Deep Learning for Sparse Signal Processing

Presented by:

Rubin Jose Peter

SPC Lab

October 20, 2018

Overview

DNN for Sparse Signal Recovery

- **Learning to optimize**
  - Signal processing algorithm is approximated by a Deep Neural Network (DNN)
  - DNN requires only simple arithmetic operations to approximate the algorithm
  - Effectiveness of the proposed approach was demonstrated by implementing WMMSE algorithm using DNN
- **Sparse signal recovery using DNN: approaches**
  - Training a DNN using ground truth(y,x)
  - Training a DNN using the input/output of a sparse recovery algorithm
  - Approximating each layer of a neural network by the input/output of an iterative sparse recovery algorithm
- **Observation**
  - Performance of the DNN based sparse signal recovery depends on the architecture of the neural network and number of training data
  - **Extended Target Detection** problem: DNN based implementation may resolve boundary and block size mismatches

Generative Adversarial Networks (GAN)

Generative Adversarial Network

- Simultaneously learn two models:[1]
  - A generative model $\mathbf{G}$: captures the data distribution
  - A discriminative model $\mathbf{D}$: estimates the probability that a sample came from training data rather than $\mathbf{G}$
- Training data: $\mathbf{x} \backsim \mathbf{p_{data}}$
- Genarator distribution: $\mathbf{G}(\mathbf{z}) \backsim \mathbf{p_g}$
- $\mathbf{D}$ maximizes: the probability of assigning correct label to both training samples and samples from $\mathbf{G}$
- $\mathbf{G}$ minimizes: $\log(\mathbf{1} - \mathbf{D}(\mathbf{G}(\mathbf{z})))$

$$\begin{aligned} &\min_{\mathbf{G}} \max_{\mathbf{D}} V(\mathbf{D}, \mathbf{G}) \\ V(\mathbf{D}, \mathbf{G}) = &\mathbf{E}_{\mathbf{x} \backsim \mathbf{p_{data}}}[\log \mathbf{D}(\mathbf{x})] + \mathbf{E}_{\mathbf{z} \backsim \mathbf{p_z(z)}}[\log(\mathbf{1} - \mathbf{D}(\mathbf{G}(\mathbf{z})))] \end{aligned} \tag{1}$$

[1] Ian J. Goodfellow et al. "Generative Adversarial Networks". In: *CoRR* (2014). arXiv: 1406.2661. URL: http://arxiv.org/abs/1406.2661.

Generative Adversarial Network

- For G is fixed, the optimal discriminator D is

$$\mathbf{D}^*_G(\mathbf{x}) = \frac{\mathbf{p_{data}}(\mathbf{x})}{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})} \tag{2}$$

  - Proof

$$V(\mathbf{D}, \mathbf{G}) = \int_{\mathbf{x}} \mathbf{p_{data}}(\mathbf{x}) \log(\mathbf{D}(\mathbf{x})) + \mathbf{p_g}(\mathbf{x}) \log\left(\mathbf{1} - \mathbf{D}(\mathbf{x})\right)\mathbf{dx} \tag{3}$$

    - Maximum of $a\log(y) + b\log(1-y)$ is at $\frac{a}{a+b}$ in $y \in [0,1]$

- $\mathbf{D}$ maximizes $P(y|\mathbf{x})$

- Y indicates whether $\mathbf{x}$ from $\mathbf{p_g}$ or $\mathbf{p_{data}}$

Generative Adversarial Network

- Cost function during the training of generator

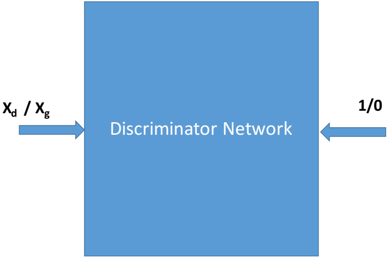
$$\mathbf{C}(\mathbf{G}) = \max_{D} V(\mathbf{D}, \mathbf{G})$$
$$= \mathbf{E}_{\mathbf{x} \backsim \mathbf{p_{data}}} \log(\frac{\mathbf{p_{data}}(\mathbf{x})}{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}) + \mathbf{E}_{\mathbf{x} \backsim \mathbf{p_g}} \log(\frac{\mathbf{p_{data}}(\mathbf{x})}{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}) \tag{4}$$

- The global minimum of $\mathbf{C}(\mathbf{G})$ is achieved if and only if $\mathbf{p_g} = \mathbf{p_{data}}$
- $C(G) = -\log 4$
  - Proof

