Progress Update – 14 Jul

Modulation classification using different DL models

Things done so far

- Trained and tested several DL models on 2016.04 RadioML dataset (11 classes)
  - FCN/DNN, basic CNNs, ResNet, Inception

Process of DL training

- Used 2016.04 RadioML dataset https://www.deepsig.ai/datasets
  - 11 classes, 162060 samples, 80% training, 20% testing
  - Each data sample in I/Q format spans across 128 time units, (2x128)
- Models taken from papers from DeepSig/ O’Shea:
  - Convolutional Radio Modulation Recognition Networks
    - Implementation: https://github.com/alyswidan/ModulationRecognition
  - Over-the-Air Deep Learning Based Radio Signal Classification
    - Implementation: https://medium.com/gsi-technology/residual-neural-networks-in-python-1796a57c2d7
- Train for 30 epochs, pick best weights within the 30 epochs (some models stop improving after a few epochs)
- All use same default learning rate, same Adam optimiser, ReLU activation

Dataset Exploration

Dataset generation (GNU radio)

- Source alphabet: voice recording (analog), Shakespeare texts ASCII (digital)
- Signal modulation: 11 types –
  - 8 Digital: 8PSK, QPSK, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64
  - 3 Analog: WBFM, AM-DSB, AM-SSB
- Simulating channel effects: center frequency offset, sample rate offset, AWGN, multi-path, fading
- Output: time series signal divided into lengths of 128 complex 32-bit samples (I/Q format)
- Dataset available in .pkl format at https://www.deepsig.ai/datasets , in the form of a dictionary with (modulation class, SNR) as keys and each 2x128 time series as values

Dataset visualization

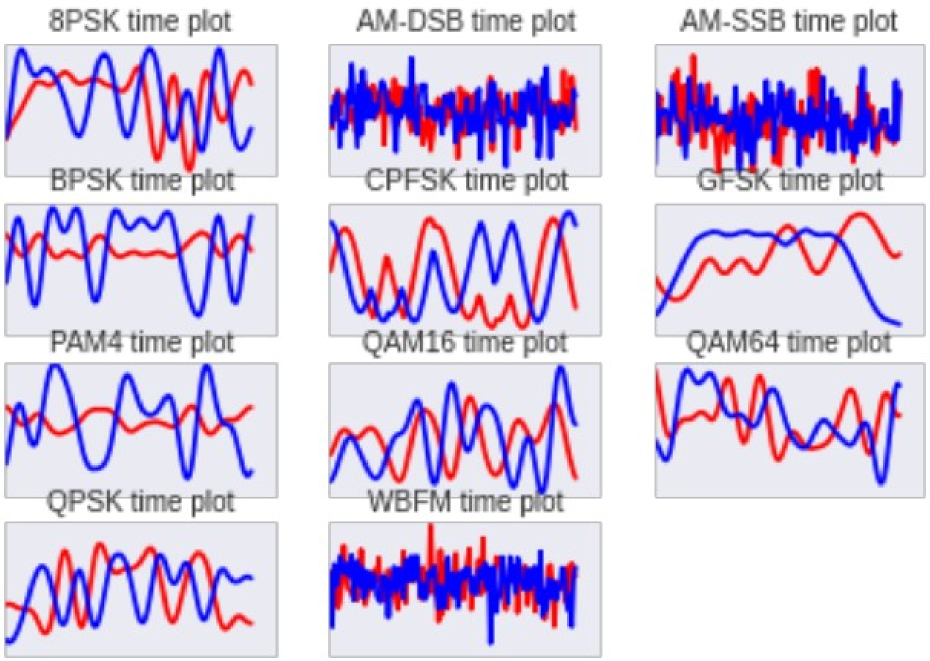


Figure 1. Time Domain of High-SNR Example Classes

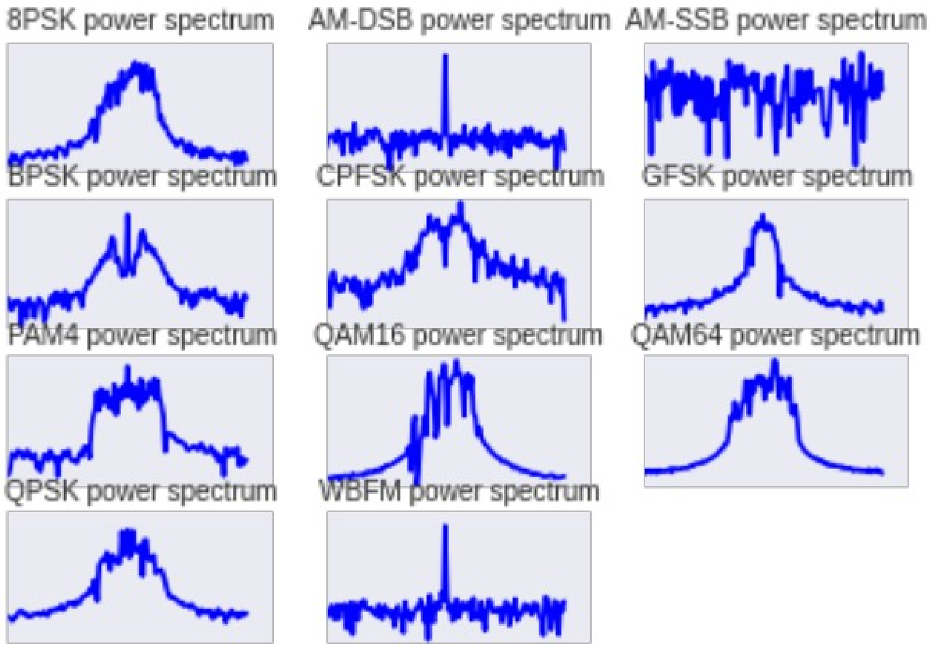
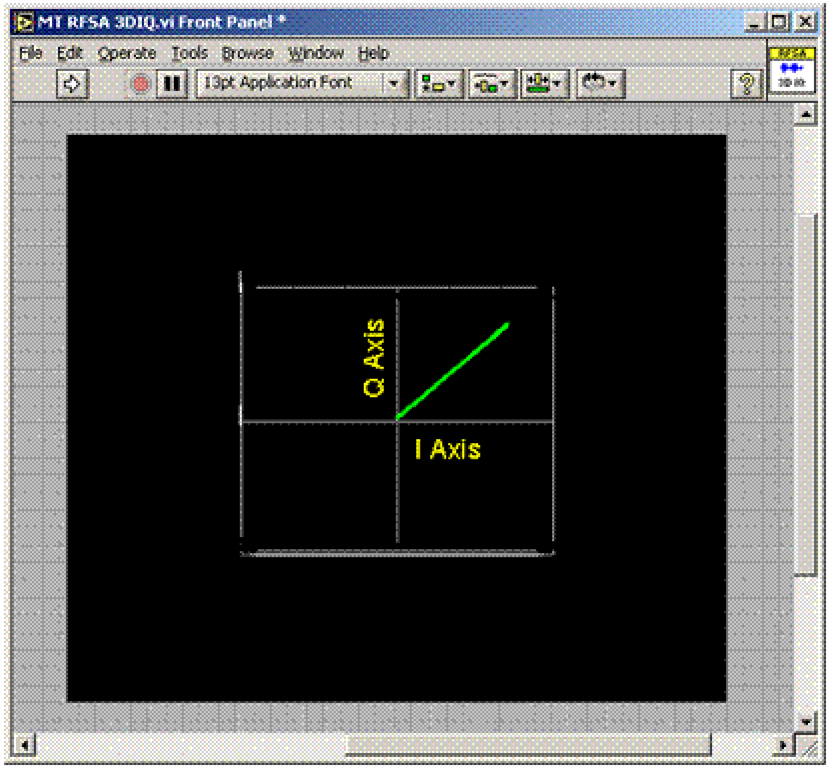
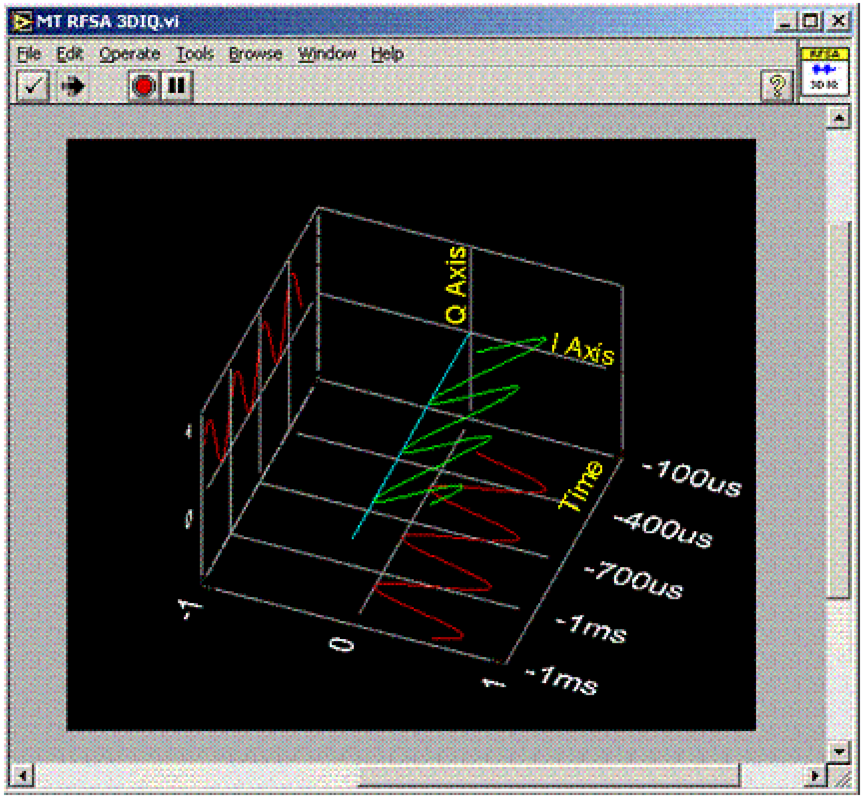


