METHODICAL DESIGN AND TRIMMING OF DEEP LEARNING NETWORKS: ENHANCING EXTERNAL BP LEARNING WITH INTERNAL OMNIPRESENT-SUPERVISION TRAINING PARADIGM

S.Y.Kung Zejiang Hou Yuchen Liu

Princeton University

ABSTRACT

Back-propagation (BP) is now a classic learning paradigm whose source of supervision is exclusively from the external (input/output) nodes. Consequently, BP is easily vulnerable to curse-of-depth in (very) Deep Learning Networks (DLNs). This prompts us to advocate Internal Neuron's Learnablility (INL) with (1)*internal teacher labels* (ITL); and (2)*internal optimization metrics* (IOM) for evaluating hidden layers/nodes. Conceptually, INL is a step beyond the notion of Internal Neuron's Explainablility (INE), championed by DARPA's XAI (or AI3.0). Practically, INL facilitates a structure/parameter NP-iterative learning for (supervised) deep compression/quantization: simultaneously trimming hidden nodes and raising accuracy. Pursuant to our simulations, the NP-iteration appears to outperform several prominent pruning methods in the literature.

Index Terms— Internal Learning, Internal Optimization Metrics (IOM), structural-parameter learning, BPOS NP-iteratom, (supervised) deep compression/quantization.

1. INTRODUCTION

By viewing the DLN as a black box, cf. Figures 1(a,b), BP is an external learning paradigm. Traditionally, BP is only used for parameter learning of DLNs, leaving the task of finding optimal structure to trial and error. To rectify this, there are recently novel approaches to structural tuning via external regulation, e.g. Louizos et al. [1] and Zhuang et al. [2].

2. SUPERVISED NET LEARNING VIA INTERNAL OPTIMIZATION METRICS (IOM)

BP learning on deep nets may suffer from vanishing/exploding gradients of an *external optimization metric*(EOM) [3]. This in turn induces the curse of depth problem [3–5]. To mitigate this problem, we propose an internal learning paradigm, allowing the hidden nodes to be directly trained. This requires that teacher labels to be sent to all the hidden nodes, just like the "Trojan-horses". This leads to an Omnipresent Supervision(OS) internal learning strategy, where *internal teacher labels* are ubiquitously accessible to all hidden nodes, as exemplified by Figures 1(a,c).

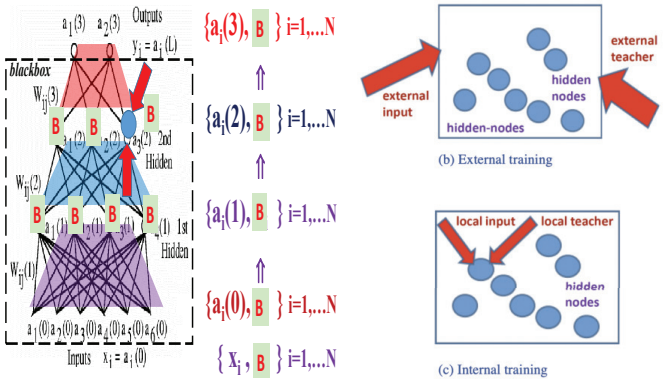


Fig. 1. For classification problem, the teacher labels can be metaphorically hidden in "Trojan-horses" and transported (along with the data) from the input layer to all hidden nodes. (a) The original label, say B, is being sent to all hidden nodes; (b) External teacher only; (c) Possible *internal teacher labels* ITLs are: granularity-adaptive (class or super-class), layer-adaptive, or end-user-adaptive to facilitate INE in XAI.

Internal learning applies only to classification problems [6], where the training dataset consists of a set of pairs denoted as $[\mathcal{X}, \mathcal{Y}] = [\mathbf{x}_1, y_1], [\mathbf{x}_2, y_2], \ldots, [\mathbf{x}_N, y_N]$ where a teacher value, y_t, is assigned to each training vector $\mathbf{x}_t$, for $t = 1, \cdots, N$. Let L denote the number of different classes with N_ℓ denoting the number of training vectors associated with the l-th class, $l = 1, \cdots, L$, and teacher values are discrete labels, i.e. $y_t \in$ **class labels**.

