Lightweight Neural Networks from PCA & LDA Based Distilled Dense Neural Networks

ICIP 2020

MEA. Seddik[1,2,*], H. Essafi[1], A. Benzine[1,3], M. Tamaazousti[1]

[1]CEA List, France
[2]CentraleSupélec, L2S, France
[3]Sorbonne University, CNRS, France

[*]http://melaseddik.github.io/

August 21, 2020

Abstract

Context:

- **Compression** of dense neural networks with the teacher-student approach.

Motivation:

- Build **lightweight** neural networks that can **fit into** edge and IoT devices with **limited resources** (memory and computation).

Proposed methods:

- We proposed **two methods** which rely on **dimension reduction** techniques (PCA and LDA).
- The dimension reduction is applied at each layer of the teacher net and then mapped to the layers of the student net using a **multi-task loss** function.

Setting

Given a Teacher Network (TN) trained on a dataset $\mathcal{D}$ with loss $\mathcal{L}_{\textbf{TN}}$

$$\boxed{\textbf{(TN)} : \begin{cases} \boldsymbol{h}^{(0)} = \boldsymbol{x} \in \mathbb{R}^{p_0} \\ \boldsymbol{h}^{(\ell)} = f_\ell\left(\boldsymbol{W}^{(\ell)}\boldsymbol{h}^{(\ell-1)} + \boldsymbol{b}^{(\ell)}\right) \in \mathbb{R}^{p_\ell} \end{cases} \quad \forall \ell \in [L]}$$

Construct a Student Network (SN) to train on $\mathcal{D}$

$$\boxed{\textbf{(SN)} : \begin{cases} \tilde{\boldsymbol{h}}^{(0)} = \boldsymbol{x} \in \mathbb{R}^{p_0} \\ \tilde{\boldsymbol{h}}^{(\ell)} = f_\ell\left(\tilde{\boldsymbol{W}}^{(\ell)}\tilde{\boldsymbol{h}}^{(\ell-1)} + \tilde{\boldsymbol{b}}^{(\ell)}\right) \in \mathbb{R}^{k_\ell} \end{cases} \quad \forall \ell \in [L]}$$

Such that

$$\boxed{k_\ell \ll p_\ell \quad \& \quad \text{Performance } \textbf{(SN)} \gtrsim \text{Performance } \textbf{(TN)}}$$

Proposed Methods (Net-PCAD & Net-LDAD)

Given **(TN)**, a data matrix $\boldsymbol{X}$ and **(TN)** loss function $\mathcal{L}_{\mathsf{TN}}$

For each layer ℓ:

1. Extract the representations $\boldsymbol{H}_\ell$ of $\boldsymbol{X}$ from **(TN)**
2. Compute a projection matrix $\boldsymbol{U}_\ell \in \mathbb{R}^{p_\ell \times k_\ell}$ through PCA or LDA on $\boldsymbol{H}_\ell$

Train **(SN)** as a **multi-task**[1] problem with

$$\mathcal{L}_{\mathsf{SN}} = \underbrace{e^{-\sigma}\mathcal{L}_{\mathsf{TN}} + \sigma}_{\text{Learning Task}} + \underbrace{\sum_{\ell=1}^{L-1} e^{-\sigma_\ell}\mathcal{L}_{\mathrm{mse}}\left(\tilde{\boldsymbol{h}}^{(\ell)}, \boldsymbol{U}_\ell^{\mathsf{T}}\boldsymbol{h}^{(\ell)}\right) + \sigma_\ell}_{\text{(SN) Hidden Layers Task}}$$

where σ and $\{\sigma_\ell\}_{\ell=1}^{L-1}$ are learnable parameters.

[1] Using the Homoscedastic loss function: A. Kendall et al. "Multitask learning using uncertainty to weigh losses for scene geometry and semantics" in Proceedings of IEEE CVPR, 2018.

Experimental Setting & Results

Layer	(TN)	(SN)
Dense 1	$p_0 \times 1024$	$p_0 \times k$
Dense 2	1024×512	$k \times k$
Dense 3	512×256	$k \times k$
Dense 4	256×10	$k \times 10$

Table: Networks architectures.

		(SN)		
Datasets	(TN)	$k = 50$	100	200
MNIST	2.23s 98%	0.38s 97%	0.45s 97.5%	0.65s 97.8%
FASHION	2.23s 88%	0.38s 87.5%	0.45s 88.5%	0.65s 88.5%
CIFAR10	4.63s 45%	0.75s 50%	0.92s 50.1%	1.35s 50.3%

Table: Networks performances.

$\Rightarrow$ $\boxed{k_\ell \ll p_\ell \quad \& \quad \text{Performance } \mathbf{(SN)} \gtrsim \text{Performance } \mathbf{(TN)}}$