Communications in Information and Systems
Volume 20, Number 3, 319–352, 2020

Augment deep BP-parameter learning with local XAI-structural learning

S. Y. Kung and Zejiang Hou

With the explosion of big data, Deep Learning has become the main stream of the machine learning and AI research and development. However, its back-propagation learning paradigm relies on the traditional optimization of externally-supervised metrics, limiting its option to improve the network design structurally. To rectify this problem, we augment the BP-Learning with a structural learning paradigm: XAI-Learning, abbreviated as **X-Learning**. In order to come up with high-performing learning models with lower structural complexity, X-Learning places its focus on local learning/ranking of individual neurons in hidden layers. It pioneers the use of backward-broadcast so that the teacher values become directly and locally accessible to all hidden layers, making feasible the so-called output-residual learning. This is conceptually dual to the input-residual learning, advocated by ResNet. The local teacher permits the computation of local optimization metrics (LOM) to facilitate the ranking of hidden neurons. Such a ranking provides a theoretical footing of our structural learning paradigm, based on a notion of structural gradient. This ultimately leads to an evolutionary X-Learning strategy to jointly learn the structure and parameters of the learning models. The purpose is to reduce the network complexity while preserving (if not outright improving) its accuracy performance.

X-Learning can be applied to numerous applications, with either classification or regression formulation. Moreover, it outperforms prominent state-of-the-art approaches. To highlight its superiority, we shall showcase three key comparisons: (a) for ImageNet classification, X-Learning edges the 2018 LPIRC winner; (b) for regression, X-Learning outperforms the 2018 PIRM winner in image super-resolution; and (c) for finger-printing, our hierarchical HSRN CNN outperforms SRGAN by 1.5 dB in PNSR. In addition, to demonstrate its broad spectrum of applications, we present more examples in classification (e.g. ImageNet), in regression (e.g. DIV2K), and in (classification/regression) mixed domain-driven problems (e.g. Oxford Flower Dataset).

1. Introduction

In the beginning, the prevailing AI inferencing approach was built upon the rule-based expert systems, i.e. AI1.0. It relied on the collective domain-knowledge from all the domain-experts. However, inadequacy of rule-based inferencing systems was quickly and widely recognized. In the 80's, researchers in neural networks have shun away from expert systems and moved wisely towards the data-driven machine learning paradigm. Neural networks have undergone two stages of technological evolution:

- **Multi-Layer Perceptron (MLP):** On the neural network fronts, the focus of mainstream research study has been placed on supervised-learning networks. Most methods follow an external learning paradigm, in which the whole neural network can be viewed as a black-box, characterized by its internal structure/parameters. Traditionally, the goal of machine learning is to probe into the black-box by means of some (externally supervised) optimization metrics.
  In the first generation, pioneering supervised learning networks include the perceptron and multi-layer perceptron [30, 26, 23, 25], among many others. The multi-layer perceptron (MLP) is traditionally trained by the Back-Propagation (BP) learning algorithm. In fact, for NN1.0, BP is exclusively adopted for the task of *label engineering*. The vital task of *feature engineering* depends fully on the domain experts on, say, speech, image etc. Namely, machine learning plays no role in the process of *feature extraction*. This is to a good degree the reason why MLP and NN1.0 failed to come up with major commercial breakthroughs.
- **Deep Learning Networks:** In the second generation, the network structure is being evolved in to a relatively more complex *Convolutional Neural Network* (CNN), comprising of ConvNet layers and MLP layers. Le Cun et al. [15] proposed to learn an end-to-end mapping throughout the CNN between input and desired response. Just like MLP, the training of CNN is again based on the Back-Propagation (BP) algorithm. It is worth stressing that CNN can effectively cope with both the tasks of *feature engineering* (performed by ConvNet) and *label engineering* (handled by the MLP). Such a configuration of the total system is depicted in Figure 1. Note that CNN training process is accomplished by back-propagating *learning gradients* from the last layer back to the first input layer.
  Since the headline event in 2016 that a top professional GO player lost to the learning-based AlphaGo, *Deep Learning* has been broadly

recognized as the state-of-the-arts machine learning technology and de facto synonymous to AI2.0. It has successfully (and effortlessly) substituted many traditional approaches to many signal/speech/image applications, and, moreover, catalyzed numerous new business adventures. It has since successfully replaced many traditional information processing tools, and, moreover, catalyzed numerous new business adventures. Nowadays, deep learning has rapidly gained a lot of attention from the AI research community and has been tremendously successful in AI industries.

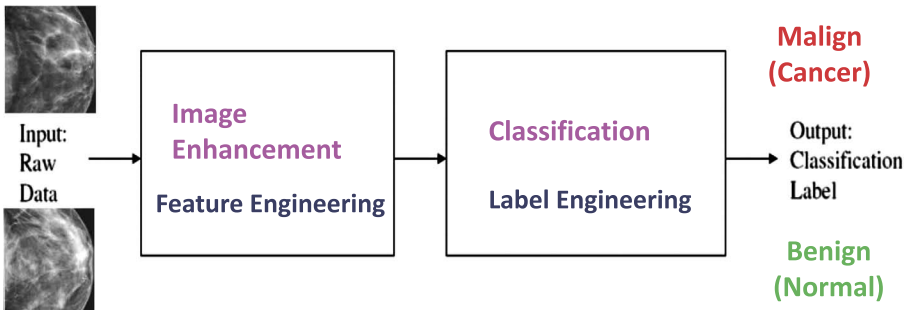


Figure 1: A total system comprise of two sub-systems: feature engineering and label engineering, respectively. The same configuration applies also to domain-driven imaging systems, discussed in Section 8.2.

2. Input/output residual learning to allay curse-of-depth

BP has been the most prevailing algorithm for parameter learning in neural networks. However, it involves propagating the error signal (the gradient) across too many layers. Consequently, it suffers from the curse of depth in deep learning which is gravely detrimental to the learning process. On the other hand, a new research front, named Explainable AI (XAI), has recently received a great deal of attention. It advocates exploring Explainability by fully harnessing the rich information embedded in the (internal) hidden neurons. XAI-Learning precipitates a local learning paradigm which is the major theme of our discussion. Furthermore, local learning is conceptually coupled with a notion of residual learning which focuses the (future) learning endeavour on what is being missed in the previous learning effort. Two types of residual learning can be exploited:

- *Input-Residual Learning.*
- *Output-Residual Learning.*

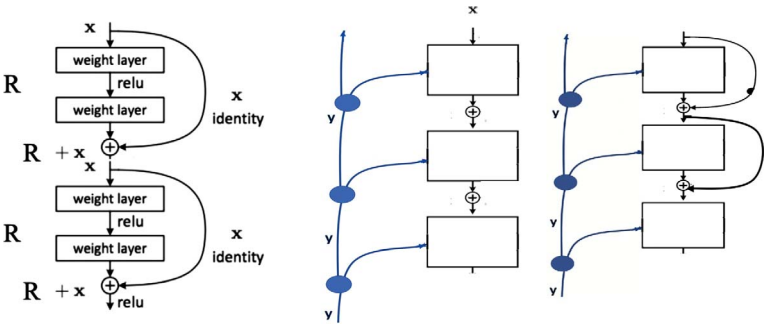


Figure 2: Three types of skips to help mitigate the curse of depth in deep learning: (a) forward skips (b) backward broadcast (BB) (c) dual skips.

2.1. Forward skips and input residual learning

He et al. [7] "reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions." This leads to an (input) "residual learning framework to ease the training of networks that are substantially deeper." With reference to the addition operation $\oplus$, shown in Figure 2(a), the input residual is defined as "input to the next layer minus the highway-skipped input of the previous layer". With the forward skips making the input of the previous layer directly accessible as a reference for the input of next layer, the residue is just the output of the preceding layer. It leads to "local residual learning" paradigm because the residual learning responsibility falls mainly on the preceding layer. In fact, He et al. [7] showed that "these residual networks are easier to optimize, and can gain accuracy from considerably increased depth." Note that the forward skips are required to be installed for both training and testing phases.

2.2. Backward broadcast (BB) and output residual learning

As a dual notion, from output perspective, residues may also be created with respect the teachers as opposed to the input. This leads to an output residual learning framework. With reference to Figure 2(b), the backward s broadcast (BB) makes the teacher directly accessible to every hidden layer. The highway-skipped output-teacher can be used as a *local teacher* to facilitate **local learning**. This induces a notion of output residues which is dual to input-residues adopted in ResNet.

Denote the *teacher matrix* by $\mathbf{Y}$ and the *neuron data matrix* by $\mathbf{A}$. The activation values in a hidden layer can be partitioned into two parts:

- The first part, denoted as $\mathbf{Span}(\mathbf{Y}/\mathbf{A})$, represents teacher-relevant information already spanned/covered by the current layer.
- The second part, denoted as $\mathbf{Res}(\mathbf{Y}|\mathbf{A})$, represents the information still impenetrable by the current layer and thus requires continued learning via the subsequent layers. This is theoretically rooted in estimation theory, with the same connotation as the *innovation process* [9].

