CS376: Computer Vision: Assignment 5 Due: Dec 2th, 11:59 PM

**Instruction:** You must answer the Short Answer Problems. You may choose either Programming Track(Image Classification) or Theory Track. 100 points in total.

# Short answer problems [30 pts]

1. What is the relation between image classification and object detection? Please give two concrete examples on how they are related (e.g., one is applied to solve another)?

Object detection is the process of creating bounding boxes around the objects in an image while image classification is the process of categorizing an image based on its content, i.e. the objects within it. Object detection can be used for robust image classification by isolating the objects from both noise and other objects and then allowing the image classification model to only run on the pixels within each bounding box, essentially classifying the objects rather than the images. Even more robust image classification can use object detection to identify features of an image to inform the classification of the entire image, i.e. using positions and classifications of multiple objects for image classifications.

1. Please compare semantic segmentation, instance segmentation and object detection. Describe the simi- larities and differences.

Semantic segmentation is the process of grouping pixels in an image into classes based on the object they seem to represent. An example of this is k-means clustering, in which we group together pixels of similar value. Instance segmentation is an extension of semantic segmentation in which each object in the same class, i.e. each “instance”, is recognized separately. For example, consider an image containing several apples and bananas. Semantic segmentation would yield two classes: apples and bananas. Instance segmentation would yield several different apples and several different bananas.

Object detection involves placing bounding boxes around the instances found in instance segmentation. Finding the locations of the extremities of the instances informs the placement of the bounds.

1. Describe at least two plausible neural network designs for the task of semantic segmentations.

One neural network design could use a series of 2d convolutions in which the result of each convolution is then padded to match the original data’s dimensions. This means that each layer in the neural network will have one neuron per pixel. Another design could be a series of convolutions followed by a series of deconvolutions. Each layer of neurons represents one convolution/deconvolution and each neuron within the layer represents an output value from the convolution/deconvolution. The final output layer is a 2d map matching the size of the image with a class assigned to each pixel coordinate. For either design, the training data set will be images with a corresponding 2d map of classes assigned to each pixel.

1. What are the applications of generative models in solving core computer vision tasks.

Generative models—models trained to create new data based on given data-- are used in several areas. One important use for them is to help train discriminator models when used in Generative Adversarial Models. The discriminator—which is trained to distinguish between types of data, is trained further using data created by a generative model. The generated data is placed in the training set as negative data—i.e. declared to not belong to the class the generator was trying to create data for. The discriminator then classifies this data and the results inform the loss functions of the discriminator and generator. As an example, suppose we wanted to create a GAN that can distinguish between real humans and fake human-like figures. The generator creates human-like figures which are classified as fake when added in the data set. The discriminator then classifies the data, and failures to distinguish the generated figures as fake force re-weightings in the discriminator model that improves it. Successes when distinguishing the generated figures as fake force re-weightings in the generator model.

Outside of helping train discriminator models, generative models can be used for image projection, completion of incomplete images, etc.

1. Discuss the differences between convolution neural networks and deconvolution neural networks.

As the names imply, convolution neural networks perform a series of convolutions while deconvolution neural networks perform a series of deconvolutions. Since convolutions result in fewer output values than input values, each layer in a convolution neural network has fewer neurons than the previous layer, yielding a final result that is less precise but has less noise. Deconvolution has the opposite effect—each layer has more neurons than the previous one, yielding a final result that has more precision but also more noise. An optimal approach would be to combine both types of networks into a single model to get a result of the same size as the input.

# Image Classification (70 points)

Our task is to perform image classification using the CIFAR-10 dataset, which consists of 50K training images and 10K testing images. We will experiment with two categories of methods: non-parametric methods and parametric methods. Our goal is to understand the performance of each method on this dataset. Our specific aims are:

* + Compare the performance between K-Nearest Neighbor classifier and Adaptive Boosting classifier;

Analyze the performance of these two methods when varying the size of the training set and the size of the testing set;

•

* + Perform cross-validation to study the hyper-parameters of each method;
  + Try different feature representations.

**Step 0: Download the CIFAR-10 data set.** The dataset can be downloaded from [https://www.cs.](https://www.cs.toronto.edu/~kriz/cifar.html) [toronto.edu/~kriz/cifar.html](https://www.cs.toronto.edu/~kriz/cifar.html).

**Step 1 (15 points): K-nearest neighbor classifier.** Implement the k-nearest neighbor classifier with *k* = 10. Each image of size 32 32 3 is represented as a vector with dimension 3072. Please report an overall accuracy as well as a confusion matrix. **Hint:** For saving computation cost, it is recommended that you can use a low-dimensional projection matrix *P* R*m×*3072 to reduce the feature dimension of each image. This projection matrix *P* may be computed using principal component analysis (via SVD) among the training images.

∈

× ×

Overall accuracy:



The following matrix is the confusion matrix with the row representing the test image class and the column representing the model’s classification. Classes are numbered as follows:

1. Airplane, 2. Automobile, 3. Bird, 4. Cat, 5. Deer, 6. Dog, 7. Frog, 8. Horse, 9. Ship, 10. Truck.

Table

Description automatically generated

**Step 2 (30 points): Adaptive Boosting classifier.** For this part, you are asked to implement an AdaBoosting method to classify the images described above. We show follow the following two major steps:

* + Weak feature computation (10 points). We will implement the features used in Viola-Jone’s face detec-

tor, i.e., the Haar Features. [https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-IJCV-](https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-IJCV-01.pdf)01. [pdf](https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-IJCV-01.pdf). See Section 2 of the paper for the detailed description. Basically, you first pre-compute an integral image from each image, you then use this integral image to generate Haar features. For efficiency, it is

suggested that you use 1K-2K Haar features rather than all the possibilities (hundreds of thousands). This can be down by choosing a random subset. Note that this random subset has to be consistent across all the training images. Moreover, the image dimension in CIFAR-10 is slightly bigger, i.e., 32 × 32 versus 24 × 24.

