CSE 547 - Assignment 1

Philip Pham

May 19, 2018

Problem 0

List of collaborators: I have not collaborated with anyone.

List of acknowledgements: None.

Certify that you have read the instructions: I have read and understood these policies.

Problem 1: Gaussian Random Projections and Inner Products

Let $\phi(x) = \frac{1}{\sqrt{m}} Ax$ represent our random projection of $x \in \mathbb{R}^d$, with A an $m \times d$ projection matrix with each entry sampled i.i.d from $N(0,1)$. Note that each row of A is a random projection vector, $v^{(i)}$.

The *norm preservation theorem* states that for all $x \in \mathbb{R}^d$, the norm of the random projection $\phi(x)$ approximately maintains the norm of the original x with high probability:

$$\mathbb{P}\left((1-\epsilon)\|x\|^2 \leq \|\phi(x)\|^2 \leq (1+\epsilon)\|x\|^2\right) \geq 1 - 2\exp\left(-\left(\epsilon^2 - \epsilon^3\right) m/4\right), \tag{1}$$

where $\epsilon \in (0, 1/2)$.

Using the norm preservation theorem, prove that for any $u, v \in \mathbb{R}^d$ such that $\|u\| \leq 1$ and $\|v\| \leq 1$,

$$\mathbb{P}\left(|u \cdot v - \phi(u) \cdot \phi(v)| \geq \epsilon\right) \leq 4\exp\left(-\left(\epsilon^2 - \epsilon^3\right) m/4\right) \tag{2}$$

Solution

Proof. First note that

$$(1-\epsilon)\|u+v\|^2 \leq \|\phi(u+v)\|^2 \leq (1+\epsilon)\|u+v\|^2$$

implies that

$$\|u+v\|^2 - 2\epsilon \leq \|\phi(u+v)\|^2 \leq \|u+v\|^2 + 2\epsilon \tag{3}$$

by triangle inequality and the assumption of the norms of u and v.

Thus, the probability of the event in Equation 3 than that of Equation 1.

Using this and taking the additive inverse, we have that

$$\mathbb{P}\left(\|\phi(u+v)\|^2 \notin \left[\|u+v\|^2 - 2\epsilon, \|u+v\|^2 + 2\epsilon\right]\right) \leq 2\exp\left(-\left(\epsilon^2 - \epsilon^3\right) m/4\right) \tag{4}$$

$$\mathbb{P}\left(\|\phi(u-v)\|^2 \notin \left[\|u-v\|^2 - 2\epsilon, \|u-v\|^2 + 2\epsilon\right]\right) \leq 2\exp\left(-\left(\epsilon^2 - \epsilon^3\right) m/4\right). \tag{5}$$

By the countable sub-additivity property of probability distributions, we have that the probability of both these events occurring is at most $4\exp\left(-\left(\epsilon^2-\epsilon^3\right)m/4\right)$. Thus, we are done if we can show $\{|u\cdot v-\phi(u)\cdot\phi(v)|\geq\epsilon\}$ subsets these two conditions.

If we have the pair

$$\|\phi(u+v)\|^2\leq\|u+v\|^2-2\epsilon\Rightarrow\|u+v\|^2-\|\phi(u+v)\|^2\geq2\epsilon \tag{6}$$

$$\|\phi(u-v)\|^2\geq\|u-v\|^2+2\epsilon\Rightarrow\|u-v\|^2-\|\phi(u-v)\|^2\geq2\epsilon, \tag{7}$$

we can use the linearity of ϕ and the expansion $\|u\pm v\|^2=\|u\|^2+\|v\|^2\pm2u\cdot v$, we can add the two inequalities to obtain

$$4\left(u\cdot v-\phi\left(u\right)\cdot\phi\left(v\right)\right)\geq4\epsilon.$$

Thus, we have that the conditions in Equations 4 and 5 imply $u\cdot v-\phi\left(u\right)\cdot\phi\left(v\right)\geq\epsilon$.

Similarly, we show that the pair

$$\|\phi(u+v)\|^2\geq\|u+v\|^2+2\epsilon \tag{8}$$

$$\|\phi(u-v)\|^2\leq\|u-v\|^2-2\epsilon \tag{9}$$

implies $u\cdot v-\phi\left(u\right)\cdot\phi\left(v\right)\leq-\epsilon$, which gives us $|u\cdot v-\phi\left(u\right)\cdot\phi\left(v\right)|\geq\epsilon$. □

Problem 2: Locality-Sensitive Hashing (LSH) for Angle Similarity

Suppose our set of n points $D=\{p_1,\ldots,p_n\}$ are vectors in d dimensions. Our problem is: given a query point q find a point $p\in D$, which has a small angle with q. Recall that the angle between two vectors a and b is $\cos^{-1}\left(\frac{a\cdot b}{\|a\|\|b\|}\right)$.

