Manuscript Details

Manuscript number	INFFUS_2019_534
Title	Unsupervised Bin-wise Pre-training: A Fusion of Information Theory and Hypergraph
Article type	Research paper

Abstract

Deep Neural Network - a multistep mathematical manipulation to process huge multidimensional data becomes inevitable in all the fields of engineering and technology as it guarantees higher accuracy and tractability. However, minimizing the training time of the Deep Neural Network still remain a significant challenge, as the parameters are huge. Training time can be minimized by optimizing and regularizing the parameters. To accomplish this, Pre-training is one of the promising techniques which is widely preferred by the researchers and considered to be a 'triggering point' of the Deep Neural Network. Though, recent research works focus on designing the efficient pre-training models, they often fail to capture the relevant information, and to maintain the stability of the learning model. Hence, this research article presents a novel unsupervised bin-wise pre-training model which fuses Information Theory and Hypergraph to speed up the learning process and to minimize the training-validation loss of the Deep Neural Network through improved feature representation. Further, a new approach of parameter updation during pre-training has been introduced that acts both as an optimizer and a regularizer. The proposed model has been evaluated using MNIST benchmark dataset and the experimental results confirm the effectiveness of the proposed unsupervised bin-wise pre-training model in terms of regularization & optimization capability and achieves competitive results compared to the state-of-the-art approaches.

Keywords	Deep Neural Network; Mutual Information; Information Theory; Partial Information Decomposition; Hypergraph

Corresponding Author	SHANKAR SRIRAM
Corresponding Author's Institution	SASTRA Deemed University
Order of Authors	Anila Glory H, Vigneswaran C, SHANKAR SRIRAM

Submission Files Included in this PDF

File Name [File Type]

Coverletter Information Fusion.pdf [Cover Letter]

Highlights.docx [Highlights]

IT-HG 24.7.19.docx [Manuscript File]

To view all the submission files, including those not included in the PDF, click on the manuscript title on your EVISE Homepage, then click 'Download zip file'.

23 – 07 – 2019

From

V. S. Shankar Sriram Ph.D,
Associate Dean, Computer Science and Engineering,
School of Computing,
SASTRA Deemed University,
Thanjavur – 613 401.
Tamil Nadu, India.

To

The Editor,
Information Fusion,
Elsevier.

Dear Sir,

Sub: Submission of the manuscript for publication– Reg.

I herewith submit the manuscript titled ***"Unsupervised Bin-wise Pre-training*: A Fusion of Information Theory and Hypergraph"** in your esteemed Journal. I also confirm the submission is original and is not being submitted for publication elsewhere.

Highlights:

- Information Theory based Hypergraph H_G construction was proposed for pre-training
- Novel parameter updation was introduced that performs optimization & regularization
- The K-helly property of H_G employed to restraint updation during pre-training
- Fusion of Information Theory and H_G improves pre-training & minimizes training time
- MNIST benchmark dataset used to evaluate the predominance of the proposed model

Thanking you

Yours Sincerely,

V. S. Shankar Sriram Ph.D
Email id: sriram@it.sastra.edu
Tel. No: +91 4362 264101 (2323)

Highlights:

- Information Theory based Hypergraph H_G construction was proposed for pre-training
- Novel parameter updation was introduced that performs optimization & regularization
- The K-helly property of H_G employed to restraint updation during pre-training
- Fusion of Information Theory and H_G improves pre-training & minimizes training time
- MNIST benchmark dataset was used to evaluate the predominance of the proposed model

Unsupervised Bin-wise Pre-training: A Fusion of Information Theory and Hypergraph

Anila Glory H[1], Vigneswaran C[1], Shankar Sriram V S[1*]
[1]Centre for Information Super Highway (CISH), School of Computing,
SASTRA Deemed University, Thanjavur, Tamil Nadu, India
sriram@it.sastra.edu[1*]

Abstract:

Deep Neural Network - a multistep mathematical manipulation to process huge multidimensional data becomes inevitable in all the fields of engineering and technology as it guarantees higher accuracy and tractability. However, minimizing the training time of the Deep Neural Network still remain a significant challenge, as the parameters are huge. Training time can be minimized by optimizing and regularizing the parameters. To accomplish this, *Pre-training* is one of the promising techniques which is widely preferred by the researchers and considered to be a '*triggering point'* of the Deep Neural Network. Though, recent research works focus on designing the efficient pre-training models, they often fail to capture the relevant information, and to maintain the stability of the learning model. Hence, this research article presents a novel *unsupervised bin-wise pre-training model* which fuses Information Theory and Hypergraph to speed up the learning process and to minimize the training-validation loss of the Deep Neural Network through improved feature representation. Further, a new approach of parameter updation during pre-training has been introduced that acts both as an optimizer and a regularizer. The proposed model has been evaluated using MNIST benchmark dataset and the experimental results confirm the effectiveness of the proposed *unsupervised bin-wise pre-training model* in terms of regularization & optimization capability and achieves competitive results compared to the state-of-the-art approaches.

