CONIX RESEARCH CENTER

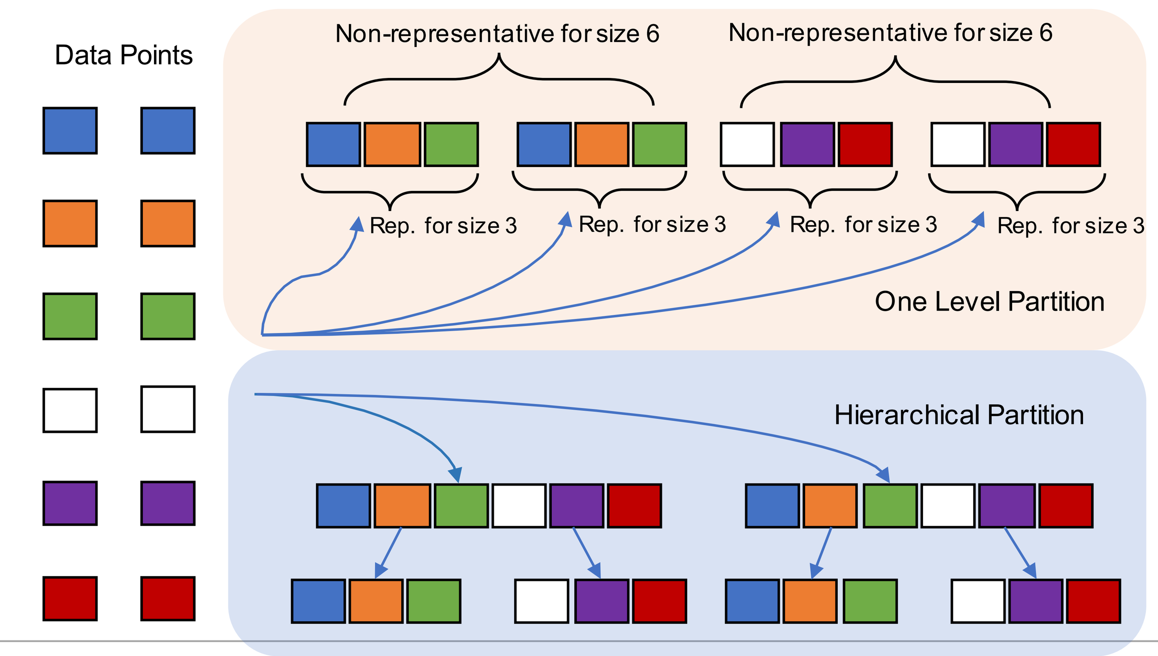
Computing On Network Infrastructure for Pervasive Perception, Cognition and Action

Fixing Mini-batch Sequences with Hierarchical Robust Partitioning

Shengjie Wang, Wenruo Bai, Chandrashekhar Lavania, Jeff A. Bilmes University of Washington, Seattle

MELODI MachinE Learning, Optimization, & Data Interpretation @ UW

- **Problem:** Stochastic gradient methods require random access to data points which can be costly, especially for resource limited devices such as mobile devices and computing at the edge. Also, the randomly sampled mini-batches may contain redundant information.
- **Approach:** We propose a general and efficient hierarchical robust partitioning framework to generate a deterministic sequence of mini-batches, one that offers assurances of being high quality, unlike a randomly drawn sequence.
- **Experiments:** We test our fixed sequences to train neural networks on CIFAR-100 and Imagenet datasets and achieve significantly improved performance compared to randomly sampled sequences.



- **Robust Submodular Partition with Cardinality Constraint:** $\max_{\pi \in \Pi(V,k)} \min_{i=1:m} f(\pi_i(V))$.

  Where f is a submodular function, $\Pi(V, k)$ is all partitions of ground set V with size k. The max-min formulation enforces the worst mini-batch to be representative of V.
- **Algorithm 1 - Greedy:** Pick the worst mini-batch and add the item with highest gain.

To generate the sequence, we order the mini-batches by decreasing function values.