As doing this exactly may be computationally expensive, let us try to do this approximately with a fast algorithm. The approximate objective is as follows: suppose there exists a point $p\in D$ which has angle less than θ with p, then our goal is return a point with angle less than $c\theta$, where $c>1$.

Let us try to do this with LSH. Let us consider the a family of hash functions, where $h(p)=\text{sign}\left(u\cdot p\right)$ where we will sample u uniformly at random from a Gaussian (or from a unit sphere).

1. Provide an exact expression for $\mathbb{P}\left(h\left(p\right)=h\left(p'\right)\right)$ based on some geometric relation between p and p'.

Solution

Define

$$\text{angle}\left(u,v\right)=\cos^{-1}\left(\frac{u\cdot v}{\|u\|\|v\|}\right), \tag{10}$$

which is the angle between two vectors.

Then,

$$\boxed{\mathbb{P}\left(h\left(p\right)=h\left(p'\right)\right)=1-\frac{\text{angle}\left(p,p'\right)}{\pi}.} \tag{11}$$

2. Provide an expression for P_1 and P_2 in terms of θ and $c\theta$. Note that since we want a small angle, we should use:

   (a) If $\text{angle}(p, p') < \theta$, then $\mathbb{P}(h(p) = h(p')) \geq P_1$.

   (b) If $\text{angle}(p, p') > c\theta$, then $\mathbb{P}(h(p) = h(p')) \leq P_2$.

   Solution

   If $\text{angle}(p, p') < \theta$, then

   $$\mathbb{P}(h(p) = h(p')) = 1 - \frac{\text{angle}(p, p')}{\pi} \geq 1 - \frac{\theta}{\pi},$$

   so $\boxed{P_1 = 1 - \frac{\theta}{\pi}.}$

   If $\text{angle}(p, p') > c\theta$, then

   $$\mathbb{P}(h(p) = h(p')) = 1 - \frac{\text{angle}(p, p')}{\pi} \leq 1 - \frac{c\theta}{\pi},$$

   so $\boxed{P_2 = 1 - \frac{c\theta}{\pi}.}$

3. Provide expressions for query time for point q, the space to store the hash tables, and the construction time of our datastructure.

   Solution

   Suppose we have L hash functions. If we use the algorithm discussed in class, to query a point, we need to compute L hashes. Then, up to 3 times, we iterate through the buckets: for each bucket, we choose a point and check how close it is to q; if it is $c\theta$ close, we stop. The worst case is that we decide there exists no point that is θ close to q. In this case, we iterate through the L buckets 3 times, so the time complexity is $O(L)$.

   For the space needed to store the hash tables, we need to store L bits for each point, so the space needed is $O(nL)$

   For construction, we need to compute L hashes for each point, so the computational complexity is $O(nL)$ as well.

Problem 3: Dual Coordinate Ascent

Consider the problem

$$\min_w L(x), \text{ where } L(x) = \sum_{i=1}^{n} (w \cdot x_i - y_i)^2 + \lambda \|w\|^2. \tag{12}$$

1. Show that the solution for Equation 12 is obtained for weights

$$w^* = \left(X^\intercal X + \lambda I\right)^{-1} X^\intercal Y \tag{13}$$

$$= \frac{1}{\lambda} X^\intercal \alpha^*, \tag{14}$$

where $\alpha^* = (I + XX^\intercal/\lambda)^{-1}$.

Solution

Proof. We can take the derivative of L in Equation 12 directly. Note that $D\left(x \mapsto Ax\right)(x) = A$ and $D\left(x \mapsto x^\intercal x\right)(x) = 2x^\intercal$. Therefore by the chain rule,

$$D\left(x \mapsto (Ax)^\intercal (Ax)\right)(x) = 2x^\intercal A^\intercal A. \tag{15}$$

We can reformulate Equation 12 as a function of w

$$\begin{aligned} l_{X,y}(w) &= (Aw - y)^\intercal (Aw - y) + \lambda w^\intercal w \\ &= (Aw)^\intercal (Aw) - 2y^\intercal Aw + y^\intercal y + \lambda w^\intercal w. \end{aligned} \tag{16}$$

