Weekly Report

Wangwon Lee, 2019/04/05

This week

- **Fine Tuning**
  - Change activation function to tanh
  - Try large filter size
  - He initializer

Next week

- **Fine Tuning**
  - Change tanh model's channel
  - Change activation function to RLelu
  - Filter initializer
- **Visualization**
  - The other model(tanh, long filter)
  - The other label
  - Activation Maximization
  - CAM(Class Activation Map)

Interesting and new finding

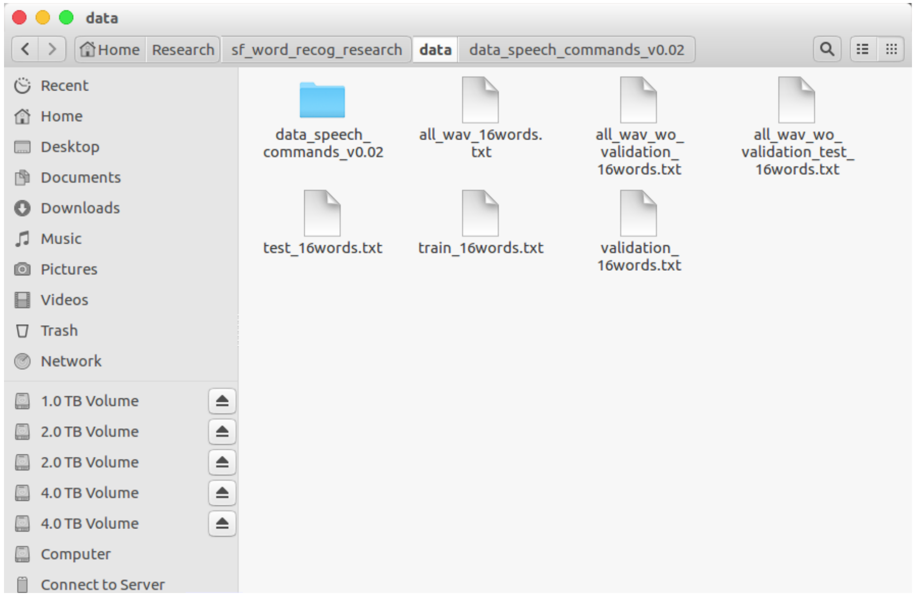
- Fine Tuning
- Visualization

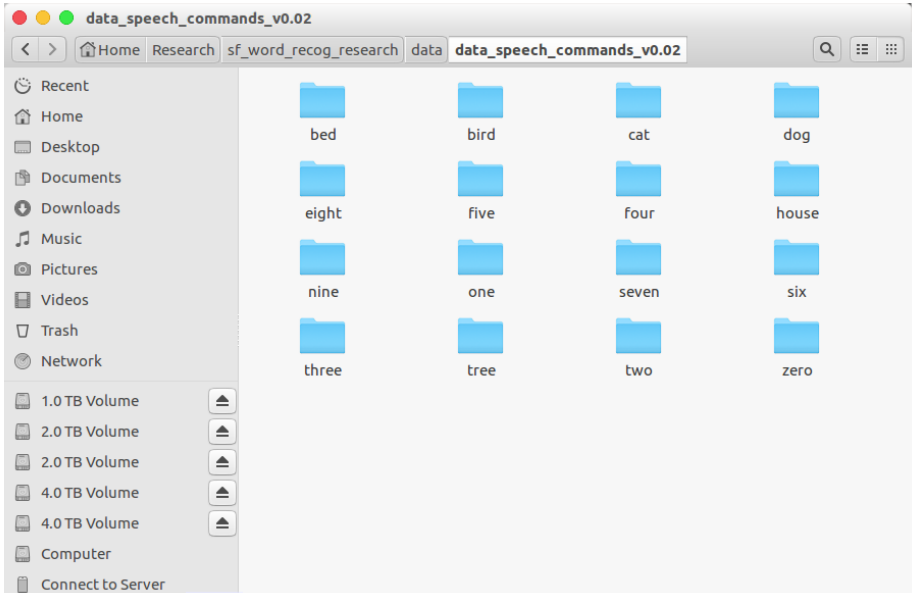
The aim of this month / Discussion

- **The aim of this month:** To investigate about CNN and visualization.

Audio Classification - Previous Work

- Data is low-waveform.
  - sec: 1, sampling rate: 16000, type: float32, channel: mono
- 16 class data.
  - 'zero', 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'bed', 'bird', 'tree', 'cat', 'house', 'dog'
- Train: 40851(≒80%), Validation: 4796(≒10%), Test: 5297(≒10%)

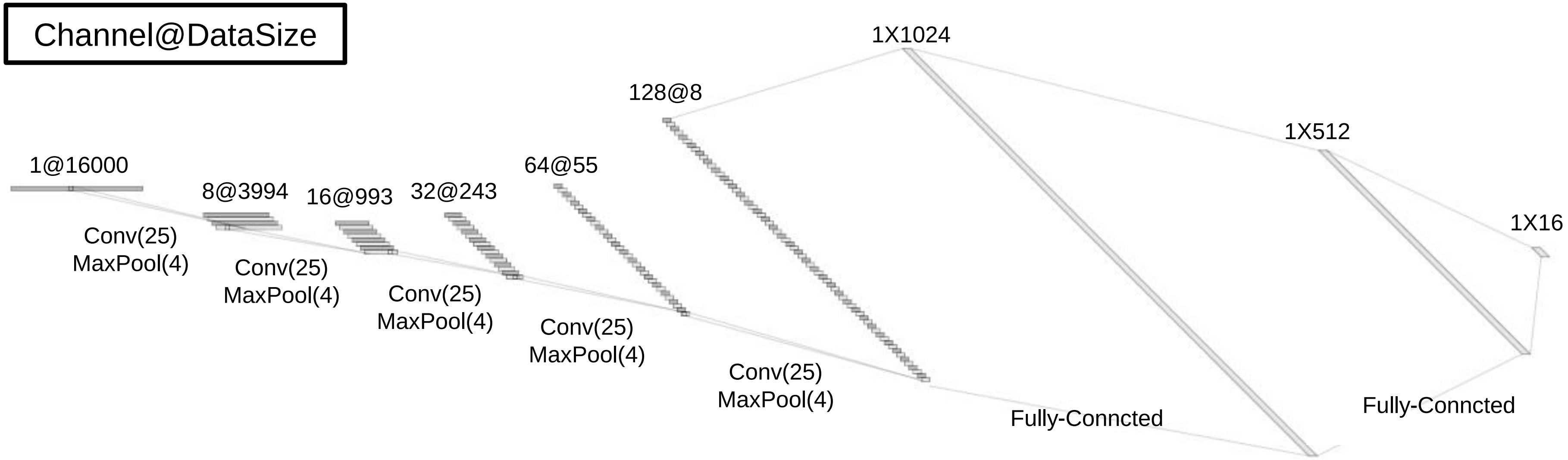




Audio Classification - Previous Work

- For example, ‘5Conv, 2FC’ baseline model’s detail.
- It just flatten 2D model. (5X5 filter->1X25 filter, 2X2 stride->1X4 stride)

