Review for Exam 1

Erik G. Learned-Miller
Department of Computer Science
University of Massachusetts, Amherst
Amherst, MA 01003

November 1, 2012

Abstract

This document reviews material you will need for Exam 1. I recommend you do each problem here. If you can do them all without referring to other material, you should be in good shape for the exam. I recommend

that you review your notes and the class documents before trying to do the material so that you know where your deficits are. .

1 Basic notation

You should understand the following types of notation.

- Summation ($\sum$). Example:

$$\sum_{i=1}^{N} X_i = X_1 + X_2 + ... + X_N. \tag{1}$$

- Product ($\prod$). Example:

$$\prod_{i=1}^{N} X_i = X_1 \times X_2 \times ... \times X_N. \tag{2}$$

- When we perform optimization of a function with respect to some changeable parameter, we are usually interested in either the minimum or maximium value of the function itself, or, alternatively, we are interested in the values of the parameter that achieved the maximum or minimum.

  Suppose we are trying to find the maximum value of a function $f(x, y)$ when y is set to the value 3 and x can be any value from some set S. To specify the maximum value of the function we write:

$$\max_{x \in S} \; f(x, y = 3). \tag{3}$$

  We frequently assign this maximum to some variable, as in:

$$F^* = \max_{x \in S} \; f(x, y = 3). \tag{4}$$

- If we are interested in the value of x which gave us this optimum, we use the arg max notation:

$$\arg\max_{x \in S} \; f(x, y = 3). \tag{5}$$

  In this case, of course, the maximizing x might be called x^*:

$$x^* = \arg\max_{x \in S} \; f(x, y = 3). \tag{6}$$

2 Basic probability

- Know all the material from the *Review of Basic Probability* handout.
- Particularly focus on marginalization and Bayes' rule.
- Suppose we are trying to classify images into one of several classes c_1, c_2, and c_3, and to do it we are using feature values f_1 and f_2. Know what is meant by the *likelihoods of the classes*, the *priors of the classes*, and the *posterior probabilty* for each class.

- Marginalization: If you are given a probability distribution for each pair of possible values of $Prob(X, Y)$, then you should be able to compute $Prob(X)$ and $Prob(Y)$ for each possible value of X and Y.
- Know the definition of statistical independence: For two random variables X and Y, X and Y are statistically independent if and only if $P(X|Y) = P(X)$ for all X and Y. Or alternatively, $P(X, Y) = P(X)P(Y)$. Be able to derive one of these formulas from the other (multiply both sides by $P(Y)$.
- Conditional probability. Be able to compute $P(X|Y)$ from $P(X, Y)$ and $P(Y)$.
- Be able to given reasonable estimates of $P(X)$, $P(X|Y)$, and $P(X, Y)$ from raw data.

3 Classification

- Understand Bayes' rule. Practice it many times so you are fluid with it. Try making up some tables of $p(x|c)$ where x is a feature value and c is a class, and then try to compute $p(c|x)$. You will also need, of course, $p(c)$ the prior probabilty of a class. *If you don't understand this, you better get help before the test.*
- If you *estimate* the posteriors for each class ($p(c|x)$) and you pick the class with the highest posterior, this is known as *maximum a posteriori* classification, or MAP classification. MAP classification is not ALWAYS the best strategy when you have *estimated* the posteriors.
- However, if you have the *exact*, true posteriors for the classes, then choosing the class with the highest posterior yields a *Bayes Optimal Classifier*, which is a classifier with the minimum probability of error. This smallest possible probability of error is also known as the Bayes error rate, or just the Bayes error.
- Given the true likelihoods (not the ones estimated from data) for each class, and the true priors (not the ones estimated from data), one can calculate the exact posteriors ($p(c|x)$), and hence, build a Bayes optimal classifier, using Bayes' rule and MAP classification. (MAKE SURE YOU UNDERSTAND THIS!)

4 Supervised learning

- Know the definition of supervised learning.
- Review the problem set on digit recognition from a single pixel. What kind of features give the best classifiers?

- Imagine you have a joint distribution $p(x, y)$ over some symptoms x of a rare disease and y, which indicates whether you have the disease or not. What is the problem with building a training set by sampling from $p(x, y)$? (Answer: you don't get enough samples of patients with the disease, since it is rare.)

- What is the solution to the above problem? Answer: Sample a data set that consists of $p(x|y = \texttt{hasdisease})$ and $p(x|y = \texttt{ishealthy})$ separately, estimate the priors $p(y)$ separately, and then use Bayes' rule to compute the posteriors.

- Know how nearest neighbor classification works (check out the wikipedia page if you need a review of this.)

- Know how k-nearest neighbor classification works. Answer: Let $k = 7$. When given a test point, find the 7 training points nearest to that test point. Among the nearest neighbors, calculate the number of training points from each class. Choose the class with the most training points (from the 7) and guess that that is the identity of the test point.

