CS 189 Introduction to Machine Learning
Fall 2017 gs3

1 Kernel PCA

Let $X \in \mathbb{R}^{n \times d}$ be a matrix with rows $x_1^T \ldots x_n^T$, and $\phi : X \longrightarrow X'$ be a feature map with associated kernel $k(x,y) = \langle \phi(x_1), \phi(x_2) \rangle$. We will attempt to perform kernel PCA on X using the feature mapping ϕ. For this problem, assume that the data is already centered.

(a) Let $\Sigma = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T$ be the sample covariance matrix. Recall that performing PCA involves solving for the eigenvectors and eigenvalues of Σ.

Show if v is a solution (an eigenvector of Σ, e.g. $\Sigma v = \lambda v$), then v is in the range of X^T, or $v = X^T \alpha$ for some α.

(b) Show if (α, λ) is a solution to $XX^T \alpha = \lambda \alpha$, then $(v = X^T \alpha, \lambda)$ solves $X^T X v = \lambda v$.

(c) Kernel PCA solves $\phi(\Sigma) v = \lambda v$, where $\phi(\Sigma) = \frac{1}{n} \sum_{i=1}^{n} \phi(x_i) \phi(x_i)^T$. Explain how to find all λ satisfying this equation without explicitly computing $\phi(x)$.

(d) In regular PCA, the projection coefficient of x onto a principal component v is $\langle x, v \rangle$. In feature space, the coefficient is $\langle \phi(x), v \rangle$. How do we compute this without explicitly computing $\phi(x)$?

(e) To get the correct projection coefficient in regular PCA, we always normalize eigenvectors v so that $\langle x, v \rangle$ gives the correct coefficient. How can we equivalently ensure proper normalization in our kernel PCA?

2 Convnet

(a) Do it! (Quick Check)

(a) Suppose the input to our CONV layer is the following 5×5 binary image:

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

which uses the following 3×3 filter with stride 1:

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix}$$

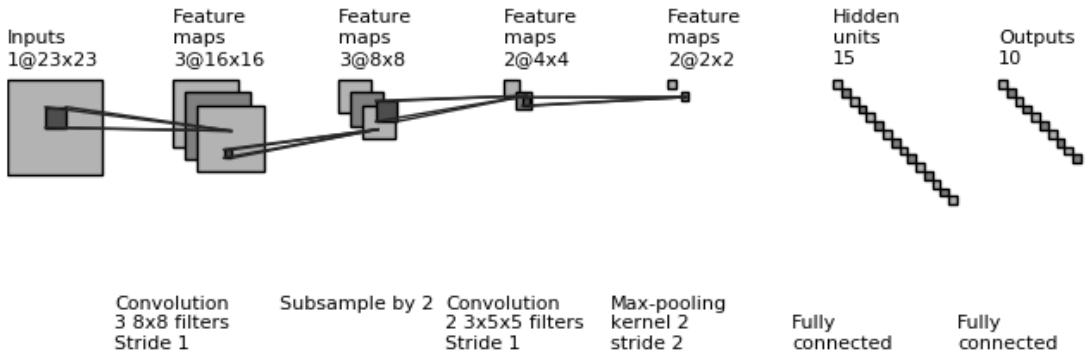
find the output of this layer for the input. How many parameters does this layer have?

(b) Now suppose we have a max pooling layer with kernel size 2 and stride 2 with the following as its input:

$$\begin{bmatrix} 1 & 5 & 8 & -1 \\ 0 & 4 & 1 & 3 \\ 9 & 8 & 1 & 2 \\ 1 & 5 & 3 & 3 \end{bmatrix}$$

find the output to this layer. How many parameters does this layer have?

(b) We have a convolutional network that takes a 23x23 image as its input and returns a 10 outputs. Assume we do not pad the image with zeros and there are no bias terms.



How many parameters are there in each layer?

(c) Suppose we have a deep convolutional neural network (a conv-net with many layers) that, given a portrait of a person, predicts whether the person in an image is Professor Sahai, or not Professor Sahai. Assume we train this classifier only with full body images of both Professor Sahai and other people. Classify the following features as "early" if they are features that are more likely to be learned in the early layers of the network, "late" if they are features that are more likely to be learned in the late layers of the network or "none" if they aren't likely to be learned in the network.