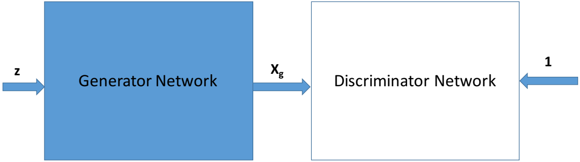
$$\mathbf{C}(\mathbf{G}) = \max_{D} V(\mathbf{D}, \mathbf{G})$$
$$= -\log 4 + KL(\mathbf{p_{data}} || \frac{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}{\mathbf{2}}) + KL(\mathbf{p_g} || \frac{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}{\mathbf{2}}) \tag{5}$$

- If **G** and **D** have enough capacity, and at each stage of the training, the discriminator is allowed to reach its optimum given **G**,then $\mathbf{p}_g$ converges to $\mathbf{p_{data}}$

- **Training of Discriminator**
- $\{x_d^{(1)}, x_d^{(2)} .... x_d^{(m)}\}$: samples from data distribution (labels 1)
- $\{x_g^{(1)}, x_g^{(2)} .... x_g^{(m)}\}$: samples from generative networks (labels 0)



- **Training of Generator**
- Discriminator is frozen
- Generator Network is trained with the desired label at the discriminator output as 1



Generative Adversarial Network

for Number of training iterations **do**

  for k steps **do**

    Sample minibatch of m noise samples $\{z^{(1)}, z^{(2)} .... z^{(m)})\}$ from noise prior $\mathbf{p_z}(\mathbf{z})$

    Sample minibatch of m example $\{x^{(1)}, x^{(2)} .... x^{(m)})\}$ from data generation distribution $\mathbf{p_{data}}(\mathbf{x})$

    Update the discriminator by ascending its stochastic gradient

$$\nabla_{\theta_D}\{\frac{1}{m}\sum_{i=1}^{m}[\log \mathbf{D}(x^i) + \log(1 - \mathbf{D}(\mathbf{G}(z^i)))]\} \tag{6}$$

  end for

  Sample minibatch of m noise samples $\{z^{(1)}, z^{(2)} .. z^{(m)})\}$ from noise prior $\mathbf{p_z}(\mathbf{z})$

  Update the generator by descending its stochastic gradient

$$\nabla_{\theta_g}\{\frac{1}{m}\sum_{i=1}^{m}[\log(1 - \mathbf{D}(\mathbf{G}(z^i)))]\} \tag{7}$$

end for

Generation of Handwritten Digits using GAN

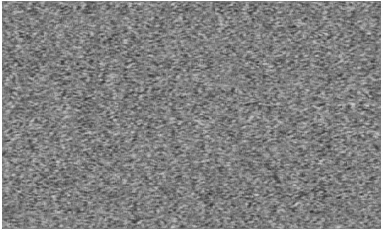


Figure: Iteration # 1

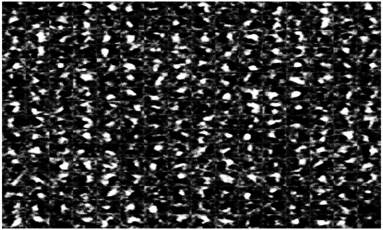


Figure: Iteration # 20

Figure: Iteration # 40

Figure: Iteration # 100

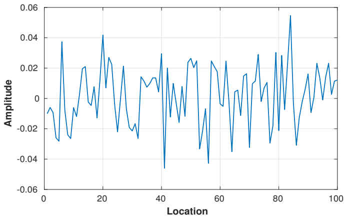


Figure: Iteration # 1

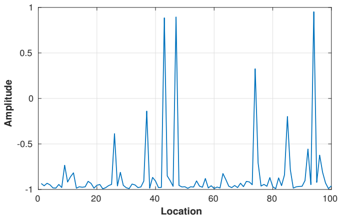


Figure: Iteration # 10

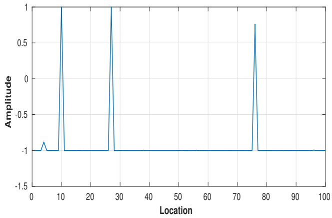


Figure: Iteration # 20

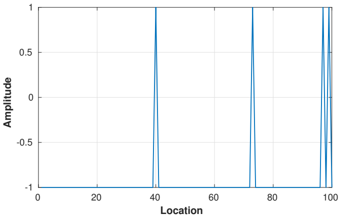


Figure: Iteration # 30

Compressive Sensing Using Generative Models

Compressive Sensing Using GAN

- System model

$$\begin{aligned} \mathbf{y} = A\mathbf{x} + \mathbf{n} \quad \mathbf{y} \in R^{m\times 1}, A \in R^{m\times n}, \mathbf{x} \in R^{n\times 1} \\ ||\mathbf{x}||_0 = k \quad k \in \{1, 2, 3...\} \end{aligned} \tag{8}$$