Taking the derivative, we have that

$$D\left(l_{X,y}\right)(w) = 2w^\intercal X^\intercal X - 2y^\intercal X + 2\lambda w^\intercal. \tag{17}$$

Setting Equation 17 to 0 and solving for w, we have

$$\begin{aligned} 0 &= 2w^\intercal X^\intercal X - 2y^\intercal X + 2w^\intercal \\ w^\intercal \left(X^\intercal X + \lambda I\right) &= y^\intercal X \\ \left(X^\intercal X + \lambda I\right) w &= X^\intercal y \\ w &= \left(X^\intercal X + \lambda I\right)^{-1} X^\intercal y. \end{aligned}$$

Since Equation 16 is a quadractic form, the problem is convex, and

$$w^* = \left(X^\intercal X + \lambda I\right)^{-1} X^\intercal Y$$

minimizes Equation 12.

Now, note that

$$\left(X^\intercal X + \lambda I\right) X^\intercal = X^\intercal X X^\intercal + \lambda X^\intercal = X^\intercal \left(XX^\intercal + \lambda I\right).$$

Multiplying on the left by $\left(X^\intercal X + \lambda I\right)^{-1}$ and on the right by $\left(XX^\intercal + \lambda I\right)^{-1}$, we have that

$$X^\intercal \left(XX^\intercal + \lambda I\right)^{-1} = \left(X^\intercal X + \lambda I\right)^{-1} X^\intercal.$$

Substituting this into Equation 13, we obtain

$$\begin{aligned} w^* &= X^\intercal \left(XX^\intercal + \lambda I\right)^{-1} y \\ &= X^\intercal \left(\lambda \left(I + \frac{XX^\intercal}{\lambda}\right)\right)^{-1} y \\ &= \frac{1}{\lambda} X^\intercal \left(I + \frac{XX^\intercal}{\lambda}\right)^{-1} y, \end{aligned}$$

which gives us the desired result. □

If $\lambda = 0$, in general, this is not true since $XX^\intercal + \lambda I$ may not be invertable when $n > d$. However, if $d \geq n$, and rank $(X) \geq n$, Equation 14 may still be well-defined.

2. Define

$$G(\alpha_1, \alpha_2, \ldots, \alpha_n) = \frac{1}{2}\alpha^\intercal (I + XX^\intercal/\lambda)\alpha - Y^\intercal\alpha. \tag{18}$$

Start with $\alpha = 0$. Choose coordinate i randomly, and update

$$\alpha_i = \arg\min_z G(\alpha_1, \ldots, \alpha_{i-1}, z, \alpha_{i+1}, \ldots, \alpha_n). \tag{19}$$

Show that the solution to the inner optimization problem for α_i is:

$$\alpha_i = \frac{y_i - \frac{1}{\lambda}\left(\sum_{j\neq i}\alpha_j x_j\right)\cdot x_i}{1 + \|x_i\|^2/\lambda}. \tag{20}$$

Solution

Proof. We can take the partial derivative of Equation 18 directly to obtain

$$\begin{aligned}\frac{\partial G}{\partial \alpha_i} &= \alpha_i + \frac{1}{\lambda}(\alpha^\intercal XX^\intercal)_i - y_i \\ &= \alpha_i + \frac{1}{\lambda}\left(\alpha_i\|x_i\|^2 + \sum_{j\neq i}\alpha_j(x_j\cdot x_i)\right) - y_i.\end{aligned} \tag{21}$$

Setting Equation 21 to 0, solving for α_i, and taking advantage of convexity, we find

$$\alpha_i = \frac{y_i - \left(\sum_{j\neq i}\alpha_j x_j\right)\cdot x_i}{1 + \|x_i\|^2/\lambda} \tag{22}$$

minimizes Equation 18 as a function of α_i, and solves the inner optimization problem. □

3. What is the computational complexity of this update, as it is stated?

Solution

The complexity of updating α_i with Equation 20 is $O(nd)$ since we need to iterate over the n rows of X, and take the d-dimensional dot product of each row with x_i.

4. What is the computational complexity of one stochastic gradient descent update?

Solution

The complexity of one stochastic gradient descent update is $O(d)$. We computed the derivative in Equation 17 for the full matrix X. In stochastic gradient descent we'd replace X by a vector by randomly sampling a row from X. Then, to compute the gradient we have to do some dot products along with scalar operations.