Figure 2. Power Spectrum of High-SNR Example Classes

I/Q format

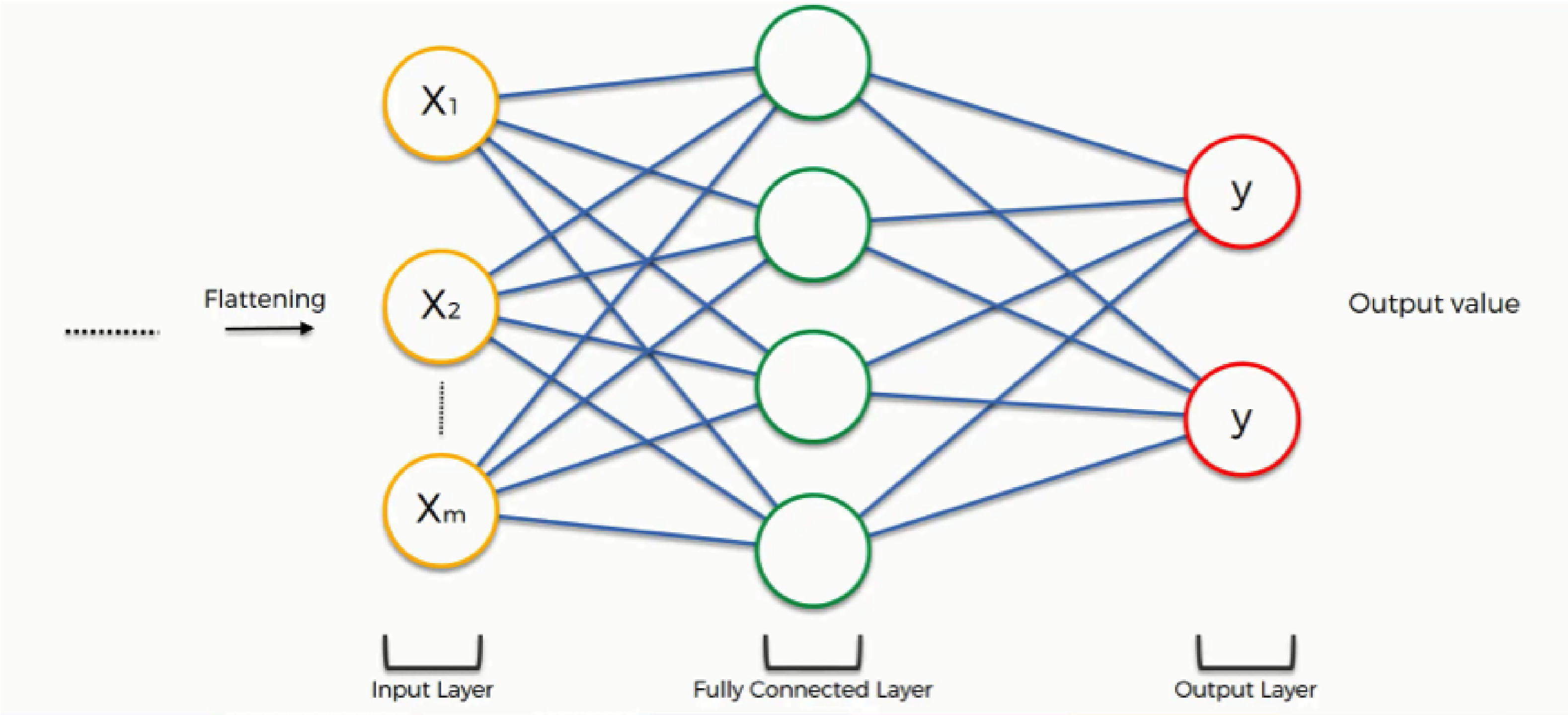
- Complex baseband representation of a signal
- decomposes a radio voltage level time-series into its projections onto the sine and cosine functions at a carrier frequency





Model descriptions

Fully Connected Networks (FCN)



- 5 layers: 186 + 128 + 64 + 32 + 11
- 3 layers: 186 + 64 + 11;   512 + 256 + 11

Convolutional neural networks (CNN)

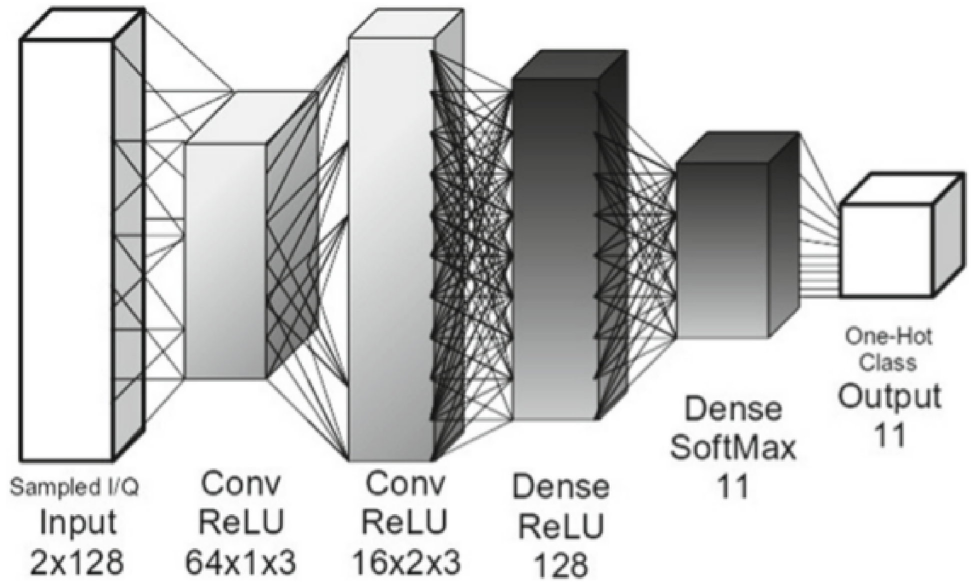
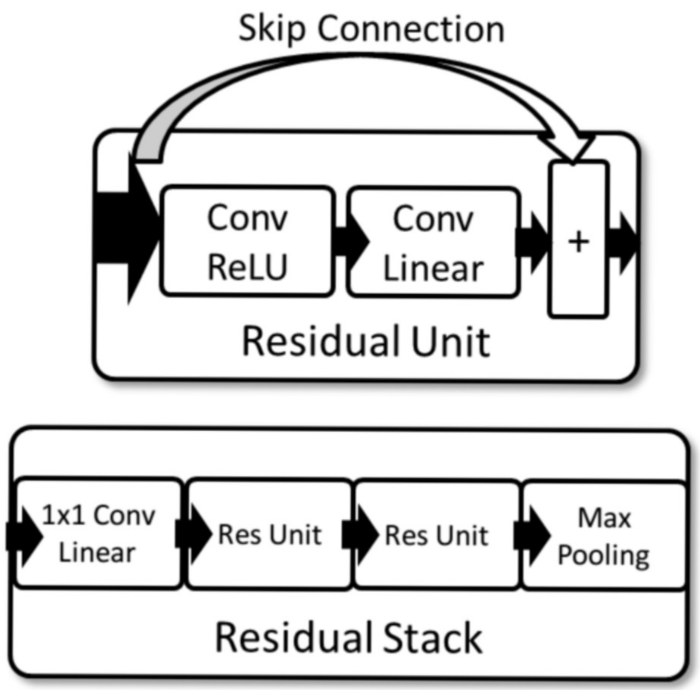


Fig. 3. CNN architecture

- See paper Convolutional Radio Modulation Recognition Networks
- 2 Conv + 2 Dense
- CNN --> matched filter?