Denote the "center-adjusted" *scatter matrix* as $\bar{\mathbf{S}} \equiv \bar{\mathbf{X}}\bar{\mathbf{X}}^T$, which can be divided into two parts [7]:

$$\bar{\mathbf{S}} = \mathbf{S}_B + \mathbf{S}_W \tag{1}$$

where within-class/between-class scatter matrix $\mathbf{S}_W$/$\mathbf{S}_B$ are:

$$\mathbf{S}_W = \sum_{\ell=1}^{L} \sum_{j=1}^{N_\ell} [\mathbf{x}_j^{(\ell)} - \vec{\boldsymbol{\mu}}_\ell][(\mathbf{x}_j^{(\ell)} - \vec{\boldsymbol{\mu}}_\ell]^T \tag{2}$$

$$\mathbf{S}_B = \sum_{\ell=1}^{L} N_\ell \, [\vec{\boldsymbol{\mu}}_\ell - \vec{\boldsymbol{\mu}}][\vec{\boldsymbol{\mu}}_\ell - \vec{\boldsymbol{\mu}}]^T \equiv \boldsymbol{\Delta}\boldsymbol{\Delta}^T \tag{3}$$

where $\boldsymbol{\Delta} \in \Re^{M \times L}$ represents the **centroids matrix**.

IOM: Discriminant Information (DI)

Internal learning requires a clear definition of IOM, i.e. an internal metric to facilitate the local learning/optimization process in each hidden node/layer. To this end, we note that maximizing SNR involves a ratio between $\mathbf{S}_B$ versus $\mathbf{S}_W$ which, thanks to Eq. [1], is equivalent to maximizing the ratio between $\mathbf{S}_B$ versus $\bar{\mathbf{S}}$. The latter leads to a new metric named **Discriminant Information** (DI), stemming from a combination of Fisher's *discriminant analysis* and Shannon's *mutual information* [7–10]. For assessing the goodness of space/subspace/node of any hidden layer, we propose:

$$\mathrm{DI}(\mathbf{W}) = \mathtt{tr}\left([\mathbf{W}^T\bar{\mathbf{S}}\mathbf{W} + \rho\mathbf{I}]^{-1}[\mathbf{W}^T\mathbf{S}_B\mathbf{W}]\right) \tag{4}$$

(The **ridge** ρ is meant to safeguard numerical inversion of $\bar{\mathbf{S}}$.) To assess the IOM of the full space of a layer, we set $\mathbf{W} = \mathbf{I}$:

$$\mathrm{DI} = \mathrm{DI}(\mathbf{I}) = \mathtt{tr}\left([\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}\mathbf{S}_B\right) \tag{5}$$

For (supervised) deep compression, we make a good use of:

$$\mathbf{W}_{i_{keep}} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ 0 & \ddots & \cdots & \cdots & 0 \\ \vdots & \cdots & 1 & \cdots & \vdots \\ 0 & \cdots & \cdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix} \quad \mathbf{W}_{i_{drop}} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & \ddots & \cdots & \cdots & 0 \\ \vdots & \cdots & 0 & \cdots & \vdots \\ 0 & \cdots & \cdots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix}$$

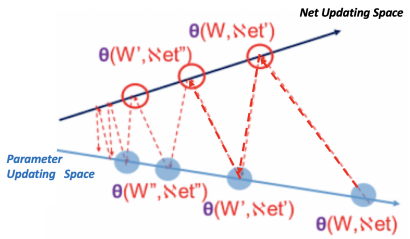
where $\mathbf{W}_{i_{keep}}$/$\mathbf{W}_{i_{drop}}$ keeps/drops only the i-th node/channel. For pruning nodes/channels in **MLP/ConvNet**, we adopt:

- **Fisher Discriminant Ratio (FDR):** FDR $= DI(\mathbf{W}_{i_{keep}})$ is the value of the i-th node/channel.
- **Dispensability of a node/channel: DI-Loss:** DILoss $\equiv \mathrm{DI}(\mathbf{I}) - \mathrm{DI}(\mathbf{W}_{i_{drop}})$ is the remaining value of the layer after removing the i-th node/channel. This reflects the **dispensability** of the i-th node/channel.

