




Image Super Resolution



ADS-project3-group3
Yuting He, Seungwook Han, Shengwei
Huang, Mengran Xia, Hongye Jiang



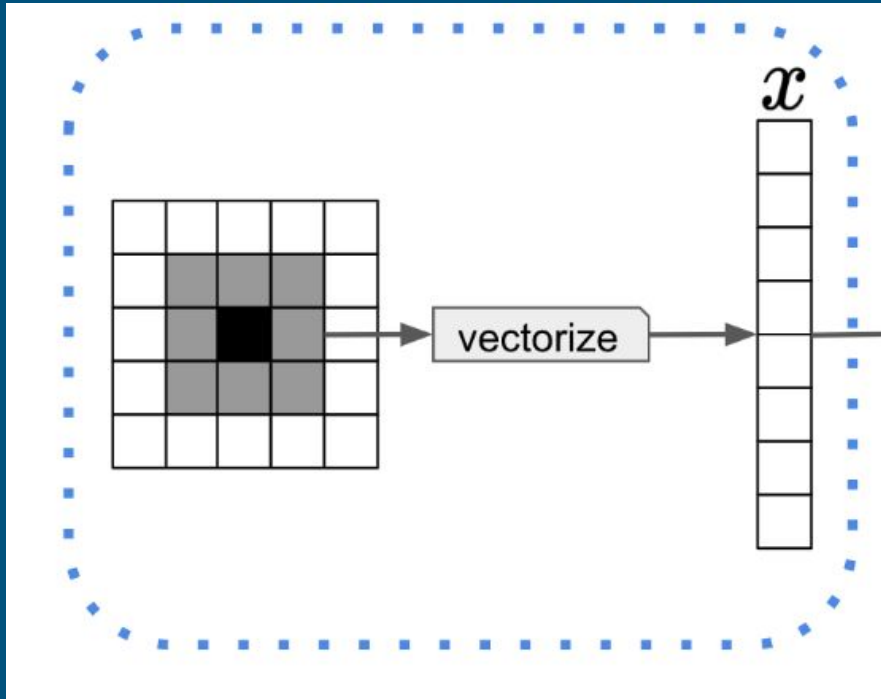
Goal

Our goal is to use machine learning method to create predictive models to enhance the blurry and low-resolution images so that we can get high resolution images.

Outline

- I. Feature Extraction
- II. Performance Measure
- III. Baseline Model
- IV. Advanced Model
- V. Outcome Comparison

I. Feature Extraction



Extract eight neighbors of every pixel as its features.

```

pixeldim <- function(imgLR,d){
  rr = nrow(imgLR)
  cc = ncol(imgLR)
  #add two row
  imgLR1 <- abind(array(0,c(1,ncol(imgLR),3)), imgLR, array(0,c(1,ncol(imgLR),3)), along = 1)
  #add two col
  imgLR2 <- abind(array(0,c(nrow(imgLR1),1,3)), imgLR1, array(0,c(nrow(imgLR1),1,3)), along = 2)

  central <- as.numeric(imgLR2[2:(rr+1), 2:(cc+1),d])

  p11 <- as.numeric(imgLR2[1:rr, 1:cc,d]) - central
  p21 <- as.numeric(imgLR2[2:(rr+1), 1:cc,d]) - central
  p31 <- as.numeric(imgLR2[3:(rr+2), 1:cc,d]) - central
  p12 <- as.numeric(imgLR2[1:rr, 2:(cc+1),d]) - central
  p32 <- as.numeric(imgLR2[3:(rr+2), 2:(cc+1),d]) - central
  p13 <- as.numeric(imgLR2[1:rr, 3:(cc+2),d]) - central
  p23 <- as.numeric(imgLR2[2:(rr+1), 3:(cc+2),d]) - central
  p33 <- as.numeric(imgLR2[3:(rr+2), 3:(cc+2),d]) - central
  pexelmattemp <- cbind(p11,p21,p31,p12,p32,p13,p23,p33)
  return(pexelmattemp)
}

featMat[,1] <- pixeldim(imgLR,1)
featMat[,2] <- pixeldim(imgLR,2)
featMat[,3] <- pixeldim(imgLR,3)