This process will be collectively referred to as Output-Residual-Learning because the leakage (i.e. residue) missed by the learning within the current layer has to be passed over to subsequent layers for further learning. With the backward broadcast (BB) making the teacher directly accessible to all hidden layers, the continued learning in all the subsequent layers can be focused on the output residue ($= \mathbf{Res}(\mathbf{Y}|\mathbf{A})$). Thus, the curse-of-depth problems (e.g. the vanishing gradients) can be greatly mitigated.

It is vital to note that the backward broadcast (BB) is needed only for the training phase. However, they will not be required in the testing phase.

3. Optimization formulations/metrics for classification and regression

ML systems are usually based on optimization formulation; so it is imperative to establish a suitable optimization metric (i.e. objective function).

3.1. Discriminant analysis for multivariate classification

Linear separability is commonly used as a guideline to define the discriminate ratio. For the scalar single-variate case, the most prominent being Fisher Discriminate Ratio (FDR). For classification, we propose a Discriminant Information (DI), a multivariate variant of FDR to measure the input layer's or a hidden-layer's discriminativeness. DI also proves to be vital for us to derive a DiLOSS metric allowing us to identify uninformative channels whose removal would be least detrimental to the performance.

3.1.1. Scatter matrices for scalar/matrix cases For classification problems, the training dataset consists of a set of input/output pairs denoted as $[X, Y] = [\mathbf{x}_1, y_1], [\mathbf{x}_2, y_2], \ldots, [\mathbf{x}_N, y_N]$ 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 additively divided into two parts [5]:

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

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 $\vec{\boldsymbol{\mu}}$ denotes the mass center, $\vec{\boldsymbol{\mu}}_\ell$ denotes the cluster centroid of the ℓ^{th} class ($\ell = 1, \cdots, L$), and J_W and $\boldsymbol{\Delta} \in \Re^{M\times L}$ represents the **between-centroids matrix**. In supervised learning, a feature's relevance is commonly represented by its SNR (Signal-to-Noise ratio) score, defined as the ratio of signal (inter-class distinction) to noise (intra-class perturbation). This leads to a notion named "Discriminant Matrix":

$$\text{DM} = [\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}\mathbf{S}_B \tag{4}$$

where ρ is the same as the ridge parameter used in the classic ridge regression, cf. Eq. (13).

3.1.2. Single-variate case: Fisher discriminant ratio (FDR) Let a_j denote the scalar activation value of one node/channel and

$$\bar{\mathbf{S}} = \bar{J} = \sum_{j=1}^{N}(a_j - \mu)^2 \tag{5}$$

where J_B are special cases of $\mathbf{S}_W$ and $\mathbf{S}_B$:

$$\mathbf{S}_W = J_W = \sum_{\ell=1}^{L}\sum_{j=1}^{N_\ell}(a_j^{(\ell)} - \mu_\ell)^2, \quad \mathbf{S}_B = J_B = \sum_{\ell=1}^{L} N_\ell\ (\mu_\ell - \mu)^2$$

Traditionally, for the scalar single-variate case, Fisher Discriminant Ratio (FDR) is defined as $FDR = \frac{J_B}{J_W}$. Note that $\bar{J} = J_B + J_W$, cf. Eq. (5). This

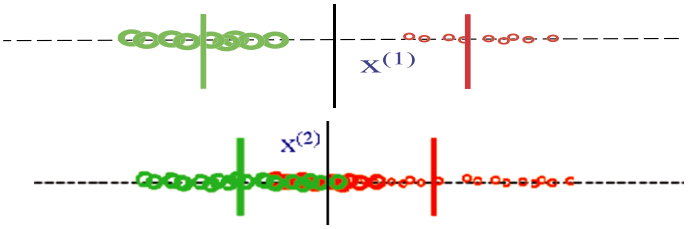


Figure 3: This figure illustrates the usage of FDR, which represents the ratio between signal to noise. (Signal is the between-class distance, and noise is the within-class spread.) In this example, $\mathbf{x}^{(1)}$ has higher FDR than $\mathbf{x}^{(2)}$.

equality allows us to adopt an equivalent denotation that $FDR = \bar{\mathbf{S}}^{-1}\mathbf{S}_B = \frac{J_B}{\bar{J}}$. Numerically, it is common to incorporate a ridge parameter ρ into the denominator $\bar{\mathbf{S}}$, resulting in $\bar{\mathbf{S}} + \rho\mathbf{I}$. In the single-variate case, this yields

$$FDR = (\bar{\mathbf{S}} + \rho\mathbf{I})^{-1}\mathbf{S}_B = \frac{J_B}{\bar{J} + \rho} \tag{6}$$

As illustrated in Figure 3, the scalar FDR ratio provides a simple score to assess how effective a feature is for the purpose of class separation.

3.1.3. Two multivariate variants of FDR For multivariate classification, there are two variants of FDRs being proposed:

- **Determinant FDR (DFDR):** C.R. Rao [24] proposed a Determinant FDR (DFDR) based on the determinant of the Discriminant Matrix in Eq. (4):

$$\text{DFDR} = \ \texttt{det}\left([\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}\mathbf{S}_B\right) \tag{7}$$

- **Trace FDR (TFDR):** In relatively recent years, a Trace-FDR (TFDR) based on the trace-norm of the Discriminant Matrix was proposed [29, 13, 11]

$$\text{TFDR} = \text{DI} = \ \texttt{tr}\left([\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}\mathbf{S}_B\right) \tag{8}$$

It can be shown that TFDR is theoretically related to Shannon's *mutual information* [27, 13]. Thus, for notational convenience, TFDR can be simply called **Discriminant Information** (DI).

Comparison of Two Multivariate FDR: DFDR vs. TFDR: Both DFDR and TFDR are transformation invariant, an imperative attribute for qualifying to be an effective metric to evaluate the discriminativeness of a layer. However, TFDR holds some additional merits over DFDR:

- **Subspace Analysis:** We note that removal of a neuron may actually increases DFDR, rendering it unsuitable for assessing the dispensability/usefulness of a neuron. For classification, the presence/absence of inter-channel redundancy hinges upon a mathematical condition called "Canonical Orthogonality" coin-phrased by C.R. Rao [24, 11]. Under such a condition, TFDR holds a vital Sum-Of-The-Parts (SOTP) property. Put simply, under the condition, the total TFDR score is equal to the sum of individual TFDR scores. The SOTP property implies that TFDR monotonically grows with the number of components. It means that removal of any neuron will result in monotocous decrease in TFDR. (Obviously, we would prefer to drop the neurons which would induce less decrease.) This makes TFDR a useful criterion to evaluate the dispensability of any individual neuron(s).
- **Derivative Analysis:** TFDR has a simple formula for it first order derivative with respect to each neuron activation function: DiLOSS, cf. Section 4.2. Obviously, we would prefer to drop the neurons with smaller DiLOSS. This makes the derivative of TFDR (named DiLOSS) a useful score to rank-order the neurons within the same layer.

3.1.4. Reformulate classification into regression problems For the regression case, the corresponding teacher values are continuous, and usually, in the *Real field* ($\Re$). Therefore, both the input and teachers have real values leading naturally to a real-valued External Optimization Metric (EOM). In order to link the classification and regression formulations, it is necessary to convert discrete teacher values into continuous teacher values.

Theorem 1 (Equivalence Between DI and RidgeLSE)**.** *Let* $\mathbf{X}$ *be the input matrix formed from all input vectors and* $\mathbf{Y}$ *be the teacher matrix formed from desired outputs prescribed by the teacher values. For both the balanced and unbalanced scenarios (i.e. with unequal sizes), equivalence between DI and Ridge LSE can be preserved by harnessing a weighted-one-hot-encoding, where the teacher values will replaced by* $\sqrt{N_i}^{-1}$ *instead of "1" as in "one-hot-encoding". Let* $\mathbf{Y}$ *denote the weighted-one-hot-encoded teacher matrix with center-adjustment, i.e. each row of* $\mathbf{Y}$ *has zero-mean. It can then be verified that* $\mathbf{S}_B = \mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T$*. Consequently,*

$$DM = [\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}[\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T] \tag{9}$$

It follows that

$$DI = \ tr(DM) = \ tr\left([\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}[\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T]\right) \tag{10}$$

When the optimal solution is reached, we have

$$RidgeLSE = \ \mathtt{tr}(\mathbf{Y}^T\mathbf{Y}) \ - \ \mathtt{tr}(\mathbf{Y}\mathbf{X}^T[\mathbf{X}\mathbf{X}^T + \rho\mathbf{I}]^{-1}\mathbf{X}\mathbf{Y}^T) \tag{11}$$

Combining Eq. (10) *and Eq.* (11)*, we have*

$$RidgeLSE = C - DI \tag{12}$$

where $C = \mathtt{tr}(\mathbf{Y}\mathbf{Y}^T) = L - 1$*, L denoting the number of different classes.*

In summary, assuming weighted-one-hot-encoded the teacher values, minimizing RidgeLSE and maximizing DI are essentially equivalent [29, 12].