Remark: For pruning channels in **ConvNet**, we use a similar DI-metric except that $\mathbf{W}_{i_{keep}}$/ $\mathbf{W}_{i_{drop}}$ must be first converted into a block-matrix form: $\mathbf{W}_{i_{keep}} \otimes I$ / $\mathbf{W}_{i_{drop}} \otimes I$.

3. NP ITERATIVE NET PRUNING METHODS

We propose an (EM-style) NP iterative learning algorithm, where N stands for Net and P for Parameter, pictorially illustrated below:



3.1. Unsupervised NP-Iterative Pruning of Links

Lacking a (supervised) IOM, we have to do with an unsupervised selection metric in implementing the N-phase of NP-iteration. Intuitively, links (weights) with lowest-magnitude are good candidates to be removed. This is the approach adopted by Han et al. and Iandola et al. [11, 12].

3.2. BPOS Iteration: Seamless Integration of EOM/IOM

Our supervised BPOS NP-method involves both (1) BP parameter-learning (based on EOM) and (2) OS net-learning (based on IOM), so we must strive for an acceptable consistency between EOM and IOM. Assuming Gaussian distribution and one-hot-encoding of the teacher values, LSE and DI metrics are essentially equivalent under the balanced scenarios(i.e. equal size per class) [10, 13].

For the unbalanced scenarios(i.e. with unequal sizes), the same equivalence can be preserved by adopting a renormalized-one-hot-encoding, where the teacher values will replaced by $\sqrt{N_i}^{-1}$ instead of "1". Such encoding assures consistency between EOM and IOM, fostering OStrim's BPOS learning, where the external BP learning [14–16] and the internal OS learning [17] are both employed.

3.3. OStrim: Supervised NP-Iterative Pruning

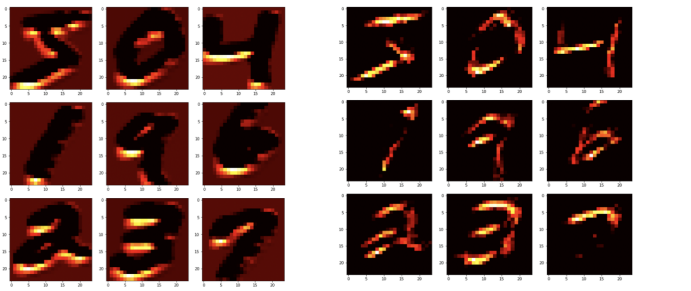
As shown in Algorithm 1, OStrim trims unwanted *nodes*, instead of *links*, by iteratively repeating : (i) **Net Updating** to remove the low-score hidden neurons based on the supervised IOM and (ii) **Parameter Updating** via external BP learning. (The iterations continue until the accuracy improvement saturates or starts to downgrade.)

Algorithm 1: NP Iterative Pruning Method

Input : Original Network Net, Pruning Ratio α
Output: Pruned Network Net_p
Out-source or BP-train a base-net Net, let $Net_p \leftarrow Net$
while $Accuracy \geq Threshold$ **do**
  Net Updating: Based on the IOM score, e.g. FDR or DILoss, drop a small fraction of lowest-scored nodes/channels.
  Parameter Updating: Based on the EOM score, apply BP to externally train Net' into Net'', let $Net_p \leftarrow Net''$
end
return Net_p

3.4. OStrim is Conducive to Implicit Regularization

To highlight the critical roles played by DI, the following figure shows (a) (selected) maximum DI channel (b) (dropped) minimum DI channel of the 1st layer in LeNet-5. By naked eyes, the former clearly yields robustified patterns of the MNIST 0-9 digits. Note that low DI nodes tend to represent overtraining nodes; consequently, drop low DI nodes/channels may actually enhance generalization performance. In short, DI is a promising metric for structural training.



