CS49/249 (Randomized Algorithms), Spring 2021 : Lecture 8

Topic: Finding Similar Pairs via Importance Sampling

Disclaimer: These notes have not gone through scrutiny and in all probability contain errors. Please discuss in Piazza/email errors to deeparnab@dartmouth.edu

---

- ***A Data-Mining Application.*** Imagine you are a data-mining company, and you have collected "features" about a large population. We will let $[n]$ denote the population, and $[d]$ denote the set of features. For us both n and d are huge (in tens of millions, say). The "feature vector" v_i associated with any individual is the $0,1$ d-dimensional vector where $v_i[r]$ is 1 if the rth feature is present in the individual and 0 otherwise. The overarching assumption is going to be

  > *These feature vectors are very sparse. Typical $||v_i||_1 = s$ where $s \ll d, n$, and for now, can be thought of as some fixed constant.*

  Challenge: We are interested in finding *pairs* of individuals whose feature vectors have a lot of intersection. More precisely, for a parameter K, we wish to find *all* pairs $(i,j) \in [n] \times [n]$ such that $\langle v_i, v_j \rangle \geq K$.

  To get a feel for this, assume that every individual has a feature vector where every feature is "turned on" with probability $\frac{s}{d}$. Then, for any pair of individuals, we expect $\frac{s^2}{d}$ features to be in common. Imagine $s \ll \sqrt{d}$, and so, this expected value $\ll 1$. In such a scenario, if a pair (i,j) have $K \approx d^{1/4}$ features in common, then that is something the data mining company would be interested in.

  The question is how fast can we do this? The *naivest* approach would be to go over all pairs of individuals, compute the dot-products of the feature vectors, and threshold. This takes $O(n^2 d) \approx O(n^3)$ time. This also is quite naive since it doesn't exploit sparsity. In today's lecture we will see (a) a matrix multiplication viewpoint of this problem, (b) how importance sampling helps solve this problem, and (c) construction of an importance sampler due[1] to Edith Cohen and David Lewis.

- ***Gram Matrix and Similar Pairs.*** A change of point of view is super helpful for solving this problem. Consider the $d \times n$ matrix A where the ith column of A is the feature vector v_i of the ith individual. Then, the dot-products of the various pairs of feature vectors are captured by the $n \times n$ Gram matrix $A^\top A$ since

$$(A^\top A)_{ij} = \sum_{r=1}^{d} A^\top_{ir} A_{rj} = \sum_{r=1}^{d} A_{ri} A_{rj} = \sum_{r=1}^{d} v_i[r] v_j[r] = \langle v_i, v_j \rangle$$

  Therefore, the challenge above can be re-casted as *find all pairs of* $(i,j) \in [n] \times [n]$ *such that* $(A^\top A)_{ij} \geq K$.

  A parameter of interest which will drive the quality of the algorithm would be sum of entries of the Gram matrix. Let us call this $\Gamma(A)$.

$$\Gamma(A) := \sum_{i=1}^{n} \sum_{j=1}^{n} (A^\top A)_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{n} \langle v_i, v_j \rangle$$

[1] *Approximating Matrix Multiplication for Pattern Recognition Tasks.* E. Cohen and D. D. Lewis, J. Algorithms 30(2): 211-252, 1999

Remark: *Note that we are also including self-dot-products. Ideally, we would remove these but for simplicity we just include them.*

Remark: *In most regimes of interest,* $\Gamma(A)$ *is small since most* $\langle v_i, v_j \text{range}$*'s are* 0 *since their features don't intersect. Indeed, the value of* K *one is interested in may be* $K := \Gamma(A)/H$*, and there can be only* H *such pairs. Perhaps, we expect a "community" of* c *highly similar people, and then* $H \approx c^2$*.*

- ***Finding Similar Pairs via Importance Sampling.*** The algorithm to solve the challenge will be via importance sampling. In particular, in the next bullet point we will see the proof of the following theorem. In plain English, the sampler returns a pair of individuals proportional to the dot-product of their feature vectors.

Theorem 1 (Cohen-Lewis Sampler)**.** Given a ***non-negative*** $d \times n$ matrix A, one can spend $O(\text{nnz}(A) + n + d)$ pre-processing time and design an importance sampler $\mathcal{A}$ which, upon query returns a *sample* (i, j) with the following property:

$$\mathbf{Pr}[(i,j) \text{ sampled by } \mathcal{A}] = \frac{(A^\top A)_{ij}}{\Gamma(A)}$$

Here, $\text{nnz}(A)$ is the number of non-zero entries in A.

Using the theorem, we can solve our challenge as follows.

1: **procedure** FINDSIMILARPAIRS($A; K$):
2: ▷ *Return all pairs of columns of* A *with dot-product* $\geq K$*.*
3: Run Cohen-Lewis Algorithm promised in Theorem 1 to get $\mathcal{A}$.
4: Sample $N := \left\lceil \frac{\Gamma(A)}{K} \cdot \ln\left(\frac{\Gamma(A)}{K\delta}\right) \right\rceil$ pairs from $\mathcal{A}$ to get R.
5: **return** R.

Theorem 2. For any non-negative matrix A, with probability $\geq 1-\delta$, FINDSIMILARPAIRS(A, K) returns all pairs i, j with $\langle v_i, v_j \rangle \geq K$.