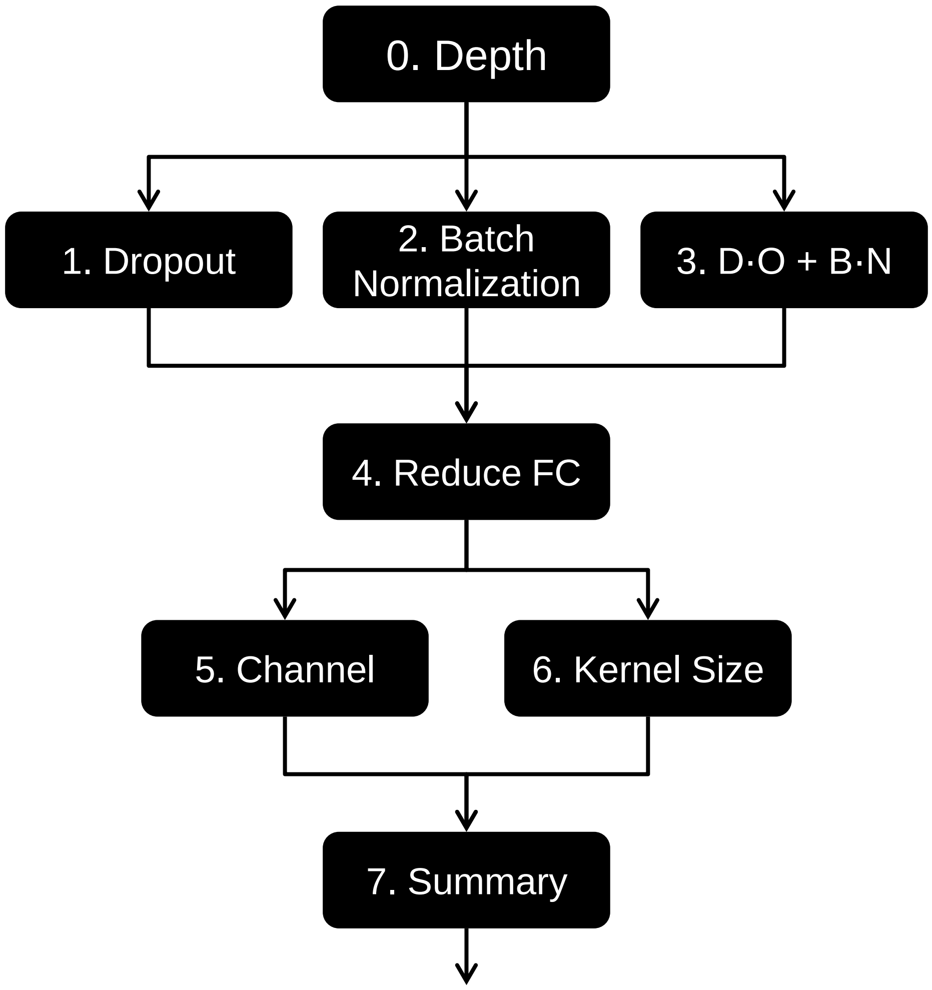
- Input: 16000X1 low waveform.
- Output:1x16 labeled one hot vector. ('zero', …, 'eight', …, 'house', 'dog')
- Loss: cross entropy loss
- Obtimizer: Adam



[Baseline Architecture]

Audio Classification - Previous Work

- Fine tuning task in 1D-CNN



0. Depth

1. Dropout

2. Batch Normalization

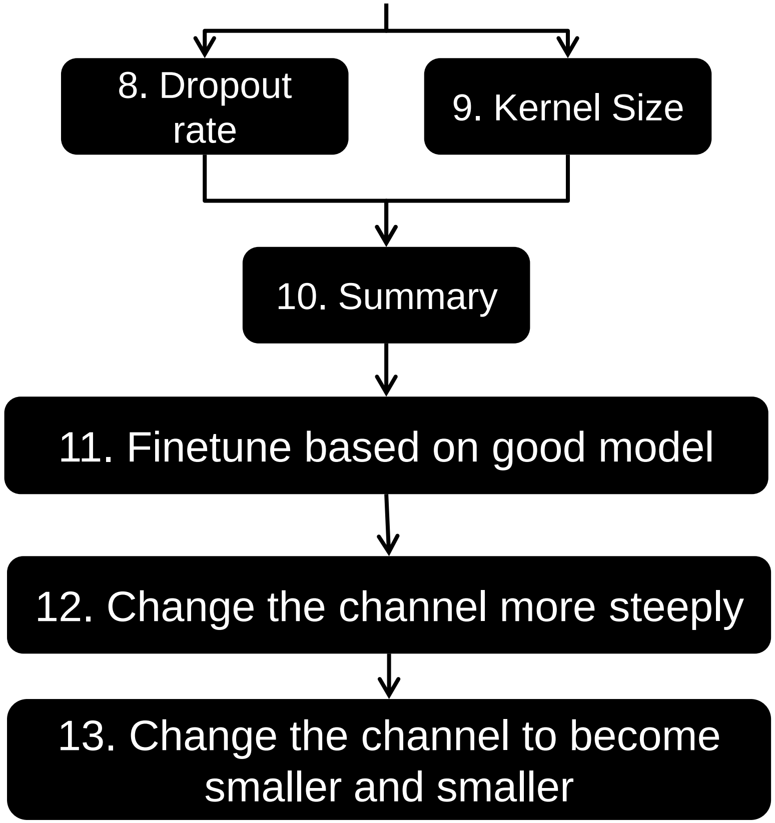
3. D·O + B·N

4. Reduce FC

5. Channel

6. Kernel Size

7. Summary



8. Dropout rate

9. Kernel Size

10. Summary

11. Finetune based on good model

12. Change the channel more steeply

13. Change the channel to become smaller and smaller

Previous Work

Audio Classification - Previous Work

- This is SOTA(State Of The Art) in current research.