- The generative models learns a mapping from low dimensional representation space $\mathbf{z} \in R^k$ to the high dimensional sample space $\mathbf{G}(\mathbf{z}) \in R^n$
- Proposed algorithm: Find a mapping between observation vectors $\mathbf{y}$ and the vectors in the latent space $\mathbf{z}$
- Mapping between measurement space and latent space is obtained by minimizing the following loss function[2]

$$V(\mathbf{z}) = ||\mathbf{AG}(\mathbf{z}) - \mathbf{y}||^2 \tag{9}$$

[2] Ashish Bora et al. "Compressed Sensing using Generative Models". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. 2017, pp. 537–546. URL: http://proceedings.mlr.press/v70/bora17a.html.

Sparse Signal Recovery using GAN

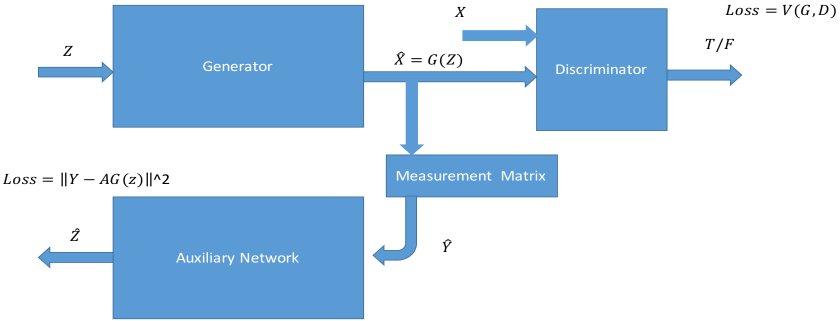
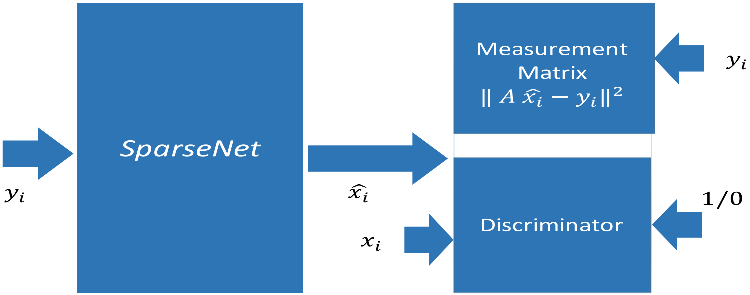


Figure: Compressive Sensing using GAN

New Framework for Sparse Signal Processing

New Framework for Sparse Signal Recovery



- *SparseNet* : DNN for sparse signal recovery
- Discriminator network can ensure sparsity
- May be useful to ensure more general features like block sparsity

Sparse Signal Recovery using New Framework

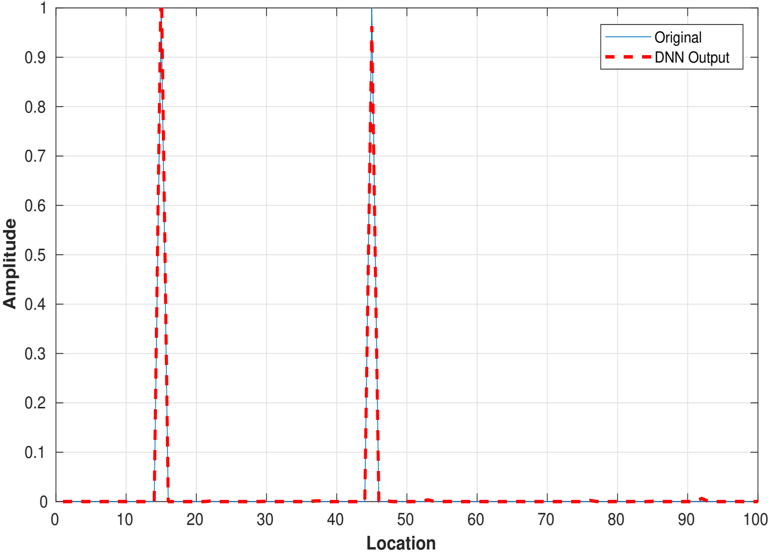
- Training of *Discriminator*
  - Discriminator : Ensures sparsity of $\mathbf{x}$
  - Trained using $\{$Data,Label$\} = \{\{\hat{\mathbf{x}}_{\mathbf{i}},0\},\{\mathbf{x}_{\mathbf{k}},1\}.... \}$