```

1. PADDING ZEROS
2. Extract neighbors
3. Centralization

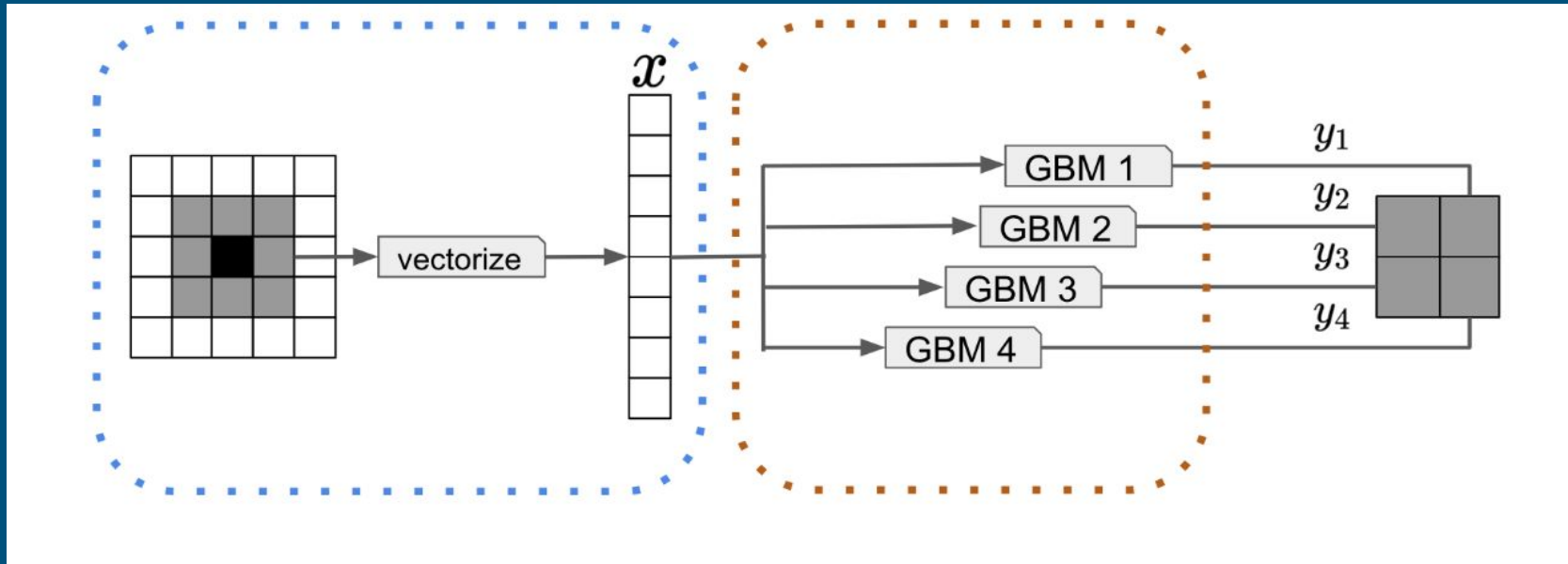
$$x = \text{vectorize}(I_{LR}) - \text{center pixel}(I_{LR})$$