- **Theoretical Guarantees:**

  Theorem 1. For submodular function f on ground set V and mini-batch size k, suppose $m = |V|/k$, Algorithm 1 gives an approximation ratio of $\frac{e-1}{(e-1)m+1}$.

  The bound almost matches the best known bound for the un-constrained robust submodular partition problem within a factor of $m/(m+1)$.

Algorithm 1: Cardinality Constrained Submodular Robust Partition (RobustPartitionK(f, V, k))

input : f, V, k
1 $m := |V|/k$ $R := V$ Let $A_1 = A_2 = \ldots = A_m = \emptyset$
while $R \neq \emptyset$ **do**
2 $j^* \in \mathrm{argmin}_{j,|A_j|<k} f(A_j)$; // least valued block.
3 $v^* \in \mathrm{argmax}_{v \in R} f(v|A_{j^*})$; // best for block.
4 $A_{j^*} := A_{j^*} \cup \{v^*\}$; // add to block.
5 $R = R \setminus \{v^*\}$
6 **end**
7 Sort A_j's by $f(A_j)$ so that $f(A_{j_1}) \geq f(A_{j_2}) \geq \ldots \geq f(A_{j_m})$
8 **return** $(A_{j_1}, A_{j_2}, \ldots, A_{j_m})$

- **Limitations of Algorithm 1:** 1) When using lazy greedy for the greedy step (line 3), the memory cost is proportional to $m|V| = |V|^2/k$. 2) if mini-batch size k is small, every mini-batch is not capable of representing the ground set V, we may get redundancies from combination of consecutive mini-batches.
- **Algorithm 2 – Run Algorithm 1 in hierarchy.**
- **Memory Efficiency and Group Representativeness:** The peak memory cost of Algorithm 2 is $\max_{i=1:r} m_i k_{i-1}$.

Even for $r = 2$, $k_1 = |V|/2$, the peak memory is halved. In addition, mini-batches grouped by the hierarchical structure are also representative.

- **Theoretical Guarantees:**

  Definition 1. We run Algorithm 1 with ground set size V', constraint size k' and $m' = |V'|/k'$, the greedy step gets executed $T = |V'|$ times, and we get a sequence of sets $Q = (A_1^T, A_2^T, \ldots, A_{m'}^T)$ as the output, with $A_{j'}^T$ having the minimal evaluation, i.e., $j' \in argmin_{i=1:m'} f(A_i^T)$. There exists an earliest greedy step $1 \leq t \leq T$ such that $|A_{j'}^T| = k'$, and $j' \in argmin_{1:m'} f(A_i^t)$, we define $\tau := \min_{i=1:m'} |A_i^t|$.

  Theorem 2. If we have $\tau \geq 2$ for every call to Algorithm 1 from Algorithm 2, then we achieve an approximation ratio of $\left(\frac{\tau-1}{2\tau-1}\right)^r \frac{k_r}{|V|}$.

Algorithm 2: Hierarchical Submodular Robust Partitioning

input : f, V, $k_1, \ldots, k_r$
1 $k_0 := |V|$; $Q_1 := (V)$; // Q_i's store sequence of sets to further partition
2 **for** $i := 1;\ i \leq r;\ i := i+1$ **do**
3 $m_i := k_{i-1}/k_i$; // m_i: number of blocks for the next partition
4 $Q_{i+1} = ()$; // Q_{i+1} initialized with an empty sequence
5 **for** $j := 1;\ j \leq |Q_i|;\ j := j+1$ **do**
6 $A_1, \ldots, A_{m_i} = \mathrm{RobustPartitionK}(f, Q_i[j], k_i)$; // $Q_i[j]$: jth set in the sequence
7 Append $A_1, \ldots., A_{m_i}$ to Q_{i+1} ; // Add partitioned blocks of $Q_i[j]$ to solution
8 **end**
9 **end**
10 **return** Q_{r+1}