- Training of *SparseNet*
  - Discriminator is frozen
  - DNN is trained by simultaneously minimizing the loss function

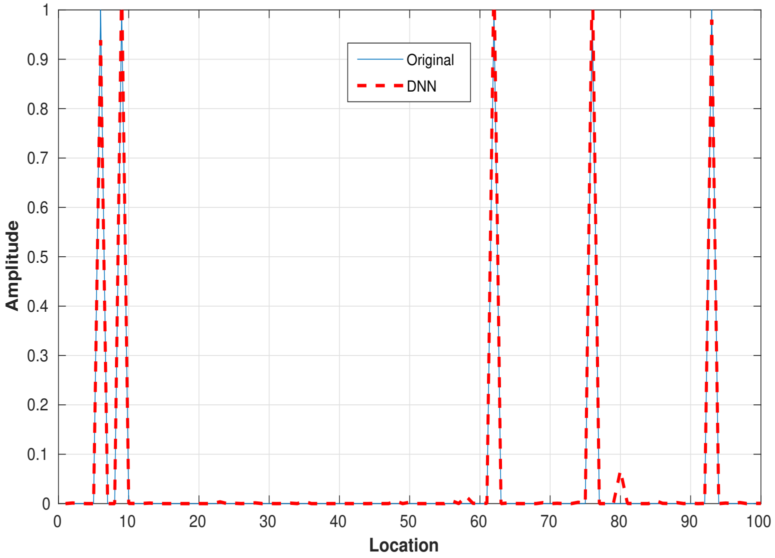
$$\begin{gathered}\min_{\mathbf{G}} \lambda_1 V(\mathbf{D},\mathbf{G}) + \lambda_2 \mathbf{E}_{\mathbf{y}\backsim\mathbf{p_y}} ||\mathbf{y} - \mathbf{AG}(\mathbf{y})||^2 \\ V(\mathbf{D},\mathbf{G}) = \mathbf{E}_{\mathbf{y}\backsim\mathbf{p_y}}[\log(\mathbf{1} - \mathbf{D}(\mathbf{G}(\mathbf{y})))]\end{gathered} \tag{10}$$

  - $\lambda_1 \& \lambda_2$: Loss weights can be specified during the training phase