	Architecture (i = 0,1,2…)	1D DO(0.5)	1D BN	1D DO+BN	Params
	baseline				
base model	5 Conv(25, $8*2^{i}$), 5 Pool(4), 2 FC	0.9090	0.9072	0.9240	1,855,056
	Accuracy and Number of parameters				
Custom channel 32 DO(0.75)	8 CONV(5, 64)	0.9533	0.9285	0.9391	94,768
Custom channel 64 DO(0.75)	8 CONV(5, 128)	0.9589	X	0.9497	363,600
Custom VGG style DO(0.75)	16 CONV(3, 128) , 8 Pool	0.9620	0.9423	0.9136	470,736
	Only Accuracy				
Custom channel 128 DO(0.25)+BN	9 CONV(5, 512)	0.9535	X	0.9701	2,071,184

Audio Classification - Previous Work

- To validate the model, We can use measure such as precision, recall, f1-score.
- We find out that It does not confuse between ‘three’ and ‘tree’.
- It just confuses ‘tree’ as ‘three’.
- And It also confuse between ‘Bed’ and ‘Bird’.

Actual class

Predict Class

```
Zero   [[369   0   5   1   3   0   1   4   0   1   0   0   1   0   0   0]
One     [  1 347   0   0   4   1   1   0   0   7   2   0   0   1   0   0]
Two     [  8   0 369   2   0   0   0   0   1   0   0   0   2   2   0   0]
Three   [  1   0   2 356   0   1   1   2   6   0   0   0   0   0   0   8]
Fore    [  2   2   1   1 359   2   0   0   0   0   0   0   1   0   0   0]
Five    [  0   6   0   3   4 386   0   2   1   3   0   1   2   0   0   0]
Six     [  0   0   0   3   1   0 366   1   2   0   1   0   0   0   0   0]
Seven   [  3   0   0   0   1   0   2 367   0   0   3   0   0   0   0   0]
Eight   [  1   0   1   2   1   0   0   0 367   0   2   0   1   0   0   1]
Nine    [  0   6   0   0   0   3   0   0   0 364   3   1   0   0   0   0]
Bed     [  1   0   2   0   0   0   0   0   6   0 166   5   2   1   0   0]
Bird    [  0   1   1   1   0   0   2   0   0   2   7 137   0   2   0   0]
Cat     [  0   0   0   0   0   0   0   1   0   0   1   0 161   4   1   0]
Dog     [  0   1   5   0   0   0   0   0   0   1   1   0   2 182   0   0]
House   [  0   0   2   0   0   1   1   0   0   1   0   0   5   1 156   0]
Tree    [  2   0   2  15   0   0   1   0   4   1   0   0   0   0   0 138]]
```

Custom channel 32 DO(0.75) **(Acc: 0.9533)**

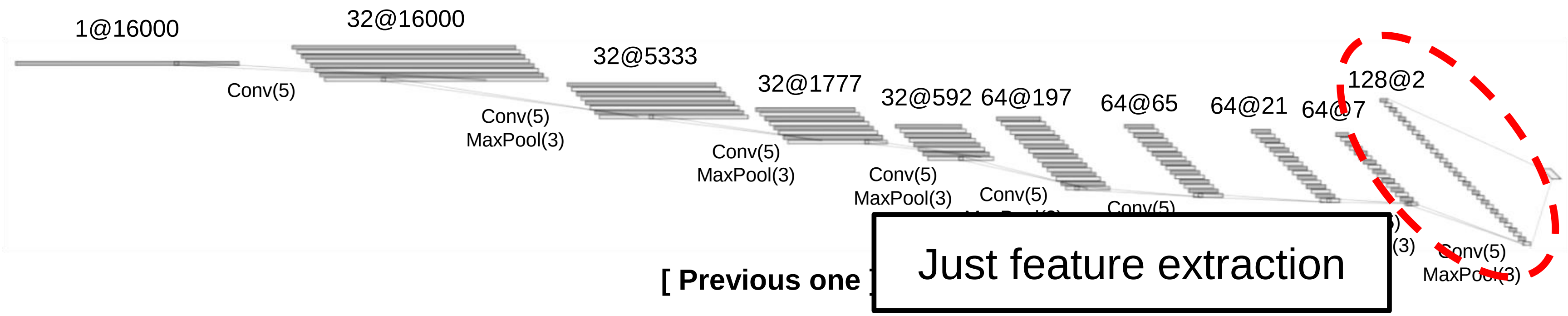
```
              Precision   Recall   F1-score   Support

       zero      0.95      0.96      0.95       385
        one      0.96      0.95      0.95       364
        two      0.95      0.96      0.95       384
      three      0.93      0.94      0.94       377
       four      0.96      0.98      0.97       368
       five      0.98      0.95      0.96       408
        six      0.98      0.98      0.98       374
      seven      0.97      0.98      0.97       376
      eight      0.95      0.98      0.96       376
       nine      0.96      0.97      0.96       377
        bed      0.89      0.91      0.90       183
       bird      0.95      0.90      0.92       153
        cat      0.91      0.96      0.93       168
        dog      0.94      0.95      0.95       192
      house      0.99      0.93      0.96       167
       tree      0.94      0.85      0.89       163

weighted avg     0.95      0.95      0.95      4815
```