3.2. Multivariate regression analysis

The extension from classification to regression hinges upon the intimate relation between *DI* and *RidgeLSE*: while linear separability can be measured by *DI*, the linear mappability can be likewise measured by RidgeLSE. Let $\mathbf{W}$ denote the optimal mapping matrix that best matches $\mathbf{X}$ to $\mathbf{Y}$, where $\mathbf{X}$ denotes the data matrix and $\mathbf{Y}$ the *center-adjusted* teacher matrix. Mathematically, $\mathbf{W}$ should be best solution to yield a minimum value of the ridge loss function:

$$E = -\text{RidgeLSE} = -\|(\mathbf{W}^T\mathbf{X} - \mathbf{Y})\|_F^2 - \rho\|\mathbf{W}\|_F^2 \tag{13}$$

where the subscript "F" denotes the Frobenius norm. This effectively converts the classification problem into a regression formulation.

The study so far points to the fact that either DI or RidgeLSE may be used interchangeably for optimization formulations. For this, we need to redefine a *Regression Matrix* for regression in lieu of the previous *Discriminant Matrix* meant for classification. Replacing $\mathbf{S}_B$ in the *Discriminant Matrix* (cf. Eq. (9)) by $\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T$, we obtain the following *Regression Matrix*:

$$\text{RM} = [\bar{\mathbf{S}} + \rho\mathbf{I}]^{-1}[\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T] \tag{14}$$

To maintain notational consistency, we again denote $\text{DI} = \ \mathtt{tr}(\text{RM})$ for the regression case. By inspection, the equivalence between DM in Eq. (9) and RM in Eq. (14) is evident. Consequently, we have

$$RidgeLSE = C - DI \tag{15}$$

where $C = \mathtt{tr}(\mathbf{Y}\mathbf{Y}^T)$ is the sum of the squared-teacher-values.

3.3. Equivalence between classification and regression

Conversely, it also means that the RidgeLSE may be used as the objective function in both problems. The only caution is that the DI-based LOM score (intended for the classification) should be replaced by an LSE-based LOM score (for regression). This formally establishes the equivalence between the DI and RidgeLSE metrics for both classification and regression analyses. Therefore, we can treat the classification and regression problems in a similar manner. This means that the same X-Learning strategy (cf. Section 5) will be valid for both classification and regression applications, i.e. "kill two birds with one stone".

3.4. Subspace/component analysis for classification/regression

There will be no surprise by now that the very same subspace/component analysis can be applied to both classification,and regression problems. Two popular approaches are elaborated below:

- **Discriminant Component Analysis (DCA).** DCA has been shown to provide an effective tool for subspace/component analysis [13, 11]. DCA components can be derived from the principal eigenvectors of DM and RM, respectively for classification and regression applications
- **Ranking Neurons: Selection/Pruning of Components.** Another component analysis involves finding the most informative or the most dispensable neurons in the layer. We shall adopt the latter approach: By trimming some deleterious neurons (evaluated by a DiLOSS metric), we can efficiently reduce learning models by a X-pruning method discussed in Section 5.

4. Local optimization metrics (LOM): DI, DiLOSS & DNs

BP is based on optimization of EOM, externally supervised by the teacher values at the output layer, often used to characterize the discrepancy of the actual response observed at the output nodes from the desired response prescribed by the teacher values. Thereafter, based on the first-order gradient of the EOM, BP can be applied to find the EOM-optimized network parameters. As such, we shall refer BP as an external learning paradigm.

While back-propagation has enjoyed great success in training parameters in various deep learning networks, it is not amenable to network structure

learning. So far, the task of designing optimal net structures is more or less left to trial and error. In order to gain high performance, the resultant networks tend to be unnecessarily bulky. This could in turn lead to costly design, rendering it unattractive for mobile/edge applications.

4.1. DI-based structural gradients

An inherent problem with BP is the assumption that all neurons trained by BP tend to be treated with equal importance, since they all work together collectively to create a desirable response. Idealistically, we want to have adequate network freedom but not to pay for unnecessary network complexity. For this, we note three key points:

- not all BP-trained hidden nodes (of the same layer) are created equal
- we must have an analytic metric to rank the neurons
- the ranking should serve as an effective tool to pick neurons to keep/drop.

To mitigate these concerns, backward broadcast (BB) let the teacher values become accessible to all the hidden layers directly and locally, paving a way to output-residual and local learning. The local teacher may now enable a direct evaluation of LOM to facilitate ranking of the individual nodes. Thereafter, X-learning can then remove low-score neurons (i.e DNs) and structurally trim each layer.

LASSO-Type LOM for Structural Learning: It is known that regularization, e.g., LASSO- and Ridge- types, leads to better generalization. For the LASSO type, we propose the following *structural energy function* for both classification and regression, c.f. Eq. (12) and Eq. (15).

$$E_S = DI_{local} - \lambda|\mathbf{a}|_0 \tag{16}$$

Model reduction via removal of low-DI nodes: For simplicity, let us momentarily assume that we remove one neuron at a time, resulting in the following before/after contrast:

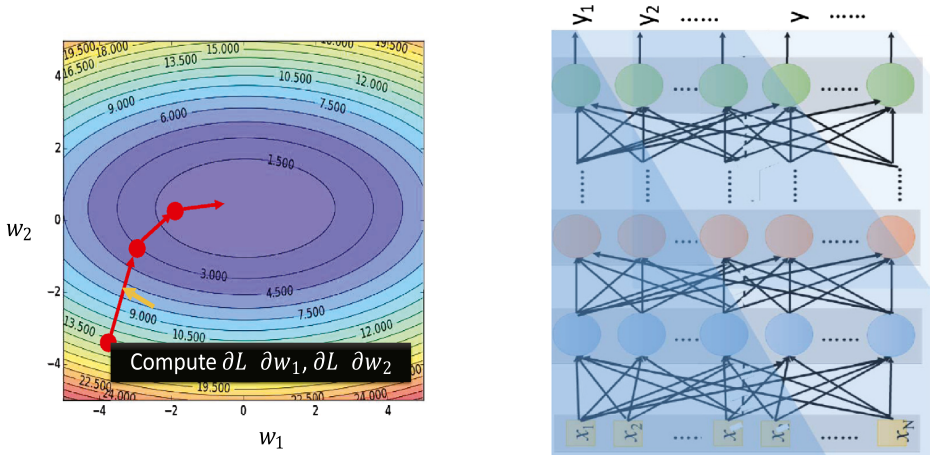
$$E_S = DI - \lambda\|\hat{\mathbf{a}}\|_0 \qquad \text{and} \qquad E'_S = DI' - \lambda\|\hat{\mathbf{a}}'\|_0 \tag{17}$$

with $\|\hat{\mathbf{a}}\|_0 = N$ and $\|\hat{\mathbf{a}}'\|_0 = N-1$ where N denotes the number of (nonzero) neurons in the layer.

Now further assume that what differentiate $\hat{\mathbf{a}}'$ from from $\hat{\mathbf{a}}$ is the removal of its i^{th} neuron. Because DI monotonically decreases with removal of neuron(s), then $\Delta DI_i = DI' - DI \leq 0$. In other words,

$$\Delta E_S = \lambda - |\Delta DI|_i \tag{18}$$

which means that E_S will increase upon removal of the i^{th} neuron only if $|\Delta DI|_i < \lambda$. This increase energy function allows a notion of greatest-ascent structural gradient, which can be combined with the conventional parameter gradient c.f. Figure 4(a). This leads to an evolutionary X-Learning strategy to jointly learn the structure and parameters of the network. As depicted in Figure 4, the structural gradient can facilitate effective structure learning.



(a) (b)

Figure 4: Two types of gradients: (a) parameter gradients and (b) structural gradient (via rank-ordering the neurons).

A greedy structural gradient method calls for an optimal structural pruning strategy. There are two intuitively sound schemes to achieve a maximal increase in E_S in each iteration:

- remove only neuron(s) with the least individual loss in $|\Delta DI|$ to maximize ΔE_S.
- remove more neuron(s) as long as it increases E_S, i.e. when $|\Delta DI| < \lambda$.

However, these two schemes contradict each other:

- To enforce the first scheme, we need to set the threshold λ lower so that only (very few of) least-loss neurons get removed in each iteration.

- To enforce the second scheme, in contrast, we need to set λ higher so that more neurons may qualify the condition that $|\Delta DI| < \lambda$.

Thus the threshold λ is usually empirically (not analytically) determined.

