Active Learning for Pixel-wise Prediction

May 22nd, 2020

Email: abdill@andrew.cmu.edu

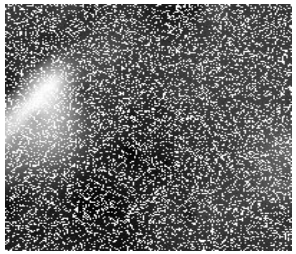
Overview of Presentation

1. Depth Estimation Task
2. Overview of Problem Set-up
3. REINFORCE Approach
4. A2C Approach
5. High Dimensional Action Space Approach
6. Gradually Eliminating Masking Network
7. Takeaways and Plans Moving Forward

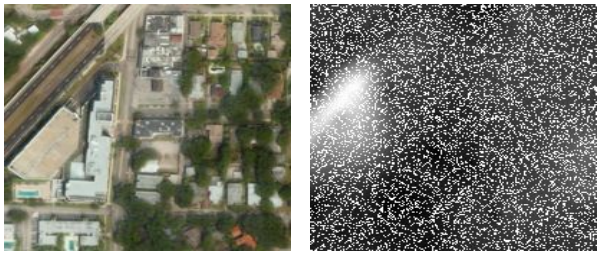
Depth Estimation Task with Active Learning

Input Image

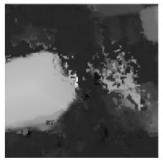
Masking



Concat



Prediction

Error = ( - )^2

Prediction

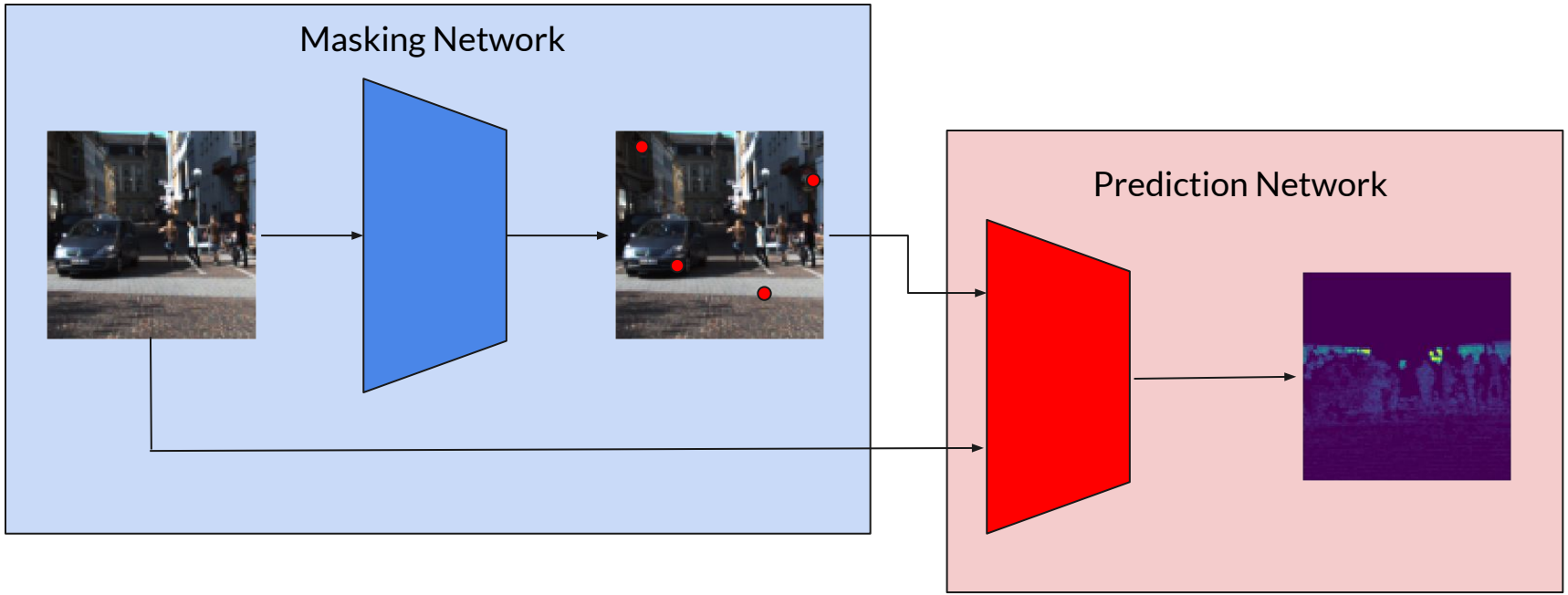
Ground Truth

KITTI Dataset Details and Preprocessing [1]

- 38,323 training images and 3,347 testing images.
- Initial images are 374 x 1238 pixels and have 3 channels intended for training self-driving cars.
- 350 by 350 subregions are cropped out.
- These square regions are downsampled to be 100 by 100 pixels in order to facilitate faster training.

General Problem Set-Up

Two Networks Cooperate to Predict the Output



Measures of Success

- MSE for per pixel depth prediction: $\frac{1}{nm}\sum_{i=0}^{n}\sum_{j=0}^{m}(D_{ij}-\hat{D}_{ij})^2$
- Count of nonzero elements in the mask: $\sum_{i=0}^{n}\sum_{j=0}^{m}\mathbb{I}\{M_{ij}\neq 0\}$

Three Essential Questions for this Approach