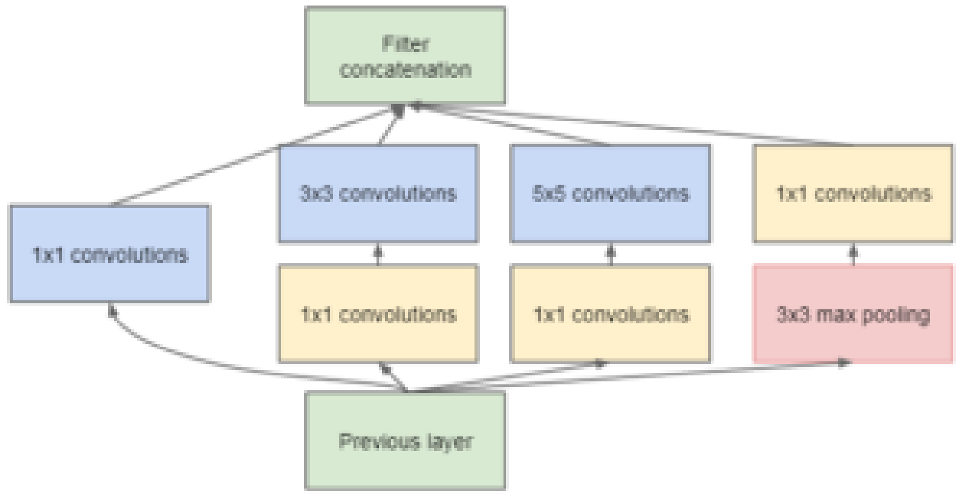
ResNet



Layer	Output dimensions
Input	2×1024
Residual Stack	32×512
Residual Stack	32×256
Residual Stack	32×128
Residual Stack	32×64
Residual Stack	32×32
Residual Stack	32×16
FC/SeLU	128
FC/SeLU	128
FC/Softmax	24

- See paper Over-the-Air Deep Learning Based Radio Signal Classification
- Res stack solves vanishing gradient problem when CNN gets too deep using skip-connection
- Skip-connection allows earlier features to operate at multiple scales and depths throughout NN
- Model contains 6 Res stacks

Inception



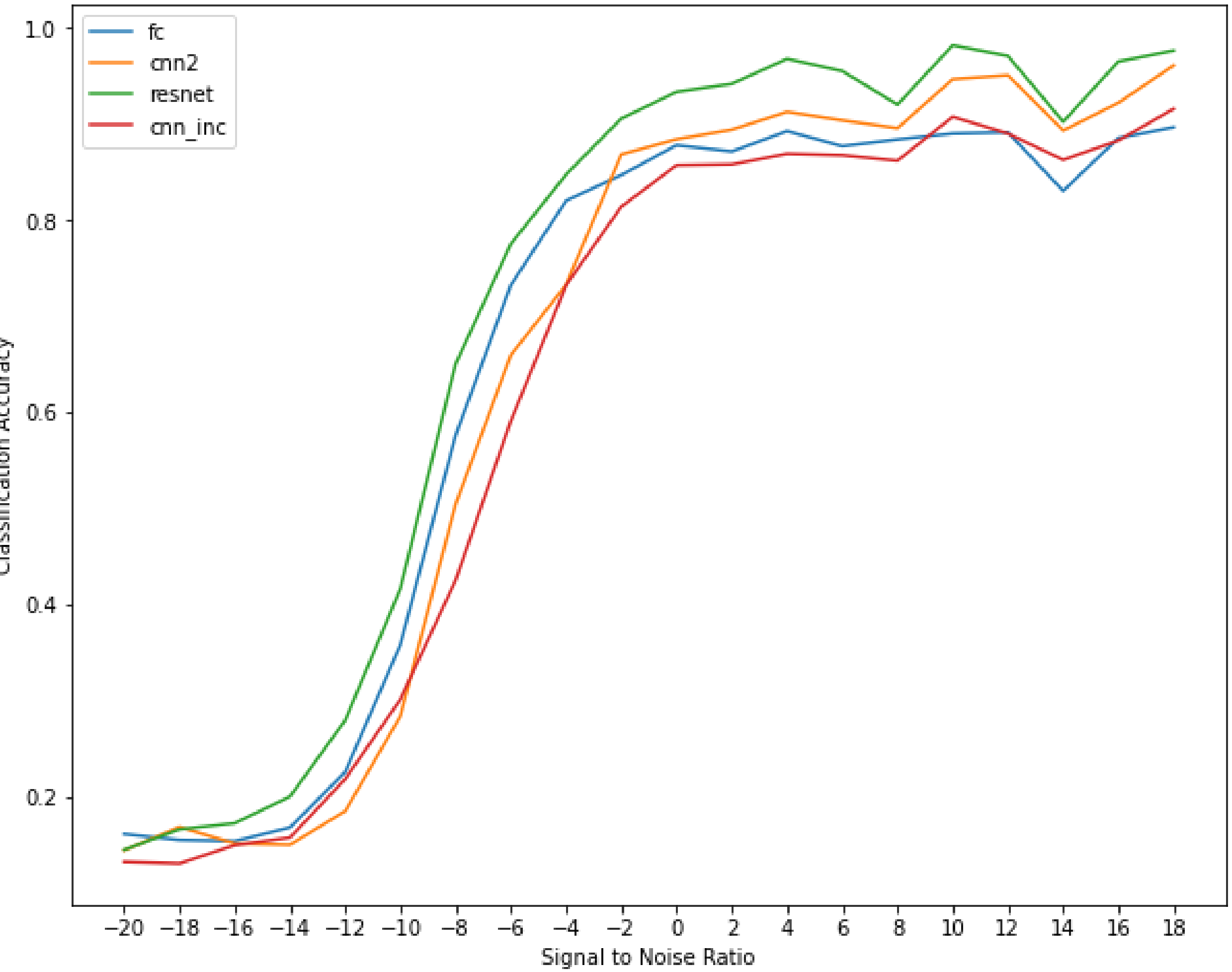
- Idea is to have filters of different dimensions in each layer --> wider instead of deeper

