100 YEARS

Rowan University

Deep-Learned Compression for RF Modulation Classification

Yagna Kaasaragadda

Presentation Outline

- Introduction
- Related Work
- Background
- Methodologies
- Experimental Results & Discussions
- Conclusions, Reflection, and Outlook
- Reference

Introduction

What ?

- How efficiently can we compress an RF signal and learn the representation of those compressed signals in order to classify the signals efficiently. Even generate new signals.

Why?

- Almost every sensor or any IOT device generates RF signals and they are processed on the cloud, or stored on a remote server.
- Send more data bits with the same bandwidth with efficient compression and reconstruction.
- Classify the signals efficiently at the receiver’s end with less information.

How?

- Using Hierarchical Quantized Auto Encoders (HQARF) for RF signals.

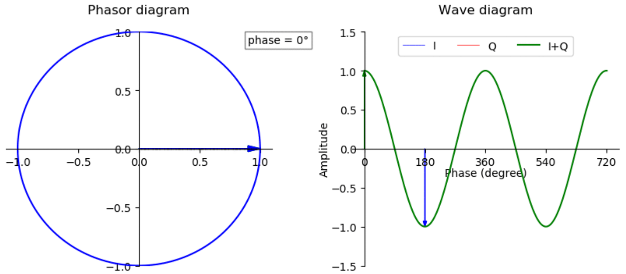
Related Work

Relevant work

- **Hierarchical Quantized Autoencoders (HQA):**
- Van den Oord, A., Vinyals, O., & Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems, 30*, 6306-6315.
- Kondo, K., Tanaka, K., & Hirai, H. (2020). Hierarchical Generative Modeling for Audio Synthesis. *arXiv preprint arXiv:2012.11861*.
- Razavi, A., Oord, A. V. D., & Vinyals, O. (2019). Vector Quantized Variational Autoencoders. *arXiv preprint arXiv:1905.10548*.
- **RF Signal Compression using Machine Learning:**
- O'Shea, T., & Hoydis, J. (2017). Deep Learning for Wireless Communications. *IEEE Transactions on Signal Processing, 65(11)*, 3551-3564.
- Ye, J., Zhang, Z., & Xu, K. (2018). Deep Learning Based Signal Processing in Wireless Communications. *IEEE Wireless Communications, 25(5)*, 59-65.
- O'Shea, T., & Hoydis, J. (2017). End-to-End Deep Learning for Physical Layer Communications. *IEEE Journal on Selected Areas in Communications, 35(10)*, 2403-2417.

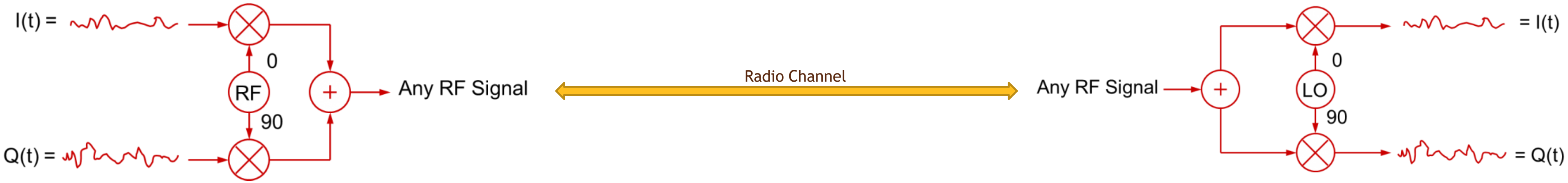
Background

- IQ Signals and Modulations.
- I – In Phase component of an RF signal
- Q – Quadrature or Out Phase component of an RF signal.



From https://en.wikipedia.org/wiki/In-phase_and_quadrature_components

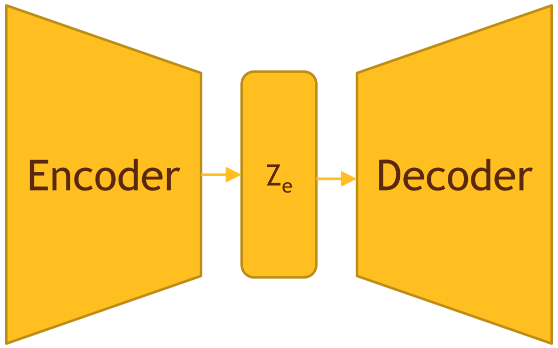
- Using the IQ values any RF modulation can be represented as a sum of the both the Signal components.
- Modulated Carrier RF = I*cos(2*π*f*t) + Q*sin(2*π*f*t)
- Complex form of RF signal: $Ae^{i\theta} = A\cos\theta + i\,A\sin\theta$



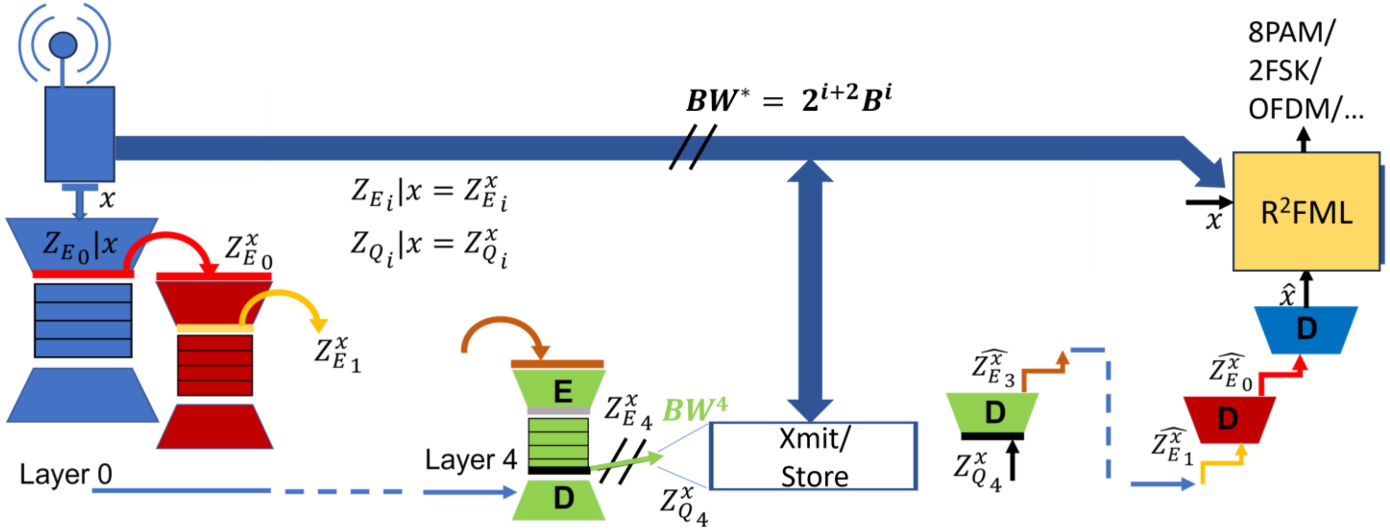
From https://www.tek.com/en/blog/quadrature-iq-signals-explained

Background (Continued..)

