Weekly Report

Wangwon Lee, 2019/03/02

This week

- **Audio Classification**
  - Previous work
  - 1D-CNN
  - Experiments
  - Fine tuning process

Next week

- **Audio Classification**
  - To investigate 1D model more specific
  - 1D, 2D CNN visualization

Interesting and new finding

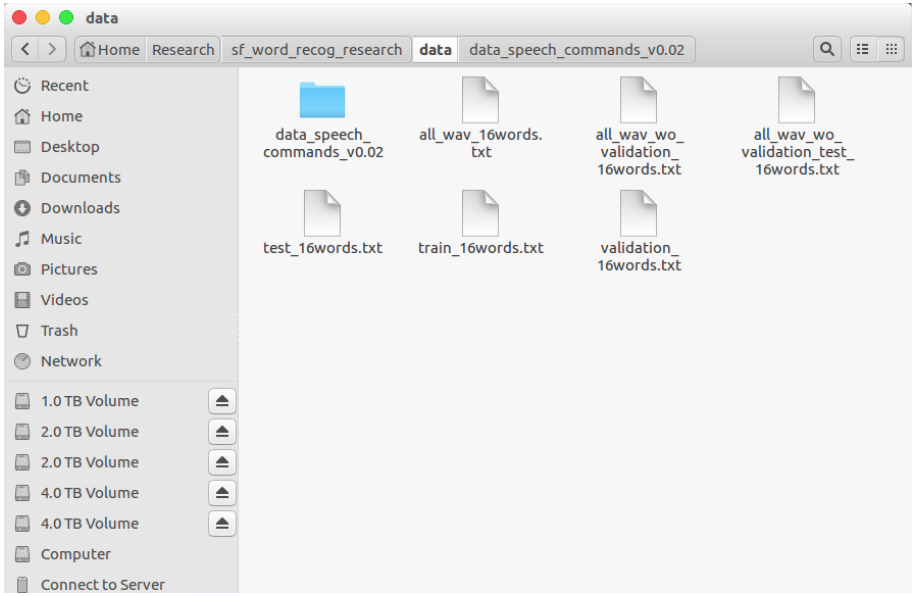
- Fine Tuning

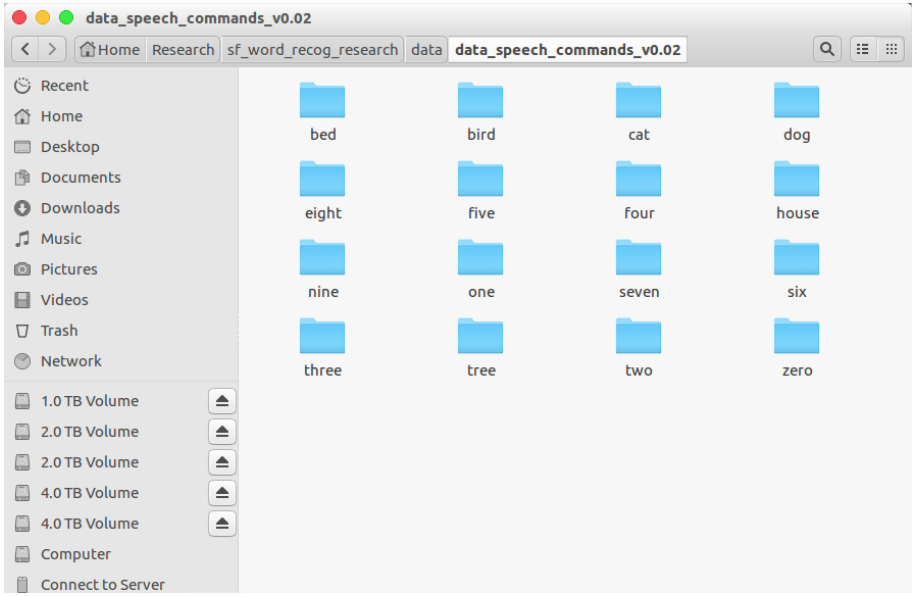
The aim of this month / Discussion

- **The aim of this month:** To study the brain and GLM, To investigate about CNN.