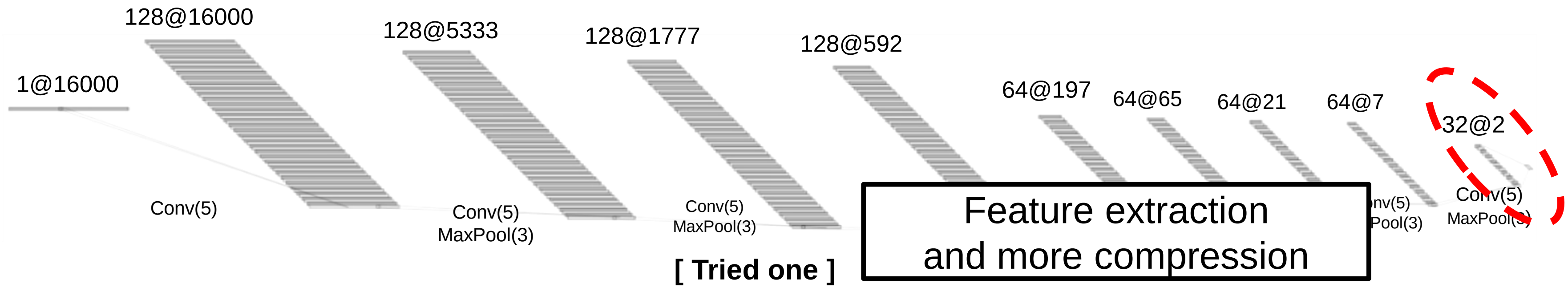
Audio Classification - Previous Work

- I think this try is very meaningful.
- Because this model focus on the feature compression than previous model.
- Like autoencoder. (16000 → 64(=2*32channel))
- This model is better when use other classifiers after extract the feature from CNN.



Just feature extraction

[Previous one]

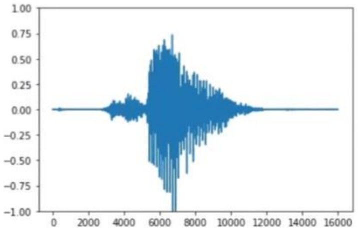


Feature extraction
and more compression

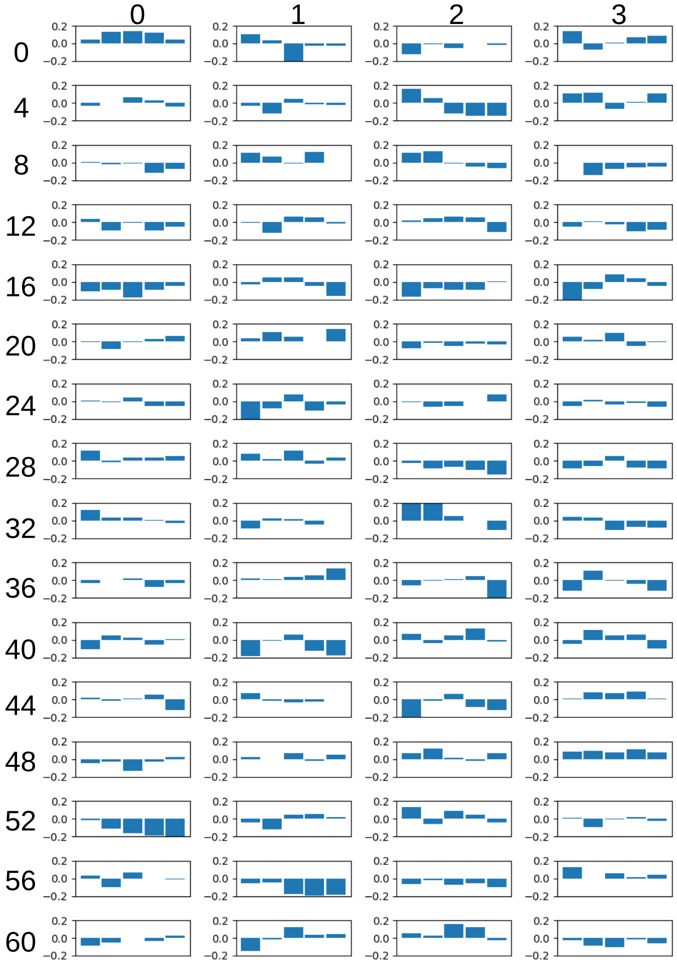
[Tried one]

Audio Classification - Previous Work

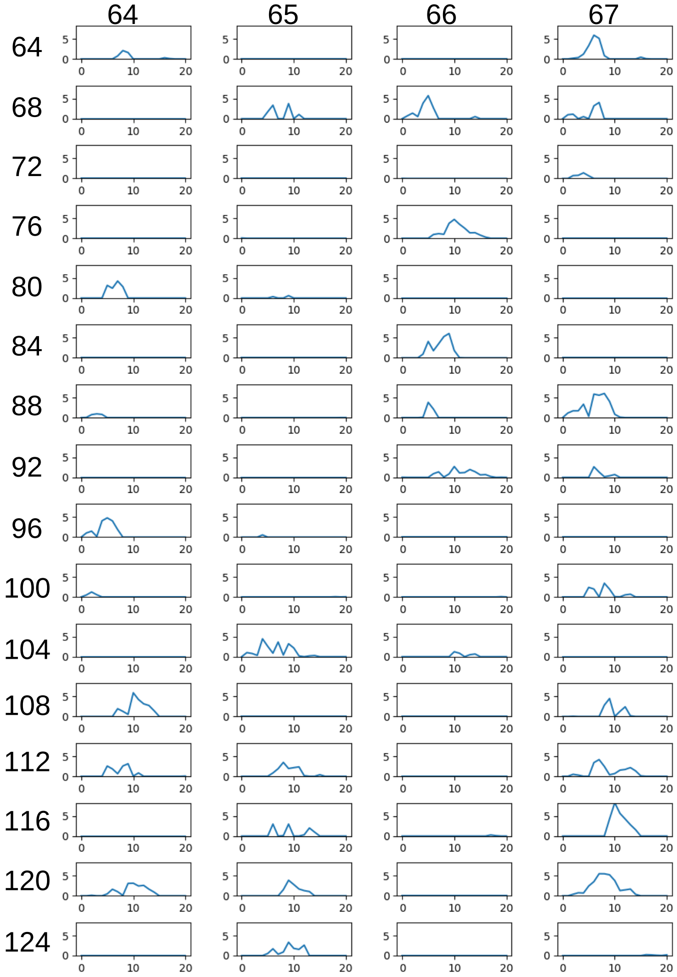
- About half is gone away...
- I don’t know whether each label has a place for feature extraction or it is originally empty