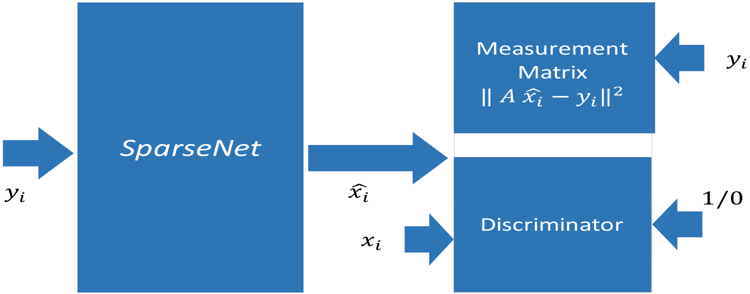
Sparse Signal Recovery



Sparse Signal Recovery

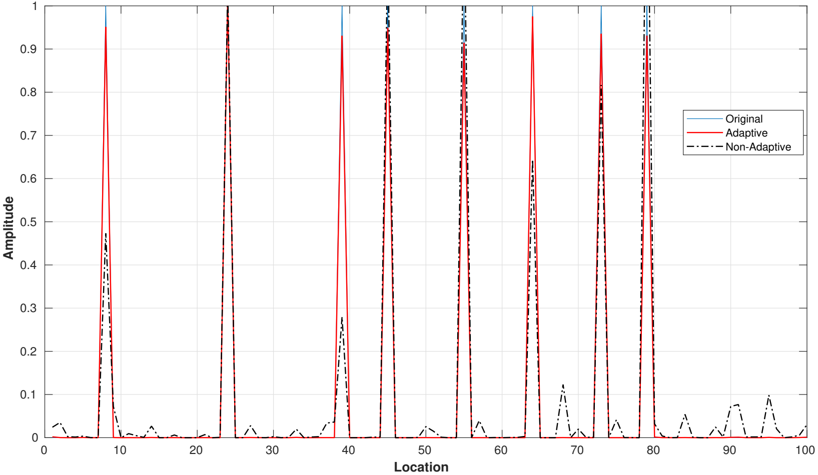


Adaptive Signal Recovery

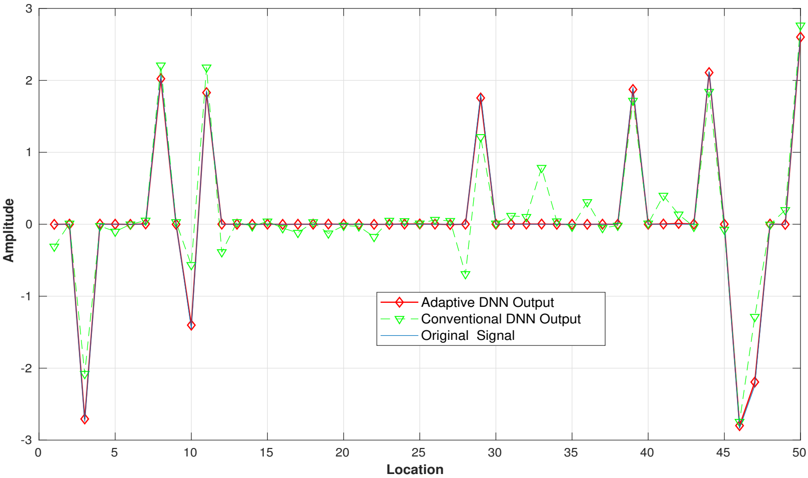


- Training of **G** does not require unknown sparse vector **x**
- Update the weights and biases of the DNN during signal recovery phase