Audio Classification

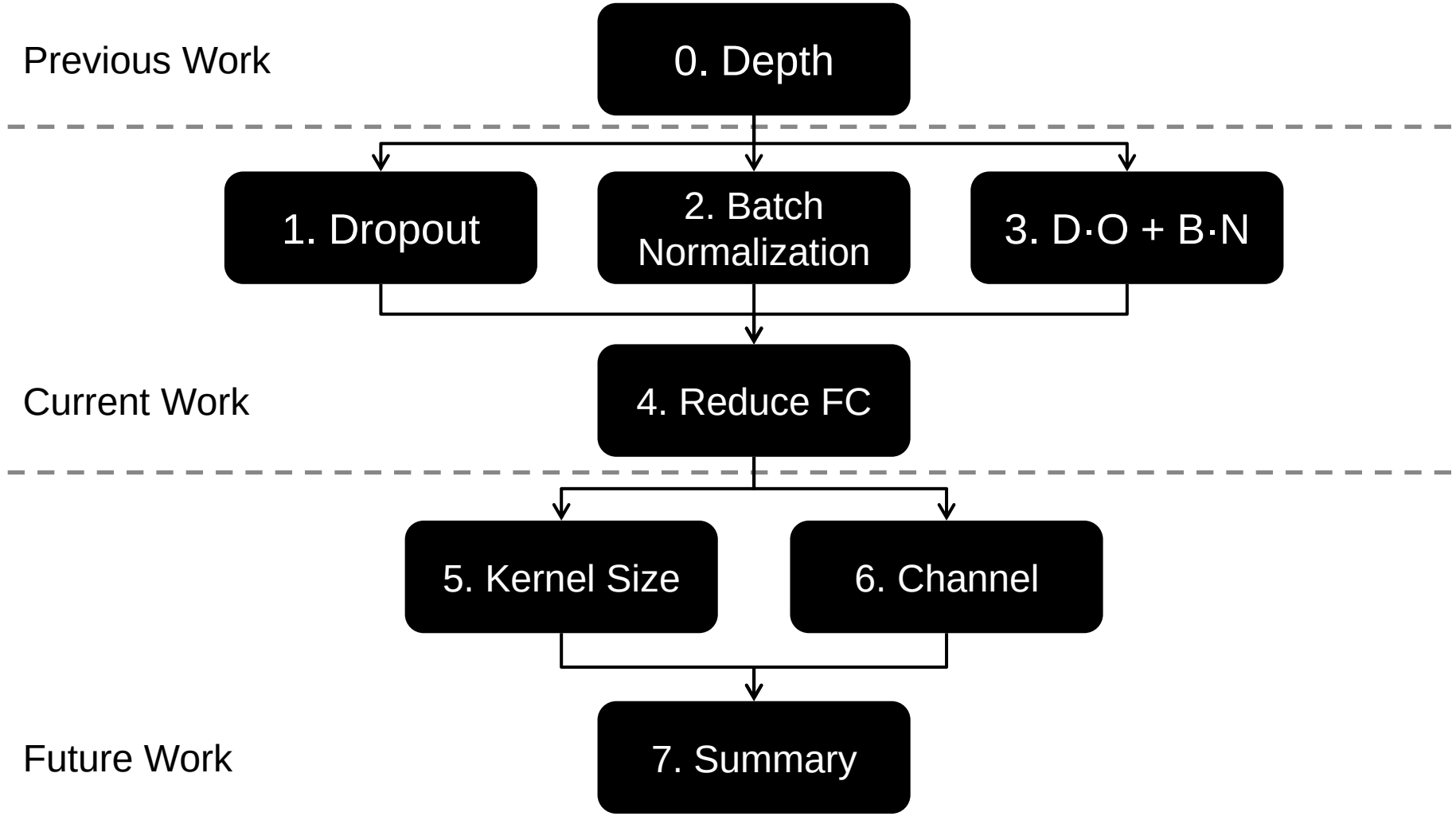
- Data is low-waveform.
  - sec: 1, sampling rate: 16000, type: float32, channel: mono
- 16 class data.
  - 'zero', 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', 'bed', 'bird', 'tree', 'cat', 'house', 'dog'
- Train: 40851(≒80%), Validation: 4796(≒10%), Test: 5297(≒10%)