Results and insights

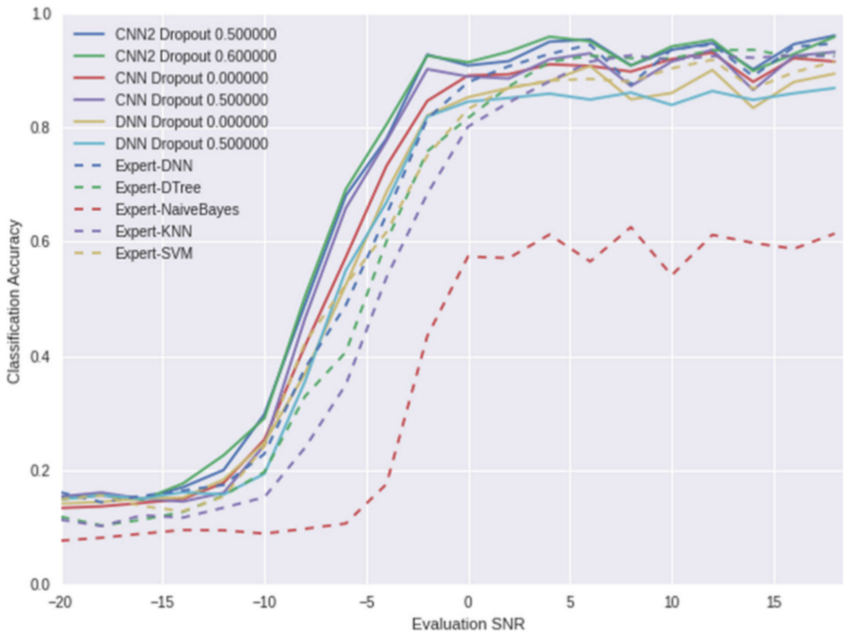
Overall insights

- ResNet appears to perform the best
- Confusion matrices
  - 8PSK vs QPSK
  - AM-DSB vs WBFM
- General trends (for FCN):
  - Deeper good, num_nodes meh
  - Batchnorm good, reg good
  - kernel_init: he_norm good

Overall performances – ResNet best



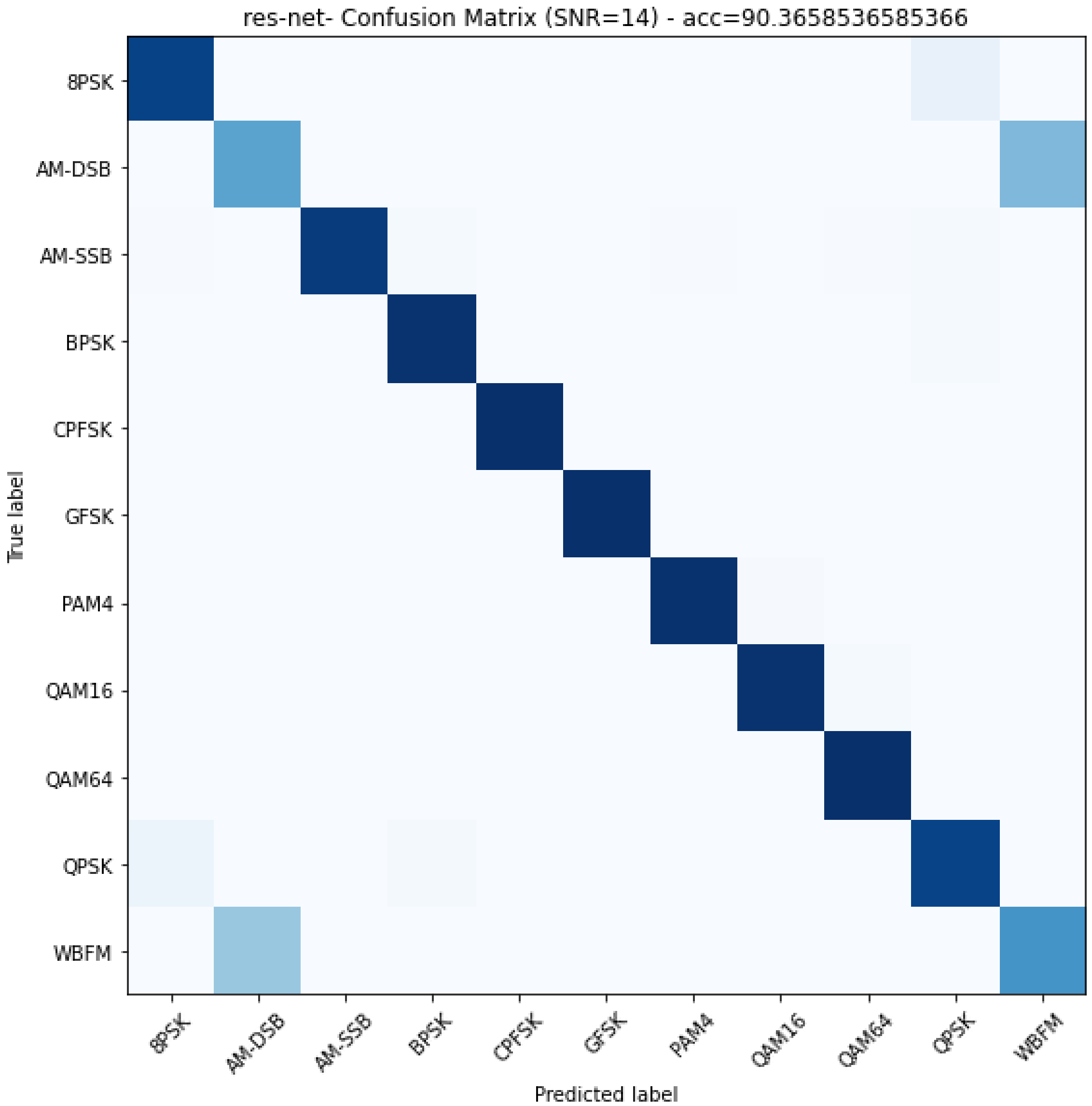
My implementations
(inception looks iffy,
maybe done wrong)



From paper
(CNN in MR)

Confusion Matrix

- 8PSK vs QPSK
- AM-DSB vs WBFM
- Same across all



Mix-ups: 8PSK vs QPSK, AM-DSB vs WPFM

- Apparently constellation diagram data format can help distinguish 16AM and 64QAM, maybe will work for these four also