(a) Edges

(b) This image contains glasses

(c) This image contains long hair

(d) The image contains stripes

(e) The person in the image is teaching 189

(f) The person in the image is Stella Yu

(d) Convolutional filters: For the following problems, assume that there is zero padding.

a) Suppose you had a n by n filter

$$\frac{1}{n^2}\begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \dots & \dots & \dots & \dots \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

What is the effect of increasing n on the result of the convolution of the filter with the input image?

b) Suppose you wanted your convolutional layer to learn vertical edge features. Using a 3 by 3 matrix as your filter, what would the parameters inside that filter be? *Hint: we want the result of the filter to be large if there is a vertical line, and small otherwise. We also want to be able to mitigate horizontal lines. Try drawing out small examples of vertical lines/horizontal lines and think about what the results should be.*

c) Suppose you had a 3 by 3 filter

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

What feature(s) does this filter learn?

d) Using the filter in part (c), give an approach for sharpening an image.

3 Decision Tree

(a) Entropy Justification

Entropy, in physical sciences, is defined as $k_B \ln W$, where k_B is Boltzmann's constant and W is the multiplicity of the coarse-grained state, or the number of ways the coarse-grained state can be arranged in. This is true under the assumption that each microstate that coarse state can be in has equal probability.

In this scenario, we have sequences of coin tosses with bias p.

(a) Boltzmann's definition of entropy is $-k_B \sum_v p(v) \ln(p(v))$. Show that in the case that each microstate v has the same probability, that this is reduced to the form given above.

(b) What is the probability of a sequence HHHHHHHH? How many ways can we have a sequence with 8 heads?

(c) What is the probability of a sequence HTHTHTHT? How many ways can we have a sequence with 4 heads?

(d) If we characterize a sequence by how many heads appeared, which case has a larger multiplicty: 4 heads, or 0 heads?

(e) We can see that a coarse-grained state with a large multiplicity has more inherent uncertainty, meaning that we are less certain about its exact specifications than we are about a coarse-grained state with a small multiplicity. In the context of decision trees, why is this useful for splitting?

(b) Decision Trees

We have a dataset with two features, labeled with classes $+$ and $-$, represented as tuples $(f_1, f_2, +/-)$:

$$\{(0,0,-),(1,0,-),(2,0,-),(0,1,+),(1,1,-),(2,1,+),(0,2,+),(1,2,+),(2,2,-)\}$$

Let L_1 be the number of class $+$ points in the left node, L_2 be the number of class $-$ points in the left node, etc. Here, we will use the splitting rule that splits on the feature that maximizes the value of $|L_1 - R_1| \cdot |L_2 - R_2|$. What is the decision tree that is generated using this splitting rule that correctly classifies the data given?

(c) More decision trees

We will now use a different cost function J(S). We will allow J(S) to be defined as the number of points not in a certain class C. Let's assume that we have two different distribution of points. Set 1 is {AX: 10, AY: 9, BX: 10, BY: 1}. Set 2 is {AX: 10, AY: 5, BX: 10, BY: 5}. Assume that the first split is on the first letter and the second split is on the second letter in the pair.

(a) According to J(S), what is the cost for both of the splits? Is this a good cost function?

(b) Compare the result of computing the split based on the cost function J(s) given above vs a split determined on entropy and surprise.

(d) Random Forests

Now we will consider a situation where we are presented with n training points in a feature space of d dimensions. Assume that we are trying to create a random forest with t trees, where each tree contains exactly k internal nodes, and we only select f out of the d features at each node. Answer the following questions

(a) Suppose that there is a particularly strong predictor-feature for the training points we are currently working with. What is the probability that this feature will not appear in any of the splits of the internal nodes?

(b) What happens to this probability as f approaches d?

(c) What happens to this probability as f approaches 1?

(d) What is the tradeoff between picking more or less features at each node in the random forest?