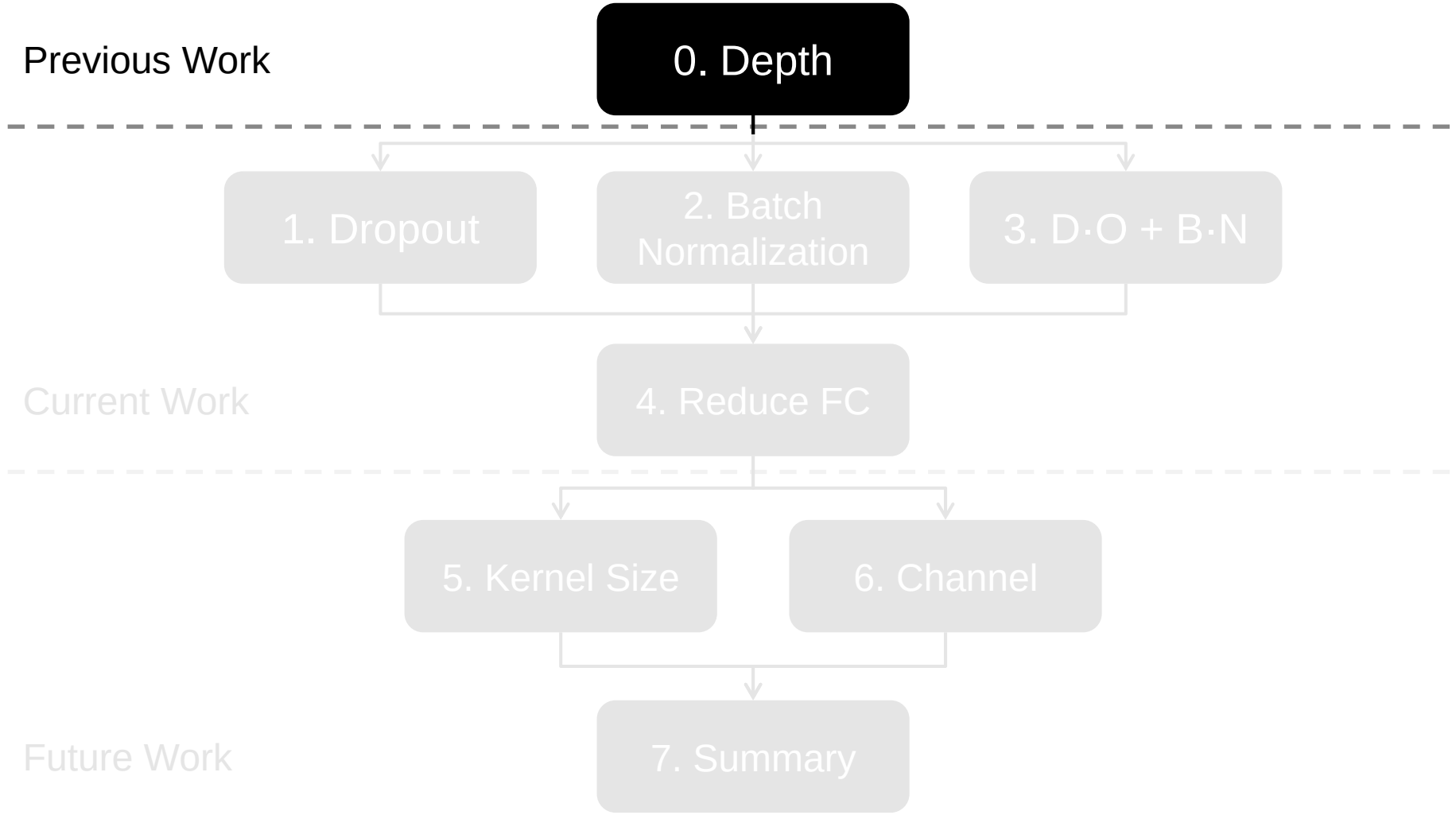
Audio Classification

- Validate task in 1D-CNN



Audio Classification

- Validate task in 1D-CNN



Audio Classification

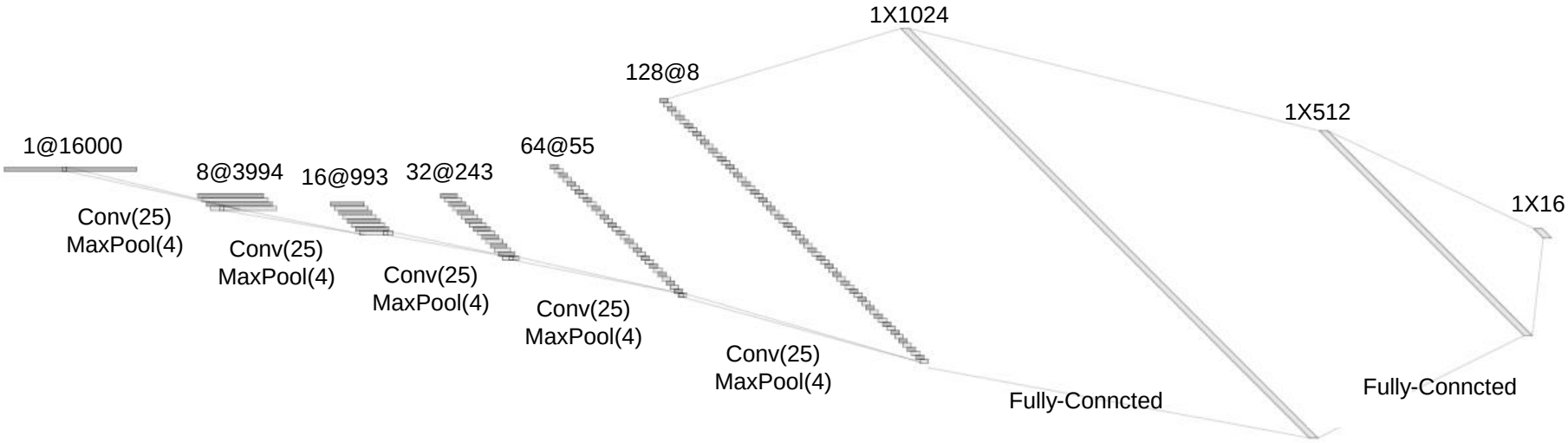
- Each model was trained while changing the depth.
- I found out that the number of parameter is important as well as accuracy
- And I asserted that FC should be reduced as much as possible