- How do we define the masking network?
- How do we train this model jointly?
- Can we use this leverage this approach to gradually use no sparse depth information?

Reinforcement Learning Perspective

- While it is easy to see how this problem could benefit from RL with its focus on learning from samples of frequently discrete action spaces, it's harder to specify the rest of the Markov Decision Process that RL relies on.

Reinforcement Learning Perspective

Essential Questions:

1. **What are the states?** $s_t^{(n)}$
2. **What are the actions?** $\mathcal{A}$
3. **What are the rewards?** $r_t^{(n)}$

What are the states?

$$s_0^{(n)} = \{I^{(n)}, \emptyset\}$$

$$s_1^{(n)} = \{I^{(n)}, D \odot M\}$$

What are the actions?

$$\mathcal{A} = \{M \in \mathbb{B}^{n \times n}\}$$

What are the rewards?

$r_0^{(n)} :=$ Negative MSE without sparse input

$r_1^{(n)} :=$ Negative MSE with sparse input plus number of pixels selected by the mask

REINFORCE Approach

REINFORCE: Let's try Monte Carlo Sampling [7]

- Measure of performance: $\mathcal{L}(M)$
- Probability distribution: $P_\theta(M)$
- Parameters we're optimizing over: θ
- Loss function we want to optimize with gradient descent:

$$\mathbb{E}_{M \sim P_\theta(M)}[\mathcal{L}(M)]$$

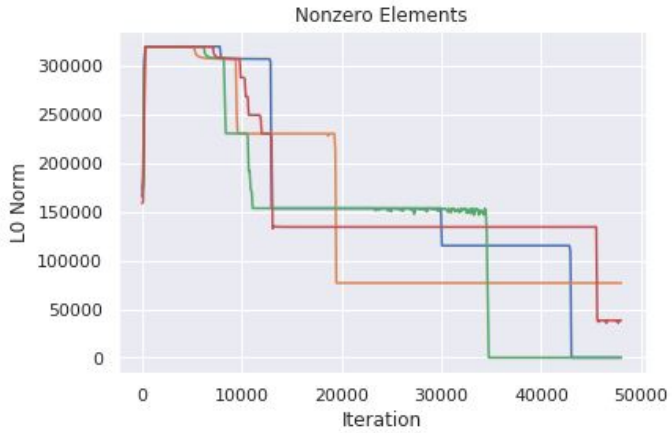
REINFORCE: Let's try Monte Carlo Sampling