- What happens if my training data in supervised learning comes from one probability distribution $p(x, y)$ and my test data comes from a very different distribution $q(x, y)$. Answer: The performance of my classifier may be arbitrarily bad, since I trained on the "wrong data". Example: My training data consists of red apples (class A) and red pears (class B). My test data consists of green apples and green pears. I cannot expect to get the test examples correct (although I may get lucky and use shape as a feature instead of color, but that would just be luck.)

- What is the fundamental problem with estimating a distribution over entire images? Describe the trade-offs between using smaller parts of the image to classify images (like a single pixel) and using larger parts, like mutliple pixels or even the entire image. (difficulty of estimation versus amount of information in the features)

5 Basic Matlab familiarity

- Know how to turn the values in a matlab matrix into a vector (use the colon operator).

- Know how to transpose a matrix and what this means (you can use either the quote operator (') or use the transpose command. This can used to turn a row vector into a column vector or vice versa.

- Understand the structure of a "grayscale" or "scalar-valued" image. It is simply a 2-dimensional array of numbers.

- Understand the structure of a "color" image: it is a 3-dimensional array of numbers in which the first "layer" represents red, the second layer represents green, and the third represents blue.
- IMPORTANT: Understand that a *grayscale* or *scalar-valued* image can be rendered in color using a *look-up-table*, which is essentially a scheme for doing color-by-number. That is, for each value in the image, the computer looks up the red-green-blue (RGB) color that should be used for that particular number.
- Know the repmat command in matlab. Know that it can be used to avoid doing for loops, and that for loops are slow in matlab.

6 Image comparison

- Understand the "sum of squared differences" between two scalar-valued images I and J:

$$SSD(I,J) = \sum_{i=1}^{\#ofpixels} (I_i - J_i)^2.$$

This is the same thing as the "Euclidean distance" except that the Euclidean distance has a square root:

$$\begin{aligned} D_{Euclid}(I,J) &= \sqrt{\sum_{i=1}^{\#ofpixels} (I_i - J_i)^2} \\ &= \left(\sum_{i=1}^{\#ofpixels} (I_i - J_i)^2 \right)^{\frac{1}{2}}. \end{aligned}$$

- Understand the "sum of absolute differences":

$$\begin{aligned} D_{abs}(I,J) &= \sum_{i=1}^{\#ofpixels} |I_i - J_i| \\ &= \left(\sum_{i=1}^{\#ofpixels} |I_i - J_i| \right)^{\frac{1}{1}}. \end{aligned}$$

Understand why one might want to use one instead of the other (sum of squared differences weighs larger errors more heavily, on average).

- Know the general formula for an L_p distance:

$$L_p(I,J) = \left(\sum_{i}^{\#ofpixels} (I_i - J_i)^p \right)^{\frac{1}{p}}. \tag{7}$$

Note that the Euclidean distance is the same as the L_2 distance ($p = 2$) and that the sum of absolute differences is the same as the L_1 distance ($p = 1$).

7 Transforms

You should understand the properties of the following sets of transformations.

- **Shifting only.** Also called **Translations**. Preserves orientation and all the properties of rigid transformations.
- **Rigid. (translation + rotation)** Preserves areas (or volumes if in 3-D) and all the properties of similarity transforms.
- **Similarity (rigid + scale)**. Preserves angles, straightness of lines, paralllelness of lines.
- **Linear transformations.** This is the family of transformations that is represented by ALL full rank 2x2 matrices. (Full rank just means the matrix doesn't squash the object to have no width or no height. For example, a matrix filled with zeroes would transform all coordinates to (0,0), so the object would effectively be shrunken to have 0 size.) Linear transformations preserve the parallelness of lines and the straightness of lines. These transformations include, as special cases, rotations, scalings, shearings, and flips. *IMPORTANT:* Note that linear transformations do NOT include translations!
- **Affine transformations (Linear + translations).** These transformations are exactly like the linear transformations, except that they also include the translations. This allows you to, say, both rotate and move an object, instead of just rotating it. This is important since linear transformations always transform with respect to the origin (0,0). So if you want to rotate something about its center, and the center is not at the origin, then you will need affine transformations.

7.1 Transforming images pixel by pixel

You should understand the programs we discussed in class to rotate images using the "forward" transformation technique and the "inverse" transformation technique. The next two pages contain two matlab functions called rotate1 and rotate2. In particular, you should understand why rotate2.m was necessary to "fill the holes" created by rotate1.m.

```
function rim=rotate1(im,angle)
r=(angle/360)*2*pi;
rotmat = [cos(r) sin(r); -sin(r) cos(r)];
[rows,cols]=size(im);
xcoords = 1:cols;
ycoords = 1:rows;
xcarray = repmat(xcoords,[rows 1]);
ycarray = repmat(ycoords,[1 cols]);
xcv = xcarray(:);
ycv = ycarray(:);
pairs = [xcv'; ycv'];
newpairs = rotmat*pairs;
newpairs = round(newpairs);
newpairs(newpairs<1)=1;
newpairs(newpairs>rows)=rows;
rim=zeros(rows,cols);
for i=1:size(newpairs,2)
     rim(newpairs(1,i),newpairs(2,i))=im(pairs(1,i),pairs(2,i));
end
```

```
function rim=rotate2(im,angle)
r=(angle/360)*2*pi;
rotmat = [cos(r) sin(r); -sin(r) cos(r)];
invmat = rotmat ^ (-1);
[rows,cols]=size(im);
xcoords = 1:cols;
ycoords = 1:rows;
xcarray = repmat(xcoords,[rows 1]);
ycarray = repmat(ycoords,[1 cols]);
xcv = xcarray(:);
ycv = ycarray(:);
pairs = [xcv'; ycv'];
oldpairs = invmat*pairs;
oldpairs = round(oldpairs);
oldpairs(oldpairs<1)=1;
oldpairs(oldpairs>rows)=rows;
rim=zeros(rows,cols);
for i=1:size(pairs,2)
     rim(pairs(1,i),pairs(2,i))=im(oldpairs(1,i),oldpairs(2,i));
end
```