(a) maximum DI (0.625) channel (b) minimum DI (0.501) channel

Table 1. Comparison with compression benchmarks

Task	Models	Accuracy %	FLOPs	Params
CIFAR100	*Mobilenet-v2 (baseline)*	73.68	1.8×10^8	2.4×10^6
	Mobilenet-v2 (DILoss)	**75.61**	$\mathbf{7.57\times10^7(42.06\%)}$	$\mathbf{1.07\times10^6(44.58\%)}$
CIFAR-10	*VGG-16 (baseline)*	93.25	6.26×10^8	1.5×10^7
	VGG-16 (Li et al., 2017)	93.41	4.12×10^8(65.81%)	5.4×10^6(36%)
	VGG-16 (FDR)	93.61	1.35×10^8(21.4%)	7.1×10^5(4.7%)
	VGG-16 (DILoss)	**94.07**	$\mathbf{1.28\times10^8(20.45\%)}$	$\mathbf{5.32\times10^5(3.55\%)}$
	ResNet-56 (baseline)	93.04	2.5×10^8	8.5×10^5
	ResNet-56 (Li et al., 2017)	93.06	1.81×10^8(72.4%)	7.3×10^5(85.88%)
	ResNet-56 (Yu et al., 2018)	93.01	1.41×10^8(56.4%)	4.94×10^5(58.12%)
	ResNet-56 (He et al., 2017)	91.9	1.25×10^8(50%)	-
	ResNet-56 (Zhuang et al., 2018)	93.49	1.25×10^8(50.25%)	4.3×10^5(50.76%)
	ResNet-56 (DILoss)	**93.84**	$\mathbf{8.38\times10^7(33.52\%)}$	$\mathbf{3.12\times10^5(35.52\%)}$
	ResNet-56 (bootstrap:DILoss+Zhuang)	**93.84**	$\mathbf{7.58\times10^7(30.32\%)}$	$\mathbf{2.81\times10^5(33.05\%)}$
MNIST	*Lenet-5 (baseline)*	99.2	4.59×10^6	4.3×10^5
	Lenet-5 (Han et al., 2015)	99.23	8.3×10^5(18.1%)	3.6×10^4(8.4%)
	Lenet-5 (Louzois et al., 2018)	99	7.85×10^5(17.1%)	1.22×10^4(2.83%)
	Lenet-5 (FDR)	99.33	2.6×10^5(5.74%)	4.9×10^3(1.1%)
	Lenet-5 (DILoss)	**99.35**	$\mathbf{2.46\times10^5(5.36\%)}$	$\mathbf{3.86\times10^3(0.89\%)}$
	Lenet-300 (baseline)	98.36	-	2.7×10^5
	Lenet-300 (Han et al., 2015)	98.41	-	2.24×10^4(8.3%)
	Lenet-300 (Louzois et al., 2018)	98.2	-	2.7×10^4(10%)
	Lenet-300-100 (FDR)	98.42	-	2.3×10^4(8.5%)
	Lenet-300 (DILoss)	**98.46**	-	$\mathbf{1.63\times10^4(6.04\%)}$

4. SUPERVISED DEEP COMPRESSION: EXPERIMENTAL RESULTS

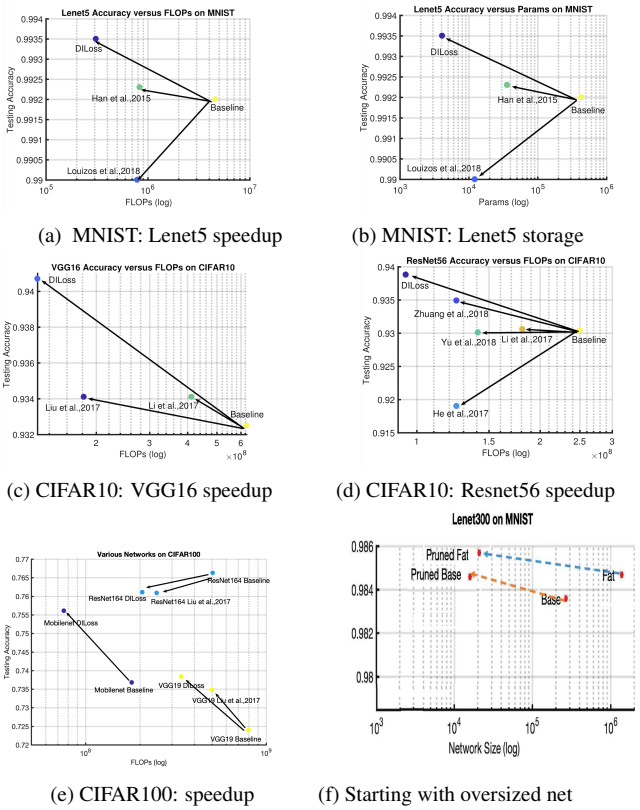
We conduct experiments on MNIST [18]/CIFAR-10 [19] datasets consisting of images of 10-class and 60,000/50,000 training samples, with 10,000 testing samples. In our study, the (supervised) deep compression seems to outperform all other pruning methods, as clearly evidenced by Table 1 and Fig. 2. (All our results, including final compressed models, have been uploaded to our OStrim-github-site [20].)