- Auto Encoders



- Hierarchical Quantized Auto Encoders



Encoder:

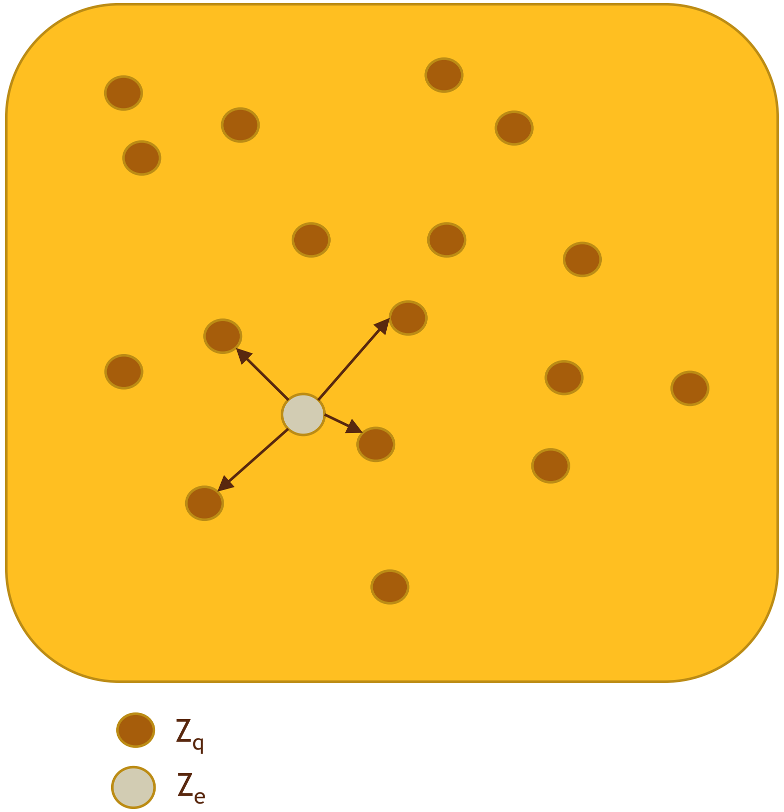
- Compresses the data into a latent space Z_e.
- Generally a series of convolution blocks

Decoder:

- Takes the Z_e as input and upscales it into the original input data.
- Generally a series of DeConv or Transpose Convolution layers.

Background (Continued..)

- Vector Quantization



- Learn a set of vectors representing the entire input dataset or entire feature set of input dataset.
- Z_q is the quantized vector.
- Z_e is the input vector.
- Each the point in the codebook is called a codeword. Each codeword is a learnable parameter.
- Pick the Z_q from the codebook that is closest to the Z_e.
- Quantization error is usually the distance between the input vector and the quantized vector.
- The distance metric is generally Euclidian distance.

Background (Continued..)

- Compression Ratio

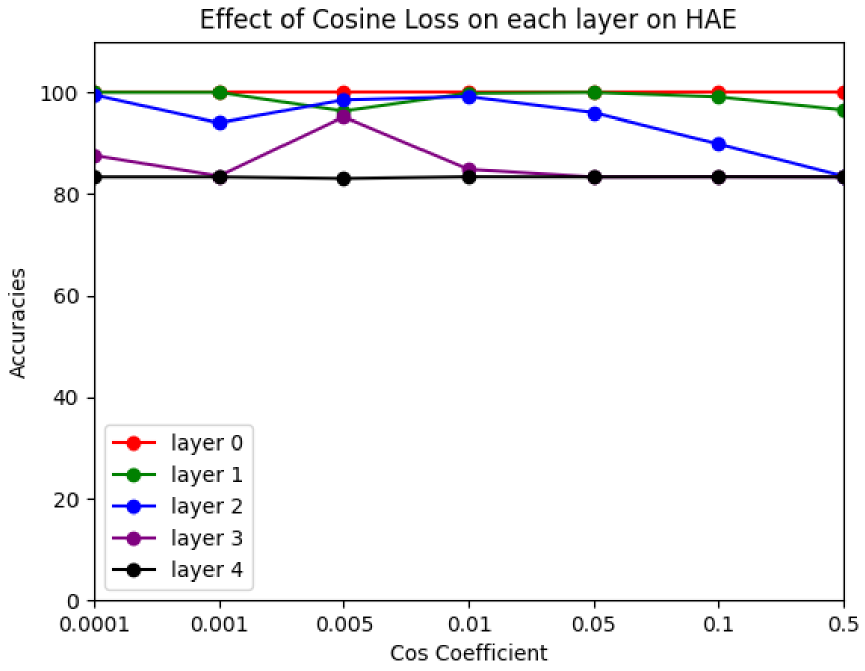
$$CR_i = \frac{B^*}{B_i} = \frac{2pH_N(X)}{d \times dim(z_{e(i)})[1]} = \frac{4.1p}{6p/2^{i+1}} = \xi \times 2^i,$$

- ξ: A constant factor equal to 1.37, derived from the other terms in the formula
- $H_N(X)$: Gaussian entropy, approximated as 2.05 for a normal distribution with unit variance
- p: Number of complex-valued samples in the original data point x.
- d: Number of bits required to represent each codeword index.
- If the codebook has 128 slots we have 128 indices which needs 7 bits to represent the codeword index.
- $dim(z_{e(i)})$ is number of features of the latent vector which is equal to the number of dimensions of each codebook vector z_q.