Architecture	2D Acc	Params	1D Acc	Params
1 Conv(5x5, 8), 1 Pool(2x2), 1 FC, 1 DO(0.5)	0.8405	49,956,064	0.6648	32,736,480
1 Conv(5x5, 8), 1 Pool(2x2), 2 FC, 2 DO(0.5)	0.8463	50,472,672	0.7078	33,253,088
2 Conv(5x5, 8), 2 Pool(2x2), 1 FC, 1 DO(0.5)	0.8717	22,368,624	0.7688	16,290,160
2 Conv(5x5, 8), 2 Pool(2x2), 2 FC, 2 DO(0.5)	0.8941	22,885,232	0.8056	16,806,768
3 Conv(5x5, 8), 3 Pool(2x2), 1 FC, 1 DO(0.5)	0.9119	8,586,128	0.8717	7,996,304
3 Conv(5x5, 8), 3 Pool(2x2), 2 FC, 2 DO(0.5)	0.9161	9,102,736	0.8702	8,512,912
4 Conv(5x5, 8), 4 Pool(2x2), 1 FC, 1 DO(0.5)	0.9113	2,640,848	0.8945	3,689,424
4 Conv(5x5, 8), 4 Pool(2x2), 2 FC, 2 DO(0.5)	0.9130	3,157,456	0.9038	4,206,032
5 Conv(5x5, 8), 5 Pool(2x2), 1 FC, 1 DO(0.5)	X	X	0.9047	1,338,448
5 Conv(5x5, 8), 5 Pool(2x2), 2 FC, 2 DO(0.5)	X	X	0.9090	1,855,056

Audio Classification

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

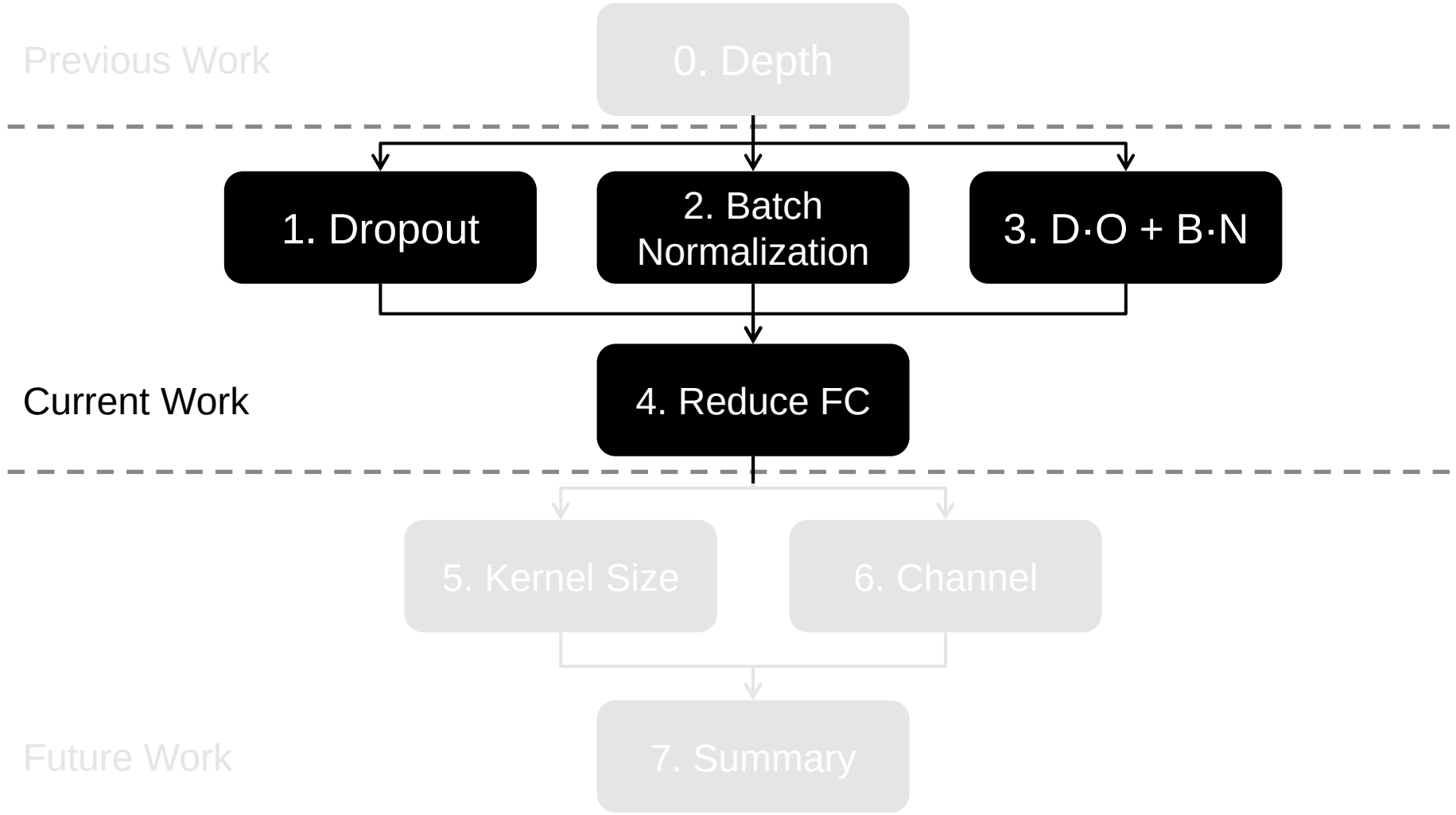
- Input: 16000X1 low waveform.
- Output:1x16 labeled one hot vector. ('zero', …, 'eight', …, 'house', 'dog')
- Loss: cross entropy loss
- Obtimizer: Adam