Original input



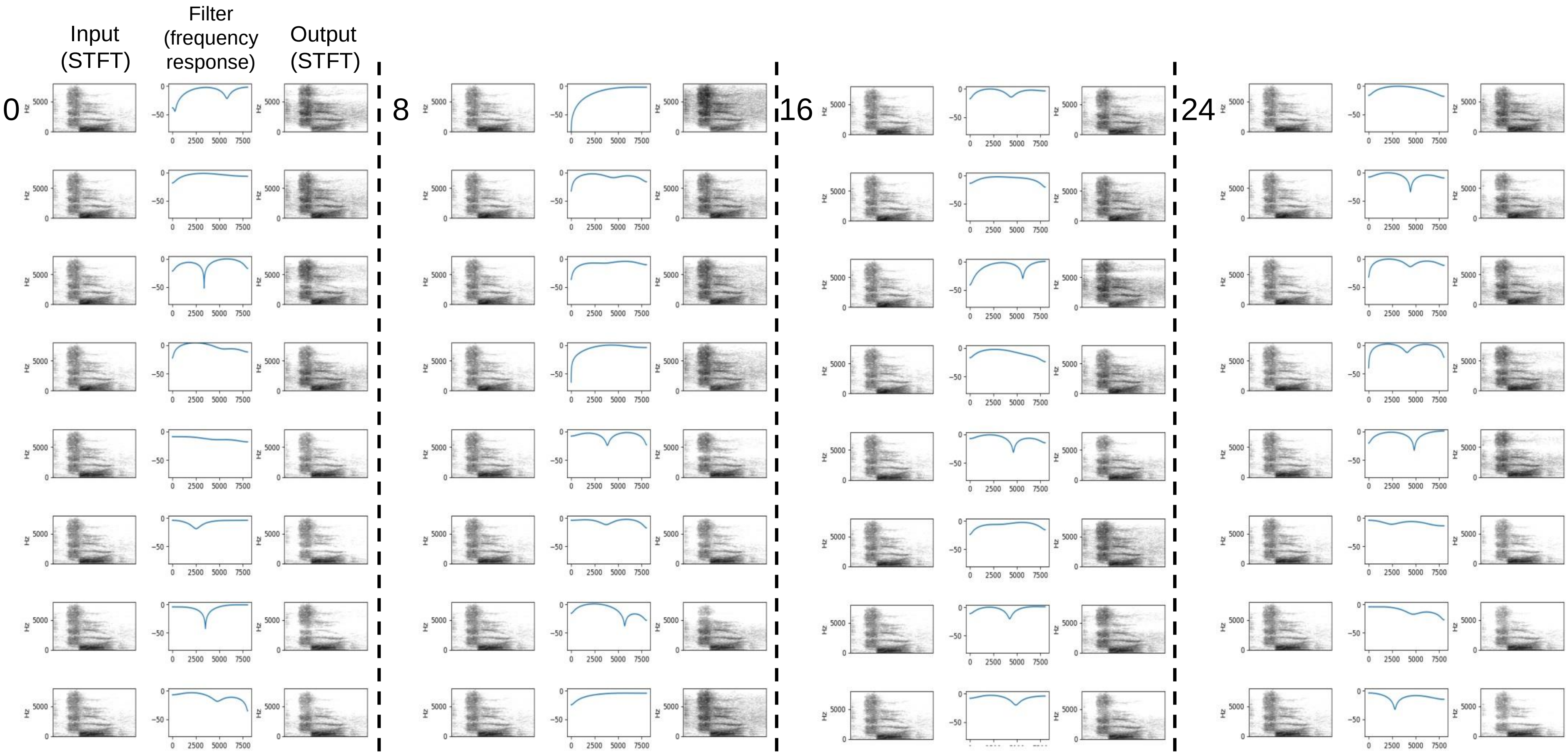
8th Layer - Filter



8th Layer – Activation map

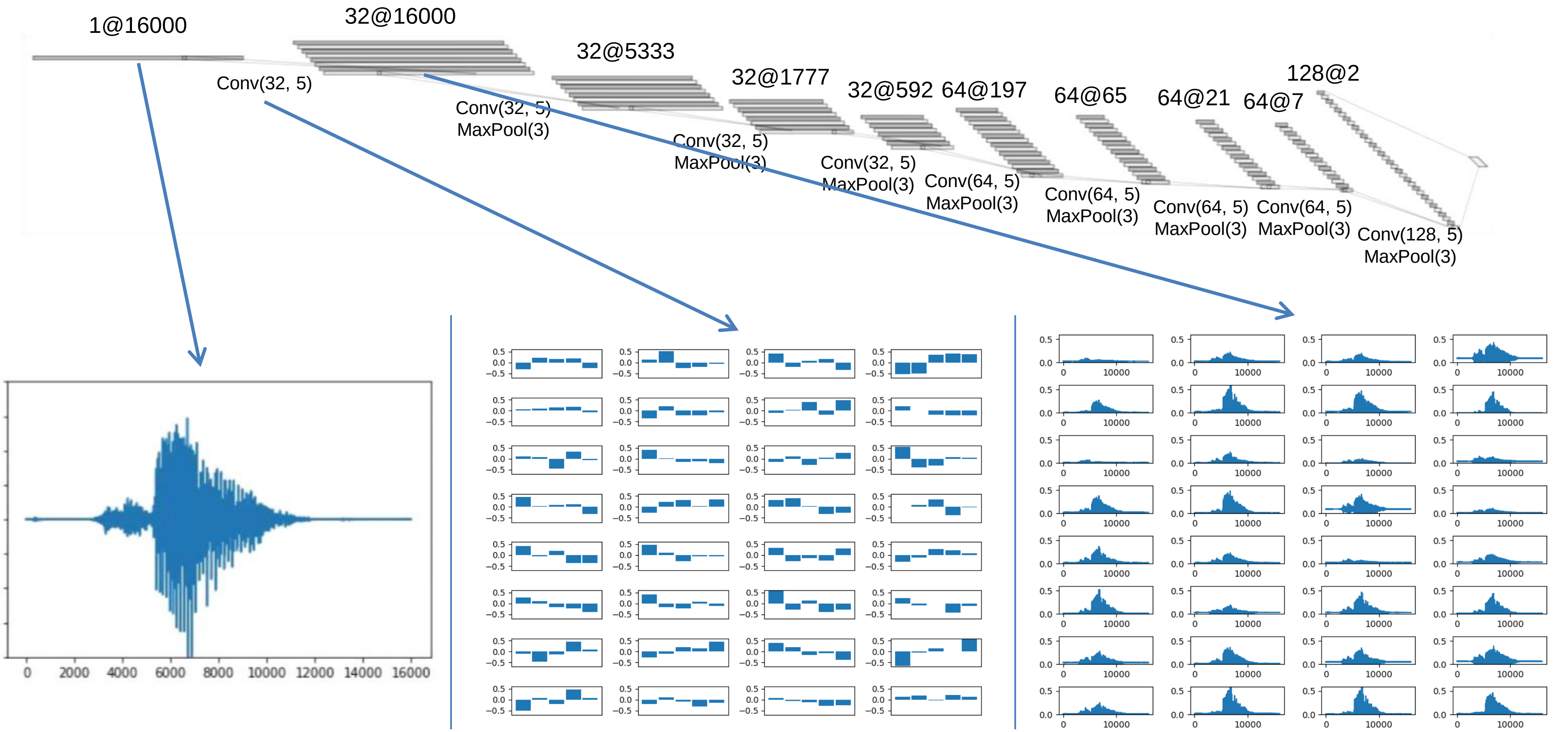
Audio Classification - Previous Work

- So I created spectrogram by Short Time Fourier Transform (STFT).
- Window Size: 512, Stride: 128
- We can see that it is applied to the shape of frequency response.



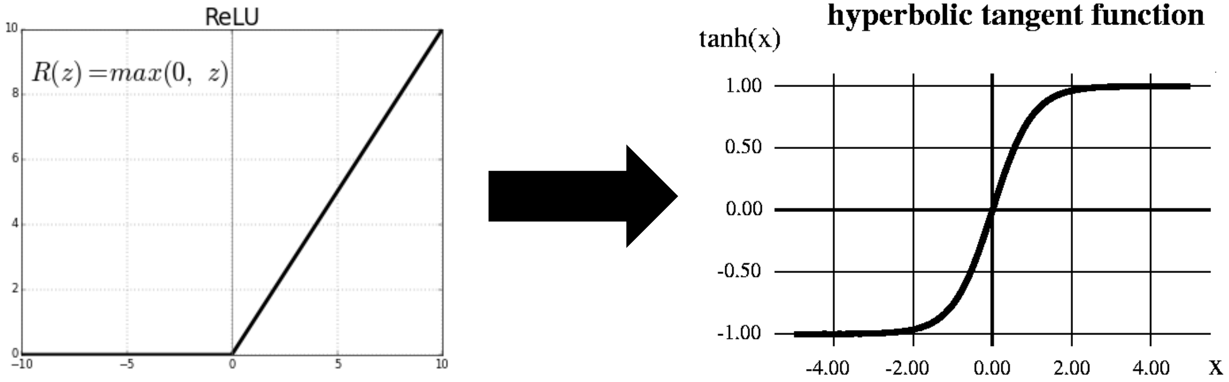
Audio Classification - Previous Work

- Visualize the filter map. (Custom channel 32 DO(0.75) Model)
- Less than zero of the waveform is removed because of ‘Relu’
- Because of this problem, it is difficult to analysis from the point of view of Signal Processing.
- So, I consider to use ‘tanh’ function.



Audio Classification

- Less than zero of the waveform is removed because of ‘Relu’
- So, I tried to change ‘Relu’ to ‘tanh’



- First of all, It is impossible to apply all of my previous task.
  - Dropout
  - Batch Normalization
  - Dropout + Batch Normalization
  - VGG style
  - Droptout rate
  - ~~Channel size~~ (First, I fix the base channel size to 64)
  - ~~etc…~~ (Maybe next time…)
- So, I tried a few task to see the results as soon as possible.

Audio Classification

- First, I applied the ‘custom CH64’ model.

Architecture	DO(0.5)	BN	DO+BN	Params
1 CONV(5, 64)	0.0953	0.0976	0.1111	16,384,400
2 CONV(5, 64)	0.3421	0.2517	0.2675	5,481,936
3 CONV(5, 64)	0.5001	0.5076	0.5350	1,861,136
4 CONV(5, 64)	0.6370	0.5518	0.6422	668,240