II. Performance Measure

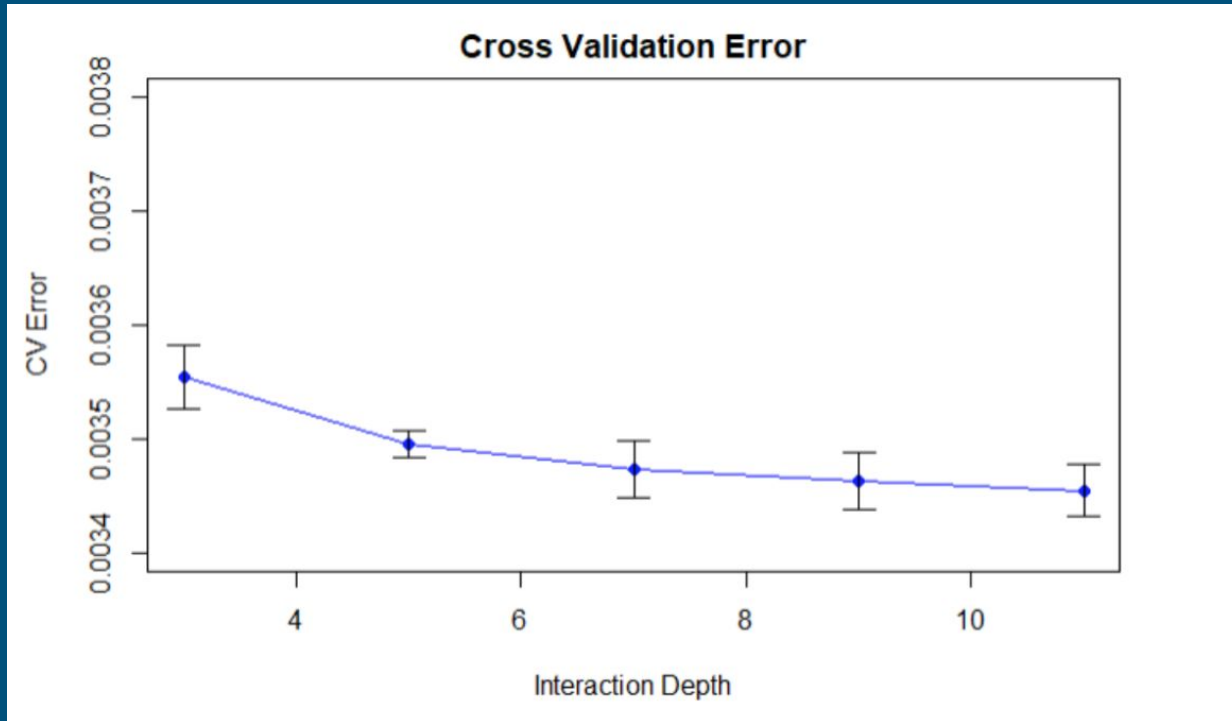
$$MSE = \frac{1}{3mn} \sum_{c=1}^3 \sum_{i=1}^m \sum_{j=1}^n [I(i, j, c) - K(i, j, c)]^2$$

$$PSNR = 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE)$$

III. Baseline Model



Cross validation



Baseline Performance

Depth = 11

Mean MSE	Mean PSNR
0.002511598	27.05948

IV. Advanced Model——XGBoost

Different booster comparison

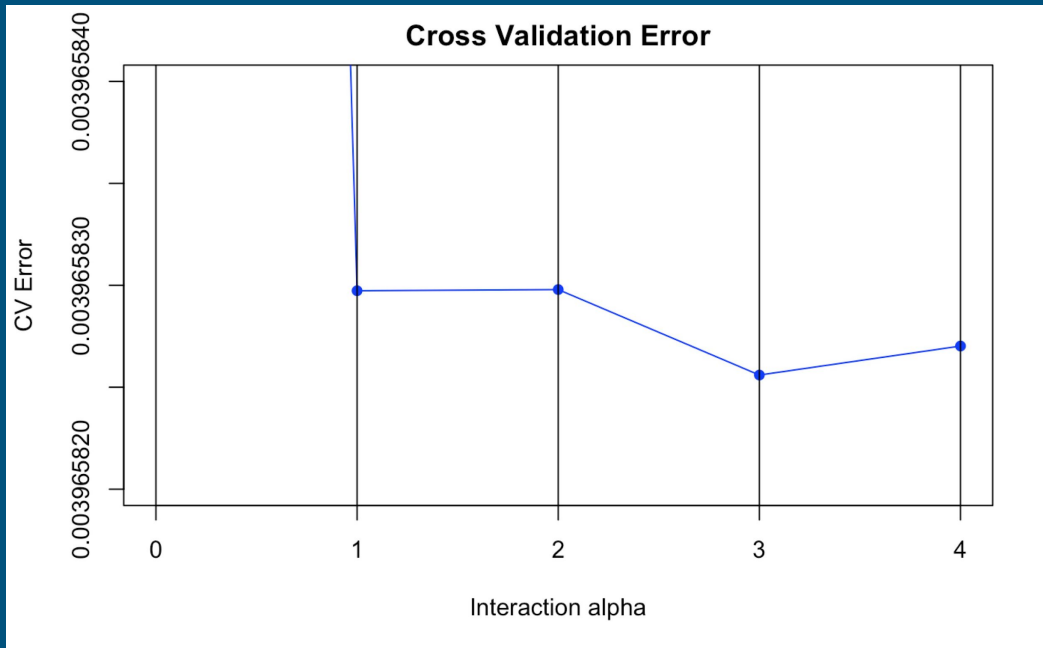
	Mean MSE	Mean PSNR
GBTREE	0.002593127	27.03771
GBLINEAR	0.002511576	27.05951
DART	0.011715	19.42583

IV. Advanced Model—XGBoost

Cross validation for alpha

Alpha = 3

L1 regularization term on weights. Increasing this value will make model more conservative. Normalised to number of training examples.