[Baseline Architecture]

Audio Classification

- Validate task in 1D-CNN



Audio Classification

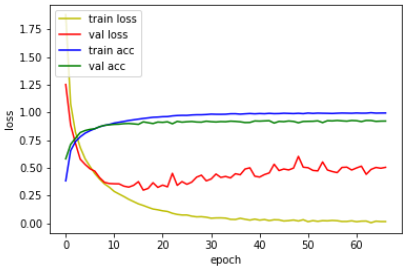
- Verify ‘Dropout(D·O)’ and ‘Batch Normalization(B·N)’
- To verify the D·O and B·N, make the model without each part and compare each case
- Of course, if the model is more deeper, the model is more better

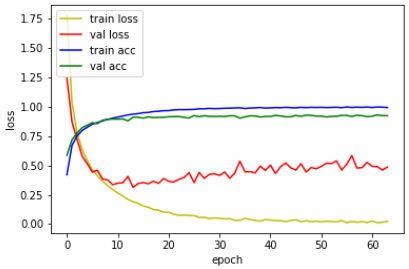
Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5) [baseline]	1D BN	1D DO+BN
1 Conv(25, $8*2^i$), 1 Pool(4), 1 FC	0.6419	0.6648	0.5979	0.6540
1 Conv(25, $8*2^i$), 1 Pool(4), 2 FC	0.6564	0.7078	0.6243	0.6800
2 Conv(25, $8*2^i$), 2 Pool(4), 1 FC	0.7275	0.7688	0.6328	0.7745
2 Conv(25, $8*2^i$), 2 Pool(4), 2 FC	0.7570	0.8056	0.7130	0.7666
3 Conv(25, $8*2^i$), 3 Pool(4), 1 FC	0.8288	0.8717	0.8110	0.8461
3 Conv(25, $8*2^i$), 3 Pool(4), 2 FC	0.8272	0.8702	0.8087	0.8744
4 Conv(25, $8*2^i$), 4 Pool(4), 1 FC	0.8506	0.8945	0.8841	0.8970
4 Conv(25, $8*2^i$), 4 Pool(4), 2 FC	0.8510	0.9038	0.8814	0.9061
5 Conv(25, $8*2^i$), 5 Pool(4), 1 FC	0.8787	0.9047	0.9026	0.9169
5 Conv(25, $8*2^i$), 5 Pool(4), 2 FC	0.8706	0.9090	0.9072	0.9240

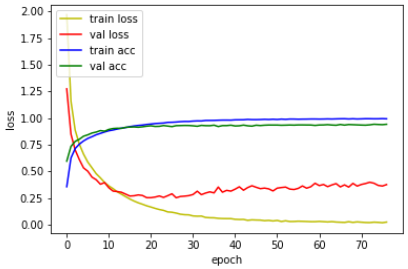
Audio Classification

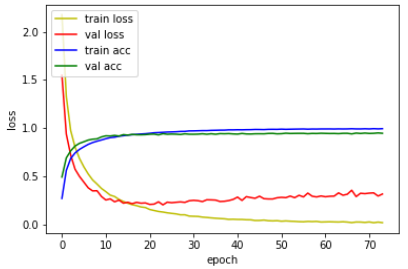
- First, I have a question “Does ‘dropout’ work well?”
- To verify this, I trained the model without D·O
- Through accuracy and learning curve, D·O makes the model better and more stable

Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5) [baseline]	1D BN	1D DO+BN
	…			
4 Conv(25, 8*2^i), 4 Pool(4), 1 FC	0.8506	0.8945	0.8841	0.8970
4 Conv(25, 8*2^i), 4 Pool(4), 2 FC	0.8510	0.9038	0.8814	0.9061
5 Conv(25, 8*2^i), 5 Pool(4), 1 FC	0.8787	0.9047	0.9026	0.9169
5 Conv(25, 8*2^i), 5 Pool(4), 2 FC	0.8706	0.9090	0.9072	0.9240









Audio Classification

- Second, I tried ‘Batch Normalization’
- As mentioned in the paper,
  B·N makes the model more stable by prevent the internal covariate shift
- Compared to the No-D·O model, B·N also makes the model better
- Compared to the D·O model, It is effective when the model is deep.

Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5) [baseline]	1D BN	1D DO+BN
…				
2 Conv(25, $8*2^i$), 2 Pool(4), 2 FC	0.7570	0.8056	0.7130	0.7666
3 Conv(25, $8*2^i$), 3 Pool(4), 1 FC	0.8288	0.8717	0.8110	0.8461
3 Conv(25, $8*2^i$), 3 Pool(4), 2 FC	0.8272	0.8702	0.8087	0.8744
4 Conv(25, $8*2^i$), 4 Pool(4), 1 FC	0.8506	0.8945	0.8841	0.8970
4 Conv(25, $8*2^i$), 4 Pool(4), 2 FC	0.8510	0.9038	0.8814	0.9061
5 Conv(25, $8*2^i$), 5 Pool(4), 1 FC	0.8787	0.9047	0.9026	0.9169
5 Conv(25, $8*2^i$), 5 Pool(4), 2 FC	0.8706	0.9090	0.9072	0.9240

Audio Classification

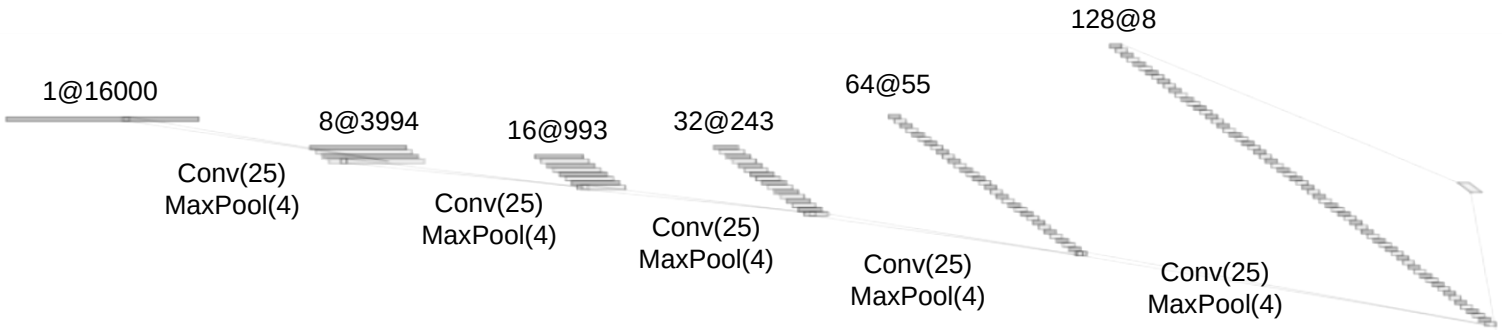
- Third, A lot of people use D·O and B·N together
- It is the best performance than the other

Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5) [baseline]	1D BN	1D DO+BN
1 Conv(25, $8*2^i$), 1 Pool(4), 1 FC	0.6419	0.6648	0.5979	0.6540
1 Conv(25, $8*2^i$), 1 Pool(4), 2 FC	0.6564	0.7078	0.6243	0.6800
2 Conv(25, $8*2^i$), 2 Pool(4), 1 FC	0.7275	0.7688	0.6328	0.7745
2 Conv(25, $8*2^i$), 2 Pool(4), 2 FC	0.7570	0.8056	0.7130	0.7666
3 Conv(25, $8*2^i$), 3 Pool(4), 1 FC	0.8288	0.8717	0.8110	0.8461
3 Conv(25, $8*2^i$), 3 Pool(4), 2 FC	0.8272	0.8702	0.8087	0.8744
4 Conv(25, $8*2^i$), 4 Pool(4), 1 FC	0.8506	0.8945	0.8841	0.8970
4 Conv(25, $8*2^i$), 4 Pool(4), 2 FC	0.8510	0.9038	0.8814	0.9061
5 Conv(25, $8*2^i$), 5 Pool(4), 1 FC	0.8787	0.9047	0.9026	0.9169
5 Conv(25, $8*2^i$), 5 Pool(4), 2 FC	0.8706	0.9090	0.9072	0.9240

