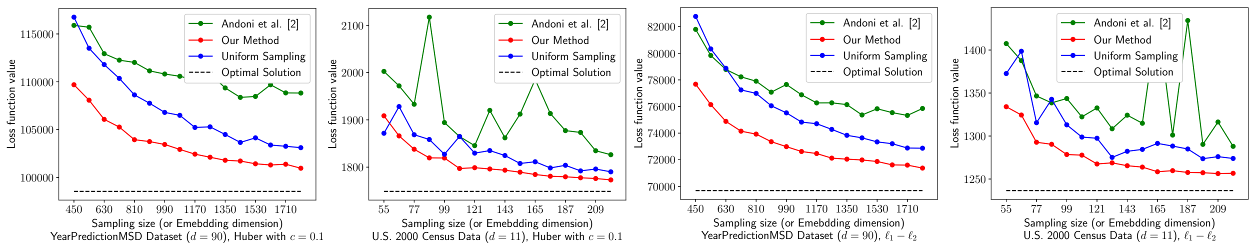
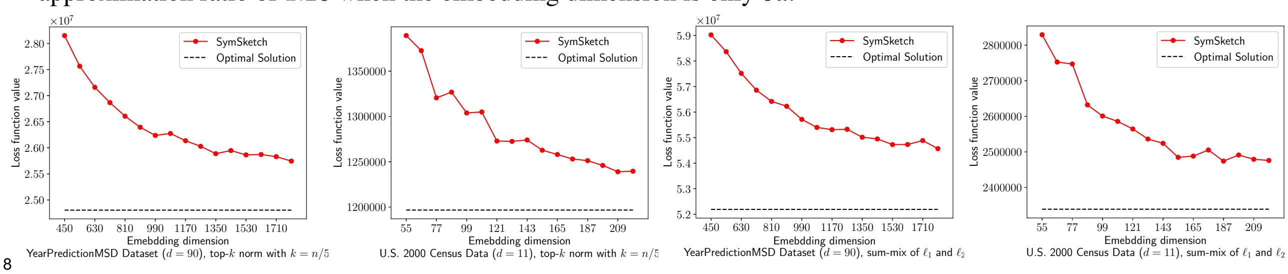
We thank all reviewers for their comments, and will incorporate suggestions in the final version. Although the goal of this paper is theoretical, we perform experiments to resolve reviewers' concern about practicality of our methods.

Experiment Setup. We compare the proposed algorithms with baseline algorithms on the U.S. 2000 Census Data containing $n = 5 \times 10^6$ rows and $d = 11$ columns and UCI YearPredictionMSD dataset which has $n = 515,345$ rows and $d = 90$ columns. All algorithms are implemented in Python 3.7. To solve the optimization problems induced by the regression problems and their sketched versions, we invoke the `minimize` function in `scipy.optimize`. Each experiment is repeated for *25 times*, and the mean of the loss function value is reported. In all experiments, we vary the sampling size or embedding dimension from $5d$ to $20d$, and observe their effects on the quality of approximation.

Experiments on Orlicz norm. We compare our algorithm in Section 2 with uniform sampling and the embedding in [2]. We also calculate the optimal solution to verify the approximation ratio. We try Orlicz norms induced by two different G functions: Huber with $c = 0.1$ and "$\ell_1 - \ell_2$". See Table 1 in our submission for definitions. Our experimental results given below clearly demonstrate the practicality of our algorithm. In both datasets, our algorithm outperforms both baseline algorithms by a significant margin, and achieves the best accuracy in almost all settings.



Experiments on symmetric norm. We compare our algorithm in Section 3 (SymSketch) with the optimal solution to verify the approximation ratio. We try two different symmetric norms: top-k norm with $k = n/5$ and sum-mix of ℓ_1 and ℓ_2 norm ($\|x\|_1 + \|x\|_2$). See Line 58-60 in our submission for definitions of these norms. As shown below, SymSketch achieves reasonable approximation ratios with moderate embedding dimension. In particular, the algorithm achieves an approximation ratio of 1.25 when the embedding dimension is only $5d$.



(Reviewer #1) Assumption 1. Our sampling algorithm in fact works for ℓ_p norms when $p > 2$. In general, suppose the function $G : \mathbb{R} \to \mathbb{R}$ satisfies that for all $0 < x < y$, $G(y)/G(x) \le C_G(y/x)^p$, for the Orlicz norm induced by G, given a well-conditioned basis with condition number κ_G, our sampling algorithm returns a matrix with roughly $O((\sqrt{d}\kappa_G)^p \cdot d/\varepsilon^2)$ rows such that Theorem 1 holds. However it is not clear how to calculate well-conditioned bases in input-sparsity time when $p > 2$. Our current method fails since it requires an oblivious subspace with $\mathrm{poly}(d)$ distortion, and it is known that such embedding does not exist when $p > 2$ [9]. Since we focus on input-sparsity time algorithms in this paper, we did not include the $p > 2$ case. We will add more discussion on this in the final version.

(Reviewer #2) Results on symmetric norm. We disagree that this is an incremental improvement. First, the previous embedding with $d \log n$ distortion only works for Orlicz norms, and in this paper we give the first subspace embedding for general symmetric norms. Second, the construction in [7] is only for streaming algorithms. To construct a subspace embedding, we need to show that (i) norms of all vectors in a subspace are preserved and (ii) there is a simple estimator in the sketch space. Neither of them can be satisfied by the construction in [7].

Comparison with [11]. First, our definitions for Orlicz norm leverage score and well-conditioned basis, as given in Definition 2 and 3, are different from all previous works and are closely related to the Orlicz norm under consideration. The algorithm in [11], on the other hand, simply uses ℓ_p leverage scores. Under our definition, we can prove that the sum of leverage scores is bounded by $O(C_G d \kappa_G^2)$ (Lemma 4), whose proof requires a novel probabilistic argument. In contrast, the upper bound on sum of leverage scores in [11] is $O(\sqrt{nd})$ (Lemma 38 in [11]). Thus, the algorithm in [11] runs in an iterative manner since in each round the algorithm can merely reduce the dimension from n to $O(\sqrt{nd})$, while our algorithm is one-shot. We will of course add a more detailed comparison with [11] in the final version.

(Reviewer #3) The uniqueness of α follows from the assumption that G is strictly increasing, in which case the two definitions are equivalent. The assumption that G is strictly increasing was also implicitly made in Andoni et al. [2].

It is indeed an interesting problem to generalize our techniques to other problems, e.g., classification problems and non-linear regression problems. We leave this as a future work.