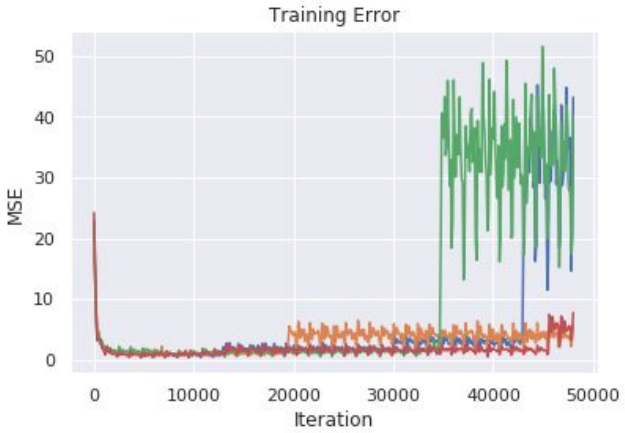
We can optimize this function with the following gradient:

$$\mathbb{E}_{M \sim P_\theta(M)} \nabla_\theta \log P_\theta(M)[\mathcal{L}(M)]$$

Note that the measure of success is never differentiated. We can therefore include any regularizer and error function on the training data without backpropping through the prediction network.

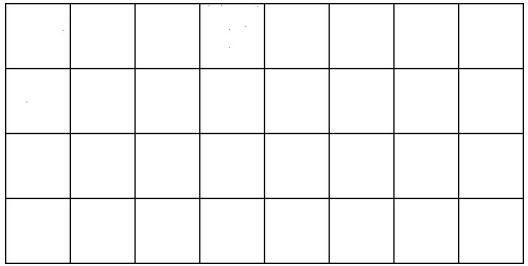
REINFORCE Results



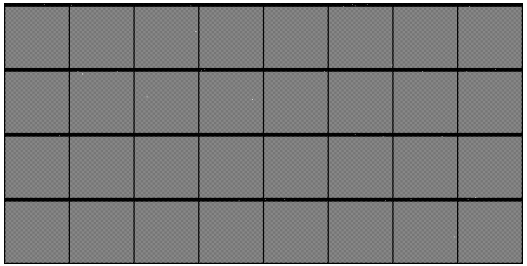


REINFORCE Results

Epoch 1



Epoch 5



Epoch 10

Advantage Actor Critic (A2C)

Introduction to A2C [8]

- Maintain an estimate of the Value function and the policy:

$$V(s_t;\theta') \qquad \pi(a_t|s_t;\theta)$$

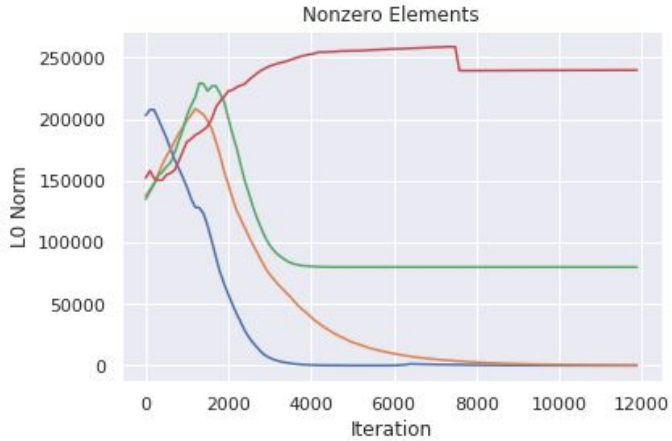
- Sample action with respect to your policy:

