Administrivia. This lecture introduces some potential ideas for projects topics. Please reach out to the course staff with questions about projects or for troubleshooting system issues with AWS/cloud credits.

1 Project Ideas

Improving training of Learned Bloom Filters. Consider Michael Mitzenmacher’s paper, “A Model for Learned Bloom Filters and Optimizing by Sandwiching.” Mitzenmacher “sandwiches” a learned oracle with two Bloom filters, and shows increased performance [Mit18]. The learned oracle was trained on the original set, but note that the oracle is actually used on the set of classified positives of the initial Bloom filter. Can we empirically improve the performance of the sandwiched Bloom filter by instead training this oracle on the classified positives of the initial Bloom filter? Is there reason to expect an improvement or not?

Scheduling algorithms of Neural Networks. In his lecture, Mohammad Alizadeh introduced Decima [Mao+18], which was a scheduler that used reinforcement learning to learn workload-specific scheduling for data processing clusters. However, examining what is fundamentally being learned may be interesting; for example, is it possible to theoretically show that standard scheduling algorithms (such as critical path) can be simulated with a neural network?

Theoretical justification for consistency/robustness tradeoff in Ski Rental problem. Manish Purohit recently presented a lecture and paper on augmenting classical online algorithms with machine learning [PSK18]. Specifically, he designed a randomized algorithm that used an oracle to solve the ski rental problem. In this algorithm, he used a hyperparameter to specify how much the oracle is trusted, which gave rise to a consistency/robustness tradeoff. Is it possible to show a lower bound that matches the consistency/robustness tradeoff of the randomized algorithm with an oracle in a meaningful way?

Examining learned partition in Ilya Razenshteyn’s k-Nearest-Neighbors-Search. In the context of the paper by Dong et al (presented in class by Razenshteyn) [Don+19], what happens if the underlying query distribution for the nearest-neighbor-search queries looks very different than the underlying dataset? How does the learned partition fare compared to other models?

Learning embeddings for k-Means clustering. Clustering is often a very difficult problem because objects are not represented/described in a way that makes them obviously separable. Consider the easy case of clustering, where you have pre-labelled red and blue points that are
separable in $\mathbb{R}^n$. Now, imagine the unsupervised case where you don’t have any labels and the structure of points is messy. Now, turn this into a learning problem: given a description of some objects, can you learn an embedding that converts the objects into some reasonable Euclidean geometry such that k-Means or other standard heuristic-based algorithms can cluster the points well? Note that k-Means doesn’t simply optimize a differentiable function, so it is hard to define a loss and run backpropagation as in a neural net. Can this be extended to other related problems like low-rank approximation?

**Improving Cache Storage with Learned Information**  A possible extension to Tim Kraska’s paper and lecture on learned data structures [Kra+17] might examine cache policy. Can we improve the time efficiency of how we access data from SQL queries with smarter cache policies? An example implementation might use a neural network to predict the data most likely to be accessed in the near future based on previous SQL queries. Thodoris Lykouris and Sergei Vassilvitskii explored this possibility by developing a framework that augments online algorithms with an ML oracle that recommends a cache policy [LV18], but developing such an oracle or extending their theory may be fruitful.

**References**