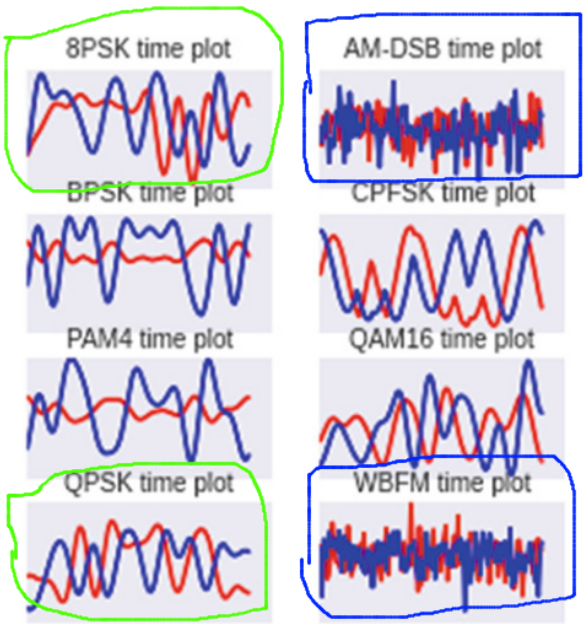


Fig. 1. Time domain of high-SNR example classes

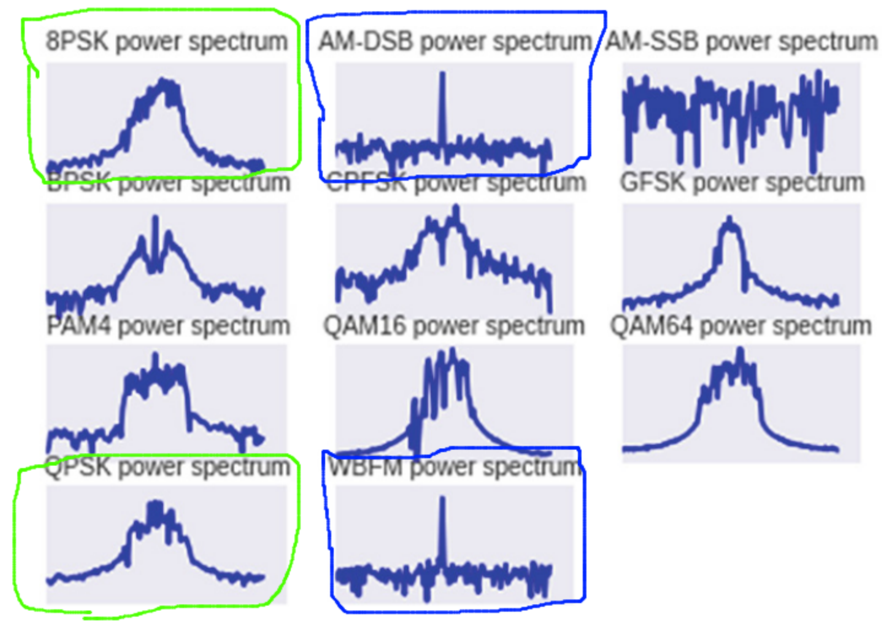


Fig. 2. Power spectrum of high-SNR example classes

Number of model parameters

FCN	CNN2	ResNet	Inception2
82,693	2,666,587	142,019	497,515

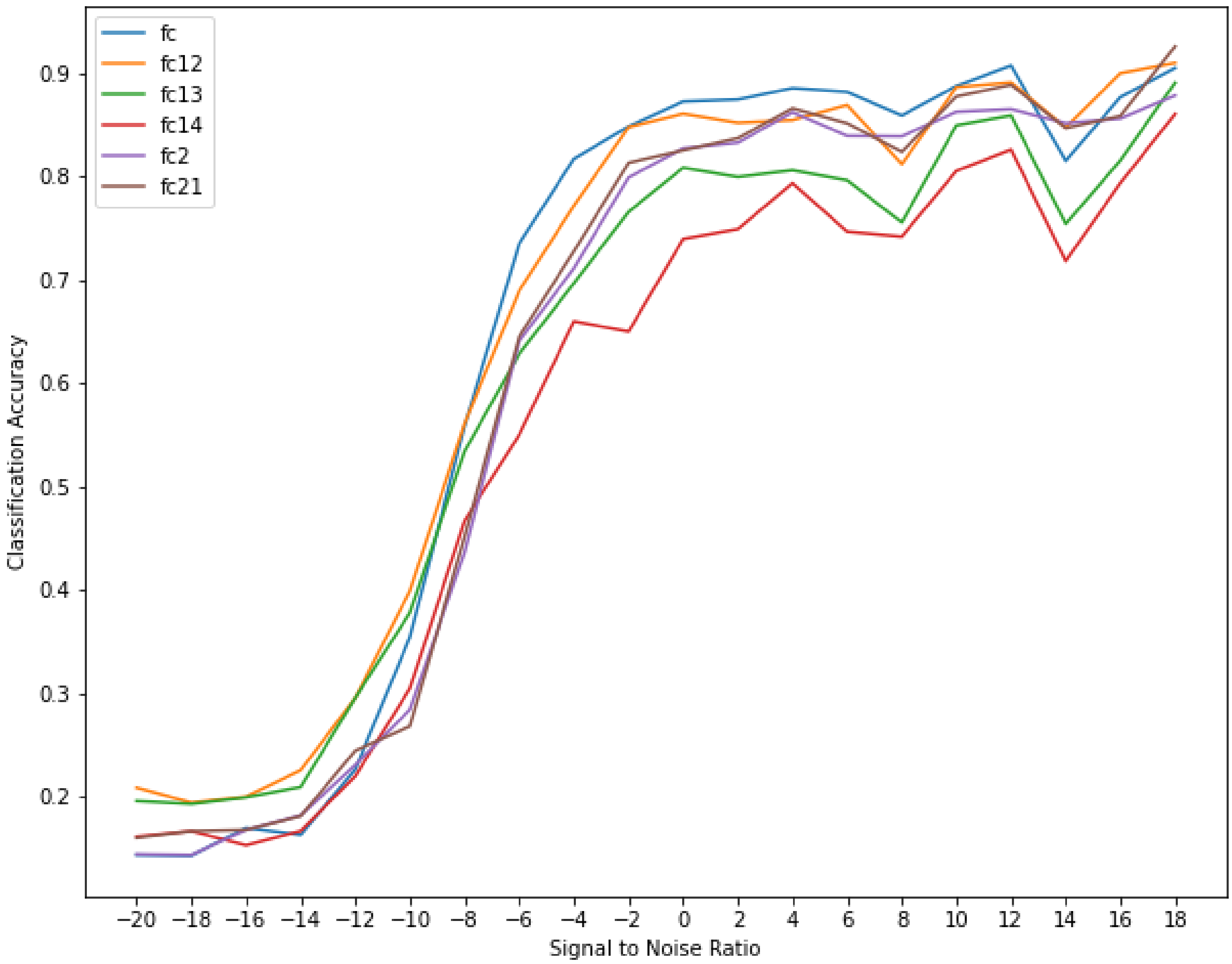
FCN and parameter testing

FCN perfomances for different params

- Params tested:
  - Regularisation of layers
  - BatchNormalisation
  - Initialisation of kernels
  - Depth of FCN (number of layers)
  - Number of nodes in layers
- Params not yet tested:
  - Dropout
  - Regularisation param

	reg	**batch norm**	**Kernel init**	**depth**	**Num nodes**
fc	Y	Y	unif	5	few
fc12	N	Y	unif	5	few
fc13	N	Y	norm	5	few
fc14	Y	N	unif	5	few
fc2	Y	Y	unif	3	few
fc21	Y	Y	unif	3	many