4.2. DiLOSS for rank-ordering neurons

In order to expedite the computation of ΔDI, let us introduce a differentiation-friendly vector $\tilde{\mathbf{a}}$, which is the same as $\hat{\mathbf{a}}$, the only difference being that its i^{th} element is scaled by a real value α_i: $\tilde{a}_i = \alpha_i a_i$. It can be verified that

- when $\alpha_i = 1$, $\tilde{\mathbf{a}} = \hat{\mathbf{a}}$
- when $\alpha_i = 0$, $\tilde{\mathbf{a}} = \hat{\mathbf{a}}'$ (now $\|\tilde{\mathbf{a}}\|_0 = N - 1$).

Since $\Delta\alpha_i = -1$, we have

$$|\Delta DI|_i \approx |\Delta\alpha_i| \frac{\partial DI}{\partial \alpha_i} = \frac{\partial DI}{\partial \alpha_i}$$

Recall that DiLOSS is defined as the differentiated DI for both classification and regression analyses:[1]

$$\text{DiLOSS}_i \equiv \frac{\partial DI}{\partial \alpha_i} \tag{19}$$

which has a closed-form formula shown below (detail omitted):

- **DiLOSS for Classification:**

$$\text{DiLOSS}_i = 2\rho \left([\bar{\mathbf{S}} + \rho \mathbf{I}]^{-1} \mathbf{S}_B [\bar{\mathbf{S}} + \rho \mathbf{I}]^{-1} \right)_{ii} \tag{20}$$

where the subscript $(\cdot)_{ii}$ denotes the $(i,i)^{th}$ entry in a matrix. Further denoting $\mathbf{G} \equiv [\bar{\mathbf{S}} + \rho \mathbf{I}]^{-1} \mathbf{\Delta}$, cf. Eq. (3), we have

$$\text{DiLOSS}_i \equiv 2\rho \left(\mathbf{G}\mathbf{G}^T \right)_{ii} = 2\rho \|\mathbf{G}_i\|^2 \tag{21}$$

where the subscript $(\cdot)_i$ denotes its i^{th} row vector.

[1] Since there is no cross-referencing between different layers, we shall drop the index $(l-1)$ or (l). Moreover, the computation of DiLOSS involves a batch method, so we can drop the superscript $(\cdot)^{(n)}$ as well.

- **DiLOSS for Regression Analysis:**

$$\text{DiLOSS}_i \;=\; 2\rho\left([\bar{\mathbf{S}}+\rho\mathbf{I}]^{-1}\mathbf{X}\mathbf{Y}^T\mathbf{Y}\mathbf{X}^T[\bar{\mathbf{S}}+\rho\mathbf{I}]^{-1}\right)_{ii} \tag{22}$$

Now, denoting $\mathbf{G} \equiv [\bar{\mathbf{S}}+\rho\mathbf{I}]^{-1}\mathbf{X}\mathbf{Y}^T$ for regression analysis, we again have $\text{DiLOSS}_i = 2\rho\left(\mathbf{G}\mathbf{G}^T\right)_{ii} = 2\rho\|\mathbf{G}_i\|^2$.

4.3. Deleterious neurons (DNs)

The Deleterious Nodes (DNs) can now be formally defined as neurons with

$$\text{DiLOSS}_i < \lambda \tag{23}$$

Consequently, removal of any deleterious neuron(s) will result in a net increase in E_S. Per our previous discussion, in order to have an optimal structural pruning strategy, the optimal threshold λ needs to be empirically and independently specified for each layer.

5. Evolutionary X-learning paradigm

A multi-layer linear basis function (LBF) network is characterized by the following equations:

$$\begin{aligned} u_j(l) &= \sum_{i=0}^{M_{l-1}} w_{ji}(l)a_i(l-1) \\ a_j(l) &= f(u_j(l)) \qquad 1 \le j \le M_l \text{ and } 1 \le l \le L, \end{aligned}$$

where the input units are represented by $x_i \equiv a_i(0)$, the output units by $y_i \equiv a_i(L)$, and where L is the number of layers and $f(\cdot)$ is conventionally a sigmoidal function or, more recently, a ReLU activation function.

5.1. Consistency between EOM and LOM

Note that we have two optimization metrics: $E_{external}$ for the external BP learning and E_{local} for the local learning. For external optimization, we minimize an external optimization metric (EOM) associated with the activation function of the output layer. It is similar to Eq. (13), except that the input data matrix ($\mathbf{X}$) is replaced by the output data matrix: $\mathbf{A}_{output}$:

$$E_{external} = -\text{RidgeLSE}_{external} = -\|(\mathbf{W}^T\mathbf{A}_{output} - \mathbf{Y})\|_F^2 - \rho\|\mathbf{W}\|_F^2 \tag{24}$$

Likewise, we need to specify the local optimization metric (LOM) based on the internal teacher labels so as to facilitate an effective ranking of neurons in each hidden layer. To derive LOM, we replace the input data matrix ($\mathbf{X}$) by the activation matrix $\mathbf{A}$ associated with a hidden layer ($\mathbf{X} \rightarrow \mathbf{A}_{local}$) in Eq. (13), resulting in

$$E_{local} = -\text{RidgeLSE}_{local} = -\|(\mathbf{W}^T\mathbf{A}_{local} - \mathbf{Y})\|_F^2 - \rho\|\mathbf{W}\|_F^2 \tag{25}$$

A high degree of consistency between EOM and LOM would best foster our iterative learning, where both the external and local objective functions need to be optimized. The equivalence relationship between EOM and LOM plays a vital role for reaching a consistent convergence. (It is dangerous when a stagecoach is pulled by two horses running towards different directions.)

5.2. PS-iterative learning: mixed BP-learning and X-pruning

In order to "kill two birds with one stone", we adopt the following objective functions:

- **Classification:** For best *linear separability*, the LOM and EOM are defined as:

$$E_{external} = DI_{output} (= DI_{external}) \text{ and } E_{local} = DI_{layer}$$

- **Regression:** For best *linear mappability*, we define EOM and LOM as:

$$E_{external} = -RidgeLSE_{output} \text{ and } E_{local} = -RidgeLSE_{layer}$$

As discussed before, both $EOM = E_{external}$ and $LOM = E_{local}$ can be jointly optimized in a mutually consistent fashion. Now we are ready to propose an (EM-style) Evolutionary and Iterative X-Learning, aiming at jointly optimizing EOM (in the parameter space) and LOM (in the structural space):

- **Parameter BP-Learning:** For parameter learning, we shall just rely on the traditional (externally-supervised) BP Learning based on optimization of the pre-specified EOM objective function.
- **Structural X-Pruning:** Locally-supervised learning based on LOM, which in turns leads to DiLOSS and designated DNs to be pruned from the network.

Evolutionary PS-iteration: In X-Learning, the network will be iteratively updated in two design-spaces: BP on the parameter space and X-Pruning on the net space. The iterations between parameter and structural spaces is pictorially illustrated by Figure 5.

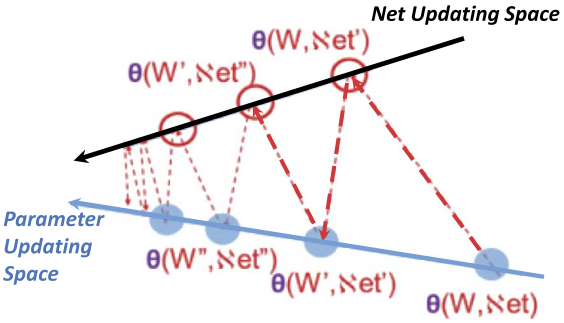


Figure 5: Pictorially illustration of Evolutionary PS-iteration.

PS-iteration: Each iteration in X-Learning contains two phases: the phase for parameter updating is called **P-phase** while the structural learning is named **S-phase**.

- **P-phase: Parameter Updating Phase.** Upon the structural pruning of any DNs, we shall retrain the remaining network by the external BP learning to recover some loss in the training accuracy. Via a BP chain-rule, we have

$$\Delta w_{ji}^{(n)}(l) \quad = \quad -\eta \frac{\partial E}{\partial w_{ji}^{(n)}(l)} = \eta \ \delta_j^{(n)}(l) f'(u_j^{(n)}(l)) a_i^{(n)}(l-1)$$

  where the *error signal* $\delta_j^{(n)}(l) \equiv -\frac{\partial E}{\partial a_j^{(n)}(l)}$.
  The BP parameter updating rule is:

$$w_{ji}^{(n+1)}(l) = w_{ji}^{(n)}(l) + \Delta w_{ji}^{(n)}(l), \forall \text{ layers}, \ l = 1, \cdots, L \tag{26}$$

- **S-phase: Structural Pruning Phase.** In each iteration, upon the completion of the parameter updating phase, we shall further remove those neurons designated as DNs (cf. Eq. (23)), i.e. $DiLOSS_i < \lambda$. The pruning process will be performed on all the $hidden-layers$, each layer with its own threshold λ, where λ will be empirically determined.