8 Understanding the sizes of sets

- How many distinct 10x10 binary images are there? Answer: 2^{100}.
- How many 1000x1000 color images are there if each color channel (red, green, blue) can take on 256 colors? First, calculate the number of colors per pixel:

$$\begin{aligned} c &= 256 * 256 * 256 && (8)\\ &= 2^8 * 2^8 * 2^8 && (9)\\ &= 2^{24} && (10)\\ &= 2^{20} * 2^4 && (11)\\ &\approx 1,000,000 * 16 && (12)\\ &= 16\ million. && (13) \end{aligned}$$

  Given about 16 million colors per pixel, the number of 1000x1000 images is

$$(16\ million^{1000*1000}) = (16\ million)^{million}.$$

  According to some estimates, there are about 2^{80} particles in the known universe.

- Given 100 x translations of an image, 100 y translations, 1000 rotations, and 500 different scales, how many different images tranformations could I produce? I'll let you do this one yourself.

9 Alignment

- The ingredients of image-to-image alignment are:
  1. Images I and J.
  2. A choice of a family of transformations. You should be familiar with all the basic families of transformations discussed in class and in this document.
  3. A criterion of alignment that assesses how well aligned images are. You should be familiar with multiple different criteria of alignment.
  4. A method for finding the best alignment over the family of transformations. These methods include exhaustive search (what do we mean by this?), gradient descent or coordinate descent, and keypoint methods. You are not responsible for keypoint methods.
- Be able to describe how gradient descent alignment works with the family of translations. Also for say, the family of rigid transformations.
- You should understand how to do alignment by maximization of mutual information (assignment 4).

10 Entropy and Mutual Information

- Entropy is a function of a *probability distribution*. Know the definition of entropy for a discrete distribution. Don't forget the minus sign!
- Mutual information is a function of the *joint distribution* of two random variables. You do NOT need to memorize the mathematical definition of mutual information. However, if I give you a joint distribution (as a table) and the definition of mutual information, you should be able to compute it. This will include computing the *marginal distributions* from the joint distribution (as in Homework 5).
- Remember that entropy is a measure of the *negative* of the *average log probability* of points sampled from a distribution. If a probability distribution just has a few possible values, each of which has a relatively large probability, then entropy will be LOW. If a probability distribution has many many possible values, each of which has a low probability, then entropy will be HIGH.
- Mutual information is 0 if and only if the random variables of the joint distribution are statistically independent. I want you to be able to show that *if* two random variables are independent, then their mutual information is 0. (You don't have to be able to show that this is the *only way* that the mutual information can be 0.)
- Mutual information should be high between two images when they are aligned. Why?
- A good feature, whose value in an image is represented by the random variable X should have *high* mutual information with the class label for an image, represented by the random variable C. If it is not, then the feature random variable doesn't give you any information about the class, which means it is a useless feature.
- Mutual information cannot be higher than the amount of information in either random variable. What do I mean by this? Suppose I am doing a classification problem with 4 classes. Then 2 bits of information is enough to tell me which class I'm in. Thus, the mutual information between any feature and the class label cannot be more than 2 bits.

11 Joint alignment

- What is congealing? Answer: a method of simultaneously aligning a bunch of images to each other.
- What is the criterion being optimized in congealing? (Answer: the sum of the estimated entropies of each "pixel stack". A pixel stack is the set of values in a particular location across all of the images.)

- Why is it sometimes easier to align a group of images than a pair of images? What can happen when we try to align a pair of images? Answer: We can stuck in local minima of the alignment function. Congealing "smooths out" the alignment landscape. (Think of the image of the "average 3" during congealing. It is smoother than a single 3.)
- Understand the basic congealing algorithm. Answer: Iterate until convergence: For each image i, for each transform parameter j, try adjusting image i with a small change to transform j and see if it reduced the sum of the pixel stack entropies (or alternatively, if it increased the probability of the pixels in the changed image with respect to the distributions of the pixel stacks). If the small change to transform parameter j for image i improved the criterion of joint alignment, keep that change and continue.
- While you don't need to understand the details of the "brightness correction congealing" that I showed in class to fix the brightness artifacts in magnetic resonance (MR) images, I want you to be aware that congealing can be done not only with spatial transformations, but also with brightness transformations.