** than the number of channels a model was trained on, we pad this sample so that it fits in the model
- Padding strategies:
  - **Reflect:** Pads with the reflection of the vector mirrored on the first and last values of the vector along each axis.
  - **Symmetric:** Pads with the reflection of the vector mirrored along the edge of the array.
  - **Constant**: Pads with a constant float value, in this case “0”.

```
array([[0, 1, 2],
       [3, 4, 5]])
```

Suppose we want to pad this sample so that it has 6 “channels”

Symmetric:

```
[[0, 1, 2],
 [3, 4, 5],
 [3, 4, 5],
 [0, 1, 2],
 [0, 1, 2],
 [3, 4, 5]]
```

Reflect:

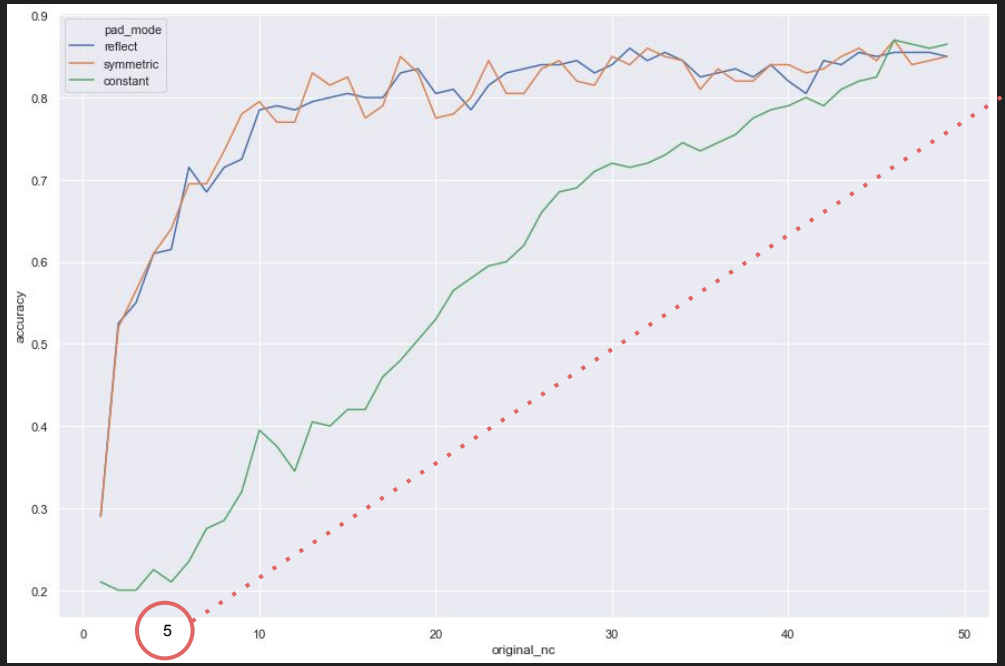
```
[[0, 1, 2],
 [3, 4, 5],
 [0, 1, 2],
 [3, 4, 5],
 [0, 1, 2],
 [3, 4, 5]]
```

Constant:

```
[[0, 1, 2],
 [3, 4, 5],
 [0, 0, 0],
 [0, 0, 0],
 [0, 0, 0],
 [0, 0, 0]]
```

Channel Padding

Accuracy vs. Number of Original Channels



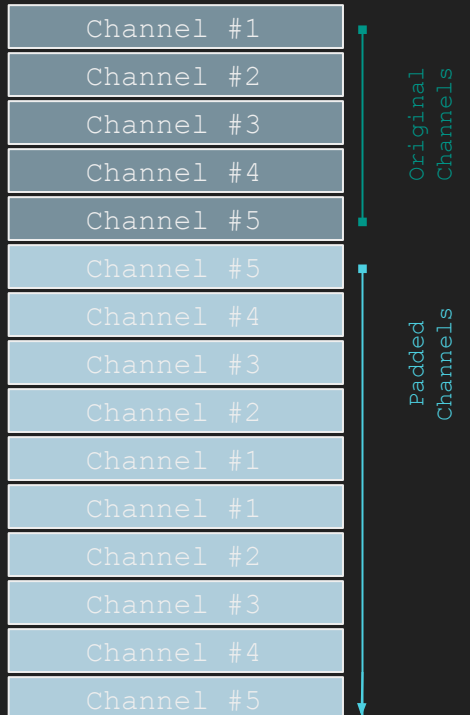
One channel, padded 49 times

No padded channels

For Example…

original_nc = 5 (45 padded)

Symmetric padding



Future Work

- Mode counting vs object detection
- Global Problem
  - Regression based approach
  - Window Slicing
- Padding Strategy vs. Training a Specific Model

Questions?