Methodologies

- Training the HQA.
  - First train the HAE
  - Transfer learn the HAE encoder and decoder weights to the HQA.
- Evaluating HQA
  - Train a classifier for validating the reconstructions.
  - Evaluate the HAE and HQA reconstructions.
  - Plot the confusion matrices at each compression level.
- Important Hyper parameters for a HAE.
  - Output feature dimension of z_e.
  - Number of Resnet blocks in the encoder and decoder.
  - Cosine loss coefficient.
- HAE Loss function:

$$L(x, \hat{x}) = MSE(x, \hat{x}) + CosCoeff \ * (1 - CosSim(x, \hat{x}))$$



Methodologies (Continued..)

- Get the best HAE possible.
- Now freeze the HAE’s Encoder and Decoder part and just try to learn the codebook of the HQA.
- HQA loss function:

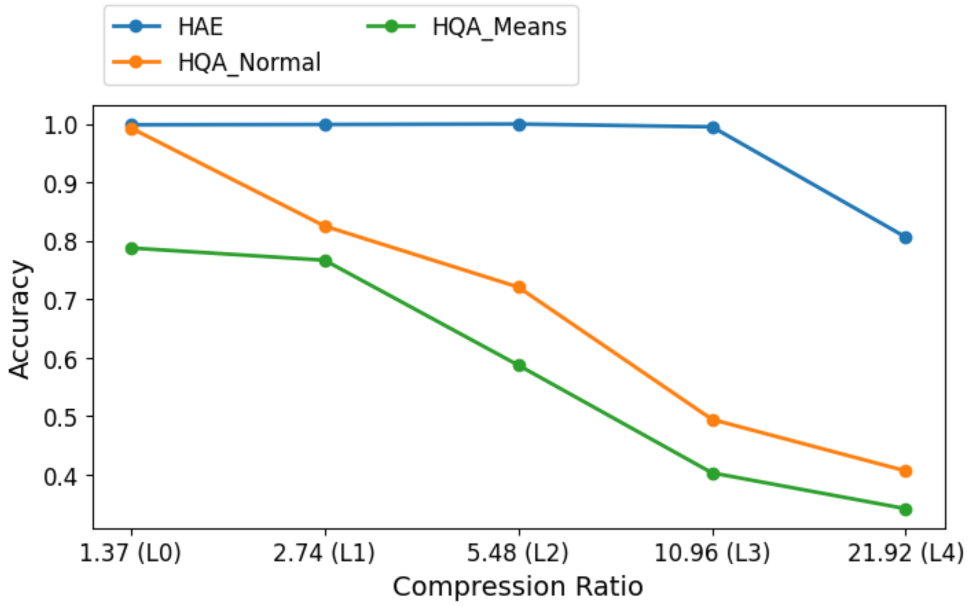
$$L(\hat{x}|x) = \mathrm{MSE}(x,\hat{x}) + Cf \times (1 - \cos(x,\hat{x})) + KL \times (p(z_q^{(i)} = j)\left(\log\left(\frac{p(z_q^{(i)} = j)}{p(z_e^{(i)} = j)}\right) + \log(K)\right)) + CL \times (p(z_q^{(i)} = j)\|\mathbf{z}_e^{(i)} - \mathbf{z}_q^{(i)}\|^2)$$

- KL Loss is the probability loss function to optimize or move the input latent vector z_e towards the codeword z_q.
- CL Loss is the commit loss which moves the quantized vector towards the input latent vector.
- In this project my focus was more on optimizing the codebook rather than optimizing the whole model.

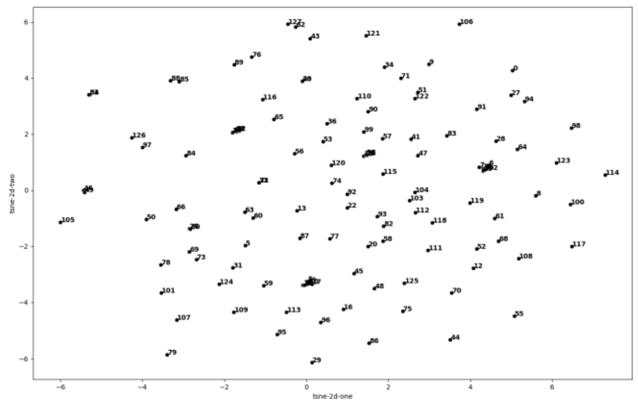
Methodologies (Continued..)

Issues for Codebook optimization

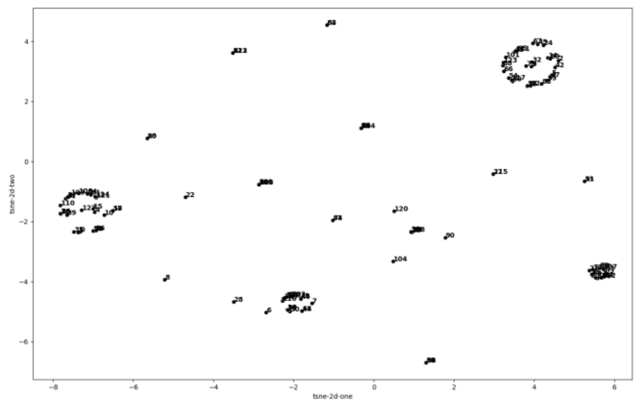
- Codebook initialization.
    - Since the codebook is a learnable parameter the initialization of the codebook can also impact the convergence of the algorithm.
    - Different Initializations
        - Uniform Distribution
        - Normal Distribution
        - From the means of the latent vectors from each class of input data.