Audio Classification

- 4th, I think that FC has a lot of problem (FC = Fully Connected)
- So I retry the model without FC layer (Now I call this model “only conv”)
- FC exists only between the last Conv layer and the output layer (for classfication)

Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5)	1D BN	1D DO+BN
1 CONV(25, 8*2^i)	0.4621	0.4889	0.4289	0.4490
2 CONV(25, 8*2^i)	0.5321	0.6449	0.5836	0.6885
3 CONV(25, 8*2^i)	0.7747	0.8239	0.7890	0.8538
4 CONV(25, 8*2^i)	0.8866	0.9215	0.8804	0.9238
5 CONV(25, 8*2^i)	0.8870	0.9277	0.9288	0.9375

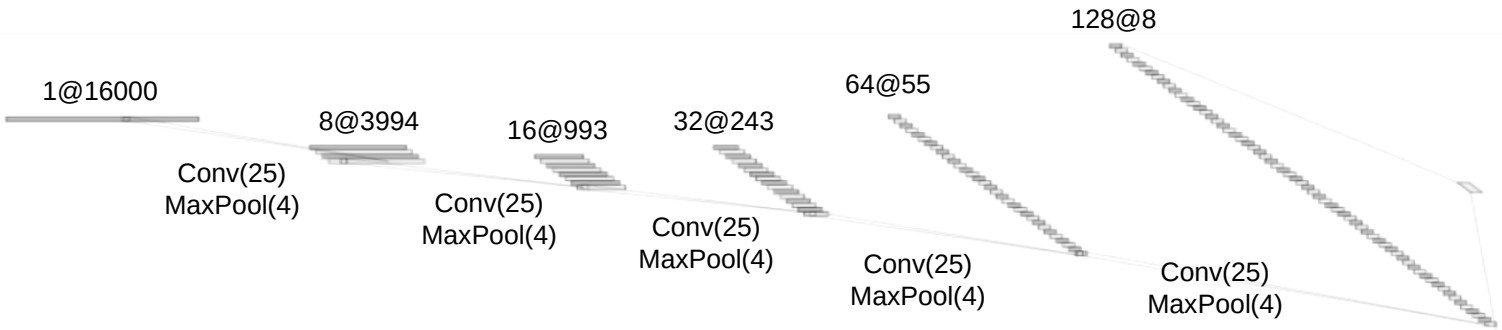


[“Only Conv” Architecture]

Audio Classification

- This model was also tested for each case.
- And also shown the best performance in ‘D·O+B·N’ model

Architecture (i = 0,1,2…)	1D No-DO	1D DO(0.5)	1D BN	1D DO+BN
1 CONV(25, 8*2^i)	0.4621	0.4889	0.4289	0.4490
2 CONV(25, 8*2^i)	0.5321	0.6449	0.5836	0.6885
3 CONV(25, 8*2^i)	0.7747	0.8239	0.7890	0.8538
4 CONV(25, 8*2^i)	0.8866	0.9215	0.8804	0.9238
5 CONV(25, 8*2^i)	0.8870	0.9277	0.9288	0.9375



[“Only Conv” Architecture]

Audio Classification

- Compare both
- Let's focus on the accuracy and the number of parameters.

base model

Architecture (i = 0,1,2…)	1D DO(0.5) [baseline]	1D BN	1D DO+BN	Params
…				
4 Conv(25, $8*2^i$), 4 Pool(4), 1 FC	0.8945	0.8841	0.8970	3,689,424
4 Conv(25, $8*2^i$), 4 Pool(4), 2 FC	0.9038	0.8814	0.9061	4,206,032
5 Conv(25, $8*2^i$), 5 Pool(4), 1 FC	0.9047	0.9026	0.9169	1,338,448
5 Conv(25, $8*2^i$), 5 Pool(4), 2 FC	0.9090	0.9072	0.9240	1,855,056

Only Conv model

Architecture (i = 0,1,2…)	1D DO(0.5)	1D BN	1D DO+BN	Params
…				
4 CONV(25, $8*2^i$)	0.9215	0.8804	0.9238	123,856
5 CONV(25, $8*2^i$)	0.9277	0.9288	0.9375	288,848

Audio Classification

- Compared to the best accuracy, It is increased. (0.9240 → 0.9375)
- Compared to the number of parameter, It is greatly decreased. (~90.8%)
- It's a great achievement beyond what I expected.