5 CONV(5, 128)	0.7423	0.6066	0.7082	506,576
6 CONV(5, 128)	0.8455	0.7583	0.7859	318,288
7 CONV(5, 128)	0.9192	0.8827	0.8687	310,224
8 CONV(5, 128)	0.9433	0.9269	0.9252	363,600
9 CONV(5, 256)	0.9477	0.9377	0.9479	521,552

Audio Classification

- Second, I tune the dropout’s rate
- As with previous results, high dropout rate model has a good performance.
- The high dropout rate contribute to the generalization of the model.

Architecture	DO(0.25)	DO(0.25)+BN	DO(0.75)	DO(0.75)+BN
1 CONV(5, 64)	0.1051	0.1020	0.1040	0.1011
2 CONV(5, 64)	0.3429	0.3246	0.3524	0.2569
3 CONV(5, 64)	0.4987	0.4295	0.5088	0.5624
4 CONV(5, 64)	0.6278	0.6079	0.7148	0.6550
5 CONV(5, 128)	0.7310	0.6947	0.7707	0.7146
6 CONV(5, 128)	0.8582	0.8150	0.8417	0.7763
7 CONV(5, 128)	0.9215	0.8922	0.9111	0.8617
8 CONV(5, 128)	0.9441	0.9169	0.9485	0.9229
9 CONV(5, 256)	0.9472	0.9439	0.9543	0.9441

Audio Classification

- 3th, I tried VGG style

Architecture	DO(0.5)	BN	DO+BN	Params
2 CONV(3, 64), 1 Pool	0.1001	0.1047	0.1032	16,396,624
4 CONV(3, 64) , 2 Pool	0.3421	0.2692	0.2721	5,498,320
6 CONV(3, 64) ,3 Pool	0.5333	0.3192	0.3927	1,881,680
8 CONV(3, 64) , 4 Pool	0.6908	0.5333	0.5952	692,944
10 CONV(3, 128) , 5 Pool	0.7659	0.5034	0.7481	564,176
12 CONV(3, 128) , 6 Pool	0.8627	0.7570	0.7666	392,400
14 CONV(3, 128) , 7 Pool	0.9246	0.8839	0.8681	400,848
16 CONV(3, 128) , 8 Pool	0.9487	0.9406	0.9364	470,736
18 CONV(3, 256) , 9 Pool	0.9516	0.9464	0.9439	760,016