Keywords: Deep Neural Network; Mutual Information; Information Theory; Partial Information Decomposition; Hypergraph;

1. Introduction:

Deep Neural Network (DNN) - an integral part of Deep Learning (DL) guarantees higher accuracy and flexibility by learning to represent "the world as a nested hierarchy of Information, with each defined in relation to simpler ones" [1]. Powerful features of DNN such as an increase in robustness and performance as the data increases, learning higher-level features from the data incrementally without feature engineering, end-to-end problem-solving capability, etc., make four among five researchers believe that the advent of DNN makes life easier [2]. However, parameter initialization is one of the major issues of DNN as it affects the rate of convergence and the generalization of the model [3–10].

To address this issue, pre-training is widely adapted in DNN as it helps in finding a better starting point in loss topology for improved Empirical Risk Minimization [11]. Pre-training is a process of adding new hidden layers for constructing a DL model and permitting the newly added layer to acquire the information from the preceding hidden layers [12]. Predominantly, Unsupervised Pre-training focuses on weight updation for

effective feature transformation and representation through layers, which reduces the high time-consuming exploration phase of the Optimization algorithm [13]. Among the existing unsupervised pre-training approaches Deep Belief Networks (DBN), Autoencoders and its variants are extensively used for pre-training [14]. However, the existing pre-training strategies suffer due to computational complexity and perform compression rather conceptualization.

Recent research works on understanding unfathomable concepts of DNN [15–20] to improve state-of-the-art methods through Information Theory has proven successful. Information Theory discloses how parameters are motivated to acquire the information from the known data and plausibly able to expound the trends observed during training [20]. This different perspective of viewing DNN helps us to answer, how model proceeds to optimize instead of a stochastic step. However, a very few have attempted to use Information Theory to solve the limitations of pre-training and yet they lack to exploit the complete significance of Information Theory.

To overcome the aforementioned issues, this paper proposes a novel pre-training model which fuses Information Theory and Hypergraph concepts in an unsupervised fashion. The novelty and major contributions of the proposed *unsupervised bin-wise pre-training model* are as follows:

1. Mutual Information and Partial Information Decomposition based Hypergraph construction has been proposed for pre-training
2. A novel parameter updation during pre-training has been introduced that performs both optimization & regularization and the proper justifications are provided
3. The K-helly property of hypergraph has been employed to restraint updation during pre-training
4. The MNIST benchmark dataset is used to evaluate the proposed model and the results have been compared with the traditional weight initialization techniques & other existing pre-training models

The rest of the paper is structured as follows: Section 1 discusses the background and importance of the proposed model with its novelty and contributions. Section 2 briefly portrays the recent and the traditional practices along with their pitfalls. Section 3 describes the necessary preambles of the proposed model. Section 4 reveals the importance and the intuition behind the proposed *unsupervised bin-wise pre-training model*. Section 5 deals with the experimental setup and presents several aids for showing the supremacy of the proposed model. Section 6 concludes the article along with the applications and the future direction.

2. Literature Review:

Owing to the difficulties faced while identifying the optimal model parameters using analytical methods, DNN can be viewed as a nonlinear optimization problem. Thus, optimal parameter initialization is the essence of DNN which helps to improve the

performance and to reduce the learning time. For suitable parameter selection two methods are extensively used by the researchers: (i) Least Squares Method (LSM) and (ii) Interval Analysis Method (IAM). The LSM takes the privilege of calculating accurate initial model parameters which reduce the initial error of the learning model. The research works reveal that there is a wide usage of LSM. Yam et al. [21] proposed a method based on linear algebra for weight initialization and the initialized weights are evaluated using LSM. Deniz Erdogmus et al. [22] applied a nonlinear LS problem with linear LS using backpropagation. Further, Deniz Erdogmus et al. [23] improved the Linear Least Square weight Initialization method for MLP via backpropagation through which the accuracy rate of the model has been increased. Liu et al. [24] employed partial LSM for the identification of appropriate number of hidden nodes and initial model parameters concurrently. Timotheou et al. [25] developed an approach to acquire linear equations with non-negativity constraints and introduced a projected gradient algorithm to find the solution for the obtained linear nonnegative least square problem. Though the aforementioned algorithms can reduce the initialization error efficiently, they often suffer due to local minima trap and fails to maintain the stability of the model.