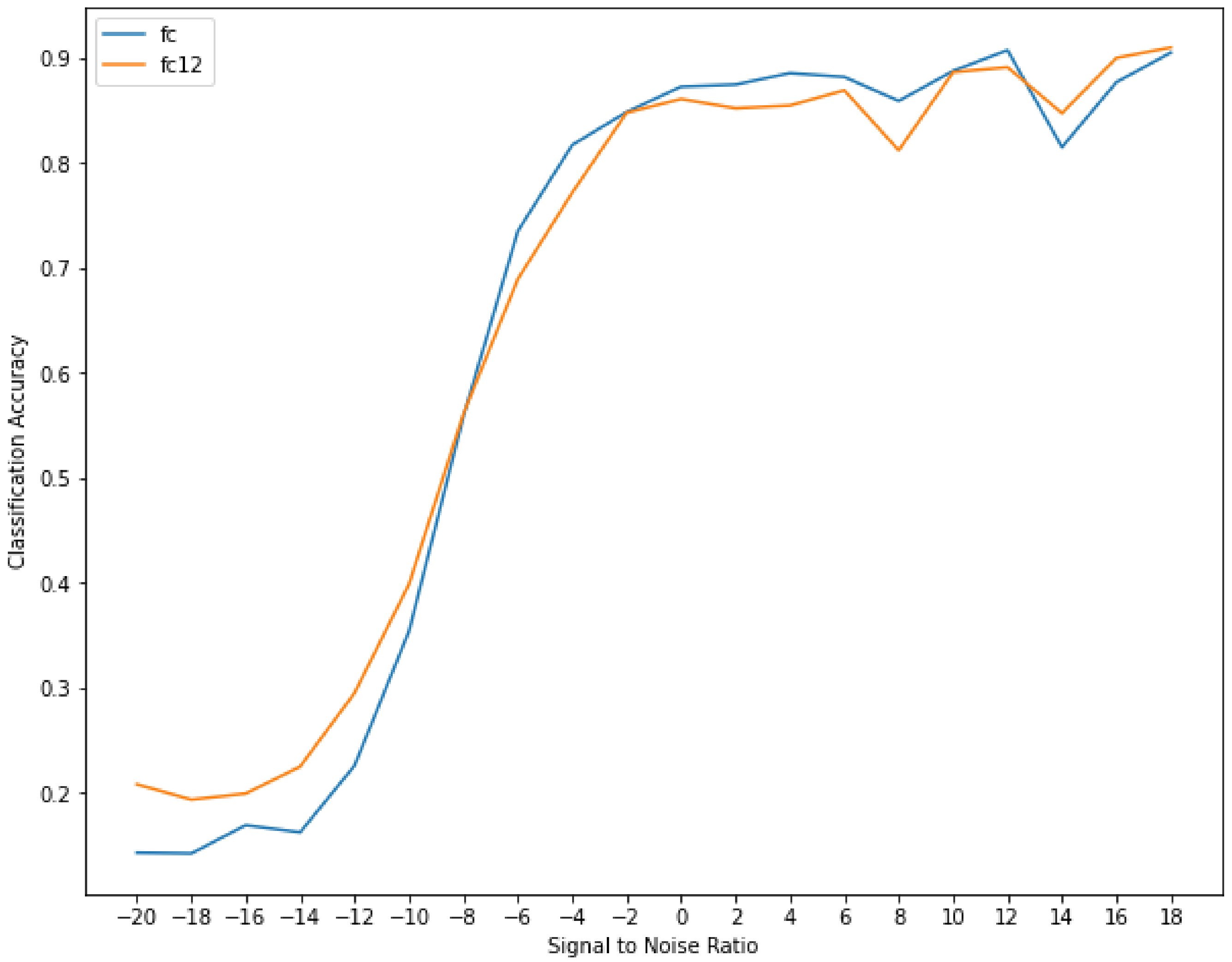
Different FCN perfomances



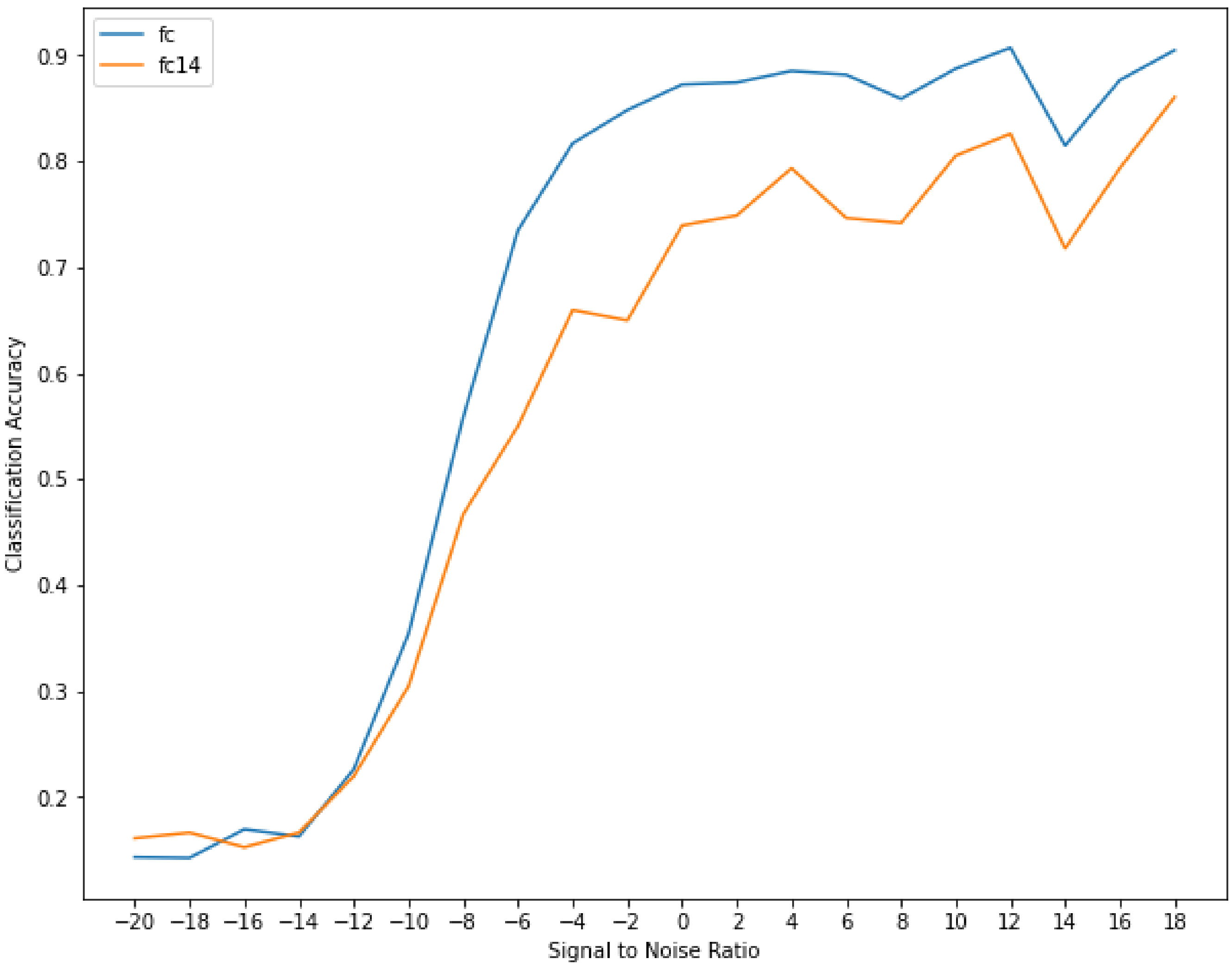
Observations on FCN

- Reg: fc vs fc12 --> reg slightly better
- Batchnorm: fc vs fc14 --> batchnorm good
- Kernel_init: fc12 vs fc13 --> unif > normal
- Depth: fc vs fc2 --> Deep good
- Num_nodes: fc2 vs fc21 --> more slightly better

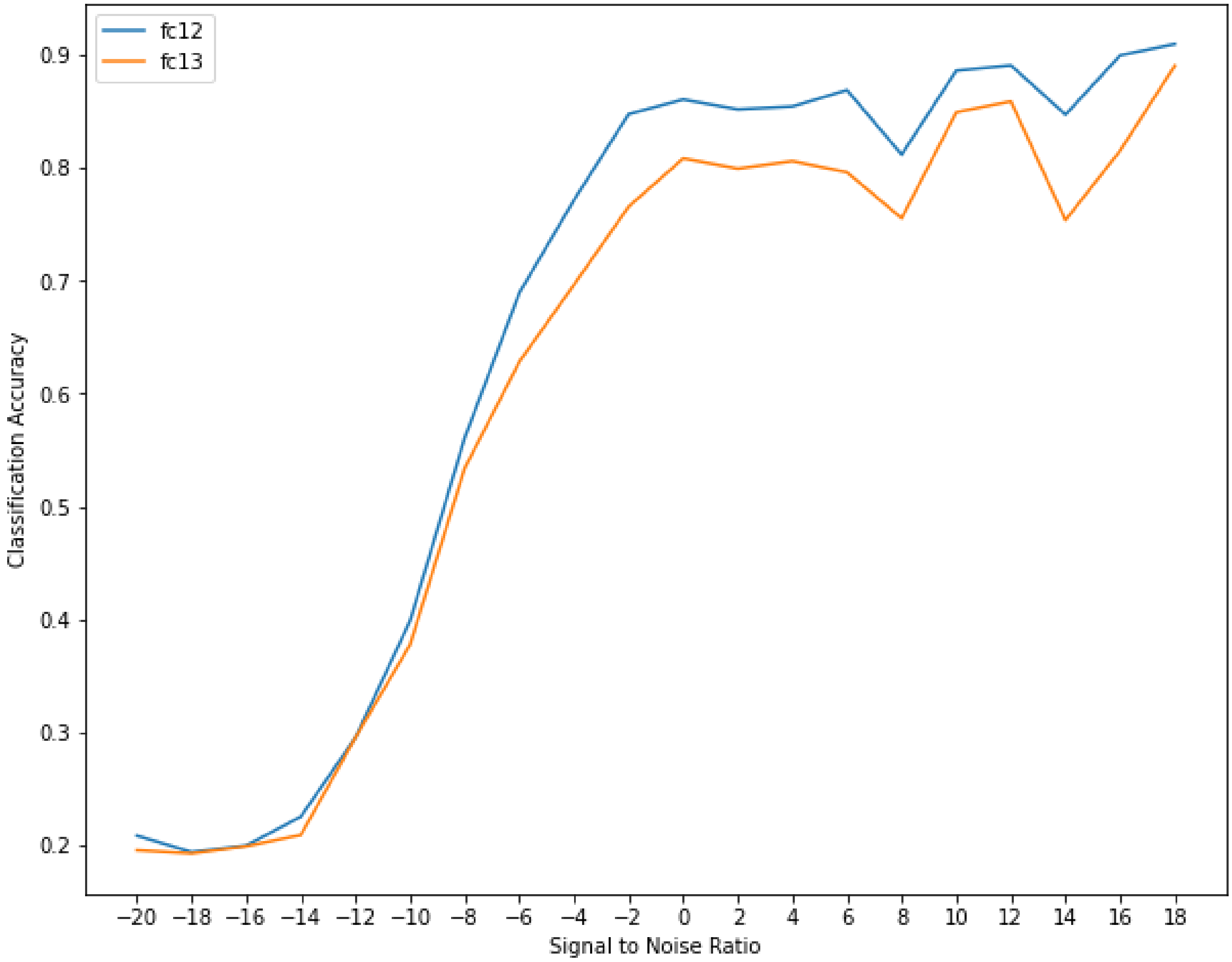
Fc (reg) vs fc12 (no reg)