Subsequently, these DNs will be pruned from the network before the external BP learning is applied to retrain the entire network again. (Once the DNs are trimmed, they will be dropped permanently.)

The procedure in X-Learning is summarized in the following pseudo-code:

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 LOM score, i.e., 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

The iterative training method imitates an evolutionary pruning process, imitating evolutionary change from the huge anhanguera to its tiny counterpart: bird. Note that the evolution process invokes not only size reduction but also functional readjustment (manifested by the net parameters).[2]

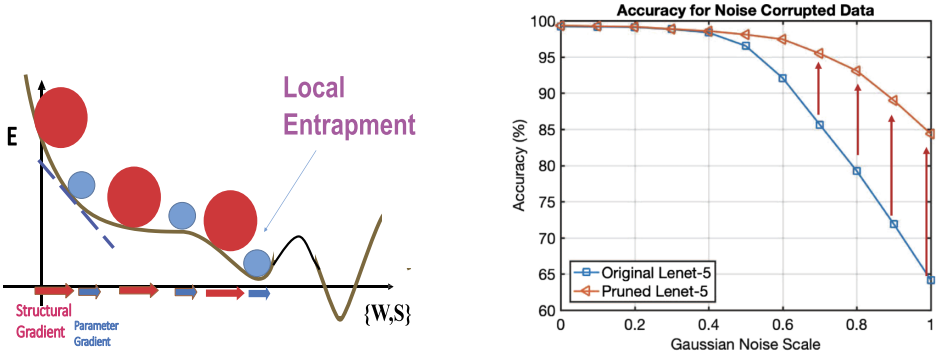
In short, the PS-iteration represents X-Learning Paradigm, for joint training of structure and parameters of the network. It embraces a versatile application domains covering both classification and regression problems. Our experiments have demonstrated that X-Learning can outperform many prominent state-of-the-arts approaches.

5.3. Avoid entrapment of local optima and mitigate overfitting problems

The X-Learning paradigm boasts two noteworthy merits:

- **Avoid Entrapment of Local Optima:** A major difficulty often encountered by the parameter gradient method (e.g. as in BP learning) is due to the entrapment of the unwanted local minimum. Although a large step size may escape local minima, the chancel is that it may actually harm (instead of help) the optimization process, according to the conventional wisdom. We now show how the X-learning (which involves

[2]Each time we gradually remove a small fraction of low-ranking nodes at one time. The subnet is then updated by backpropagation, before the next round of elimination. This has an important implication that the structural ranking will be constantly and adaptively re-adjusted.



(a) (b)

Figure 6: (a) The larger ball represents the the move via structural gradient while the smaller ball represents the move via parameter gradient. (b) It shows that there is significant difference/gap (in prediction performance) between the baseline full model and X-pruned model. Moreover, the gap rapidly grows with increasing variance on the additive Gaussian noises.

a mixed learning combining BP-Learning and X-pruning) may numerically offers a viable path to escape from such entrapment: A structural move means resetting the magnitude of all DN-linked weights (i.e. $w'_{ij} \to 0$). This represents a big move compared with a conventional parameter updating. For the latter, the amount of weight updating (i.e. $w_{ij} \to w_{ij} + \Delta w_{ij}$) is usually infinitesimal. This phenomena can be further illustrated by Figure 6(a).

Note that there are two different sizes of balls: the larger ball symbolizes a daring move via structural gradient, while the smaller ball represents the tiny move via parameter gradient in the structural space. The larger ball corresponds to a "giant leap" in the parameter space.

- **Mitigate Overfitting via Removals of DNs:** The following demonstrates that there exists plenty of weak neurons in the original model, which do more harm than good. These weak neurons are vulnerable to noise attack, making it detrimental to the prediction performance. As evidenced by our experiment on the MINIST Dataset, cf. Figure 6(b), low-DI DNs become noticeably vulnerable when we increase the power of additive noise on the input. (Note that the performance gap grows rapidly with the increasing power of the additive noise.)

6. Experimental results of X-learning for classification

We shall present our experimental results obtained by applying X-learning to some popular classification tasks/datasets. The focus of this section will be placed on MobileNet and ResNet models. For other CNN models, see [14] for more discussion on applications of X-Learning.

Structure-wise, we adopt a dual-skip connection (forward skip and backward broadcast) which allows us to take a full advantage of both input and output residual learning. Consequently, local teachers are now directly accessible to all the hidden layers so that each layer's DI-score (Eq. (8)) can be locally computed. Subsequently, pursuant to Eq. (21), DiLOSS for every neurons become readily computable as well. Based on DiLOSS, we can identify DNs to be pruned in the S-phase in the PS-Iterative algorithm. To show its broad appeal to classification applications, we cover numerous popular datasets, including MINIST, CIFAR10, CIFAR100, and ImageNet.

6.1. X-pruned classifiers for ImageNet dataset

We aim at two separate design purposes: one for low-power and the other for high-performance. As such, we shall respectively apply X-Learning to MobileNet and ResNet.

6.1.1. Low-power models: X-learning on MobileNet As shown in Table 1, X-learning can reduce MobileNetV1&V2 to yield faster latency, and smaller hardware, making it amenable to edge devices. More specifically, at 30ms/image real-time speed (required by LPIRC 2019), X-MobileNetV1 delivers 68.2% top-1 accuracy on ImageNet, a 3% improvement compared with 65.2% reported by the winner of 2018 LPIRC (MobileNet).

Table 1: Comparison with winner of 2018 LPRIC on ImageNet

Model	Top-1 Acc.	FLOPs	Param.
MobileNetV2	71.80%	300M	3.47M
WM (Sandler et al., 2018)	69.80%	210M	2.61M
X-MobileNetV2	70.80%	210M	2.33M
LPIRC 2018	65.20%	186M	–
ThiNet (Luo et al., 2017)	65.44%	170M	2.57M
DCP (Zhuang et al., 2018)	65.91&	170M	2.57M
X-MobileNetV1	68.20%	170M	1.90M

6.1.2. High-performance models: X-learning on ResNet Our X-ResNet50 achieves 75.66% top-1 accuracy, a 0.5% gain over the baseline, while accelerating the inference speed by 2x. This result outperforms the current state-of-the-art method (Molchanov et al., 2019) by around 1% in accuracy with similar speeds, as shown in Table 2. Note that in Table 2, we provide results of three X-ResNet50 under different FLOPs reduction ratios.

Table 2: Comparison with other methods in pruning ResNet50 on ImageNet

Model	Top-1 Acc.	FLOPs	Param.
ResNet50 (baseline)	75.15%	4.09B	25.60M
X-ResNet50	76.34%	2.63B	17.00M
SSS (Huang et al., 2018)	71.82%	2.33B	15.60M
ThiNet (Luo et al., 2017)	71.04%	2.44B	16.94M
GDP (Lin et al., 2018)	72.61%	2.24B	–
SFP (He et al., 2018)	74.61%	2.42B	–
X-ResNet50	76.10%	2.31B	16.72M
Taylor (Molchanov et al., 2019)	74.60%	2.00B	–
X-ResNet50	75.60%	2.00B	15.60M

6.2. X-learning on MNIST, CIFAR10, and CIFAR100 datasets

There are many other popular datasets, we shall highlight our results on MNIST, CIFAR10 and CIFAR100 datasets. Compared with the baseline models, X-Learning can simultaneously compress the structure and raise the accuracy. More significantly, X-Learning outperforms many of existing pruning methods, as elaborated in below.

- **MINIST Dataset** We conduct experiments on MNIST [16] dataset consisting of 60,000 training samples and 10,000 testing samples in 10 classes. On Lenet-5, we achieve 20x in speedup and 100x in storage reduction, compared to 10x reduction from Han et al. [6].
- **CIFAR-10 Dataset** CIFAR-10 [10] dataset consisting of 50,000 training samples and 10,000 testing samples in 10 classes. As shown in Table 3, X-Learning is effective on more compact and advanced ResNet56 using CIFAR10 datasets. More precisely, X-Learning yields an accuracy improvement of 0.8% (93.04% to 93.84%). In addition, X-ResNet56 offers 3x speedup compared with the baseline (ResNet56).
- **CIFAR-100 Dataset** CIFAR-100 [10] dataset consisting of 50,000 training samples and 10,000 testing samples in 100 classes. 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 achieve

Table 3: Comparisons of pruning ResNet56 using CIFAR10

Model	Top-1 Acc.	FLOPs	Params.
ResNet56 (baseline)	93.04%	250M	0.85M
L1-norm (Li et al., 2017)	93.06%	181M	0.73M
CP (He et al., 2017)	91.90%	125M	–
DCP (Zhuang et al., 2018)	93.49%	125M	0.43M
X-ResNet56	93.84%	83.8M	0.31M

77.70% accuracy, which is 1% higher than the baseline model at 185M FLOPs and 0.62M parameters (equivalent to 2.72x FLOPs reduction and 2.74x storage size compression). Moreover, our X-ResNet164 outperforms Network Slimming method [19] by 1% accuracy and 25% less FLOPs.