On the contrary, Interval Analysis Method (IAM) identifies the appropriate range of initial model parameters in order to avoid premature convergence and to keep the hidden nodes active. The appropriate range is evaluated from diverse interpretations. In order to improve the convergence rate Drago et al. [26] calculated the maximum magnitude of the model parameters using statistical analysis. Thimm et al. [10] conducted several experimentations to find a suitable range of initial model parameters for deeper networks. Using Cauchy's inequality, Yam et al. [3] estimated the initial parameters and applied linear algebraic approach to ensure the correctness of the output nodes. Further Yam et al. [4] developed an approach to guarantee the full utilization of the activation function using multidimensional geometry. In 2008 [27] Yang et al. introduced a parameter initialization approach based on adjustment quantities. In 2014 [8] Adam et al. developed a linear interval tolerance model, in which the input of each hidden node should be in the active part of an activation function (sigmoid). Though the rate of accuracy has been improved, still these approaches couldn't mitigate the problem of local minima trap and the stability issues.

Further, a concept of pre-training is evolved in the field of deep learning (i) to evade local minima trap by reducing the exploration phase (ii) to increase the rate of convergence and (iii) to initialize optimal model parameters [14]. During pre-training, the model parameters are tuned and near optimal parameter values are identified. Thus pre-training can be considered as a kind of parameter initialization approach. Pre-training is broadly classified into two categories namely, (i) supervised pre-training and (ii) unsupervised pre-training [12]. The former refers to the pre-training of the whole neural network based on decision class labels, latter refers to the unsupervised learning phase using unlabelled samples followed by the supervised fine-tuning phase. The latter is widely preferred as it learns the information representation of the input data and uses the representation for supervised fine-tuning phase [28]. In 2010 [12] Dumitru Erhan et

al. analysed the need for unsupervised pre-training and concluded with many fruitful observations such as it (i) enhances the generalization, (ii) avoids the pitfalls of random initialization, and (iii) straighten out the impact of pre-training on local minima trap.

The recent surge of unsupervised DL concepts such as Stacked Autoencoders (SAE) and Deep Belief Networks (DBN) are commonly used for unsupervised pre-training [14]. However, the existing unsupervised pre-training approaches are used to reduce the computational burden, none of the pre-training techniques have effectively addressed the desired task. DBN is suitable for binary inputs and performs contrastive divergence & probability distribution. It has to perform mean field interference for every new input which is expensive [29]. Similarly, the main objective of SAE is conceptualization, still it fails to capture the relevant information rather it behaves as a compression model [30]. The goal of the pre-training model is conceptualization i.e., to extract significant information from the input data.

Subsequently, in recent days Mutual Information (MI), an important quantity of information theory gains the attraction of the researchers who are working in the field of Machine Learning [31]. From the term itself, the reader can infer that MI reveals the relationship between two different entities. In 2018 [18] Shrihari Vasudevan proposed dynamic hyperparameter (learning rate) tuning using Mutual Information. In the same year, Kairen Liu et al. [20] employed Mutual Information to visualize the importance of individual neuron and observed that, there is a positive correlation between MI and the classification performance of the learning model. Consecutively, Marylou Gabrie et al. [32] applied information theory concepts in deep neural networks and analysed the importance & deeds of MI and Entropy throughout the learning process. Nevertheless, none of the above research works has employed MI for pre-training.

Similarly, Partial Information Decomposition (PID) plays a vital role in identifying the contribution of single variable to categorize the target variable by segregating the complete state space as synergy, redundancy and unique which gains much popularity in recent days. In 2010 [33], Williams and Randall proposed a novel approach based on Partial Information Decomposition and formulated the new definition for redundancy to evade the negative decomposition of multivariate data. In 2017 [34], Daniel and Stefano extended the same concept in different aspect and derived several conclusions such as information gain or loss leads to unique decomposition and it provides depth interpretation about the synergy & redundancy. However, none of the researchers viewed the relationship between the neuronal interactions in PID perspective.