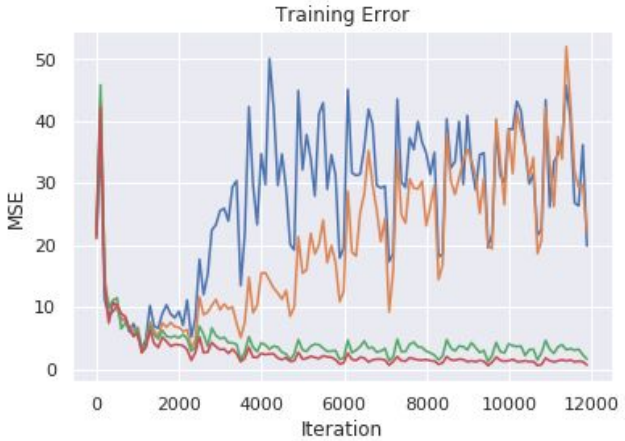
$$a \sim \pi(\cdot|s_t;\theta)$$

- Train the estimators:

$$\mathcal{L}_\theta = \log(\pi(a_t|s_t;\theta))(\mathcal{L} - V(s_t;\theta')) \quad \mathcal{L}_{\theta'} = \frac{1}{2}(\mathcal{L} - V(s_t;\theta'))^2$$

A2C Results

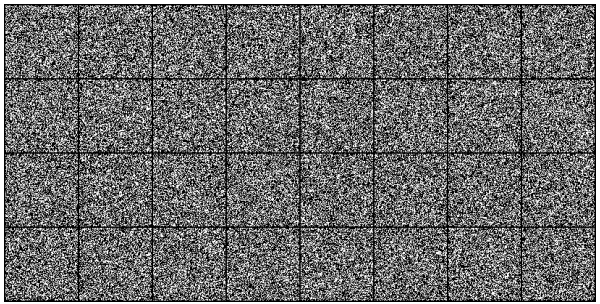

Nonzero Elements
L0 Norm
250000
200000
150000
100000
50000
0
0
2000
4000
6000
8000
10000
12000
Iteration


Training Error
MSE
50
40
30
20
10
0
0
2000
4000
6000
8000
10000
12000
Iteration

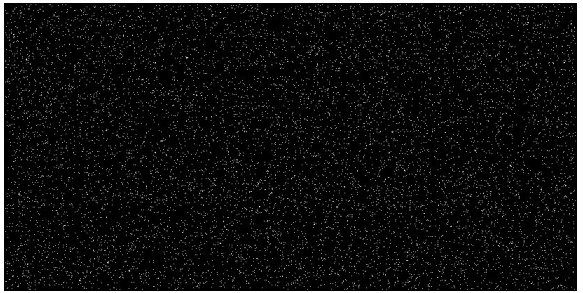

Testing Error
MSE
50
40
30
20
10
0
0
200
400
600
800
1000
Iteration

A2C Results

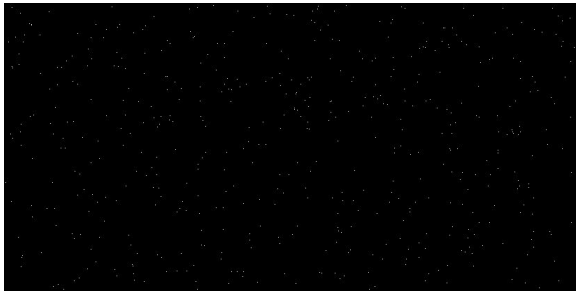
Epoch 1



Epoch 5



Epoch 10



Trading Discrete for Continuous Action Spaces

Sampling in High Dimensional Spaces

- It’s a well-known issue that simple sampling schemes suffer in high dimensional spaces…
- Even with MCMC techniques, it can be difficult to know at what point “mixing” has been achieved…
- This makes PGM-based models like the Ising Model particularly unattractive in a deep learning setting.

RL and High Dimensional Action Spaces

- Our action space is the set of binary masks, this gives us 2^(100x100) actions.
- The RL methods we've used typically are tested on much smaller action spaces, closer to tens of actions.
- Not all hope is lost though, as similar techniques have worked for even continuous spaces!

Deep RL in Large Discrete Action Spaces [10]

Following Dulac-Arnold et al, we will *embed* our discrete action space into a continuous action space to use algorithms built for large action space exploration, such as Deep Deterministic Policy Gradient [9].