6.2.1. Bootstrapping Note that X-Learning and a recently proposed method DCP [32] are both based on discriminant analysis. Since they are complementary, it is natural to further improve performance by bootstrapping each other. We can further reduce both storage (from 0.31M to 0.28M) and FLOP (from 83.8M to 75.8M) of ResNet56.

6.3. Soft X-pruning

For X-Quantization, we invent a closed-form quantization matrix (Q), to optimally control the quantization level of channels: higher-precision (more-bits) for more informative channels. By cascading X-Pruning and X-Quantization, we can reduce the size of ResNet-110 by 20x (3x reduction by X-Pruning and 7x reduction by X-Quantization) without any loss of classification accuracy on CIFAR10.

Moreover, we propose a codebook quantization method, by it self (without X-Pruning), it can compress compress the ResNet18 model by 20x on the ImageNet dataset, and achieves 67.5% accuracy. It represents a win-win performance over ABC-Net (64% accuracy with 15x reduction [18]) as well as LR-Net (63.5% accuracy with 15x reduction [28]).

7. Regression scenarios: X-learning for SR imaging systems

The objective of a typical imaging problem is that, given an (low-quality) input image (poor quality in resolution and/or color rendition, etc.), find a (nonlinear) mapping to transform the input image into a desired output image, e.g. one restoring/resembling the GT and/or one enhancing the image

visualization. Our main focus will be placed on the application of fidelity-based SR imaging systems, which tackles the problem of recovering a high-resolution image from a single low resolution image.

To learn the prior, recent state-of-the-art methods mostly adopt the example-based strategy, most of them learn mapping functions from external low- and high-resolution exemplar pairs. This approach is closely related to recent example-based methods based on deep convolutional neural networks (CNN). In fact, Dong et al. [4] pointed that the sparse-coding based SR methods [2] can be viewed as a deep convolutional neural network. The deep learning method can be formulated for generic image super-resolution, where the teacher will be set as the desired high-resolution images. There is evidence showing the same techniques can be naturally carried over to other types of applications with their own training samples, such as enhancing dim-light images to brighter image, where the teacher will be set as the desired brighter images.

X-learning can be effectively applied to most regression applications. Moreover, most restoration and enhancement problems can be formulated as regression-type applications. As a specific example, the generic imaging systems are naturally amenable to regression analysis as they involve continuous-valued input and teachers (i.e. ground truth) in image fields.

7.1. X-learning approach to low-power SR imaging systems

Image super-resolution and image enhancement have numerous practical applications. In this problem, the original images are low-quality images, and the objective is to restore the low-quality images to a high-fidelity ones. Both the external and local teachers will simply be the ground-truth high-quality images.

Training/Testing Datasets: The XSR-ResNet was applied to super-resolution image enhancement with DIV2K as the training dataset. The learned SR-imaging models are then applied to several test datasets, including Set5, Set14, BSD100, and Urban100.

Improvement Over the Baseline (SRGAN): Our baseline model is SRGAN, a predominant CNN-based model for SR imaging systems. Because the original ResNet is bulky and X-reduction-friendly, X-pruning is ideal for model reduction. A key challenge, however, lies in that SRGAN employs the Subpixel layers to up-scale the LR images to the final HR output. This scheme is by itself not amenable to X-Learning. To circumvent this problem,

we need to replace the subpixel layers by the nearest-neighbor interpolation (NNI) with learnable convolution filters for up-scaling. Thereafter, it becomes feasible that these upsampling layers also undergo PS-iterations, with both BP-learning and X-learning. This results in a very impressive reduction, with a saving factors of 16x in model size and 14x in FLOPs, while suffering from a negligible performance loss of −0.2 db.

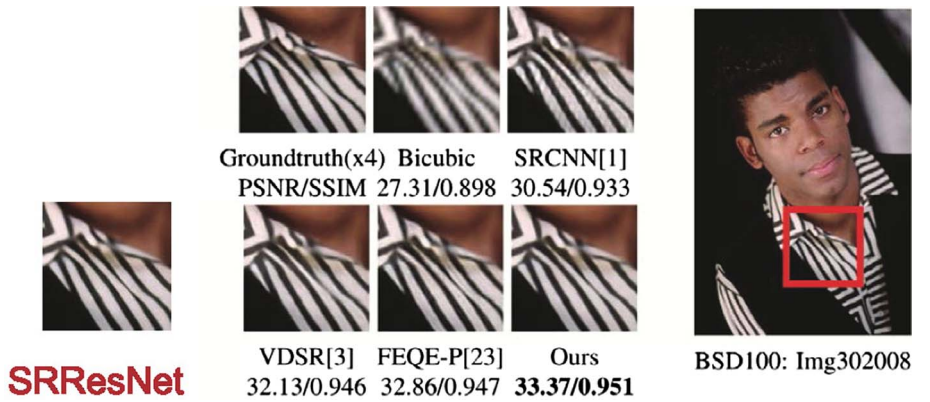


Figure 7: We show SR-imaging on one of the BSD100 images under 4×4 bibubic down-sampling setting. The results of SR-RESNET, FEQE, and XSR-RESNET are displayed. (Note the ground truth and ours do not show the artifact appearing in the third rightmost white stripe in FEQE.)

Improvement Over Winner of PIRM Challenge (FEQEnet): In addition, our X-SRResNet outperforms the FEQEnet, the winner of 2018 PIRM Challenge. More precisely, X-SRResNet delivers an advantage of shorter latency by more than two folds while holds a .3 db gain in PSNR over the FEQEnet, as shown in Table 4. Visual results of SR-RESNET, FEQE, and XSR-RESNET (ours) are displayed in Figure 7.

7.2. X-learning for high-performance SR-imageing systems

We also make a comparison between the X-Pruning based network compression and a network binarization method for the SR system [20], where the weights of the original CNN are converted to $[-1, +1]$ to save model size and computation complexity. For baseline CNN, we use LapSRN and SRResNet. Table 5 summarizes the PSNR/SSIM quantitative performance of our X-Pruning models versus binarized models, where we can observe that X-Pruning outperform SR binarization by a large margin.

Table 4: Comparison with Winner of PIRM Challenge (FEQEnet). We report values of peak signal-to-noise ratio (PSNR) / structural similarity index measure (SSIM) for each method on four image SR benchmarks

Dataset	SRCNN	VDSR	FEQE-P	Ours
Set5	30.47/0.8610	31.53/0.8840	31.53/0.8824	**31.84/0.889**
Set14	27.57/0.7528	28.42/0.7830	28.21/0.7714	28.38/0.775
BSD100	26.89/0.7108	27.29/0.7262	27.32/0.7273	**27.40/0.730**
Urban100	24.51/0.7232	25.18/0.7534	25.32/0.7583	**25.51/0.765**

Model	#Params.	FLOPs	GPU Latency
SRCNN	69K	128B	0.04 s
VDSR	668K	1231B	0.16 s
FEQE-P	96K	11B	0.01 s
Ours	92.6K	**9.6B**	**0.004 s**

Table 5: Comparison of X-Learning with SR-binarization method. We report values of peak signal-to-noise ratio (PSNR) / structural similarity index measure (SSIM) for each method on four image SR benchmarks

Dataset	Scale	Bicubic	LapSRN	LapSRN Binary	X-LapSRN	SRResNet	SRResNet Binary	X-SRResNet
Set5	2	33.66/0.930	37.25/0.957	–	37.35/0.959	37.90/0.959	35.66/0.946	**38.04/0.961**
	4	28.42/0.810	31.33/0.881	30.21/0.857	31.60/0.887	32.05/0.891	30.34/0.864	**32.20/0.894**
Set14	2	30.24/0.869	32.96/0.910	–	33.20/0.916	33.44/0.915	31.56/0.897	**33.62/0.918**
	4	26.00/0.703	28.06/0.768	27.13/0.751	28.19/0.774	28.49/0.780	27.16/0.756	**28.62/0.781**
BSD100	2	29.56/0.843	31.58/0.892	–	31.73/0.894	32.12/0.899	–	**32.20/0.900**
	4	25.96/0.668	27.22/0.724	–	27.33/0.730	27.58/0.735	–	**27.60/0.737**
Urban100	2	26.88/0.840	30.25/0.907	–	30.34/0.910	31.80/0.925	28.76/0.882	**32.10/0.928**
	4	23.14/0.657	25.02/0.747	24.31/0.720	25.26/0.761	25.90/0.782	24.48/0.728	**26.05/0.784**

Moreover, in Table 6, we report in details the network complexity and actual inference acceleration of our X-Pruning model, compared with the network binarization method. In summary, X-Pruning achieves higher reduction factor of model size, FLOPs, and latency.