MINIST Dataset: We gain around 20x in speedup on Lenet-300. On Lenet-5, we achieve 20x in speedup and 100x in storage reduction, cf. Fig.2(a-b).

CIFAR-10 Dataset: DILoss-based OStrim is effective on ResNet56, VGG16, and Lenet using CIFAR100 and CIFAR10 datasets. We improve the accuracy by around 0.8% with nearly 5x in speedup on VGG16 as compared with the baseline, cf. Fig.2(c). Even for the more compact and advanced structure like ResNet56, OStrim yields an accuracy improvement of 0.8% with 3x in speedup compared with the baseline, cf. Fig.2(d).

CIFAR-100 Dataset: Fig. 2(e): On Mobilenet, a reduction of around 2.5x in speedup is observed while improving accuracy by from 73.68% to 75.61%. On ResNet164, we gain 2.5x in speedup. Finally, on VGG19, OStrim achieve around 2.5x in speedup while improving the accuracy by from 72.4% to 73.84%.

Start OStrim with Oversized Nets: Note that the update learning tends to converge rapidly due to the good initial condition we use for fine tuning the net. To verify the vital importance of initial condition, we purposely retrain the optimally-reduced LeNet-5, but using randomized initial condition. Not surprisingly, it yields an accuracy of 97.28%, far poorer than 99.35% by starting OStrim-DILoss with the base. It has been reported by some optimization theoreticians that a somewhat oversized (fat) network may bring about desirable numerical convergence [21]. As evidenced by Fig.2(f)



(a) MNIST: Lenet5 speedup (b) MNIST: Lenet5 storage

(c) CIFAR10: VGG16 speedup (d) CIFAR10: Resnet56 speedup

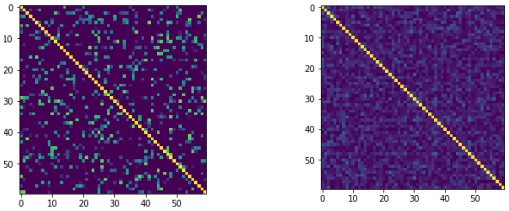
(e) CIFAR100: speedup (f) Starting with oversized net

Fig. 2. The broad range of reduction (100x) implies a very comprehensive coverage of NP structural learning, i.e. covering a lot of potential structural options.

starting the NP iteration with a fat DLN, we have a higher design and flexibility with a broader range of size-performance tradeoff/optimization.

Bootstrapping: Note that DILoss NP iteration and Zhuang's method [2] are both based on discriminant analysis. Since they are complementary, it is natural to further improve performance by bootstrapping each other. As shown in Table.1, we can further reduce both storages and FLOPs somewhat by applying several rounds of bootstrapping.

Advantage of DILoss in Redundancy Mitigation: On MNIST Lenet300 experiment, we note that OStrim-DILoss outperforms OStrim-FDR. This may be attributed to the fact that DI can better account for inter-feature redundancy than FDR. Indeed, the nodes selected via DI tend to be less correlated than those selected by FDR, as evidenced by the following Pearson-correlation analysis [22]:



(a) OStrim-FDR (b) OStrim-DI

Savings on Input Features: Note that OStrim may also be viewed as a tool for input feature reduction. It can reduce Lenet300's (MLP{784-300-100-10}) into $MLP\{314-60-20-10\}$ while improving accuracy by .1%, cf. Table 1. It means that merely 40% of informative features selected by OStrim-DILoss can well preserve the useful information in the original MNIST images. From the discriminant analysis' perspective, such feature reduction represents a kind of lossless compression. For certain sensor array applications, it may be applied to save hardware/human costs unnecessarily wasted on raw-data acquisitions.