Successively in real time scenario, it is very challenging to model the multiple relationships that exist between the conditional attributes. There arises a need for a tool that expresses the n-ary relationship among the elements [35]. Hypergraph is a mathematical tool that describes the complex relationships between the attributes. The concept of Hypergraph and its properties (Hyper clique, Helly, Minimum Transversal etc.) are popular and attractive among the research community since it offers minimum time complexity [36]. Hypergraph has a wide range of applications in multiple domains

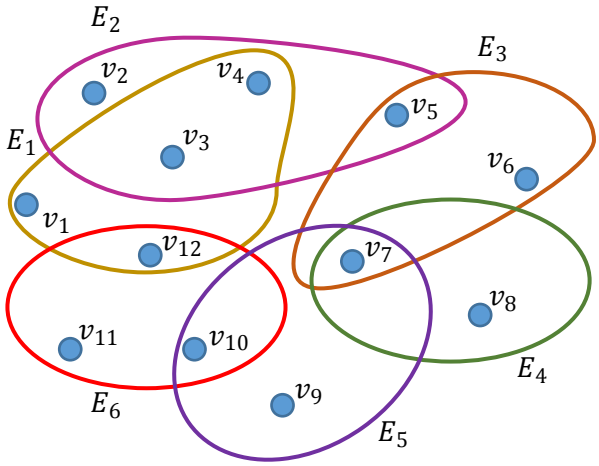
such as social network analysis, service-oriented architecture, system modelling, cloud service selection, intrusion detection systems etc. For instance, in 2017 Gauthama Raman et al. [37] proposed a feature selection model based on helly property of Hypergraph for Intrusion Detection Systems. In 2018, Nivethitha Somu et al. [38] employed helly property to identify trustworthy Cloud Service Providers. Though, the Hypergraph and its properties have many significant benefits none of the research works exploited Hypergraph for pre-training.

Thus, in order to design a pre-training model which learns the significant information from the input data and tunes the model parameters according to the learnt feature representation, this paper proposes a novel *unsupervised bin-wise pre-training model* which fuses the benefits of Hypergraph Concepts and Information Theory.

3. Preliminaries:

3.1 Hypergraph:

A graph is said to be a hypergraph (H_G), if it contains set of points denoting the vertices $V \leftarrow \{v_1, v_2, v_3, \ldots\ldots, v_m\}$ and the hyperedges $E \leftarrow (E_1, E_2, E_3, \ldots\ldots, E_n)$ denoted via a continuous curve linking two or more elements (Fig. 1.). Simply in other words, if the cardinality of each hyperedge is ≥ 2. H_G can also be defined as an incidence matrix $H = (h_i^j)$ with rows as vertices & columns as edges where $h_i^j = 0$ if $v_j \notin E_i$ & $h_i^j = 1$ if $v_j \in E_i$ (Fig.1.) [39]



	E_1	E_2	E_3	E_4	E_5	E_6
v_1	1	0	0	0	0	0
v_2	0	1	0	0	0	0
v_3	1	1	0	0	0	0
v_4	1	1	0	0	0	0
v_5	0	1	1	0	0	0
v_6	0	0	1	0	0	0
v_7	0	0	1	1	1	0
v_8	0	0	0	1	0	0
v_9	0	0	0	0	1	0
v_{10}	0	0	0	0	1	1
v_{11}	0	0	0	0	0	1
v_{12}	1	0	0	0	0	1

Fig. 1. Hypergraph Structure and its corresponding Incidence Matrix
$H_G(V,E); V \leftarrow \{v_1, v_2, v_3, \ldots\ldots, v_{11}, v_{12}\}; E \leftarrow (E_1, E_2, E_3, E_4, E_5, E_6, E_7)$
$E_1 \leftarrow \{v_1, v_3, v_4, v_{12}\}$; $E_2 \leftarrow \{v_2, v_3, v_4, v_5\}$; $E_3 \leftarrow \{v_5, v_6, v_7\}$
$E_4 \leftarrow \{v_7, v_8\}$; $E_3 \leftarrow \{v_7, v_9, v_{10}\}$; $E_3 \leftarrow \{v_{10}, v_{11}, v_{12}\}$

3.1.1 Helly Property:

A Hypergraph is said to have the helly property if the hyperedges of the hypergraph are having the non-empty intersection and it should not form the pair, sequence or loop (Fig.

2). Hypergraph which satisfies the helly property is termed as helly hypergraph (Algorithm 1) [40].

In other words, let $E_1, E_2, \ldots, E_n$ be the hyperedges of the hypergraph $H_G(V,E)$ if the intersection among the hyperedges $E_i \& E_j, \forall\, i,j \in \{1,2,\ldots,z\}, z \in Z$ is non-empty, then the helly property holds, where Z is a positive integer.