Algorithm 1 Wolpertinger Policy

State $\mathbf{s}$ previously received from environment.
$\hat{\mathbf{a}} = f_{\theta^\pi}(\mathbf{s})$ {Receive proto-action from actor.}
$\mathcal{A}_k = g_k(\hat{\mathbf{a}})$ {Retrieve k approximately closest actions.}
$\mathbf{a} = \arg\max_{\mathbf{a}_j \in \mathcal{A}_k} Q_{\theta^Q}(\mathbf{s}, \mathbf{a}_j)$
Apply $\mathbf{a}$ to environment; receive $r, \mathbf{s}'$.

Deep RL in Large Discrete Action Spaces

We must answer the follow questions for this technique:

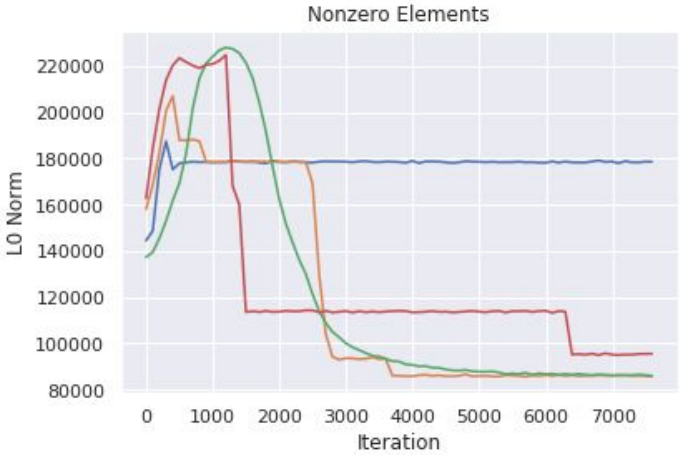
1. **How do we embed our action space?**

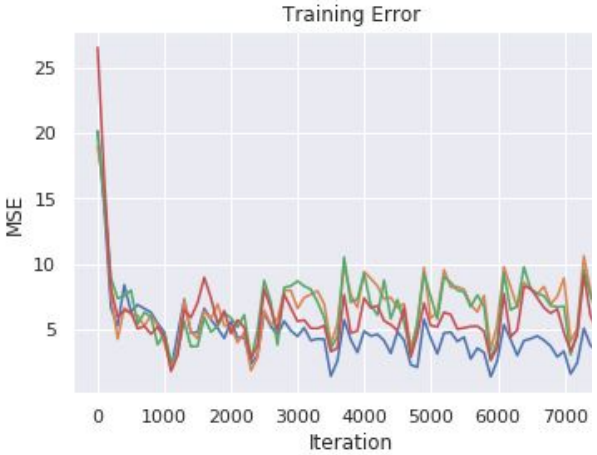
   Our probability matrix is a continuous version of our action space.

2. **How do we find “nearby” actions for our selected continuous action?**

   One possibility is to simply sample from the mask distribution or threshold probability values.