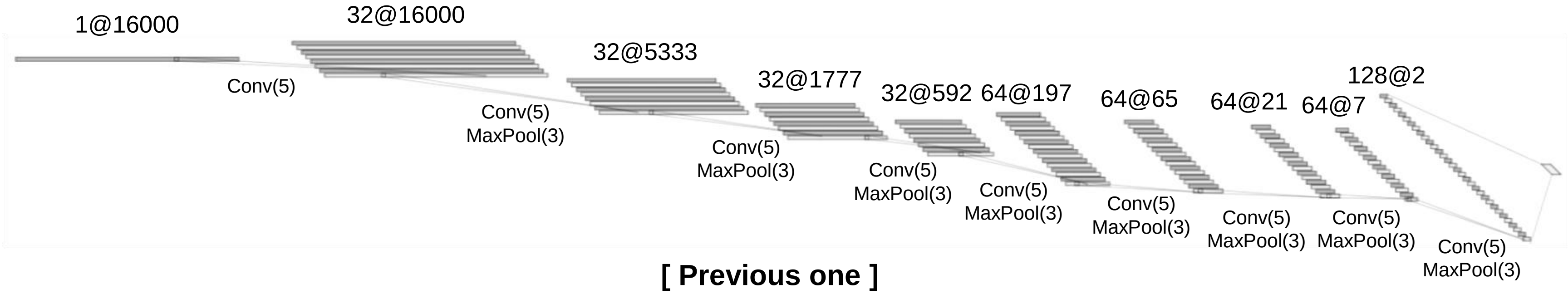
Audio Classification

- 4th, I tune the dropout's rate
- In this result, 'custom' model is better than 'VGG like' model
- Compared to Relu, overall performance dropped by 1-2%.

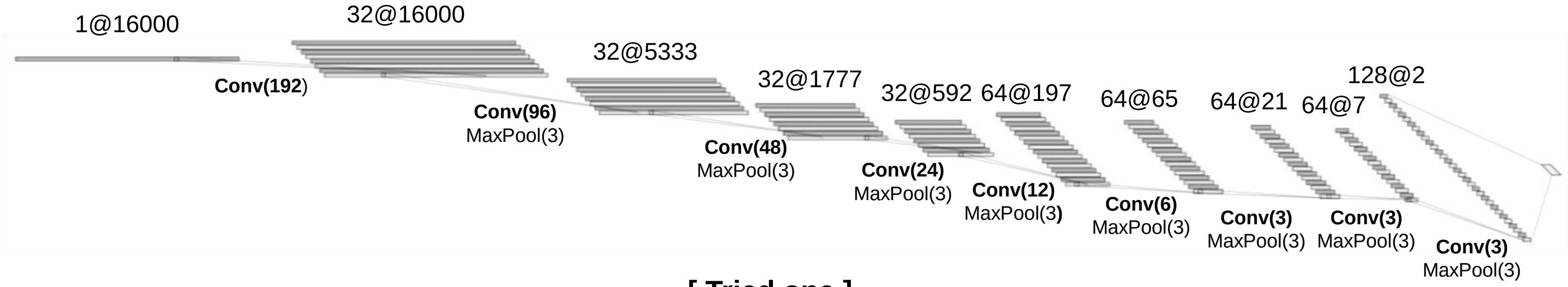
Architecture	DO(0.25)	DO(0.25)+BN	DO(0.75)	DO(0.75)+BN
2 CONV(3, 64), 1 Pool	X	X	X	X
4 CONV(3, 64) , 2 Pool	X	X	X	X
6 CONV(3, 64) ,3 Pool	0.5157	0.4347	0.5736	0.5753
8 CONV(3, 64) , 4 Pool	0.6474	0.6565	0.7410	0.7279
10 CONV(3, 128) , 5 Pool	0.7564	0.7248	0.7848	0.7607
12 CONV(3, 128) , 6 Pool	0.8677	0.7780	0.8696	0.7896
14 CONV(3, 128) , 7 Pool	0.9277	0.9001	0.9304	0.8924
16 CONV(3, 128) , 8 Pool	0.9487	0.9304	0.9404	0.9269
18 CONV(3, 256) , 9 Pool	0.9475	0.9489	0.9522	0.9448

Audio Classification

- I tune the filter size more bigger.
- Filter size: 192 → 96 → 48 → 24 → 12 → 6 → 3 → …



[Previous one]



[Tried one]

Audio Classification

- I didn’t try early case because the results were probably not good.
- Start channel size: 32

Architecture	DO(0.5)	BN	DO+BN	Params
1 CONV(192, 32)	X	X	X	X
2 CONV(96, 32)	X	X	X	X
3 CONV(48, 32)	0.7765	0.5678	0.7506	1,063,536
4 CONV(24, 32)	0.8249	0.7329	0.8424	481,424
5 CONV(12, 64)	0.8758	0.8042	0.8856	404,688
6 CONV(6, 64)	0.9213	0.8785	0.9302	294,160
7 CONV(3, 64)	0.9340	0.9115	0.9385	261,456
8 CONV(3, 64)	0.9483	0.9292	0.9468	259,472
9 CONV(3, 128)	0.9445	0.9466	0.9398	281,104

Audio Classification

- Start channel size: 64
- If filter size is more bigger, model become more heavy.
- It take a lot of time… So, I will prepare the other case until the next time.