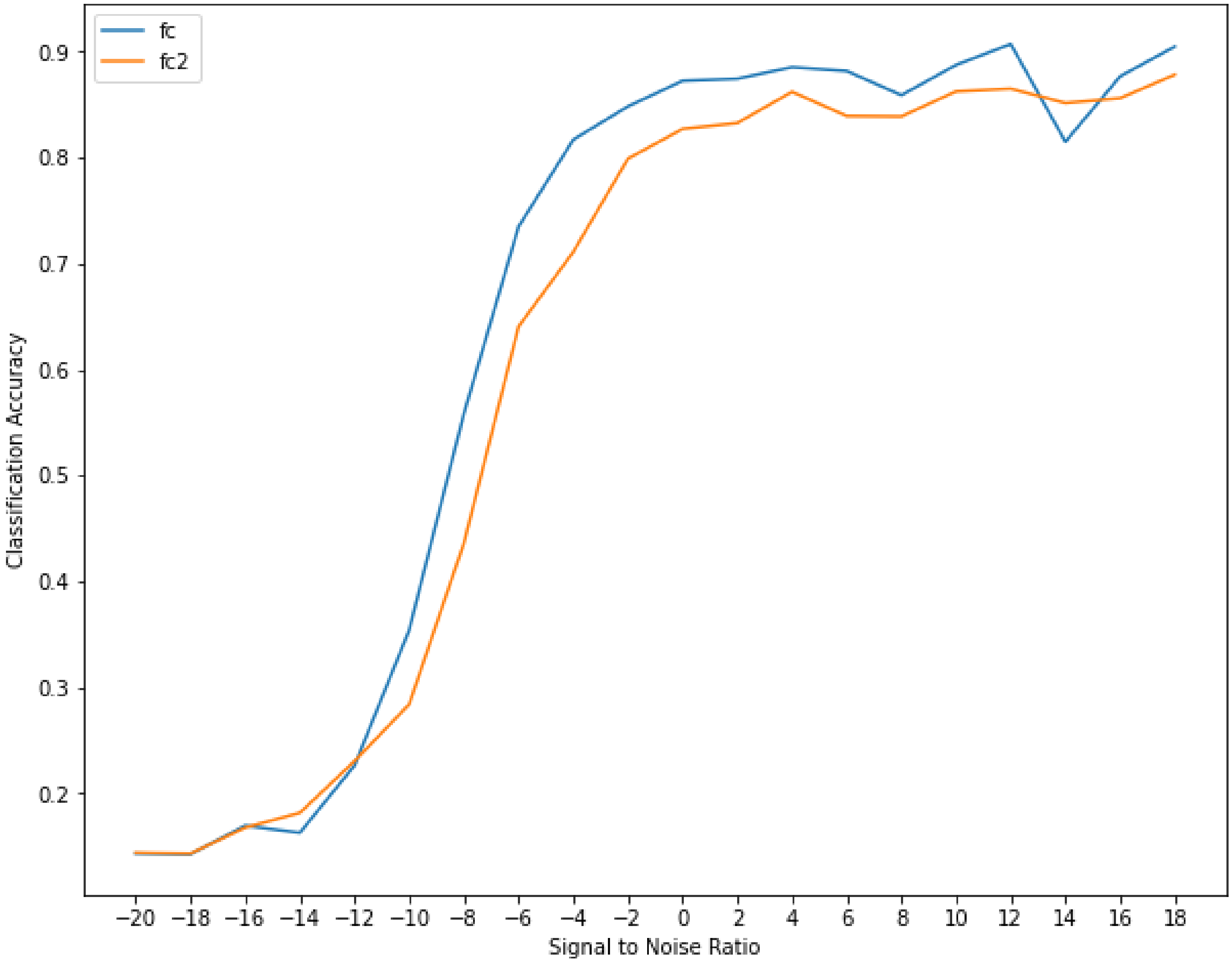
fc (batchnorm) vs fc14 (no bn)



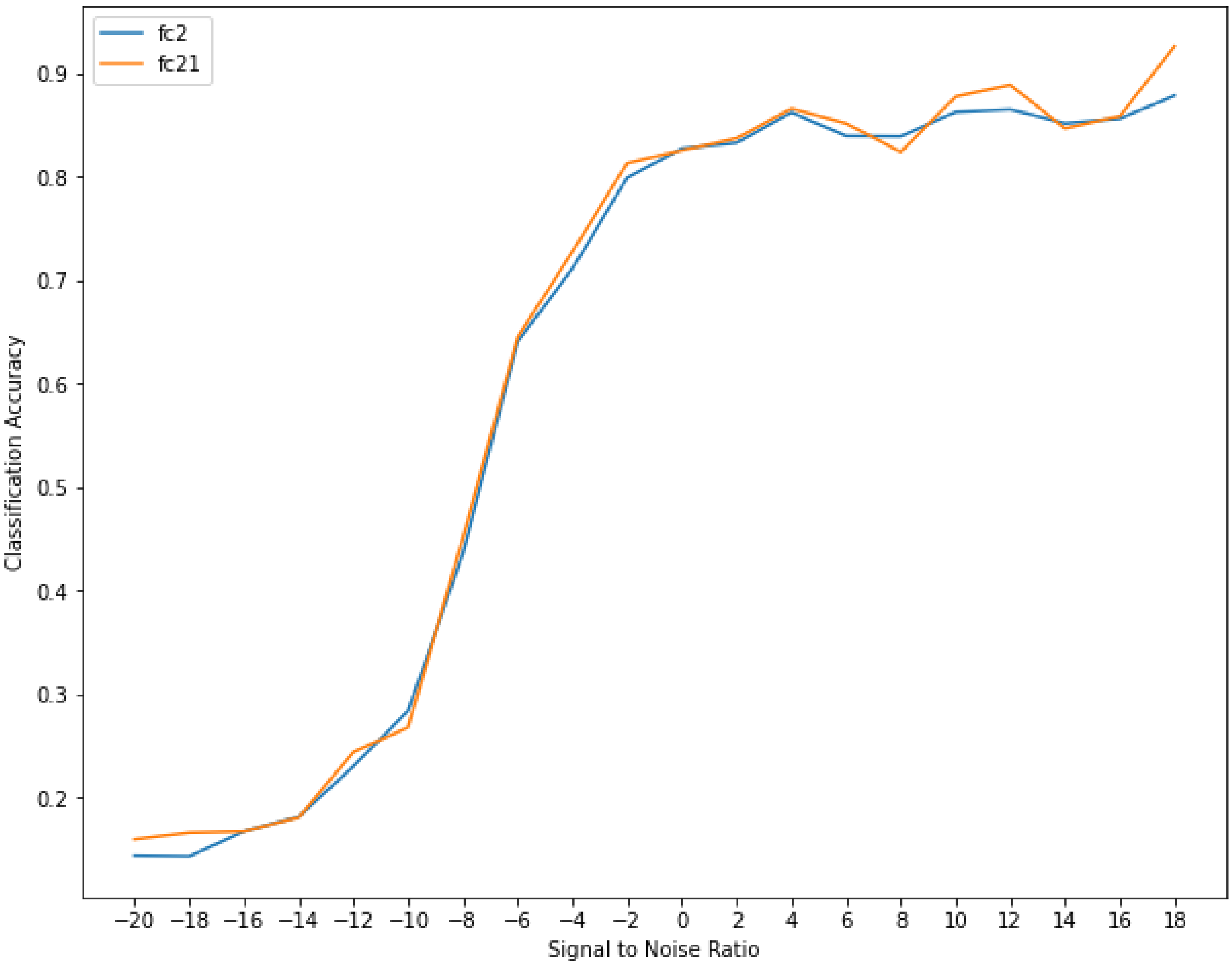
Fc12 (unif init) vs fc13 (norm init)



fc (deep) vs fc2 (shallow)