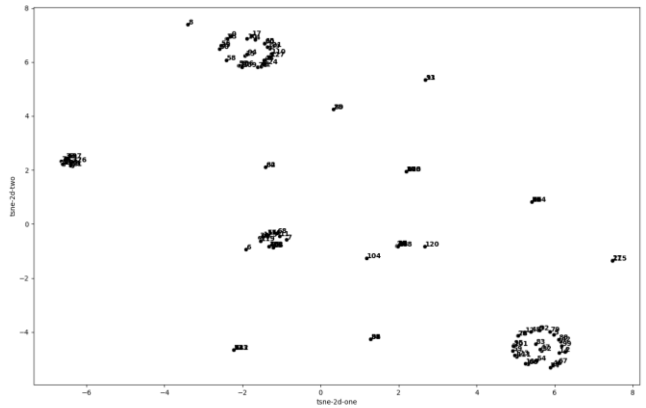
Uniform Distribution



Normal Distribution



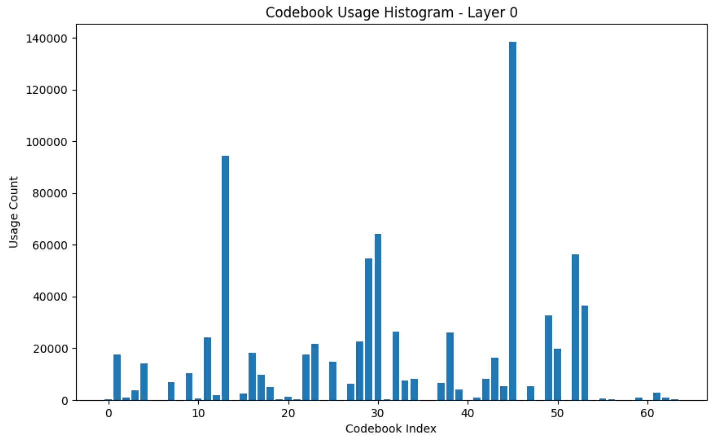
Distribution of Means of Ze's

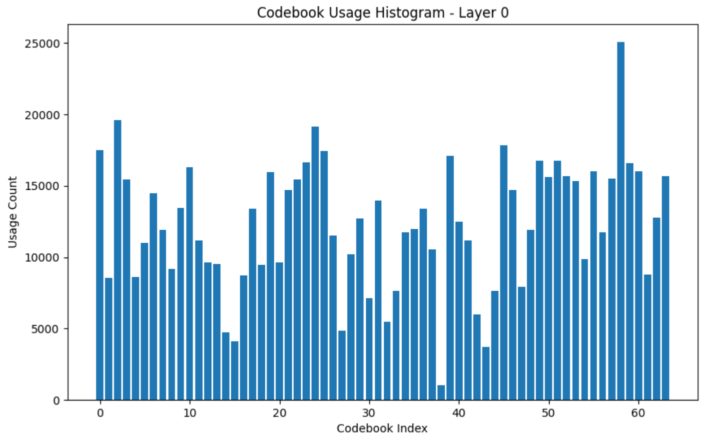


Methodologies (Continued..)

Issues for Codebook optimization

- Codebook Saturation
  - Same Indices might be used multiple times for different input data.
  - Can't explore the entire codebook dimensions.
  - Reset the least used codeword at a frequent interval during training (which can be a hyper parameter).



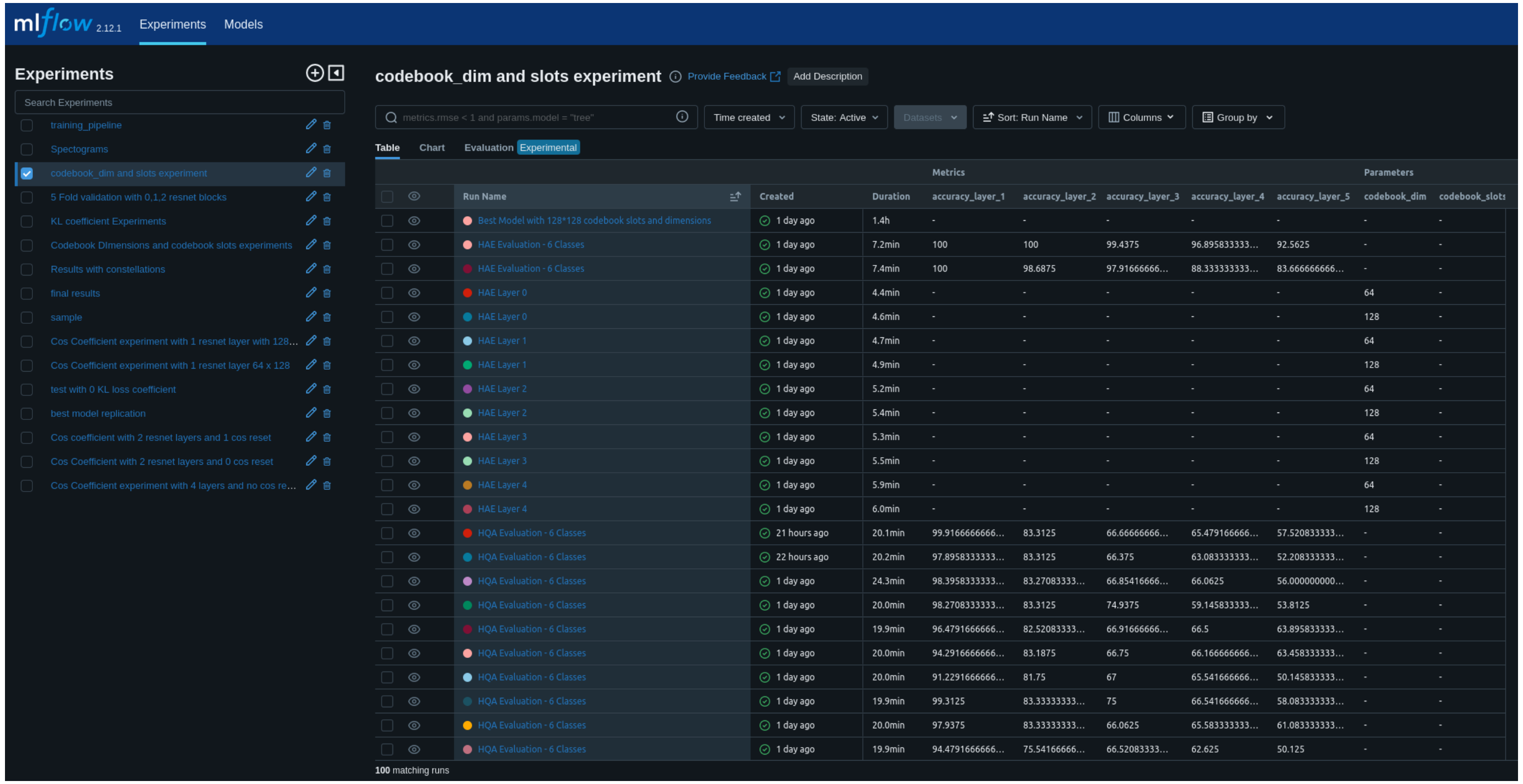


Experimental Results

- Dataset: TorchSig
    - A Digital modulated signals library which has 53 different modulations that can be synthetically generated.
    - I chose to use only 6 of the modulations. They are:
        - 4ask
        - 8pam
        - 16psk
        - 32qam-cross
        - 2fsk
        - Ofdm256
    - Dimension of the dataset 2 –channels (I and Q) since the we can’t use complex signals for neural networks we use complex2D transformation.
    - The dataset is already normalized since it’s a synthetic dataset.
    - The total number of samples of a signal for the experiments is 1024 IQ samples. Which gives the dimension of input signals as 2x1024.