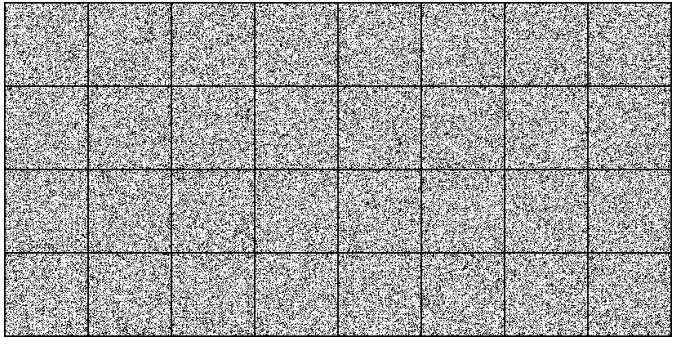
DDPG Results



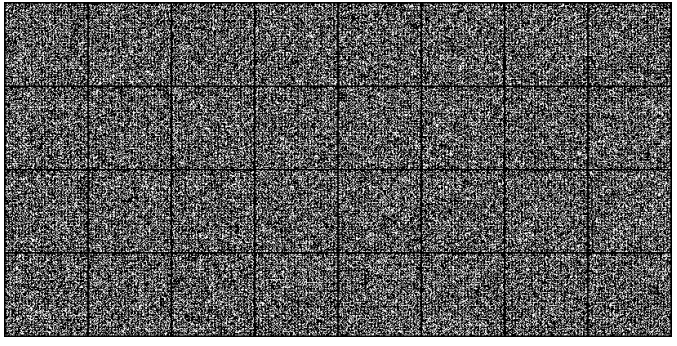


DDPG Results

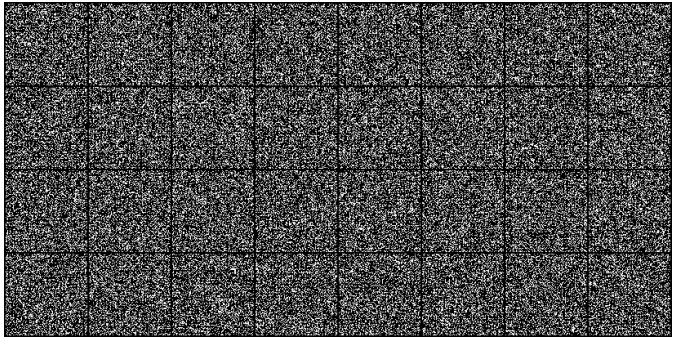
Epoch 1



Epoch 5



Epoch 10



Gradually Eliminating Masking Network

Gradually Eliminating Masking Network

Two of the major goals of this research project are:

1. Can active learning help address the ambiguity of monocular depth estimation?
2. Can active learning be used to provide a curriculum for learning so that at the end of training no sparse data has been used?

Gradually Eliminating Masking Network

Two of the major goals of this research project are:

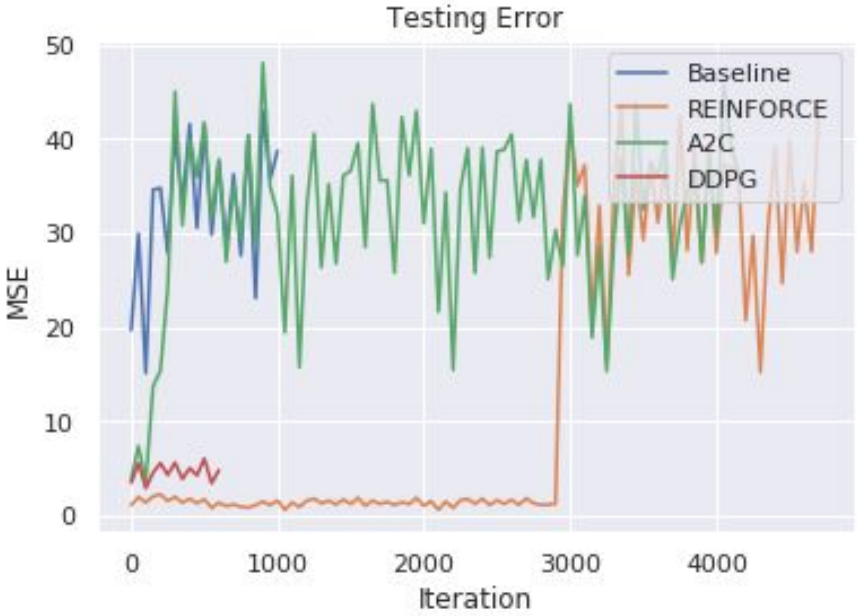
1. **Can active learning help address the ambiguity of monocular depth estimation?**
2. Can active learning be used to provide a curriculum for learning so that at the end of training no sparse data has been used?