5. Now consider the procedure.

   - Start with $\alpha = 0$, $w = \frac{1}{\lambda} X^\intercal \alpha = 0$.
   - Choose coordinate i randomly and perform the following update:
     - Compute the differences:

$$\Delta\alpha_i = \frac{(y_i - w \cdot x_i) - \alpha_i}{1 + \|x_i\|^2 / \lambda} \tag{23}$$

     - Update the parameters as follows:

$$\begin{aligned} \alpha_i &\leftarrow \alpha_i + \Delta\alpha_i \\ w &\leftarrow w + \frac{\Delta\alpha_i}{\lambda} x_i. \end{aligned} \tag{24}$$

   Prove that the update rule in Equation 24 is valid.

Solution

Proof. Let α' and w' be the result of updating coordinate i of α. Assume that $w = \frac{1}{\lambda} X^\intercal \alpha$. This is true when $\alpha = 0$. We will show that this invariant holds as α is updated.

To see that, the update rule for w is valid, we can rewrite

$$w = \frac{\alpha_1}{\lambda} x_1 + \cdots + \frac{\alpha_i}{\lambda} x_i + \cdots + \frac{\alpha_n}{\lambda} x_n, \tag{25}$$

so

$$\begin{aligned} w' &= w + \frac{\Delta\alpha_i}{\lambda} x_i \\ &= \frac{\alpha_1}{\lambda} x_1 + \cdots + \frac{\alpha_i + \Delta\alpha_i}{\lambda} x_i + \cdots + \frac{\alpha_n}{\lambda} x_n \\ &= \frac{\alpha_1}{\lambda} x_1 + \cdots + \frac{\alpha_i'}{\lambda} x_i + \cdots + \frac{\alpha_n}{\lambda} x_n \\ &= \frac{1}{\lambda} X^\intercal \alpha'. \end{aligned}$$

Thus, the w update is valid.

To see that the α update is valid, we show that Equations 20 and 24 are equivalent. Both algorithms initiate $\alpha = 0$, so they are equivalent at the initial step.

By using the definition $w = \frac{1}{\lambda} X^\intercal \alpha$,

$$\begin{aligned} \alpha_i' &= \alpha_i + \Delta\alpha_i \\ &= \frac{(y_i - w \cdot x_i) - \alpha_i}{1 + \|x_i\|^2 / \lambda} + \frac{\alpha_i + \alpha_i \|x_i\|^2 / \lambda}{1 + \|x_i\|^2 / \lambda} \\ &= \frac{1}{1 + \|x_i\|^2 / \lambda} \left(y_i - \frac{1}{\lambda} \left(\sum_{j \neq i} \alpha_j x_j \right) \cdot x_i - \frac{1}{\lambda} \alpha_i \|x_i\|^2 + \frac{1}{\lambda} \alpha_i \|x_i\|^2 \right) \\ &= \frac{y_i - \frac{1}{\lambda} \left(\sum_{j \neq i} \alpha_j x_j \right) \cdot x_i}{1 + \|x_i\|^2 / \lambda}, \end{aligned}$$

so both update rules are equivalent. □

6. What is the computation complexity of the update defined by Equations 23 and 24?

Solution

The computation complexity is $O(d)$. Computing the dot product when computing $\Delta\alpha_i$ and updating w are both $O(d)$ operations. Everywhere else, we do scalar operations. This is much faster than the $O(nd)$ update for Equation 20.

Problem 4: Project Milestone

Build a simple object dection model. Our object detector will comprise of a binary classifier per category: given features of an image patch corresponding to a bounding box, does this patch contain an object of the category of interest?

Solution

Experimental Setup

I treated this problem as a multi-label classification problem. Each image patch may contain objects from one of 17 categories that are part of the `vehicle` or `animal` supercategories in the COCO dataset[1].

Region proposal for patches were found with the selective search fast algorithm[2] Then, the bounding box coordinates were projected and turned into features with adaptive max pooling.

Negatives could then just be patches that don't contain a specific category. Additional patches that contain objects of no categories could be added, but I didn't find this helpful. Mining hard negatives didn't seem to do much either.

After training, validation, and test datasets were created, various models were tried. The best model was that with the lowest cross-entropy loss on the validation dataset. Models tried were logistic regression, a multi-layer perceptron with 1 layer of hidden units, and a multi-layer perceptron with 2 layers of hidden units. L_2 regularization was applied in all the models.