Empirical Results

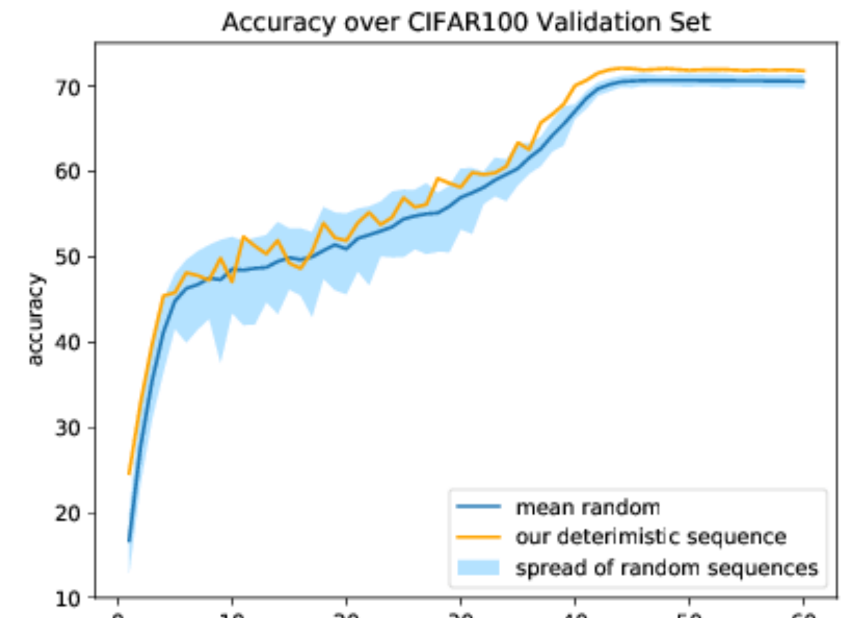
- **Choice of Submodular Function:** Nearest Neighbor Submodular Function (a special case of a facility location function):

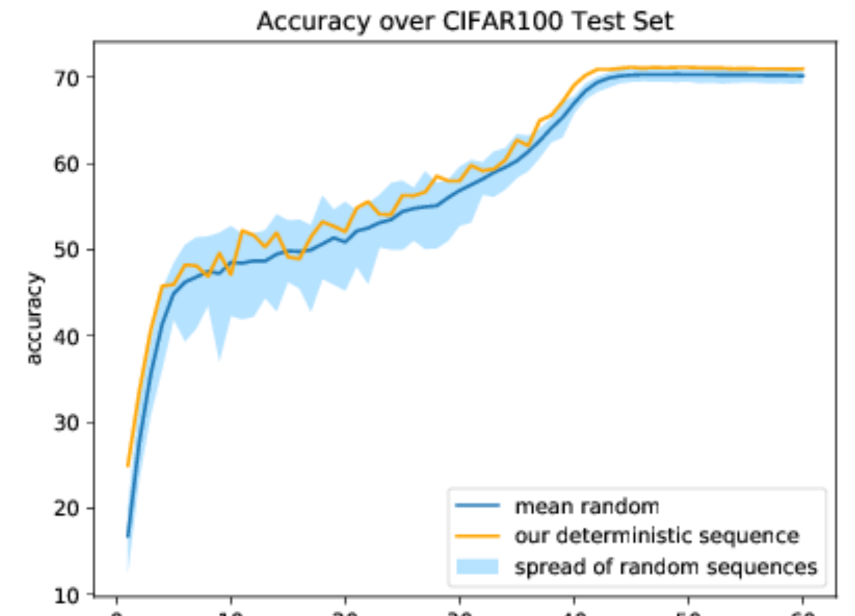
$$f_{NN}(S) = \sum_{v \in V} \max_{v' \in S} \mathrm{sim}(v, v'), \qquad \mathrm{sim}(v_1, v_2) = e^{\frac{-||x(v_1)-x(v_2)||_2}{\sigma}}.$$