(10 points) Train an Adaboost classifier for each class, i.e., using the one-versus-all mode. Since for each classifier, the number of negative instances is nine times more than the number of positive instances, you have to balance the initial weight. Please refer to slide 10 of Lecture 19 regarding the Adaboosting algorithm.

•

(10 points) Combine these one-versus-all classifiers into a multi-class image classifier. Please report the overall accuracy.

•

Initializing, adaboosting for 5 weak classifiers, and testing all on the same set of 1000 images yields:

Overall accuracy



Class Accuracies



**Step 3 (5 points): Comparison.** Please compare the performance of the K-nearest neighbor classifier and the Adaboosting classifier. Which classes does K-nearest neighbor do better and which classes does Adaboosting do better, and why? How about running time?

The adaboosting classifier has an overall accuracy that is greater than the k-nearest neighbor classifier, but a much more extreme distribution in individual class accuracies. Half of the adaboosting classifiers have an accuracy between 85% and 90% while the other half have an accuracy between 10% and 15%. The high accuracy classes were airplanes, birds, dogs, ships, and trucks while the low accuracy classes were automobiles, cats, deer, frogs, and horses. I am not sure what the reason for this is. My guess would be that the lower accuracy classes have fewer easily identifiable distinguishing features, but this can be remedied by more training iterations and more data. Perhaps I simply haven’t trained the model enough.

K-nearest had a more even distribution of accuracies all ranging from 15% to 52%, with planes and ships being significantly more accurate than average and cats and dogs being significantly less accurate. My best guess as to why k-nearest seems to do better with commercial-grade machines rather than consumer-grade machines and domestic animals is because of variations within the classes. Since planes and ships aren’t consumer-level machines, they don’t have as much variation amongst them as machines like automobiles do (e.g. a jeep and a prius look much more dissimilar to eachother than a boeing 737 and an Airbus A350 do to eachother). Domestic animals also have much more variation because selective breeding has created that variation (e.g. a Maltese vs a German Shepherd).

Total runtime for adaboost, both training and testing, is many times longer than that of k-nearest. However, if we exclude training from the adaboost runtime, adaboost is faster than k-nearest. There are very few computations being conducted when testing adaboost—a simple series of classifiers each conducting a few operations. The training, however, takes time. The initialization of all the possible classifiers and tuning their weights is an extremely space and time consuming process. Since k-nearest doesn’t actually undergo machine-learning, it can avoid a training phase but cannot improve.

**Step 4 (10 points): Cross-validation.** Perform 5-fold cross-validation for the K-nearest neighbor classi- fier. Report the optimized hyper-parameter *K* and the corresponding confusion matrix.

Each fold had k set to the column number x5. Improvements completely diminished after k = 10.



Confusion matrix for k = 10:

Table

Description automatically generated

**Step 5 (10 points): Cross-validation II.** Perform 5-fold cross-validation for the AdaBoosting classifier. Report the optimal value for the number of weak classifiers.

Class accuracies at

1 - 8 weak classifiers:



9-10 weak classifiers:



It seems that with my implementation, 1 weak classifier is optimal. No changes in accuracy occurred when adding more classifiers until I reached 9 weak classifiers, at which point accuracy dropped. This is likely due to limitations of my hardware I had to work around. The training sets were relatively small but still took time to sift through, and there were occasions where I had to compress or cut out data to make this feasible. I also used a very limited amount of haar features, 89—much much fewer than the recommendation of this paper (but still 3 times the amount clarified on piazza).

**Tips**: Please check piazza post for tips.

# Theory Track (70 points)

**Problem 2: (20 points)** Define the Harr feature of an image. Describe an efficient algorithm to compute Harr features. Discuss

* + (10 points) Runtime complexity of the algorithm.
  + (10 points) Space complexity of the algorithm.

**Problem 3: (20 points)** Describe an algorithm to perform instance segmentation using the following building blocks:

* + An approach for generating object proposals (bounding boxes).
  + An approach for computing dense pixel-wise descriptors.

**Problem 4: (15 points)** So far we have talked about either the k-nearest neighbor classifier or a parametric approach (e.g. Adaboosting), think about how to combine the strength of both approaches. Hint: use parametric approaches to compute feature descriptors. Please write out the pseudo-code of your algorithm. **Problem 5: (15 points)** If you have a large collection of unlabeled images and a small collection of labeled images. Describe an algorithm to train a neural network that can leverage the unlabeled images for training. For simplicity, you may assume that the classification task is binary. Again, please write out the pseudo-code of your algorithm.

# Submission instructions:

Create a single **zip** file so submit on Canvas that includes

* + Your well-commented code, including the files and functions named as specified above.
  + A **PDF** writeup of your results with embedded figures where relevant. Please do not include any saved matrices or images etc. within your zip file.