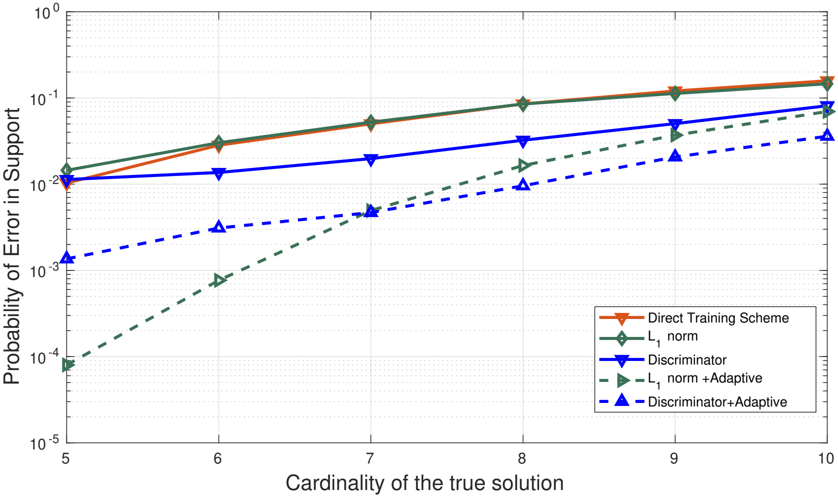
Comparison of Adaptive Vs Non-Adaptive



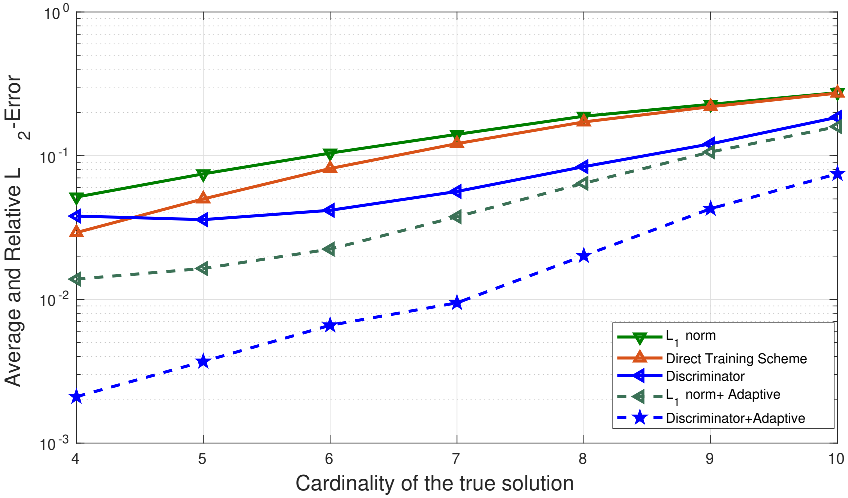
Comparison of Adaptive Vs Non-Adaptive



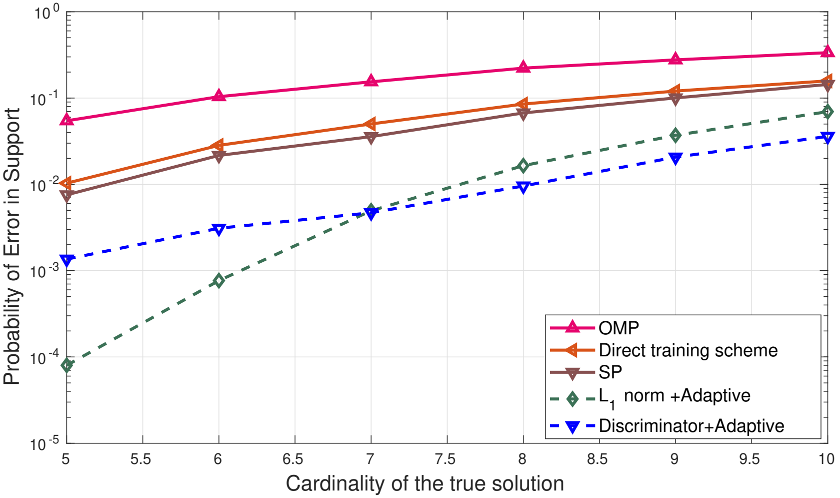
Comparison of Adaptive Vs Non-Adaptive



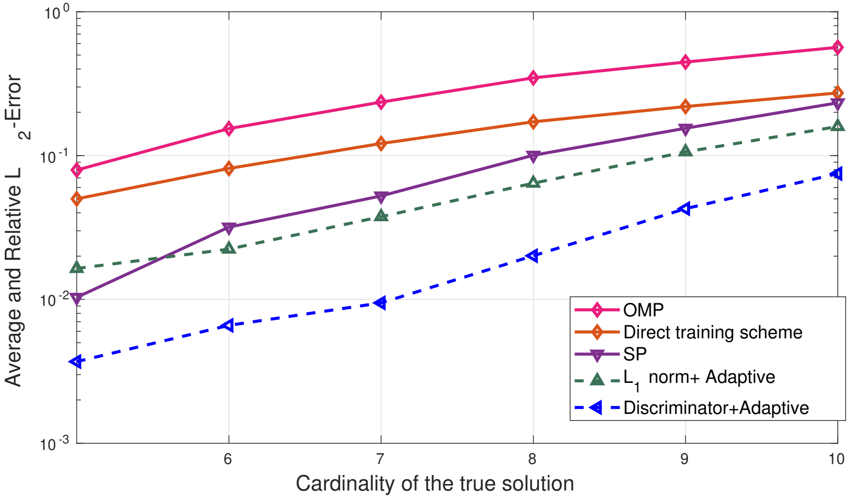
Comparison of Adaptive Vs Non-Adaptive



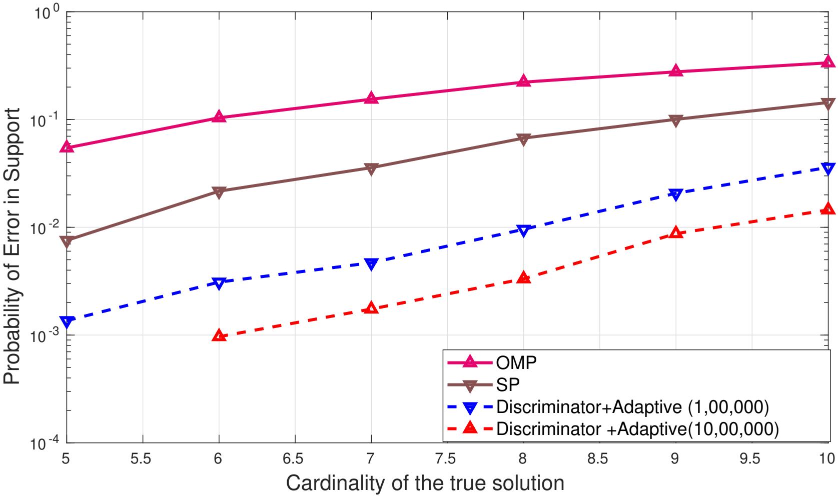
Comparison with Other Algorithms



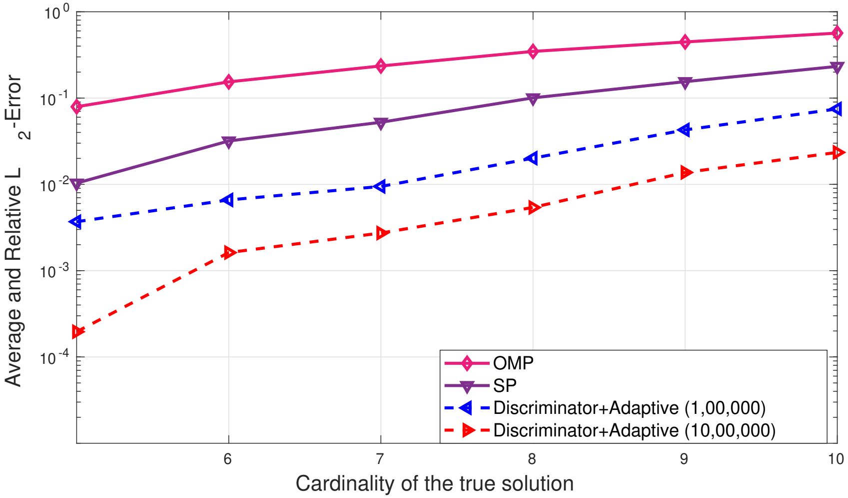
Comparison with Other Algorithms



Training Sets with Different Cardinality



Training Sets with Different Cardinality



Intuitive Explanation Under Bayesian Framework

- **Signal Model**:

$$\mathbf{y} = A\mathbf{x} + \mathbf{n} \quad \mathbf{y} \in R^{m\times 1}, A \in R^{m\times n}, \mathbf{x} \in R^{n\times 1},$$
$$||\mathbf{x}||_0 \leq \; K \quad \mathbf{n} \sim \mathbf{N}(\mathbf{0}, \frac{\mathbf{I}}{\lambda}) \tag{11}$$

- **Likelihood** term is given by,

$$p(\mathbf{y}|\mathbf{x},\lambda) = (\frac{\lambda}{2\pi})^{\frac{m}{2}} \exp(-\frac{\lambda}{2}||\mathbf{y} - A\mathbf{x}||^2)$$
$$\log(p(\mathbf{y}|\mathbf{x},\lambda)) = -\frac{\lambda}{2}||\mathbf{y} - A\mathbf{x}||^2 + f(\lambda) \tag{12}$$

- **Maximum Likelihood Estimation** of **x** with sparsity constraint is

$$\hat{\mathbf{x}} = \underset{\mathbf{x}\in\mathbf{S}}{argmax} \quad p(\mathbf{y}|\mathbf{x},\lambda)$$
$$S = \{\mathbf{x} : ||\mathbf{x}||_0 \leq \; K \; \} \tag{13}$$

- For G is fixed, the optimal discriminator D is

$$\mathbf{D}^*_G(\mathbf{x}) = \frac{\mathbf{p_{data}}(\mathbf{x})}{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})} \tag{14}$$

- The global minimum of $\mathbf{C}(\mathbf{G})$ is achieved if and only if $\mathbf{p_g} = \mathbf{p_{data}}$.

$$\begin{aligned}\mathbf{C}(\mathbf{G}) &= \max_D V(\mathbf{D}, \mathbf{G}) \\ &= -\log 4 + KL(\mathbf{p_{data}} || \frac{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}{\mathbf{2}}) + KL(\mathbf{p_g} || \frac{\mathbf{p_g}(\mathbf{x}) + \mathbf{p_{data}}(\mathbf{x})}{\mathbf{2}})\end{aligned} \tag{15}$$

- Discriminator ensures $\mathbf{p_g} = \mathbf{p_{data}} \implies \hat{x} = G(y) \in S$

- Optimization problem during the training phase becomes

$$\min_{\mathbf{G}:\mathbf{G}(\mathbf{y})\in S} (-log4)\lambda_1 + \lambda_2 \frac{1}{m} \sum_{i=1}^{m} ||\mathbf{y_i} - \mathbf{AG}(\mathbf{y_i})||^2 \tag{16}$$

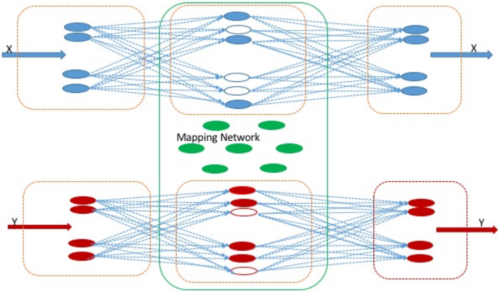
m : Number of samples in a minibatch

- Above cost function is proportional to the log likelihood of $\{\mathbf{y_1}, \mathbf{y_2} \dots \mathbf{y_m}\}$
- $\mathbf{P_X}(x)$: uniform prior over S
- The new framework tries to give a MAP estimate of $\mathbf{x}$ with prior distribution $\mathbf{P_X}(x)$ or ML estimate on the set of k sparse vectors