Experimental Results

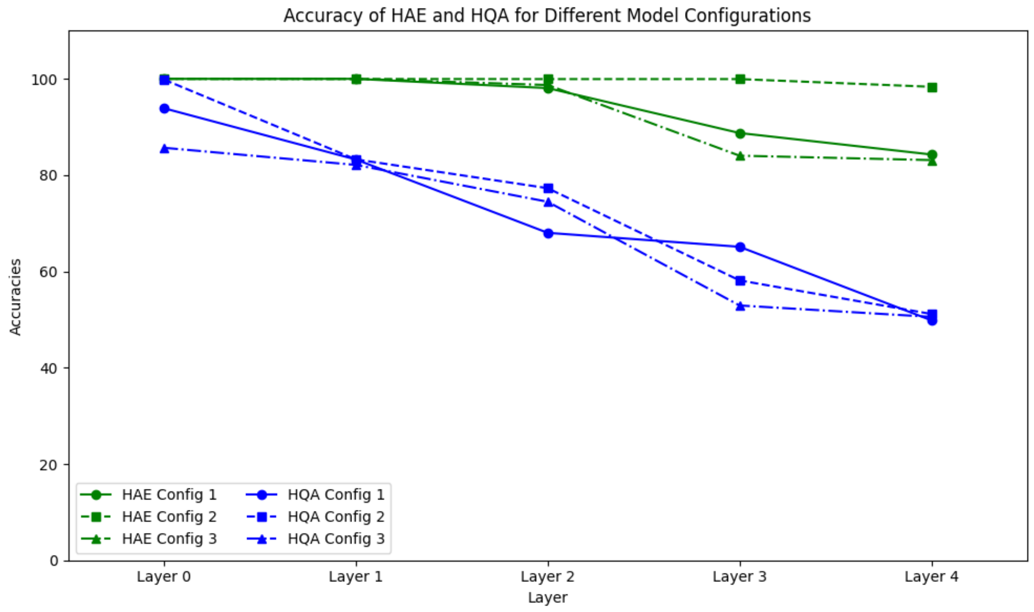
- The whole project code is written using MLOPs methodologies, to save all the articrafts like Constellation Diagrams, Spectrograms, confusion Matrices, etc.
- The HQA takes about an hour to train on only one Nvidia A6000 Gpu with IQ samples of 1024 per signal and 8000 samples per signal for about 10 epochs.



Experimental Results

Best HAE results.

- Found the optimal set of Cos Coefficient for different layers.
- Find the optimal set of Resnet blocks for each Encoder and Decoder Configuration.
- Best Cos Loss coefficient for layers 0,1,2,3,4
  [0.1,0.05,0.01,0.005,0.0001]
- These are the best model accuracies out of 5 different runs with standard deviation of around 2-3%
- Config-1
    - 0 Resnet blocks
- Config-2
    - 1 Resnet blocks
- Config-3
    - 2 Resnet blocks



Experimental Results

Compression with HQA.

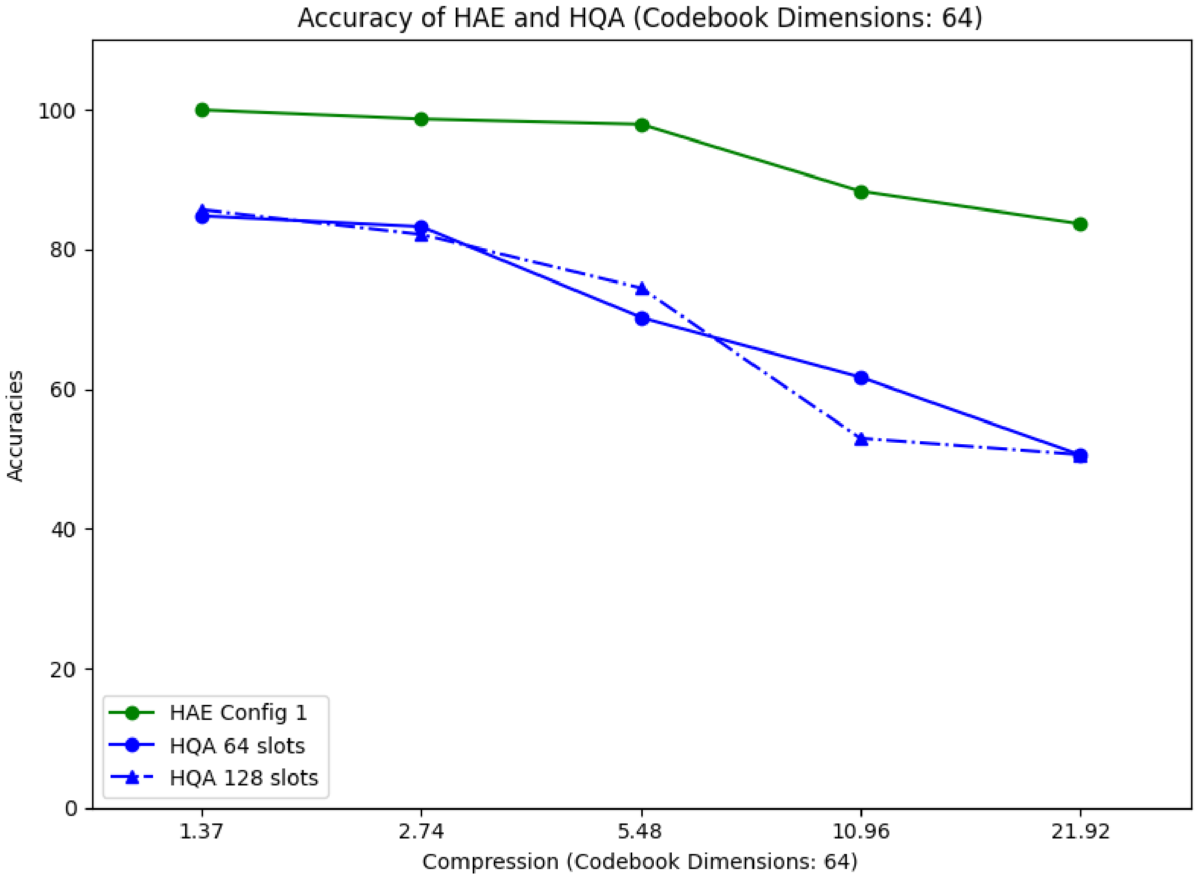
- The compression levels are calculated as in the formula yielded the following compression on different layers of model.