Table 6: Comparison of X-Learning with SR-binarization method in terms of network complexity

Model	Scale	Weight	#Params.	FLOPs	Model Size	CPU/GPU Latency
SRResNet	2	float32	1.375M	6.369B	5.499 MB	7.29 s/137 ms
SRResNet Binary	2	$\{-1, +1\}$	1.377M	b-ops	0.928 MB	–
X-SRResNet	2	float32	**0.154M**	**0.743B**	**0.631 MB**	**1.99 s/117 ms**
SRResNet	4	float32	1.522M	9.185B	6.089 MB	1.65 s/38 ms
SRResNet Binary	4	$\{-1, +1\}$	1.524M	b-ops	1.518 MB	–
X-SRResNet	4	float32	**0.301M**	**3.548B**	**1.222 MB**	**0.68 s/30 ms**
LapSRN	4	float32	0.870M	8.549B	3.509 MB	2.66 s/98 ms
LapSRN Binary	4	$\{-1, +1\}$	0.512M	b-ops	1.494 MB	–
X-LapSRN	4	float32	**0.121M**	**0.981B**	**0.517 MB**	**0.85 s/74 ms**

7.3. Hierarchical CNN for high-performance SR-imageing systems

7.3.1. Network architecture Inspired by the observation that the effective receptive field increases whenever the feature goes through a convolution operation, we propose a hierarchical aggregation to achieve multiple equivalent receptive fields and enrich the feature scales in the final output. In the prior design of MSRN [17], one block consisting of two by-pass with 3x3 convolution kernel and 5x5 convolution kernel, respectively. Outputs from two by-pass are concatenated and sent into 1x1 convolution for feature fusion. In contrast, the novelty in our model hinges upon wider windows due to our hierarchical multi-stage design (refered to as HSRN), as depicted in Figure 8. For example, by cascading 3x3 windows twice we obtain effectively a 9x9 window. Moreover, by cascading 3x3 windows thrice we obtain a 27x27 window. It is intuitive that the widened windows can lead to greater flexibility and, ultimately, higher performance.

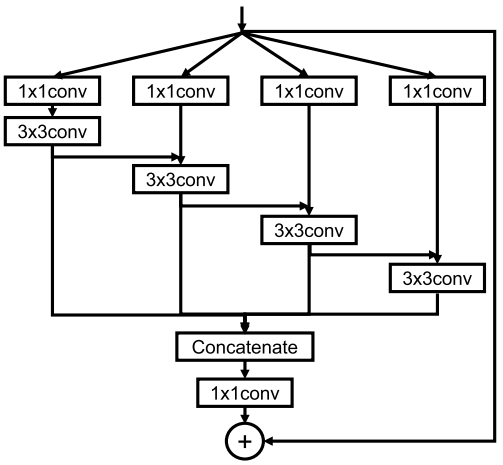


Figure 8: Hierarchical Network proposed for SR-imaging systems.

7.3.2. Performances on SR benchmarks We use the 2K-resolution dataset DIV2K [1] (800 training images and 100 validation images) to train our network. For testing, we evaluate our models on four standard benchmarks: Set5 [3], Set14 [31], BSD100 [21], and Urban100 [8]. LR images by generated by down-sampling the corresponding HR images using the bicubic kernel function.

Evaluation metrics: For quantitative evaluation, we calculate PSNR and SSIM between the reconstructed image and the ground-truth on Y-channel of the YCbCr color space. Figure 9 shows a visual comparison of results via different SR-imaging systems.

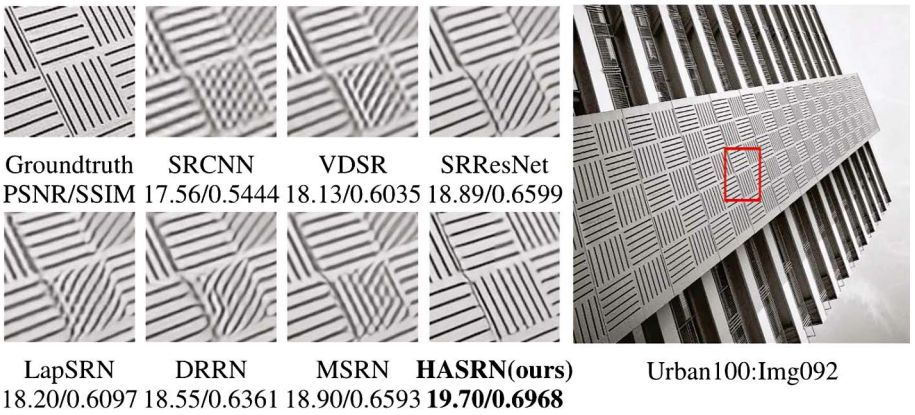


Figure 9: Hierarchical SR Imaging for Urban100 Image092.

7.3.3. Performances on finger-print dataset Figure 10 shows a visual comparison on fingerprint image super-resolution. Also, quantitative comparison is provided in Table 7, where our method outperforms SRGAN, which represents the state-of-the-art on the FVC2000 dataset, by 1.5 dB in PNSR.

Table 7: Comparison of HSRN (ours) and SRGAN on fingerprint SR

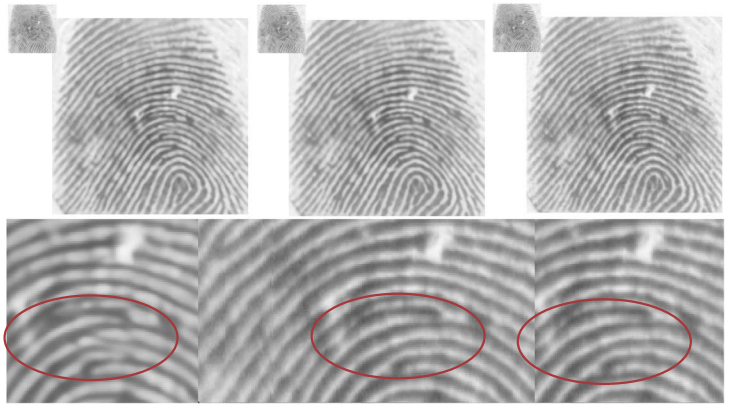
Dataset	Scale	SRGAN	Ours Multi-scale
FVC2000	2	42.93/0.980	42.96/0.981
	4	34.98/0.916	36.51/0.934

7.4. Other imaging systems

Based on the DIV2K dataset, we apply the proposed HSRN to other image restoration results. For denoising application, refer to Table 8. For deblocking application, refer to Table 9.

8. From XAI-learning to domain-driven learning models

This section points to some future research fronts: (1) X-Learning could be instrumental/amenable to the development of Logical Deep Learning



(a) (b) (c)

Figure 10: The LR image is shown on the top-left corner. Visual comparison of differnet results on fingerprint enhancement:(a) SRResNet, (b) HSRN (ours), (c)Ground-truth.

Table 8: Performance of HSRN on denoising application/datasets

Dataset	Noise σ	BM3D	TNRD	RED	DnCNN	MemNet	IRCNN	FFDNet	RNAN	HMSDAN
Kodak24	30	30.89	28.83	29.71	31.39	29.67	31.24	31.39	31.86	**33.35**
	50	28.63	27.17	27.62	29.16	27.65	28.93	29.10	29.58	**30.96**
BSD68	30	29.73	27.64	28.46	30.40	28.39	30.22	30.31	30.63	**32.16**
	50	27.38	25.96	26.35	28.01	26.33	27.86	27.96	28.27	**29.72**
Urban100	30	30.36	27.40	29.02	30.28	28.93	30.28	30.53	31.50	**33.29**
	50	27.94	25.52	26.40	28.16	26.53	27.69	28.05	29.08	**30.58**

which include DARPA's XAI-Learning as a most promising special case. (2) Domain-driven systems featuring double-CNN learning: one CNN for feature engineering and another for label engineering.

8.1. XAI-learning and logical deep learning

X-Learning appears to be naturally amenable to DARPA's XAI initiative on End User Explainability, where the main focus is placed on "explanations of higher-level decisions that would be relevant to the end user and the missions he/she needs to manage"

It is vital to have end-user-adaptive label(s) so that a learning model may be re-purposed to active learning environments. In X-Learning, LOM can facilitate quantitative analysis on the explainability of hidden nodes. The LOM-selected channels may rapidly retrieve information of grave con-

Table 9: Performance of HSRN on deblocking application/datasets

Dataset	q	JPEG	ARCNN	TNRD	DnCNN	MemNet	IACNN	RNAN	HMSDAN
LIVE1	10	27.77/0.7905	28.98/0.8217	29.15/0.8111	29.19/0.8123	29.45/0.8193	29.34/0.8199	29.63/0.8239	**29.90**
	20	30.07/0.8683	31.29/0.8871	31.46/0.8769	31.59/0.8802	31.83/0.8846	31.73/0.8848	32.03/0.8877	**32.30**
	30	31.41/0.9000	32.69/0.9166	32.84/0.9059	32.98/0.9090	–	33.19/0.9132	33.45/34.47	**33.71**
Classic5	10	27.82/0.7800	29.04/0.8111	29.28/0.7992	29.40/0.8026	29.69/0.8107	29.43/0.8070	29.96/0.8178	**30.92**
	20	30.12/0.8541	31.16/0.8694	31.47/0.8576	31.63/0.8610	31.90/0.8658	31.64/0.8628	32.11/0.8693	**33.16**
	30	31.48/0.8844	32.52/0.8967	32.78/0.8837	32.91/0.8861	–	32.93/0.8874	33.38/0.8924	**34.48**

cern/threat to the end-user, e.g. an imminent drone attack, cf. Figure 11. In this case, X-Learning allows us to rapidly identify the most relevant neurons to form quick decision/response in the run-time.

