Abstract

The LC Toolkit is an open-source library written in Python and PyTorch that allows to compress any neural network using several compressions including quantization, pruning, and low-rank. In this paper, we utilize the LC toolkit's common algorithmic base to take a deeper look into \( \ell_0 \)-constrained pruning problems defined as follows: given a budget of \( \kappa \) non-zero weights, which weights should be pruned in the final network?

We observe that \( \ell_0 \)-pruned networks have a different connectivity structure compared to pruning results using \( \ell_1 \) norm. Additionally, our \( \ell_0 \) formulation has the advantages of both methods: we can specify the amount of pruning precisely (unlike \( \ell_1 \) formulation), while experiencing a shrinkage effect (unlike \( \ell_0 \) formulation).

Reference