Model Selection

The best model was the multi-layer perceptron with 2 layers of hidden units. The first layer had 512 units, and the second layer had 256 units. The loss and average precision score on the training and validation dataset can be seen in Figures 1 and 2. The best metrics are summarized in Table 1.

For the L_2 regularization parameter, $\lambda = 0.004$ was found to work best. A smaller parameter led to overfitting as seen by a big gap between loss on the training dataset and validation dataset. A larger parameter led to a degredation in performance on the validation dataset.

For the optimization, stochastic gradient descent was used with Nesterov's momentum. The learning rate was 0.05, and momentum was 0.9. The batch size was 16.

[1] I chose to use these 17 categories instead of all 80 to simplify the problem and reduce training time.

[2] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. International journal of computer vision, 104(2):154-171, 2013.

Figure 1: The loss between the training and validation dataset is rather small, which indicates the choice of λ was appropriate.

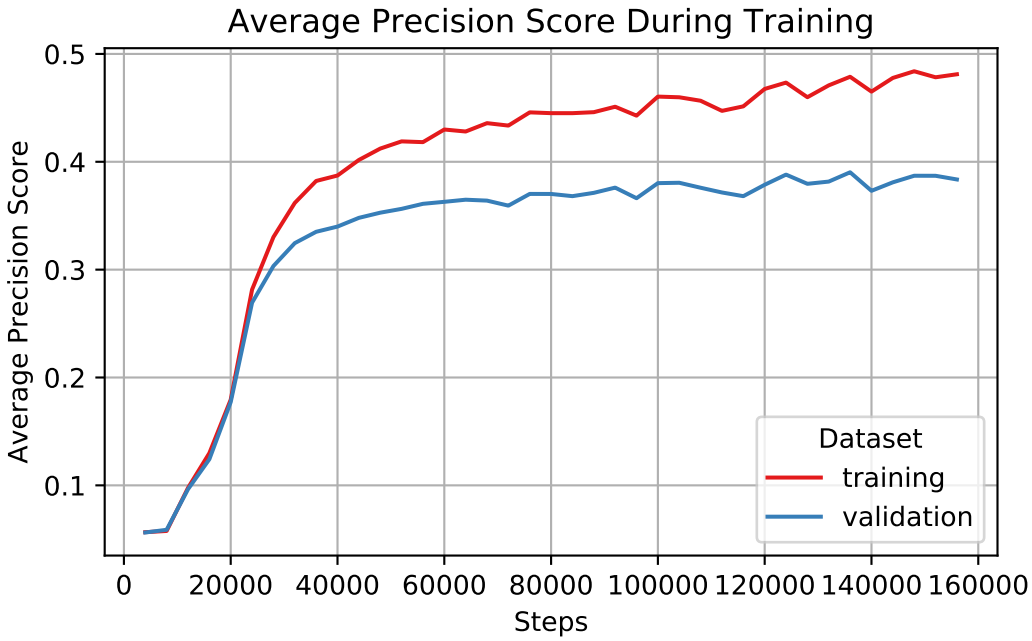

Figure 2: The average precision score is averaged over the categories without weighting.

	Average Precision Score	Loss
Training	0.483937	0.136209
Validation	0.390272	0.148008

Table 1: The best loss and average precision score achieved during training.

Label	Training Observations	Test Observations	Average Precision Score
bicycle	2166	416	0.140143
car	12179	2476	0.476441
motorcycle	2630	614	0.381392
airplane	3045	447	0.500307
bus	2972	543	0.313419
train	873	160	0.224836
truck	6058	1238	0.215611
boat	2961	650	0.262890
bird	6485	1473	0.428072
cat	2516	569	0.514017
dog	3387	562	0.227955
horse	4802	1074	0.332912
sheep	7702	1378	0.533573
cow	7733	1582	0.380756
elephant	5177	853	0.566383
bear	851	113	0.215011
zebra	5147	1153	0.680173
giraffe	3908	826	0.747567

Table 2: Generally, less frequently occuring categories have lower scores.

Evaluation

The best model was evaluated against the test dataset. On the test dataset, the average precision score was 0.396748. The numbers are broken out by category in Table 2.

Discussion

The linear model essentially does a logistic regression for each category. The advantage of the multi-layer perceptron is that the weights for the non-terminal layers are shared between each categories. Thus, if one believes that the hidden units are learning an intermediate representation of the image, categories with not so many examples can leverage what is learned from the more abundant categories.

In reality, the multi-layer perceptron only gives us a 0.03 boost in average precision score over the linear model, so it's questionable if the additional complexity is worthwhile.