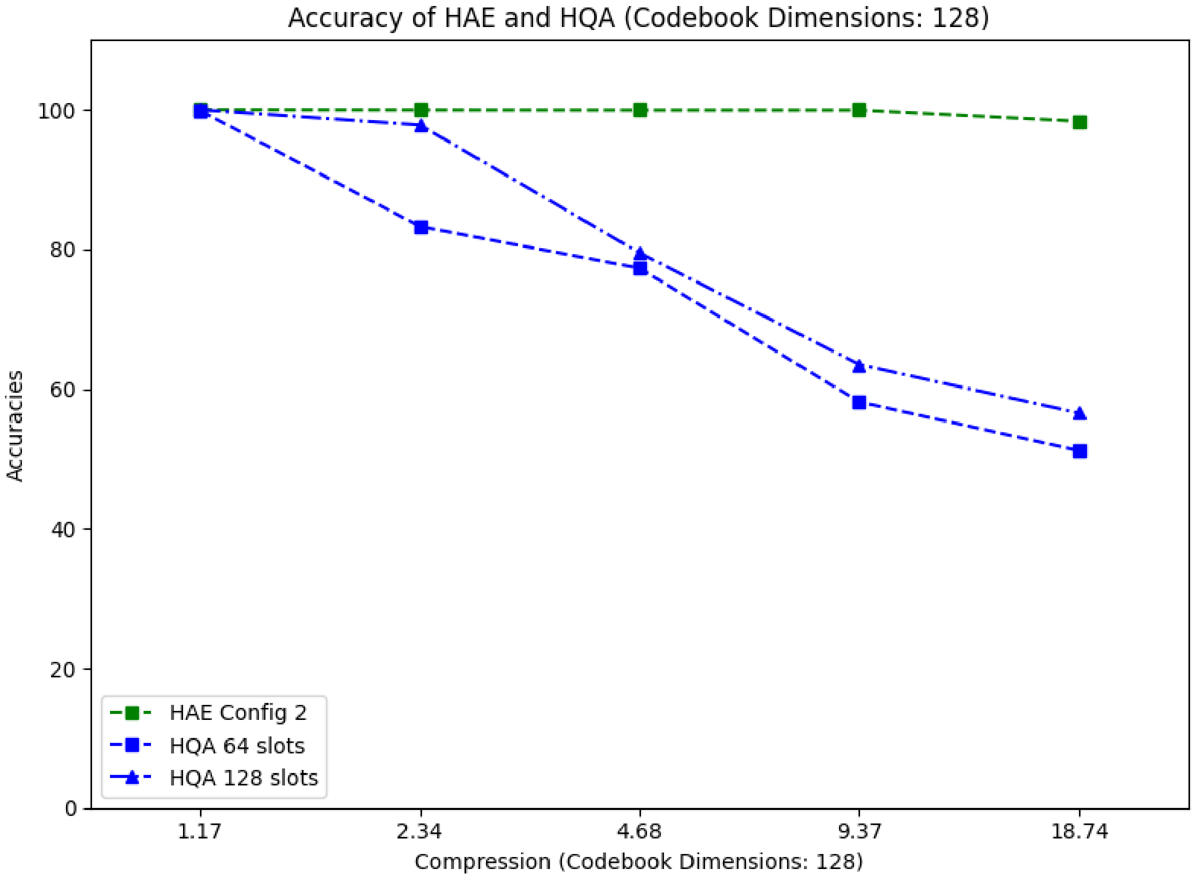
$$CR_i = \frac{B^*}{B_i} = \frac{2pH_N(X)}{d \times dim(z_{e(i)})[1]} = \frac{4.1p}{6p/2^{i+1}} = \xi \times 2^i,$$

Layer	Input Dimensions (x or z_e)	Output Dimensions (z_e)	Compression Ratio (CR_i)
L0	2 × 1024	64 × 512	1.37
L1	64 × 512	64 × 256	2.74
L2	64 × 256	64 × 128	5.48
L3	64 × 128	64 × 64	10.96
L4	64 × 64	64 × 32	21.92

Experimental Results

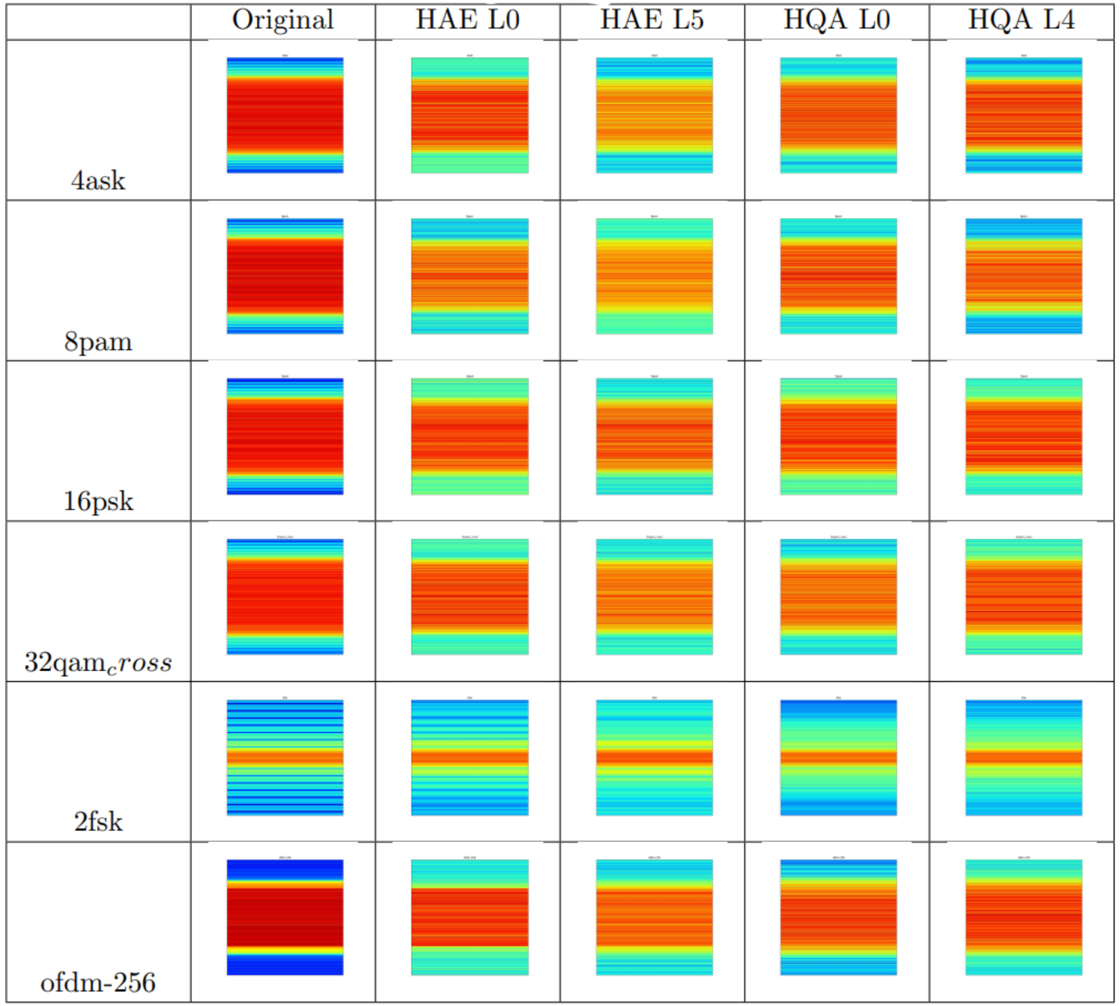
Accuracy at different compression levels and codebook Dimension configurations.