5. ALGORITHMS/ARCHITECTURE CO-DESIGN

The hardware costs estimated/reported above are based on the following analysis. Note that a full characterization of LeNet-5 (containing CNN and MLP layers) is:
$\{28^2 \rightarrow [f_1\ n_1\ m_1\ \mu_1] \rightarrow [f_2\ n_2\ m_2\ \mu_2] - n_3 - n_4 - n_5\}$
where we denote $f_i \times f_i$ as the filter kernel size of the i-th CNN layer; n_i as the channel number. Here, WLOG, we shall fix (1) the ratio of the sizes of feature-maps before/after sub-sampling to $(m_i \times m_i)/(\mu_i \times \mu_i) = 4$ and (2) $m_i = \mu_{i-1} - f_i + 1$ since we retain only the completely filtered portion after convolution. This leads to a simplified characterization:

$$\{[f_1\ n_1] \rightarrow [f_2\ n_2] - n_3 - n_4 - n_5\}$$

For LeNet-5, we have: $\{[5\ 20] \rightarrow [5\ 50] - 800 - 500 - 10\ \}$. Denote the number of parameters and FLOPs in layer i as p_i and $FLOP_i$, then the total storage P and total FLOPs are:

$$P = \sum_{i=1}^{L} p_i \qquad FLOP = \sum_{i=1}^{L} FLOP_i \tag{6}$$

where (1) for MLP layers:

$$p_i = n_{i-1} \times n_i \qquad FLOP_i = 2 \times n_{i-1} \times n_i$$

and (2) for ConvNet layers:

$$p_i = n_{i-1} \times n_i \times f_i^2 \qquad FLOP_i = 2 \times n_{i-1} \times n_i \times f_i^2 \times m_i^2$$

The above formula may be useful for hardware optimization strategy: It can be analytically be shown that, relatively speaking, we can achieve more FLOP saving by trimming ConvNet and more parameter saving by reducing MLP.

Parallel Architectures: Note that hardware implementation for dense networks (as obtained by OStrim) is simpler than sparse network (as in Han's). First, dense nets may enjoy the speed/power advantages by cleverly harnessing caches as memory units [23]. In addition, dense nets can fully take advantage of their smooth dataflow to facilitate parallel/pipelined processing [16].

6. DI-BASED DEEP QUANTIZATION

When applications shift from high accuracy to low power, we may want to further quantize hidden nodes and net on the OS-trimmed net. We applied 16-bit fixed point OStrim to Lenet-300-100 on MNIST dataset, and obtained the same accuracy of 98.42%, cf. OStrim-github-site [20].

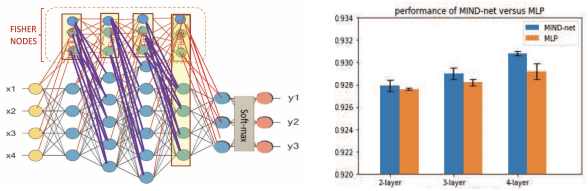


Fig. 3. (a) The configuration of MINDnet. (b) Accuracies.

- By cherry-picking a small fraction of the highest FDR nodes, we can achieve 100x saving in storage from baseline while yielding an accuracy of 94.42%
- By reducing 2/3 of the weaker nodes from 16-bit to 8-bit, we yield 150x saving with accuracy = 94.22%

Note that, in the worst case, the quantization-error-variance is amplified by the spectrum norm of the weight matrix whenever it traverses across any layer. We conclude that quantization on lower layers tends to induce greater side-effect. To verify this, we compare three reduced nets (all with 100x acceleration): 190-13-10-10; 110-20-16-10; and 75-26-20-10 and observe the accuracies dropping from 94.42%, 93.21%, to 89.85%, pursuant to what theoretically predicted.

7. DI-BASED NETWORK GROWING: MINDNET

In additional to structural learning, DI-based IOM may also be utilized to train a DI-boosting structure. Figure 3(a) shows a general configuration of MINDnet (Monotonic INcreasing Discriminant Network).