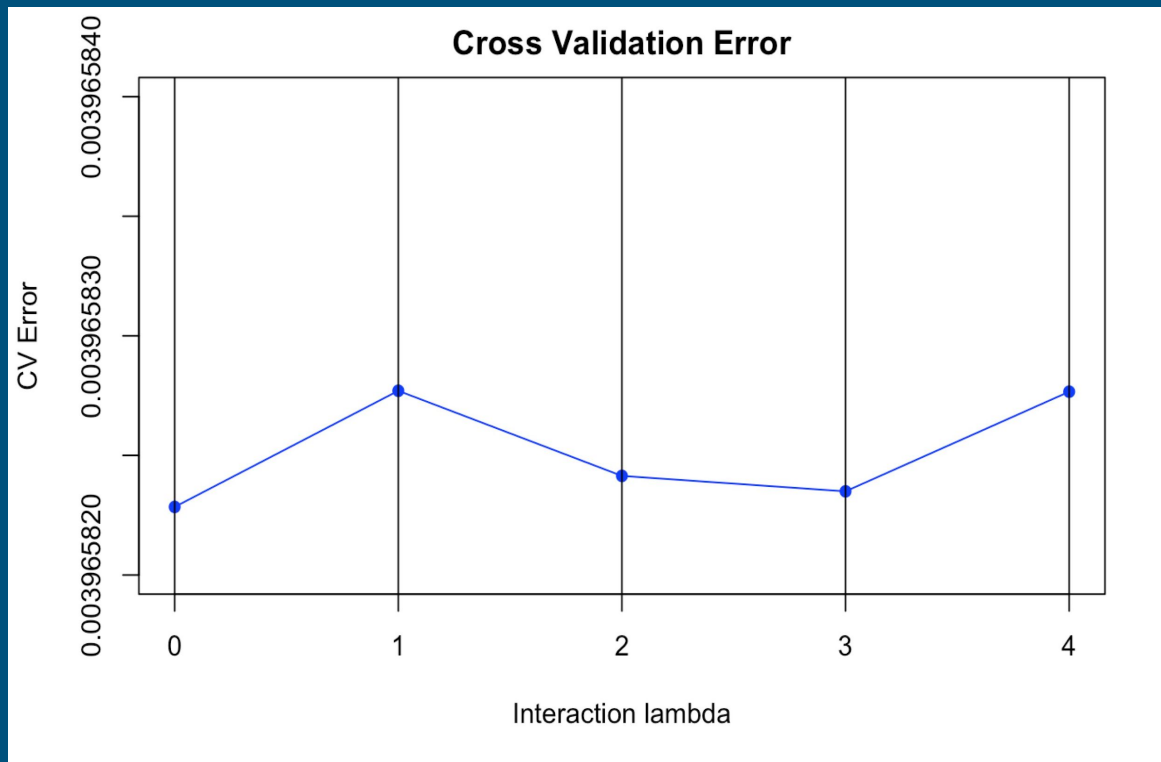


IV. Advanced Model—XGBoost

Cross validation for lambda

Lambda = 0

L2 regularization term on weights. Increasing this value will make model more conservative.
Normalised to number of training examples.



V. Outcome Comparison

	Baseline Model	XGBoost		Baseline Model	XGBoost
Mean MSE	0.002511598	0.002511576	Training time	> 5 h	47.453 s
Mean PSNR	27.05948	27.05951	Testing time	102.248 s	11.602 s

V. Outcome Comparison



Low resolution



Baseline model



XGBoost model



High resolution

Why XGBoost fast and accurate?

Speed:

- XGBoost utilizes OpenMP which can parallel the code on a multithreaded CPU automatically.
- XGBoost has defined a data structure DMatrix to store the data matrix. This data structure will perform some preprocessing work on the data so that the latter iteration is faster.

Accuracy:

- The main reason for the improvement of the accuracy is because the newly-defined regularization term and the pruning approach which makes the learned model more stable.

Thanks!

