COMPSCI 514: Problem Set 2

Released: 9/17.

Due: 9/28 by 8:00pm in Gradescope.

Instructions:

- You are allowed to, and highly encouraged to, work on this problem set in a group of up to three members.
- Each group should **submit a single solution set**: one member should upload a pdf to Gradescope, marking the other members as part of their group in Gradescope.
- You may talk to members of other groups at a high level about the problems but **not work through the solutions in detail together**.
- You must show your work/derive any answers as part of the solutions to receive full credit.

1. Random Hashing for Streaming Computation (12 points + 4 bonus)

Let $x_1, x_2, \ldots, x_n$, be a stream of inputs. Each is an integer in $[m]$.

1. (4 points) Give an algorithm using as little memory as possible that selects a random distinct element from the stream. E.g., if the stream is $x_1 = 3$, $x_2 = 5$, $x_3 = 2$, $x_4 = 3$, then the algorithm should output $2, 3$, or 5, each with probability $1/3$. How much memory does your algorithm require? You may assume for simplicity that any single number takes $O(1)$ units of memory to store.

2. (4 points) Modify your algorithm to output a random element from the stream with probability proportional to its frequency. E.g., if the stream is $x_1 = 3$, $x_2 = 5$, $x_3 = 2$, $x_4 = 3$, then the algorithm should output 2 with probability 1/4, 3 with probability 1/2, and 5 with probability 1/4. Your algorithm should work even if n is not known ahead of time.

3. (4 points) Let $f(v)$ be the frequency of some value v in the stream. I.e., $f(v) = |\{i \in [n] : x_i = v\}|$. Consider running t independent instantiations of the algorithm from part (2) and estimating $f(v)$ as $\tilde{\mathbf{f}}(v) = \mathbf{s}(v) \cdot \frac{n}{t}$, where $\mathbf{s}(v)$ is the number of times that v is sampled out of the t samples taken. How large must you set t so that with probability $\geq 1 - \delta$, $\left|\tilde{\mathbf{f}}(v) - f(v)\right| \leq \epsilon n$?

4. **Bonus (4 points)**: For some constant $c > 0$, it is possible to create 2^{cm} subsets of $[m]$, each with $m/2$ elements, such that no two of the subsets share more than $3m/8$ elements in common. Use this fact to argue that any *deterministic algorithm* that guarantees to approximate the number of distinct elements d in a data stream of n items with error less

than $d/16$ (i.e., the algorithm outputs $\hat{d}$ with $|\hat{d} - d| < d/16$) must use $\Omega(d)$ bits of memory. How does this compare to the memory usage of the MinHash algorithm presented in class? **Hint:** Think about how many different states an algorithm using $O(d)$ bits can be in.

2. Random Group Testing (15 points)

Suppose, hypothetically, that we are facing a global pandemic and wish to control the spread by testing as many individuals as possible. Unfortunately testing is expensive. A common method to address is issue is to test individuals in *groups*. I.e., the biological samples (e.g., nose swabs) from multiple patients are combined into a single sample and tested for the disease all at once. If the test returns negative, it means that all individuals in the group are negative. If the test comes back positive, it means that at least one individual in the group has the disease. We'll see here how this strategy can be used to significantly save on the number of tests required to identify a small subset of positive individuals in the population.

1. (4 points) Consider the following two-stage testing scheme: we divide a population of n individuals into C arbitrary groups. We then test each of these groups in aggregate. For any group that comes back positive, we retest all members of the group individually. Show that there is a choice for C such that, if p individuals in the population are positive, we can find all of those individuals with $\leq 2\sqrt{np}$ tests. You may assume that p is known in advance (often it can be estimated accurately from the positive rate of prior tests).
2. (1 point) Say we are testing the UMass Amherst student body. $n = 30,000$ and there is a 1% positivity rate, so $p = 300$. How many tests does the strategy of part (1) save over simply individually testing each member of the student body?
3. (6 points) Consider the following improved randomized scheme: collect k samples from each individual. Then, repeat the following process k times: randomly partition the population into C groups (i.e., each individual is assigned independently to group i with probability $1/C$ for $i = 1, 2, ..., C$), and test each group in aggregate. Once this process is complete, report that an individual is positive if *every group they were part of tested positive.* Report that an individual is negative if *any of the groups they were part of tested negative.* Show that for $C = O(p)$ this scheme finds all truly positive patients, and that each negative patient is marked positive with probability $\leq \frac{1}{2^k}$.
4. (2 points) Show that if we set $k = O(\log n)$, then with probability $\geq 9/10$ the method of part (3) yields no false positives, no false negatives, and requires just $O(p \log n)$ tests.
5. (2 points) What algorithm covered in class does the scheme from part (3) resemble?

3. Locality Sensitive Hashing in Use (8 points + 5 bonus)

We would like to use locality sensitive hashing to search for similar handwritten digit images from the MNIST dataset. We will measure similarity using cosine similarity and use the SimHash method. Throughout the problem, use the data provided in the `mnist.mat` file. It is helpful to initially normalize all images to have unit Euclidean norm. Include printouts of any code in your problem set submission.

Given an input image x, you would like to identify any image y with cosine similarity $\langle x, y \rangle \geq .95$. Your task is to pick a number of table repetitions t and a hash signature length r so that any image close to x is identified with probability at least 98%. At the same time, you would like to minimize the number of false positives in your hashing scheme.

1. (2 points) Using that for two images x and y, $\Pr[SimHash(x) = SimHash(y)] = 1 - \frac{\theta}{\pi}$ where θ is the angle between x and y in radians, determine for each $r \in \{1, \ldots, 30\}$ the number of repetitions t required to achieve the desired false negative rate of 2%. You may want to write code to solve this problem, but please also describe in words/equations how you determined the required t for a given r. Assume that x and y are unit norm through this problem.

2. (2 points) Given a fixed value of r and t, what is the expected number of collisions between images x and y with cosine similarity $\langle x, y \rangle = s$ across t hash tables? Give an equation in terms of r, t, and s. Count collisions happening in different tables as different collisions. Assume that a collision only occurs if x and y have matching SimHash signatures (i.e., ignore additional collisions that occur when inserting to the hash table).

3. (4 points) For each $r \in \{1, \ldots, 30\}$, use the $10,000$ images in `testX` to estimate the expected number of collisions that will be encountered for an image x when using enough hash tables to ensure a 2% false negative rate (as determined in part (1)). Plot the expected number of collisions as a function of r. Discuss the trend, and how you might use this information to choose r for an application. Include both a description with words/equations along with any code used. **Hint:** For a given image x in `testX`, compute the expected total number of collisions that will occur with other images in `testX`. Then average over all $10,000$ possibilities for x in the set to get your estimate.

4. **Bonus (5 points)**: To get a feel for how SimHash is working, set $r = 35$ and compute SimHash signatures for the $60,000$ images in `trainX`. For simplicity, do not worry about using any repetitions (i.e., use $t = 1$.) Focusing on a single digit type, run a few near neighbor queries in order to find a few (maybe 6 or so) sets of images that hash to the same signatures. Plot these sets of colliding images. What do you notice? Are the colliding images similar? Are there many false positives?