Architecture	DO(0.5)	BN	DO+BN	Params
1 CONV(192, 64)	X	X	X	X
2 CONV(96, 64)	X	X	X	X
3 CONV(48, 64)	0.7886	0.5346	0.7117	2,421,968
4 CONV(24, 64)	0.8482	0.7423	0.8656	1,306,896
5 CONV(12, 128)	0.8760	0.8258	0.9094	1,202,576
6 CONV(6, 128)	0.9225	0.9088	0.9202	1,030,672
7 CONV(3, 128)	0.9350	0.9321	0.9524	989,840
8 CONV(3, 128)	0.9489	0.9472	0.9535	1,010,448
9 CONV(3, 256)	0.9479	0.9516	0.9570	1,102,864

Audio Classification

- In last lab seminar, we know that ‘he_uniform initializer’ has good performance
- So, I found the paper of ‘he initializer’
- ‘He initializer’ is improvement of ‘xavier initializer’.
- They said, If we use ‘relu’ function, it is more better than ‘xavier initializer’.

$$\sigma = \frac{1}{\sqrt{n_{\text{in}}}} \quad \Rightarrow \quad \sigma = \frac{\sqrt{2}}{\sqrt{n_{\text{in}}}}$$

< Xavier > < He >

n_{in}: Number of neurons feeding into given neuron

Xavier:

Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010.

He:

He, Kaiming, et al. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." Proceedings of the IEEE international conference on computer vision. 2015.

- In tensorflow, Conv1D ‘s default initializer is ‘xavier initializer’.
- So I tried ‘he initializer’ to my model.

Audio Classification

- Compare to previous result.
- The Most of case is improved a little.
- I think ‘He initializer’ is little better than ‘Xavier initializer’.

Architecture	DO(0.5)	BN	DO+BN		DO(0.5)	BN	DO+BN
1 CONV(5, 64)	X	X	X		X	X	X
2 CONV(5, 64)	X	X	X		X	X	X
3 CONV(5, 64)	0.5844	0.4893	0.5726		0.5904	0.4773	0.6033
4 CONV(5, 64)	0.7128	0.6291	0.7333		0.7221	0.6145	0.7329
5 CONV(5, 128)	0.7732	0.7022	0.8073		0.7859	0.6872	0.8019
6 CONV(5, 128)	0.8901	0.8233	0.8818	➡	0.8928	0.8087	0.8802
7 CONV(5, 128)	0.9400	0.9061	0.9302		0.9433	0.9126	0.9252
8 CONV(5, 128)	0.9560	0.9508	0.9458		0.9587	0.9383	0.9531
9 CONV(5, 256)	0.9537	0.9464	0.9470		0.9541	0.9524	0.9529

< custom model > < Xavier > < He >

Audio Classification

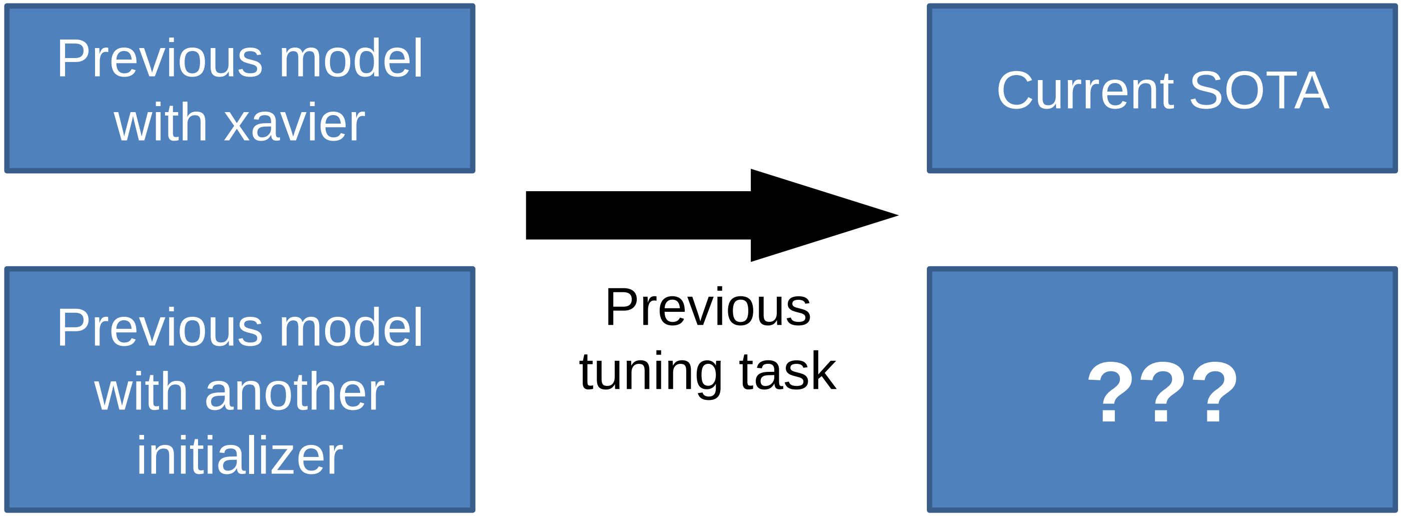
- Compare to previous result.
- It works well for a few cases, but in some cases it is not effective.
- Is it effective when the filter size is large?

Architecture	DO(0.5)	BN	DO+BN		DO(0.5)	BN	DO+BN
2 CONV(3, 64), 1 Pool	X	X	X		X	X	X
4 CONV(3, 64) , 2 Pool	X	X	X		X	X	X
6 CONV(3, 64) ,3 Pool	0.5844	0.4893	0.5726	→	0.5680	0.5047	0.5755
8 CONV(3, 64) , 4 Pool	0.7128	0.6291	0.7333		0.6793	0.6442	0.7080
10 CONV(3, 128) , 5 Pool	0.7732	0.7022	0.8073		0.7759	0.6947	0.8027
12 CONV(3, 128) , 6 Pool	0.8901	0.8233	0.8818		0.8831	0.8368	0.8939
14 CONV(3, 128) , 7 Pool	0.9400	0.9061	0.9302		0.9458	0.9178	0.9418
16 CONV(3, 128) , 8 Pool	0.9581	0.9508	0.9458		0.9593	0.9398	0.9543
18 CONV(3, 256) , 9 Pool	0.9537	0.9464	0.9470		0.9533	0.9387	0.9506

< VGG like model > < Xavier > < He >

Audio Classification

- There is a more initializer (ex. xavier_normal, he_normal, etc)
- What if I apply another initializer at the same previous task?
- Because ‘he initializer’ shown such better performance, I expect about it.



Previous model with xavier

Previous model with another initializer

Previous tuning task

Current SOTA

???

Any Question?

Thank you