Remark: *If* $K = \Gamma(A)/H$ *and* δ *were a constant, then we only need* $O(H \ln H)$ *many samples :* $O(\ln H)$ *for every such highly similar pair.*

Remark: *One issue of the algorithm is that it may return pairs which are not similar, that is, could have false positives. But this can be taken care of, and is a very instructive exercise (see problem set). In particular, one can add multiply* N *by another* $\Theta(1/\varepsilon^2)$ *and take care of the false positives as follows : in the theorem statement we can state for* every *pair returned one has* $\langle v_i, v_j \rangle \geq K(1-\varepsilon)$*.*

Proof. The proof follows by a union bound. Let R^* be the collection of all pairs (i,j) such that $\langle v_i, v_j \rangle \geq K$. How many pairs can there be in R^*? Since the sum of all $\langle v_i, v_j \rangle$'s is $\Gamma(A)$, the number of pairs where the dot-product is $\geq K$ is at most $|R^*| \leq \frac{\Gamma(A)}{K}$. Furthermore, for *every* $(i,j) \in R^*$, the probability $\mathbf{Pr}[(i,j) \text{ sampled by } \mathcal{A}] = \frac{(A^\top A)_{ij}}{\Gamma(A)} \geq \frac{K}{\Gamma(A)}$. Therefore, for this fixed $(i,j) \in R^*$,

$$\begin{aligned}
\mathbf{Pr}[\text{none of the } N \text{ samples are } (i,j)] &\leq \left(1 - \frac{K}{\Gamma(A)}\right)^N \\
&\leq \left(1 - \frac{K}{\Gamma(A)}\right)^{\frac{\Gamma(A)}{K} \cdot \ln\left(\frac{\Gamma(A)}{K\delta}\right)} \\
&\leq e^{\ln\left(\frac{\Gamma(A)}{K\delta}\right)} = \frac{K}{\Gamma(A)} \cdot \delta \qquad (1)
\end{aligned}$$

This is an upper bound on the probability a fixed $(i,j) \in R^*$ is missed out. Therefore, the probability that ***any*** $(i,j) \in R^*$ is missed out is by the union bound

$$\leq \sum_{(i,j) \in R^*} \mathbf{Pr}[\text{none of the } N \text{ samples are } (i,j)] \leq |R^*| \cdot \frac{K}{\Gamma(A)} \cdot \delta \leq \delta \qquad \square$$

- ***Cohen-Lewis Important Sampler for Non-negative Matrix Multiplication.***

  Upon input the $d \times n$ matrix A, in the pre-processing step, one computes the following score for every *row* (feature) : for every $r \in [d]$, we compute $\mathsf{score}(r) := \sum_{i=1}^{n} A_{ri}$, that is, $\mathsf{score}(r)$ is the ℓ_1-norm of the rth row. This clearly takes $O(\mathrm{nnz}(A))$ time.

  Given these scores, here is the description of the sampler.

  1: **procedure** COHEN-LEWIS(A; $\mathsf{score}(r) : r \in [d]$):▷ *Return (i,j) with probability proportional to $(A^\top A)_{ij}$*
  2: Sample $r \in [d]$ with probability $\frac{\mathsf{score}^2(r)}{\sum_{r=1}^{d} \mathsf{score}^2(r)}$. ▷ *Proportional to* **square** *of score.*
  3: Sample $i \in [n]$ with probability $\frac{A_{ri}}{\mathsf{score}(r)}$.
  4: Sample $j \in [n]$ with probability $\frac{A_{rj}}{\mathsf{score}(r)}$.
  5: **return** (i,j).

The following lemma completes the proof of Theorem 1

Lemma 1. The probability that (i,j) is returned by COHEN-LEWIS is exactly $\frac{(A^\top A)_{ij}}{\Gamma(A)}$.

Proof.

$$\begin{aligned}\mathbf{Pr}[(i,j) \text{ sampled}] &= \sum_{r=1}^{d} \mathbf{Pr}[(i,j) \text{ sampled} \mid r \text{ sampled}] \cdot \mathbf{Pr}[r \text{ sampled}] \\ &= \sum_{r=1}^{d} \frac{A_{ri}}{\mathsf{score}(r)} \cdot \frac{A_{rj}}{\mathsf{score}(r)} \cdot \frac{\mathsf{score}^2(r)}{\sum_{r=1}^{d} \mathsf{score}^2(r)} \\ &= \frac{\sum_{r=1}^{d} A_{ri}A_{rj}}{\sum_{r=1}^{d} \mathsf{score}^2(r)} \quad = \frac{(A^\top A)_{ij}}{\Gamma(A)} \end{aligned} \tag{2}$$

where the last equality follows from (a) the definition of $(A^\top A)$, and the following observation

$$\Gamma(A) = \sum_{i=1}^{n}\sum_{j=1}^{n}(A^\top A)_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{r=1}^{d} A_{ri}A_{rj} = \sum_{r=1}^{d}\left(\sum_{i=1}^{n} A_{ri}\right)\left(\sum_{j=1}^{n} A_{rj}\right) = \sum_{r=1}^{d} \mathsf{score}^2(r) \quad \square$$

Exercise: *How much time does* COHEN-LEWIS *take? That is, how efficiently can we sample from a distribution over a support of size N, whose marginals are given as input? You should be able to show that this can be done with $O(N)$ preprocessing time and $O(\log N)$ time per query.*

Learning Tidbits:

- Algorithm Design: *In a universe where elements have weights, and we want to obtain the items with "heavy weight", then it suffices to be able to sample items proportional to that weight. Importance Sampling is precisely what does that.*
  For the case of sampling a pair proportional to $(AB)_{ij}$ for two non-negative matrices, Cohen-Lewis got away without multiplying the matrices completely. This is because $(AB)_{ij}$ can be written as $\sum_r A_{ir}B_{rj}$. Once something can be written as a sum-of-products, one can often try to sample an r (with the correct probability), and then sample the A_{rj}, B_{rj}'s. Very powerful.
- Analysis: *(a) Union Bound, again. (b) Although not covered in the lecture, the second remark after Theorem 2 can be done via another application of Chernoff bound. Highly recommended exercise. See problem set.*