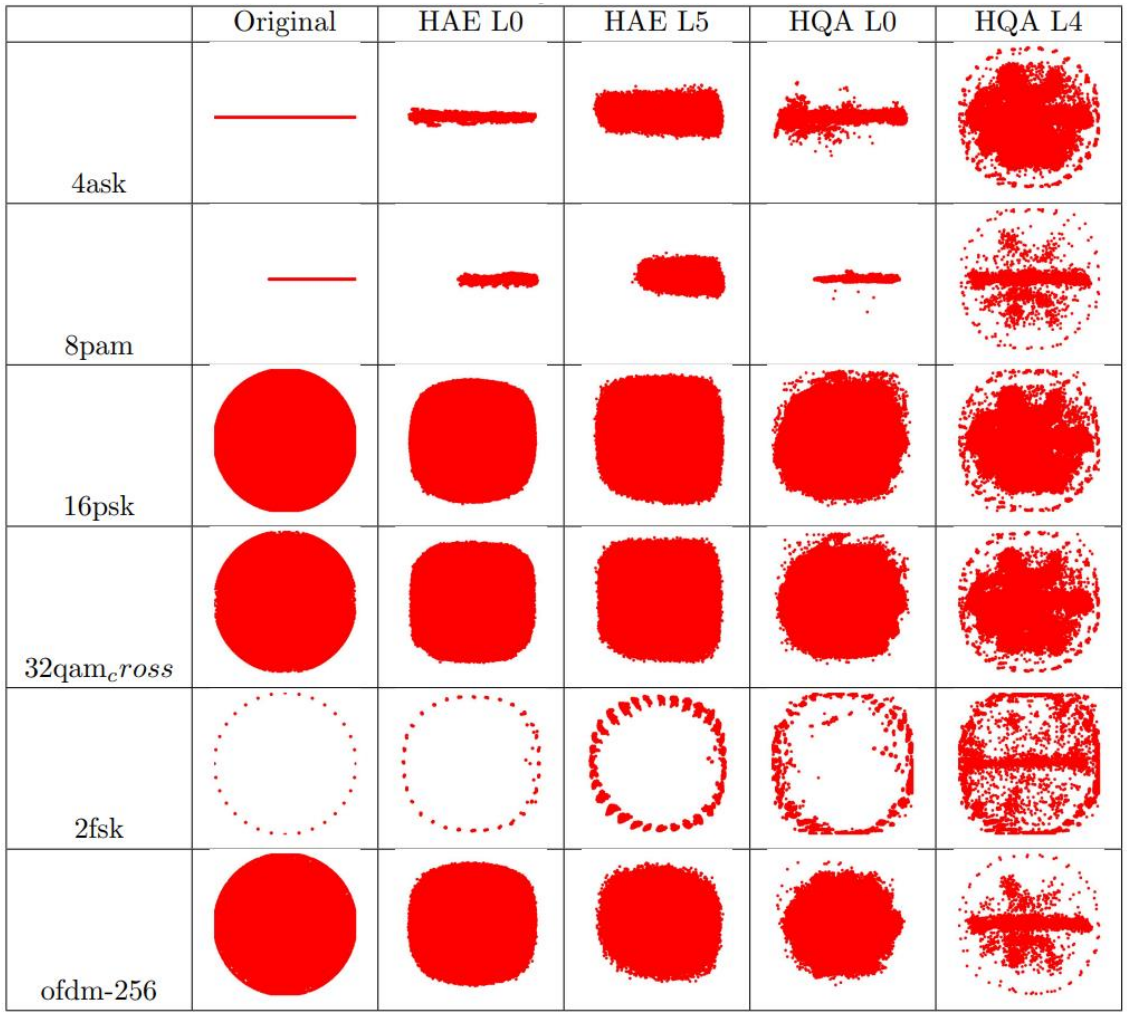


Experimental Results

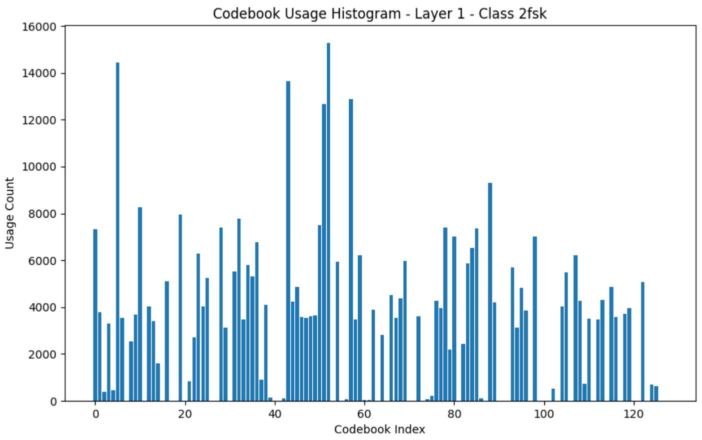
Spectrograms

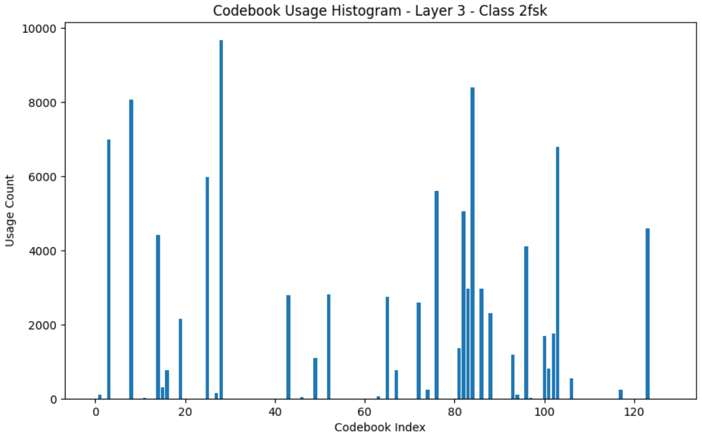


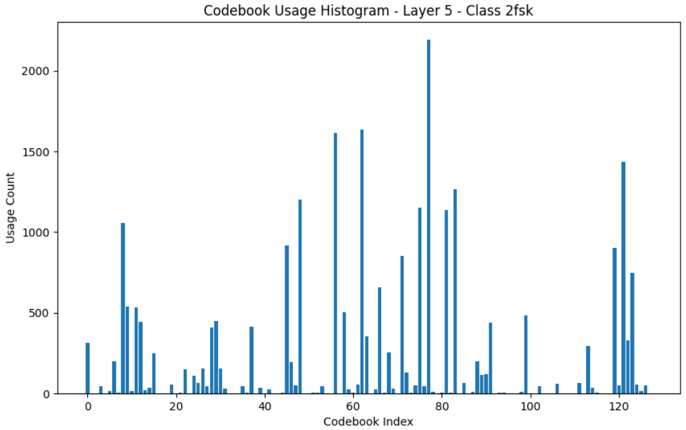
Constellations

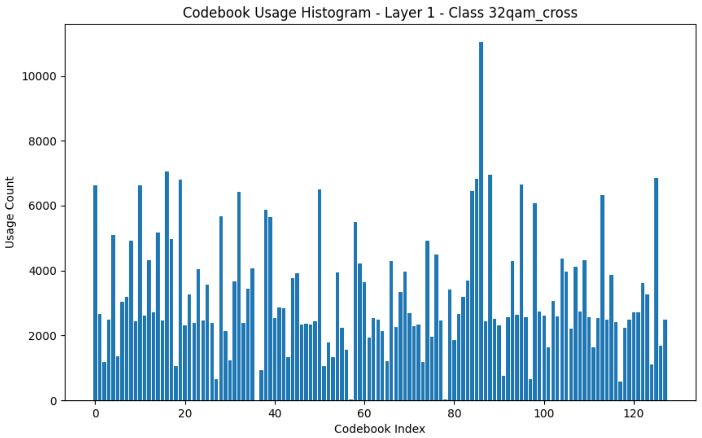


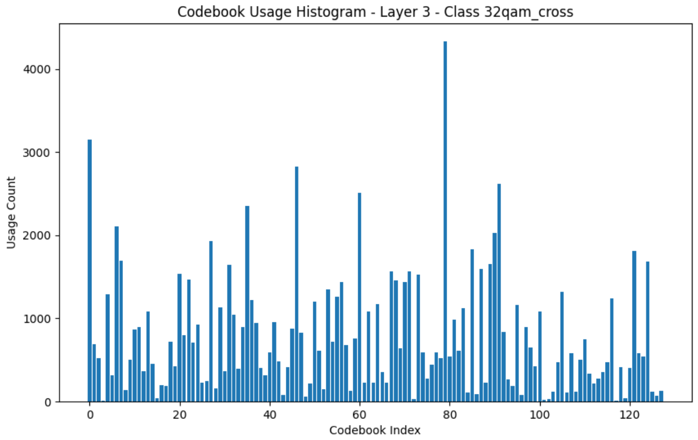
Codebook Usage of Histograms

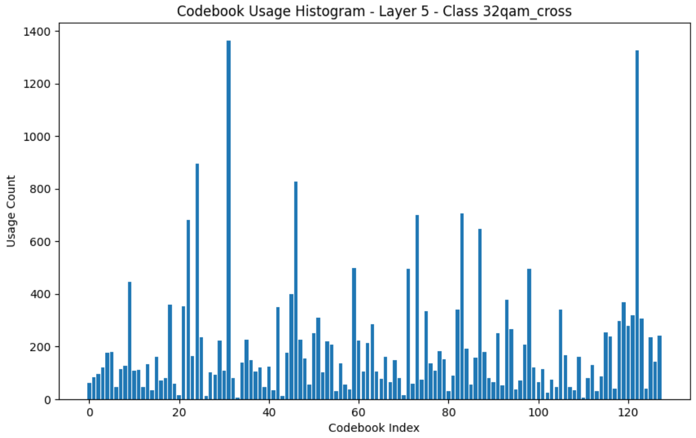












Conclusions, Reflections, & Outlooks

- Trade- off between Reconstruction Accuracy and classification
- Aim is to compress beyond certain threshold.
- Generate New signals.
- De- Noising properties autoencoders.
- Effect of different Cosine Loss on different layers

References

- Van den Oord, A., Vinyals, O., & Kavukcuoglu, K. (2017). Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems, 30*, 6306-6315.
- Kondo, K., Tanaka, K., & Hirai, H. (2020). Hierarchical Generative Modeling for Audio Synthesis. *arXiv preprint arXiv:2012.11861*.
- Razavi, A., Oord, A. V. D., & Vinyals, O. (2019). Vector Quantized Variational Autoencoders. *arXiv preprint arXiv:1905.10548*.
- O'Shea, T., & Hoydis, J. (2017). Deep Learning for Wireless Communications. *IEEE Transactions on Signal Processing, 65(11)*, 3551-3564.
- Ye, J., Zhang, Z., & Xu, K. (2018). Deep Learning Based Signal Processing in Wireless Communications. *IEEE Wireless Communications, 25(5)*, 59-65.
- O'Shea, T., & Hoydis, J. (2017). End-to-End Deep Learning for Physical Layer Communications. *IEEE Journal on Selected Areas in Communications, 35(10)*, 2403-2417.

Questions?