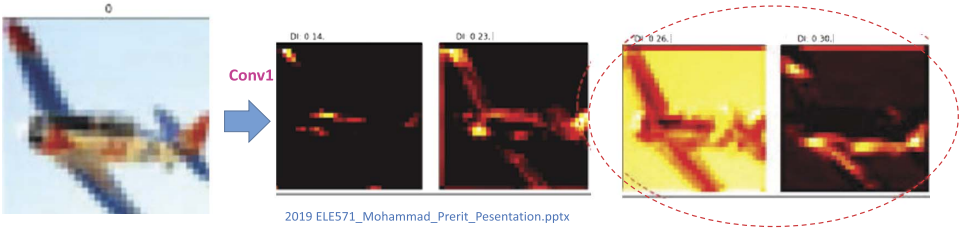
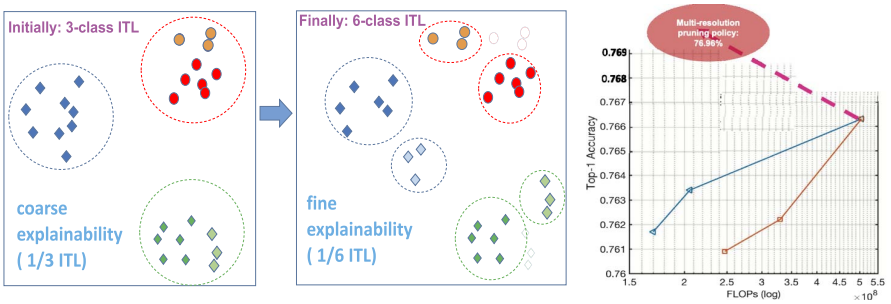


Figure 11: Deleterious versus intelligent neurons for the end-user's ABC-query: Airplane, Bird, or Car?

Example: Multiple-Granularity Labeling Boosts CIFAR-100 Performance: The layered-structure of deep learning network is meant to gradually map the input space to the output space (layer by layer). To facilitate such gradual conversion, we adopt a layer-dependent designation of Local Labels (LLs): gradually from coarse-granularity LL to fine-granularity LL, cf. Figure 12(a). In our experimental study on CIFAR-100, which has naturally two types of labels, we have found that the adoption LL-based teachers can further boost the final prediction performance. More exactly, we have observed an improvement of around 1% over the basic X-learning results, Figure 12(b).



(a) (b)

Figure 12: Multiple-granularity labeling may boost prediction performance.

8.2. Domain-driven learning example: image-based classification systems

With reference to a Domain-Driven imaging for mammogram detection, sketched in Figure 1, we have conducted a domain-driven experiment on a popular domain-specific Oxford-Flowers dataset [22]. More specifically, we adopt a slimmer version of SRResNet (e.g., 10 layers) for super resolution part, to reduce the computation overhead. For the classification part, we adopt the most commonly used deep CNN, e.g., VGG and ResNet. For training, we adopt two scenarios:

- **Scenario 1:** two sets of supervising teachers are used: one for super resolution, and one for classification. This also explains why the domain-driven imaging system can be called as doubly-driven systems
- **Scenario 2:** an end-to-end training of both SR parts and classification parts by the classification teacher labels only

As shown in Table 10, our domain-driven imaging systems in both scenarios achieve better accuracy than the low-resolution baseline. More importantly, we can see that the end-to-end training (i.e. scenario 2) achieves the best classification results. This suggest that domain-specific classification labels can better guide the training of SR to produce useful high-resolution images to achieve better accuracy.

Table 10: Domain-driven SR classification on Oxford-Flowers dataset

Dataset	Model	Input	Acc.
Flowers	VGG	50x50	67.82%
	Scenario-1 (VGG)	227x227	68.03%
	Scenario-2 (VGG)	227x227	**69.17%**

Note that Table 10 compares the performances between single-CNN and double-CNN: double-CNN has more parameters and FLOPs than the single CNN. It makes sense to conduct comparative study based on equal hardware ground. To allay this concern, we have done experiments under the same FLOPs requirement between single-CNN and double-CNN on CIFAR100. As shown in Table 11, our method outperforms the ResNet-1001 by 1.2% of accuracy under the same FLOPs condition (i.e. 1.2B)

For future extension, some additional domain-driven systems are: (1) SR imaging tool (feature enhancement for low-resolution images), (2) Denoising tool (feature enhancement for noisy images), (3) Light-Enhancing tool (feature enhancement for night shots).

Table 11: Domain-driven SR classification on CIFAR100 dataset

Dataset	Model	Input	Acc.	#Params.	#FLOPs
ClFAR100	ResNet-1001 (He et at. 2016b)	32x32 LR	77.29%	10.2M	1.2B
	Domain-driven ResNet-164	64x64 SR	78.49%	2.1M	1.2B

Acknowledgement

The authors wish to thank Professor Feng Jiang (HIT), Professor Ying Li, and Dr. Mert Al, Yuchen Liu (Princeton University) for their invaluable suggestions and assistance.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017.

[2] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[3] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding, 2012.

[4] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2015.

[5] Ronald A Fisher. The statistical utilization of multiple measurements. *Annals of Eugenics*, 8(4), 1938.

[6] 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*, pages 1135–1143, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[8] Zehao Huang and Naiyan Wang. Data-driven sparse structure selection for deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 304–320, 2018.

[9] Thomas Kailath, Ali H Sayed, and Babak Hassibi. *Linear Estimation*. Prentice Hall, 2000.

[10] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *Online*: http://www.cs.toronto.edu/kriz/cifar.html, page 4, 2014.

[11] S. Y. Kung. *Kernel Methods and Machine Learning*. Cambridge University Press, 2014.

[12] S. Y. Kung. Compressive privacy: From information/estimation theory to machine learning [lecture notes]. *IEEE Signal Processing Magazine*, 34(1):94–112, 2017.

[13] S. Y. Kung. Discriminant component analysis for privacy protection and visualization of big data. *Multimedia Tools and Applications*, 76(3):3999–4034, 2017.

[14] S. Y. Kung, Zejiang Hou, and Yuchen Liu. Methodical design and trimming of deep learning networks: Enhancing external bp learning with internal omnipresent-supervision training paradigm. In *ICASSP 2019 – 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8058–8062. IEEE, 2019.

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

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

[17] Juncheng Li, Faming Fang, Kangfu Mei, and Guixu Zhang. Multi-scale residual network for image super-resolution. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 517–532, 2018.

[18] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In *Advances in Neural Information Processing Systems*, pages 345–353, 2017.

[19] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks

through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017.

[20] Yinglan Ma, Hongyu Xiong, Zhe Hu, and Lizhuang Ma. Efficient super resolution using binarized neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[21] David Martin, Charless Fowlkes, Doron Tal, Jitendra Malik, et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. ICCV Vancouver, 2001.

[22] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006.

[23] David B. Parker. Learning logic technical report tr-47. *Center of Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA*, 1985.

[24] C. Radhakrishna Rao. Large sample tests of statistical hypotheses concerning several parameters with applications to problems of estimation. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 44, pages 50–57. Cambridge University Press, 1948. MR0024111

[25] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958. MR0122606

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

[27] Claude Elwood Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948. MR0026286

[28] Oran Shayer, Dan Levi, and Ethan Fetaya. Learning discrete weights using the local reparameterization trick. *arXiv preprint* arXiv:1710.07739, 2017.

[29] Andrew R. Webb and David Lowe. The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis. *Neural Networks*, 3(4), 1990.

[30] Paul Werbos. Beyond regression: new tools for prediction and analysis in the behavioral sciences. *Ph.D. Dissertation, Harvard University*, 1974. MR2940620

[31] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *International Conference on Curves and Surfaces*, pages 711–730. Springer, 2010. MR2889812

[32] 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*, pages 883–894, 2018.

S. Y. Kung
Princeton University
Princeton
USA
E-mail address: kung@princeton.edu

Zejiang Hou
Princeton University
Princeton
USA
E-mail address: zejiangh@princeton.edu

Received January 31, 2020