IN5400 – Machine Learning for Image Analysis

Alex

Week 04

1 Coding

Learning objectives:

- custom data loaders
- prediction with a pretrained neural network from torchvision. No data augmentation here yet.

Another take away: prediction with a neural net runs fairly fast. It is not a must to use GPU for prediction.
Take the 2500 Imagenet val images, unpack them. The labels are in `ILSVRC2012_bbox_val_v3.tgz`. `getimagenetclasses.py` has example routines to get the label for one image.

Link is in mattermost for security reasons.

- write a dataset class for this dataset. `https://pytorch.org/tutorials/beginner/data_loading_tutorial.html`.
  - You need to implement __len__ and __getitem__
  - Suggestion: do not load all images in the __init__ method of the dataset class. That does not scale well if you have 500000 images :) . Load the filenames and the labels instead into a list or the like. load an image in `getitem` of your Dataset-derived class!
- rescale the images so that the smaller side is $s = 224$ pixels and perform a center crop of 224×224
- use on top a generic Dataloader class from PyTorch.
- neural network: Choose one from the torchvision model zoo. Initialize the weights of the chosen neural net so that you load weights from the so-called torchvision model zoo.

- compare performance to the case when you do not subtract the mean and normalize the subpixels. If too slow, use only the first 500 images, report the performance difference
- visualize an image and its top-5 predicted classes ... matplotlib can help.
- hint: use a net with little parameters, avoid VGG, Alexnet unless you got a GPU and want to use it as a heating (... a rather expensive one).

2 Disastrous Derivatives

- Compute the directional derivative $Df(X)[H]$ in direction H for:

$$\begin{aligned} f(X) &= Xa, & X \in \mathbb{R}^{d\times k}, a \in \mathbb{R}^{k\times 1}, \\ f(X) &= XX^\top, & X \in \mathbb{R}^{d\times n} \end{aligned}$$

- What will be the shape of the direction H in $Df(X)[H]$ for these two? Is it a real number, a vector or a matrix? Express it as $\mathbb{R}^{1\times 1}$ if you think it will be a scalar, as $\mathbb{R}^{d\times 1}$ if you think it is a vector, or as $\mathbb{R}^{d\times e}$ if you think it is a matrix.
- What will be $Df(X)[H]$? Hint: you can write it as product of matrices if you like it (instead of summing in the flavor of $\sum_{ijk} c_{ijk}$).
- Compute the directional derivative $Df(X)[H]$ in direction H for:

$$\begin{aligned} f(X) &= XCX, & X \in \mathbb{R}^{d\times d} \\ f(X) &= CXBX^\top AX, & X \in \mathbb{R}^{d\times d}, \{A, B, C\} \in \mathbb{R}^{d\times d} \end{aligned}$$

Hint: remember in class the product rule.

- Compute the directional derivative $Df(X)[H]$ in direction H for:

$$f(X) = \begin{pmatrix} 1 & x_2 \end{pmatrix} \begin{pmatrix} 1 & x_2^3 \\ \sin x_2 & x_1 \end{pmatrix}$$

3 n-dim Hyperplanes are a piece of bunny!

Here I want you to think a bit geometrically about these hyperplanes without the need to draw them.

3.1 3 dims

Suppose you have in 3 dimensional space the following dataset: $D_n = \{(x_1 = (-5, 1, 2), y_1 = +1), (x_2 = (1, 2, -3), y_2 = -1)\}$.
Find a linear classifier $f(x) = wx + b$ (means its parameters) which predicts all points correctly.

Hint:
It can help to think of a hyperplane defined in terms of an orthogonal vector and a bias term. Get the orthogonal vector to the hyperplane right first, bias you can determine then afterwards by plugging in points, and solving for the bias.
For two points, what would be a good orthogonal vector for a hyperplane if you have two points which need to be separated??