7.1. Fisher Weights for Inheritance Nodes

MINDnet relies on the inheriting Fisher nodes, (shown as the upper nodes in Figure 3), $\mathbf{W} = \bar{\mathbf{S}}^{-1}\mathbf{\Delta}$, derived by the OS training strategy. (The full-space DI can be retained by the Fisher projection matrix $\mathbf{W}$ *iff* it fully spans the $(L-1)$-dimensional subspace of $\mathbf{S_B}$.) Note that the discriminant information of any hidden layer can be fully transferred to the next layer via a small number ($\leq L$) of Fisher-nodes. By further augmenting the Fisher nodes with the traditional hidden-nodes (shown as the lower-nodes in Figure 3), it guarantees that the DI will monotonically increase from layer to layer.

7.2. MINDnets

MINDnet may be constructed layer-by-layer (Sequential MINDnet) or simultaneously (Parallel MINDnet). MINDnet has its own BPOS learning, involving a recursive loop with two updating sub-steps:

(i) For all layers, compute the fisher nodes of the next layer : $\bar{\mathbf{S}}^{-1}\mathbf{\Delta}$ (cf. Eqs. [1 - 3]).

(ii) Apply the external BP learning across all hidden layers.

In Figure 3(b), we compare sequential MINDnet with MLP for CIFAR-10 using 64-dimensional input vectors pre-extracted by ResNet56. The accuracy improves as we sequentially grow the MINDnet layer by layer. Note that MINDnet gives a slightly better prediction accuracy (93.1%) than the original ResNet56 (92.78%). Using multi-kernel Fisher nodes [24], we can further arrive at an accuracy of 93.28%.

Acknowledgement: We thank Mert Al, W.Y. Chang, Drs. Y. Li and T. Chanyaswad for their invaluable contributions.

Dedication: In memory of Dr. Jan Larsen 1965-2018.

8. REFERENCES

[1] Christos Louizos, Max Welling, and Diederik P Kingma, "Learning sparse neural networks through l_0 regularization," *arXiv preprint arXiv:1712.01312*, 2017.

[2] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu, "Discrimination-aware channel pruning for deep neural networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 883–894.

[3] Sepp Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, 1998.

[4] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010.

[5] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber, "Training very deep networks," in *Advances in neural information processing systems*, 2015.

[6] Sotiris B Kotsiantis, "Supervised machine learning: A review of classification techniques," 2007.

[7] Ronald A Fisher, "The statistical utilization of multiple measurements," *Annals of eugenics*, vol. 8, no. 4, 1938.

[8] Claude Elwood Shannon, "A mathematical theory of communication," *Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[9] S.Y. Kung, *Kernel methods and machine learning*, Cambridge University Press, 2014.

[10] S.Y. Kung, "Compressive privacy: From information/estimation theory to machine learning [lecture notes]," *IEEE Signal Processing Magazine*, vol. 34, no. 1, pp. 94–112, 2017.

[11] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, 2015.

[12] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[13] Andrew R Webb and David Lowe, "The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis," *Neural Networks*, vol. 3, no. 4, 1990.

[14] Paul Werbos, "Beyond regression:" new tools for prediction and analysis in the behavioral sciences," *Ph. D. dissertation, Harvard University*, 1974.

[15] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning internal representations by error propagation," Tech. Rep., California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[16] S.Y. Kung, *Digital neural networks*, PTR Prentice Hall Englewood Cliffs, 1993.

[17] S.Y. Kung, "Systematic (analytical and empirical) optimization/generalization of deep learning network," in *Short Course in DeepLearn*, Genova, 2018.

[18] Yann LeCun, Corinna Cortes, and CJ Burges, "Mnist handwritten digit database," *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010.

[19] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton, "The cifar-10 dataset," *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014.

[20] "https://github.com/zejiangh/ostrim-np-iterative-deep-compression," .

[21] Jason Lee, "private communication," 2018.

[22] Karl Pearson, "Note on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, 1895.

[23] S.Y. Kung, *VLSI Array Processors*, Prentice Hall Information and System Science Series, 1988.

[24] Thee Chanyaswad, J Morris Chang, and S.Y. Kung, "A compressive multi-kernel method for privacy-preserving machine learning," in *Neural Networks (IJCNN), 2017 International Joint Conference on*. IEEE, 2017, pp. 4079–4086.