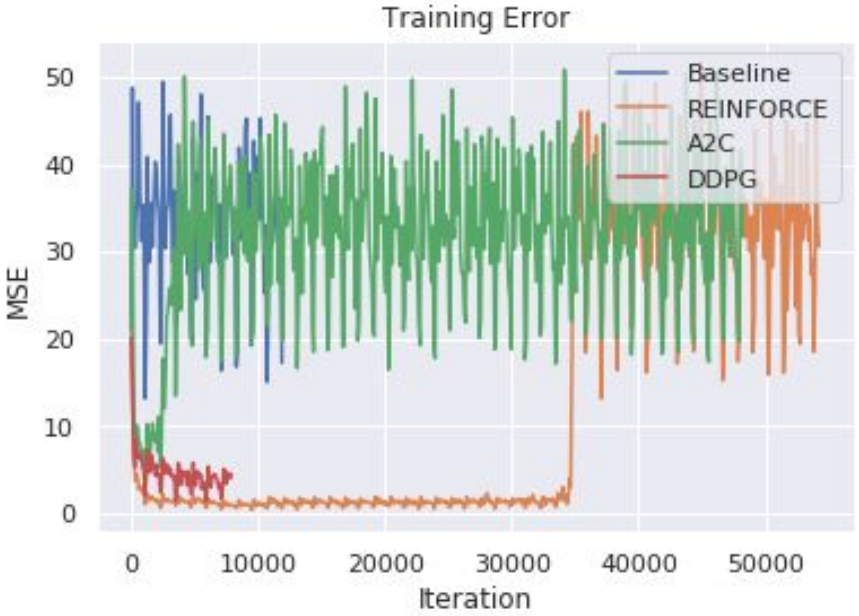
Gradually Eliminating Masking Network

Two of the major goals of this research project are:

1. Can active learning help address the ambiguity of monocular depth estimation?
2. **Can active learning be used to provide a curriculum for learning so that at the end of training no sparse data has been used?**

Best Runs vs. Baseline





Conclusion

Takeaways

- RL techniques allow for greater reduction in sparsity than Gumbel Softmax or similar techniques.
- In terms of variance: REINFORCE < A2C < DDPG.
- It appears that active learning techniques can improve the final estimator learned for monocular depth estimation, at the cost of higher variance.

Plans Moving Forward

1. Continue to experiment with continuous RL methods and recent additions to DDPG to stabilize performance.
2. Experiment on larger image patches to test how that degrades performance / training time.
3. Conduct more in depth testing to see how “plausible” the masks generated are at each training step.

Github Repositories

Public Github Links

- Sparse Aerial Depth Completion:

  https://github.com/austinbdill/sadc

- Active Learning for Pixelwise Prediction:

  https://github.com/austinbdill/pixelwise_al

References

References

1. KITTI Depth Completion Dataset

   http://www.cvlibs.net/datasets/kitti/index.php

2. Playing Atari with Deep Reinforcement Learning

   https://arxiv.org/abs/1312.5602

References

3. Categorical Reparameterization with Gumbel Softmax

   https://arxiv.org/pdf/1611.01144.pdf

4. Learning Selection Masks for Deep Neural Networks

   https://arxiv.org/pdf/1906.04673.pdf

References

5. Learning Associative Markov Networks

   http://robotics.stanford.edu/~koller/Papers/Taskar+al:ICML04.pdf

6. Undirected Graphical Models - Probabilistic Machine Learning

   https://www.cs.ubc.ca/~murphyk/MLbook/pml-print3-ch19.pdf

References

7. Policy Gradient Methods for Reinforcement Learning with Function Approximation

   https://papers.nips.cc/paper/1713-policy-gradient-methods-for-reinforcement-learning-with-function-approximation.pdf

8. Asynchronous Methods for Deep Reinforcement Learning

   https://arxiv.org/abs/1602.01783

References

9. Continuous Control with Deep Reinforcement Learning

   https://arxiv.org/abs/1509.02971

10. Deep Reinforcement Learning in Large Discrete Action Spaces

    https://arxiv.org/abs/1512.07679

Github References

11. PyTorch Straight Through Gumbel Softmax Implementation:

https://gist.github.com/yzh119/fd2146d2aeb329d067568a493b20172f