3.2 5 dims I

Suppose you have in 5 dimensional space the following dataset: $D_n = \{(x_1 = (1, 0, 1, -2, 6), y_1 = +1), (x_2 = (3, 1, -3, 5, 1), y_2 = -1)\}$.
Its the same thing as above, now in 5 dims. Find a linear classifier $f(x) = wx+b$ (means its parameters) which predicts all points correctly.

3.3 5 dims II

Suppose you have in 5 dimensional space the following dataset: $D_n = \{(x_1 = (1, 0, 1, 6, 3), y_1 = +1), (x_2 = (3, 1, -3, 5, 1), y_2 = -1), (x_3 = (2, -1, 0, 4, 2), y_2 = -1)\}$.
Good news: you cant draw it but still solve it. An alternative way is to project these three points into a 2-dim subspace, by the way, and solve it in the subspace.

How to solve it:
You can draw a line between the two points x_2, x_3 of the same label. This line can be part of many hyperplanes (its a 4d space of all possible hyperplanes). The vector $x_2 - x_3$ is parallel to this line .
These are the steps:

I **Now choose a vector w:**

- which is **not** parallel to $x_2 - x_3$,
- which is **not** orthogonal to $x_1 - x_3$ (that avoids some degenerate cases)

and which will serve as candidate for the orthogonal of the hyperplane to be found.

II **Answer for yourself:**

- what is a possible mathematical criterion to test that w is not parallel to the line $x_2 - x_3$?
- what is a possible mathematical criterion to test that w is not orthogonal to the line $x_1 - x_3$?

III Next, **make this vector w into a vector w_2 which is orthogonal** to the line $z = x_2 - x_3$ by using the following:

$$w_2 = w - (w \cdot z)\frac{z}{\|z\|^2}$$

Why do I want you to do it?

$$w_2 \cdot (x_2 - x_3) = 0$$

implies that:

$$w_2 \cdot x_2 + b = w_2 \cdot x_3 + b$$

means $f(x_2) = f(x_3)$ no matter what choice of bias. Once you choose a bias, both points will lie on the same side of the hyperplane! Also note that the projection formula has a more symmetrical writing form which makes clearer what it does:

$$w_2 = w - (w \cdot \frac{z}{\|z\|})\frac{z}{\|z\|}$$

We remove from w the component in the direction of $\frac{z}{\|z\|}$. The amount of the component is $w \cdot \frac{z}{\|z\|}$

IV **Use w_2 to for your classifier as weight/direction, and determine a bias b which separates points x_1 from the other two.**

V If w would be parallel to $x_2 - x_3$. Then applying step III would result in the zero vector which is useless. **Answer for yourself:** If you would choose a w which is parallel to $x_2 - x_3$ and **use it without step III** to try to find a separating hyperplane, can you think of a case where x_1 is positioned in such a way, that w is unsuitable as a separating hyperplane ? Think of it in 2 or 3 dimensions to produce an answer.

Off the tasks: This gives you a hint, why I asked you in step III to remove the component in w parallel to $x_2 - x_3$ before solving for a bias ;-).

Offtopic: this works for any K points z_k in D dims for $K <= D$ which are linearly separable from one single point x, such that the z_k are to be classified against x.

- find a hyperplane spanning these K points z_k – for this you can consider the span of $z_2 - z_1, z_3 - z_1, \ldots, z_k - z_1$.

  If $x - z_1$ lies in the hyperplane, then you cannot separate it, otherwise:

- find a vector w_2 which is orthogonal to the hyperplane ($D - K + 1$ dim space of those orthogonals, it must be orthogonal to all the $z_i - z_1$).
- use that w_2 to solve for the bias.

3.4 just for reading: Gram-Schmid-Orthogonalization

`https://en.wikipedia.org/wiki/Gram%E2%80%93Schmidt_process` for a short version `https://www.math.tamu.edu/~yvorobet/MATH304-2011A/Lect3-05web.pdf` for a longer version