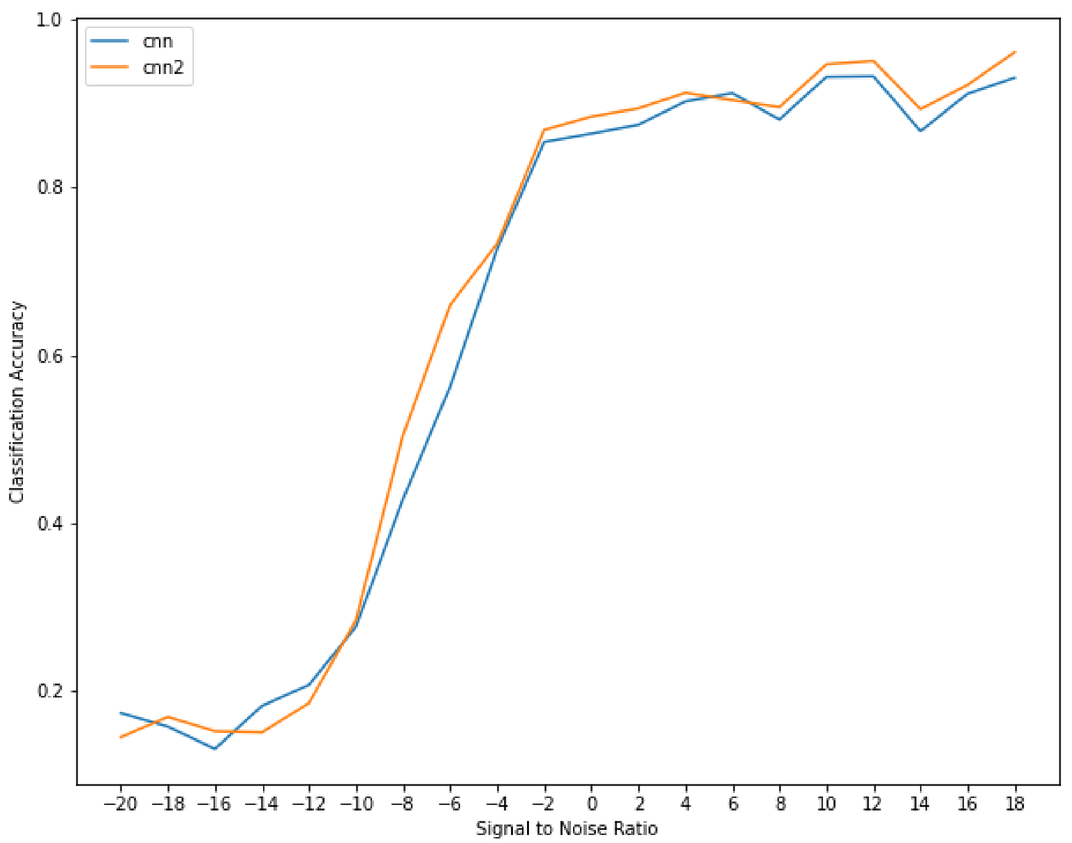
Fc2 (few nodes) vs fc21 (more nodes)



Basic CNN performances

CNN perf comparison

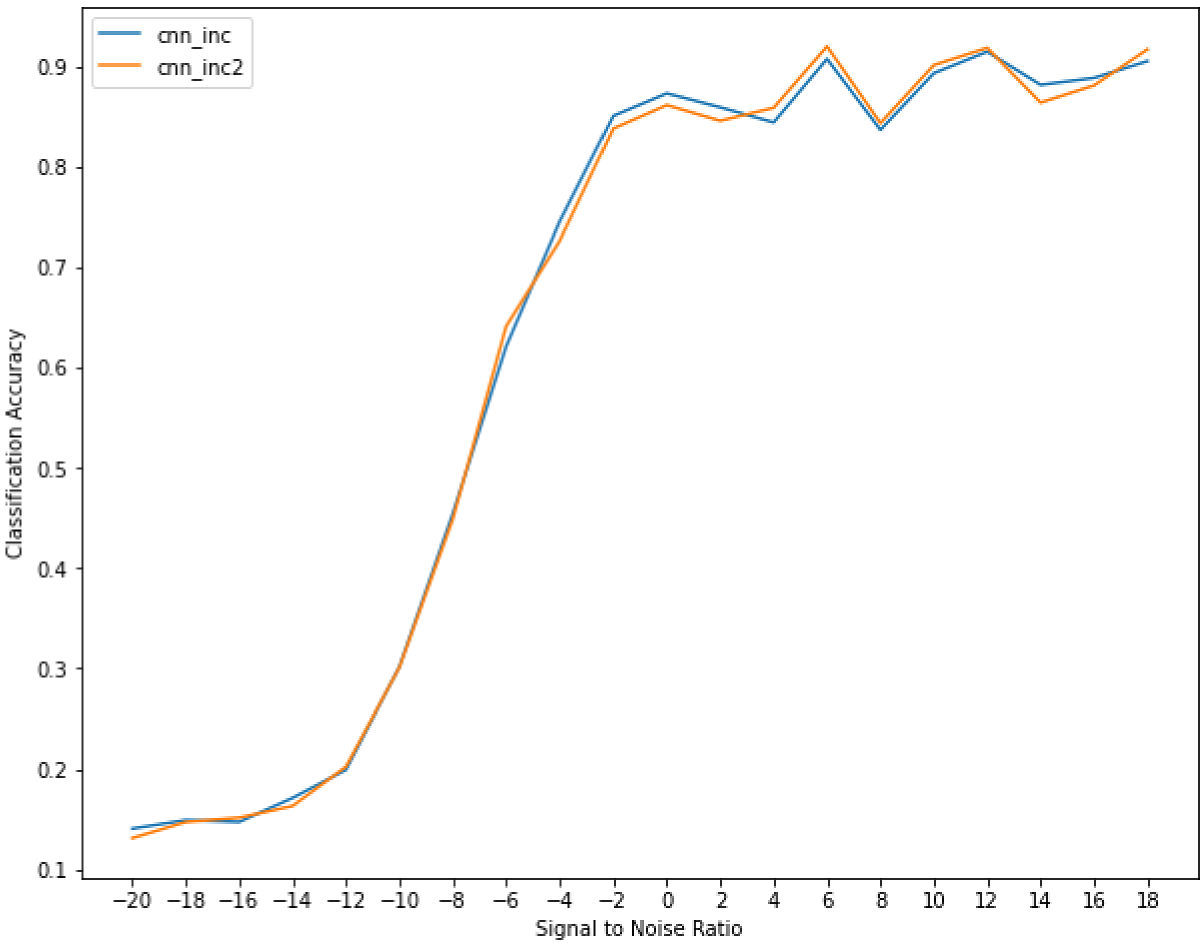
- CNNs both 2 Conv + 2 Dense, only different in number of nodes
- Cnn has ~500k, cnn2 ~ 2.5mil
- Cnn2 slightly better than cnn, but much longer to train



Inception modules

Number of inception modules stacked

- cnn_inc: 1 mod, cnn_inc2: 2 mods
- Num mods doesnt appear to be significant



Things to do

- Continue working on classification
  - 2018 radioML dataset (24 classes)
  - Try other DL models: 08645696 Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios
    - Additional CNN with constellation diagram as input to distinguish indistinguishable mod schemes
  - Try extract other features (fft, deriv, integral, moments etc) and use non-DL methods?
- Understanding and visualising radioML dataset
  - try GNUradio
  - Learn how to modulate, demodulate
  - Try generating own dataset, maybe other formats?
- Other areas to explore
  - Improving SNR (but how to measure SNR?)
  - Adversarial examples?

Useful resources

- Datasets: https://www.deepsig.ai/datasets
- Papers by O'Shea/ DeepSig
  - Radio Machine Learning Dataset Generation with GNU Radio
  - Convolutional Radio Modulation Recognition Networks
  - Over-the-Air Deep Learning Based Radio Signal Classification
- Implementations and tutorials
  - Example Classifier Jupyter Notebook: https://github.com/radioML/examples/blob/master/modulation_recognition/RML2016.10a_VTCNN2_example.ipynb
  - Student report on MR, with problem statement, codes, report: https://github.com/alyswidan/ModulationRecognition
  - Tutorial with codes for ResNet classifier: https://medium.com/gsi-technology/residual-neural-networks-in-python-1796a57c2d7