Coupled Dictionary Learning

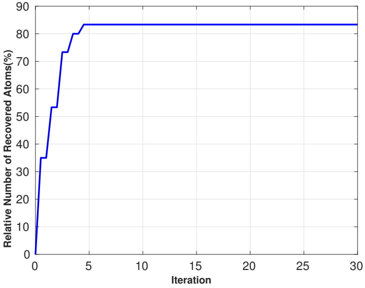
Coupled Dictionary Learning using DNN

- Dictionary Learning of **x** and **y**
  - $\mathbf{z_x}$: Sparse representation of **x**
  - $\mathbf{z_y}$: Sparse representation of **y**
- Coupled Dictionary Learning
  - Train a mapping network between $\mathbf{z_x}$ and $\mathbf{z_y}$
  - $\mathbf{z_x}$: Sparse representation of **x** and **y** with respect to coupled dictionary
  - Dictionary for **x** :Decoding network of **x**
  - Dictionary for **y** :Decoding network of **y** and mapping network



$$\% \text{ of Recovered Atoms} = \frac{R}{n}100, \quad R = \sum_{i=1}^{n} \mathbb{1}_{(x<.01)}, \quad x = 1 - \max(\ \hat{\mathbf{d}}_{\mathbf{i}}{}^{T}\mathbf{d}_{\mathbf{j}}\ )$$

$$\text{Average representation error, } E = \frac{1}{N}\sum_{i=1}^{N} \frac{||\mathbf{Y}_{\mathbf{i}} - \hat{\mathbf{D}}\hat{\mathbf{Z}}_{\mathbf{i}}||^2}{m} \tag{17}$$



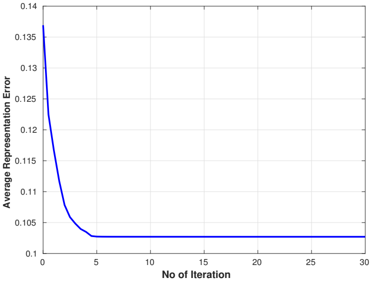
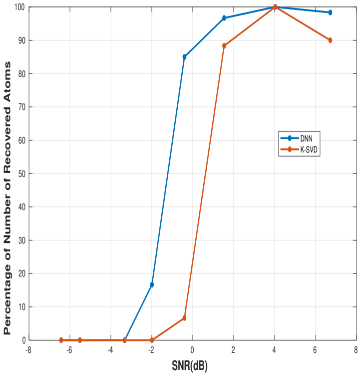


Figure: K-SVD



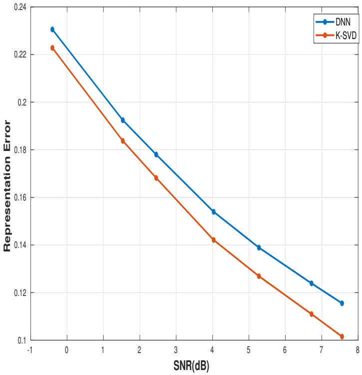
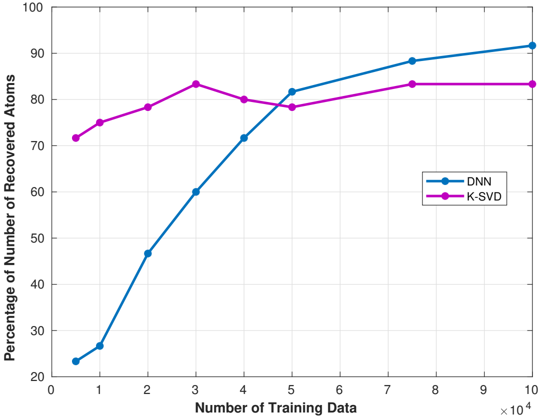
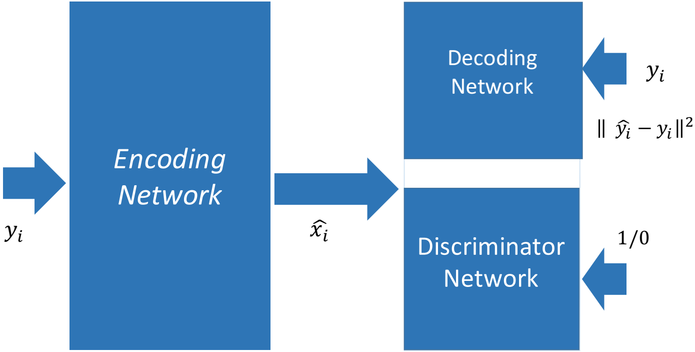


Figure: K-SVD / DNN

Dictionary Learning using DNN



Dictionary Learning under new Framework



- Decoding network is a single layer MLP with linear activation functions

Conclusion

- The new framework allows to update the inverse function during testing phase (Adaptive Signal Recovery)
- The proposed discriminator based scheme can be extended for arbitrary prior distribution
- More general features like block sparsity may be ensured using adversarial training
- The new framework may be useful for other sparse signal processing applications like dictionary learning, coupled dictionary learning etc.