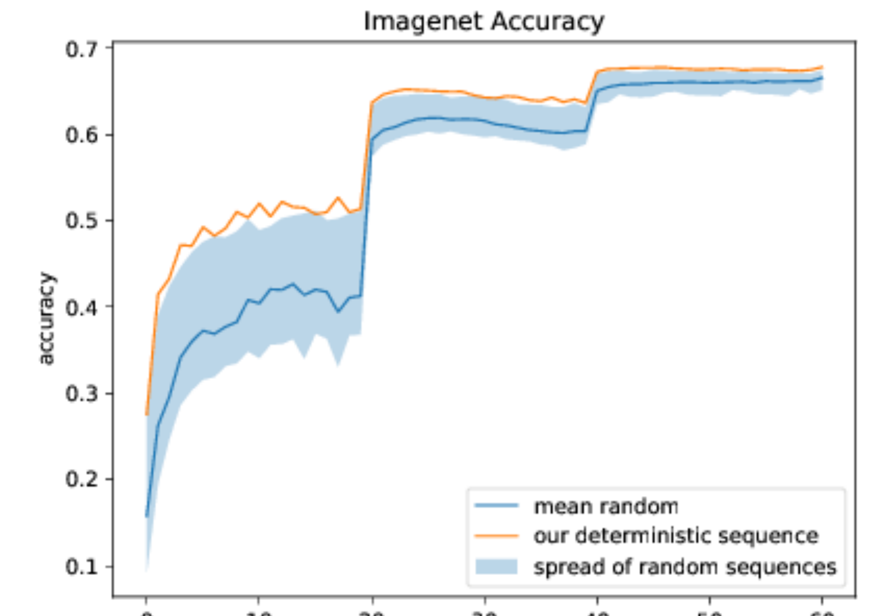
f_{NN} naturally captures the maximum likelihood estimates over the given data set for a nearest-neighbor classifier. As f_{NN} only requires similarities between data points within the same class, for large dataset such as Imagenet, we can afford to compute such sparse similarity graph.

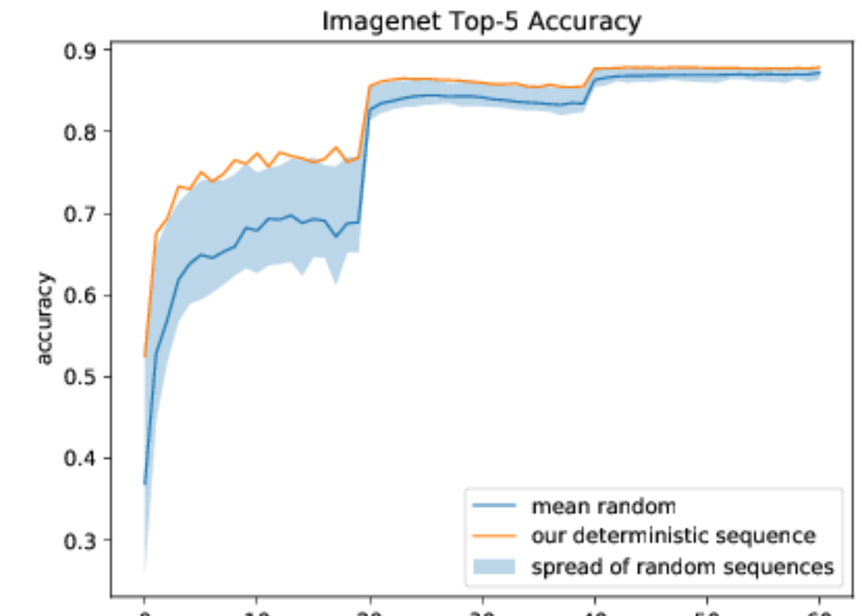
- **Experiments on CIFAR-100 and Imagenet:**

For CIFAR-100, we use the WRN-28-8 network, and for Imagenet, we use the Resnet-18. Compared to 30 randomly generated sequences on CIFAR-100, and the p-value for on test set accuracy is 0.0009. For Imagenet, we compare to 15 randomly generated sequences and the p-value for top-1 accuracy is 0.0151.









SRC JUMP

This work was supported in part by the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.