base model

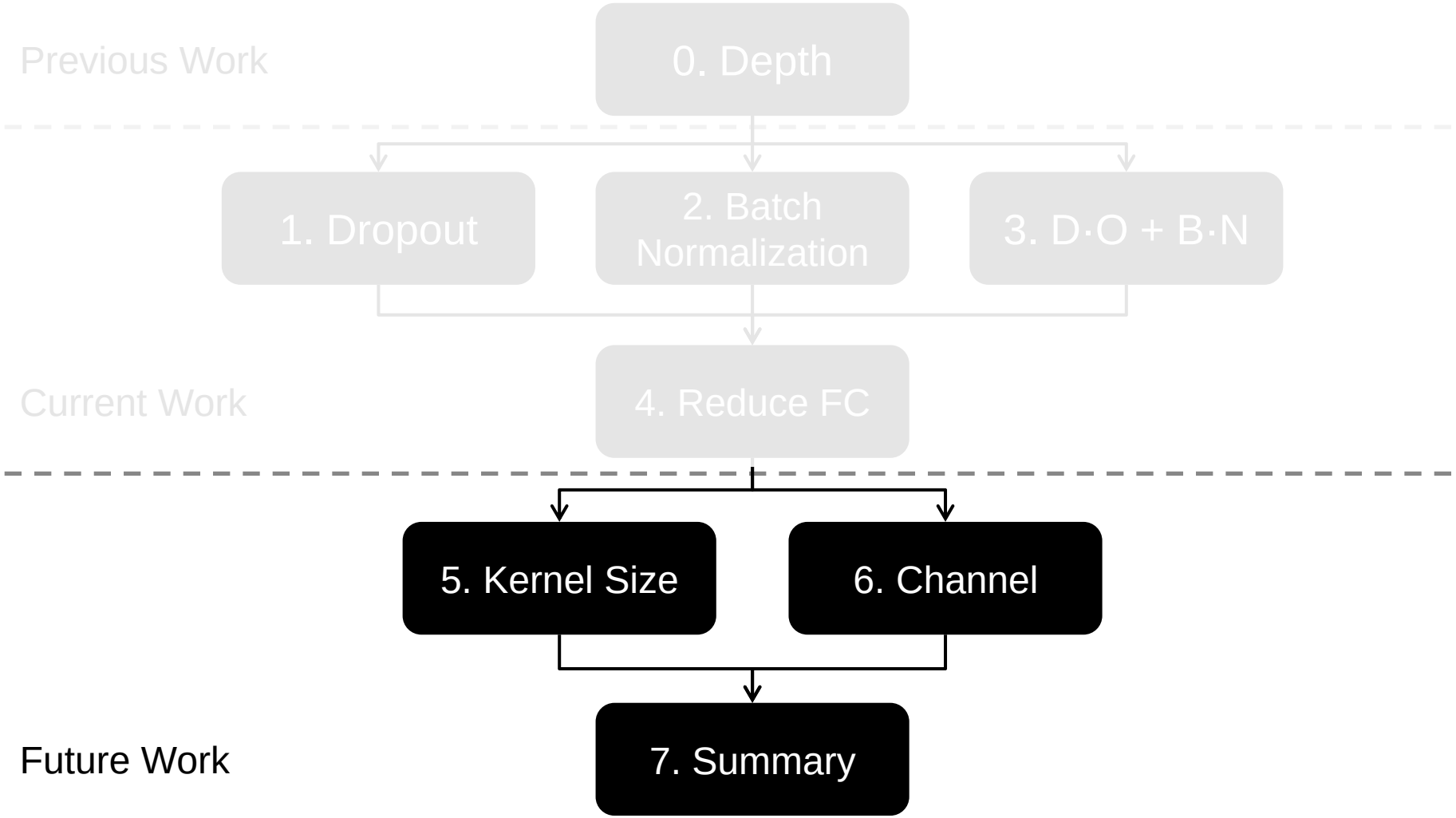
Architecture (i = 0,1,2…)	1D DO(0.5) [baseline]	1D BN	1D DO+BN	Params
…				
4 Conv(25, $8*2^i$), 4 Pool(4), 1 FC	0.8945	0.8841	0.8970	3,689,424
4 Conv(25, $8*2^i$), 4 Pool(4), 2 FC	0.9038	0.8814	0.9061	4,206,032
5 Conv(25, $8*2^i$), 5 Pool(4), 1 FC	0.9047	0.9026	0.9169	1,338,448
5 Conv(25, $8*2^i$), 5 Pool(4), 2 FC	0.9090	0.9072	0.9240	1,855,056

Only Conv model

Architecture (i = 0,1,2…)	1D DO(0.5)	1D BN	1D DO+BN	Params
…				
4 CONV(25, $8*2^i$)	0.9215	0.8804	0.9238	123,856
5 CONV(25, $8*2^i$)	0.9277	0.9288	0.9375	288,848

Audio Classification

- Validate task in 1D-CNN



Any Question?

Thank you