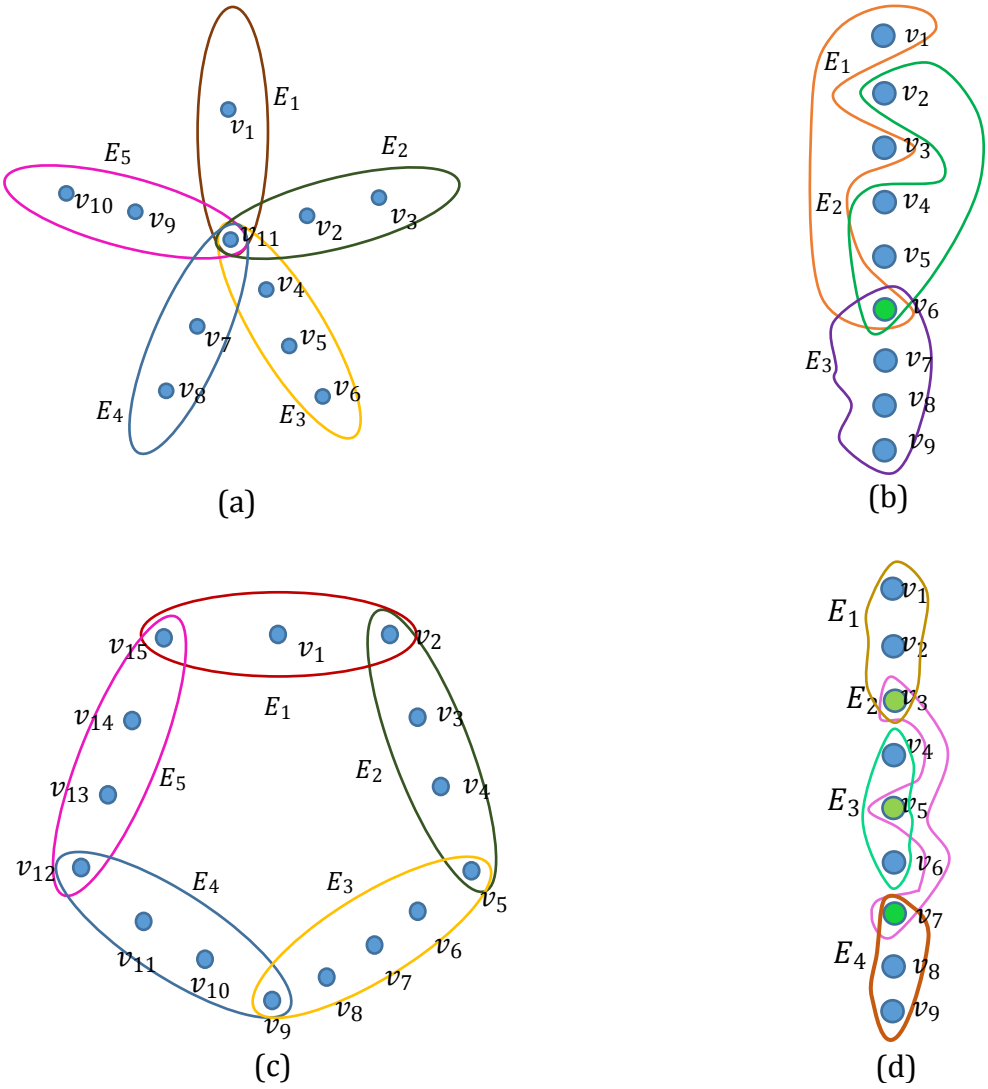


Fig. 2. (a) Hypergraph $H_G(V,E)$ with 5 hyperedges (E) and 11 vertices (V), v_{11} is the intersecting vertex of all the hyperedges (presence of helly property) (b) Illustration of Neurons as $H_G(V,E)$ which satisfies the helly property. (c) Hypergraph $H_G(V,E)$ with 5 hyperedges (E) and 15 vertices (V), there is no common intersection among all the hyperedges and it forms a loop (Lack of helly property) (d) Illustration of Neurons as $H_G(V,E)$ which violates helly property.

Algorithm 1:
Data: Vertices & Hyperedges of a Hypergraph $H_G(V,E)$
$\forall$ *pairs of vertices* V_a *and* V_b *of* $H_G(V,E)$ *do*
$X_{V_aV_b} :\rightarrow$ *the hyperedges containing both* V_a *and* V_b
$\forall$ *vertices* v *of* $H_G(V,E)$ *do*
If vertices V_a *and* V_b *are neighbours of* v *then*
$X_{V_aV_v} :\rightarrow$ *all* E *containing* V_a *and* V_v
$X_{V_bV_v} :\rightarrow$ *all* E *containing* V_b *and* V_v
$X :\rightarrow X_{V_aV_b} \cup X_{V_aV_v} \cup X_{V_bV_v}$
If $\cap X \neq \emptyset$ *then*
$H_G(V,E)$ *holds the* ***helly property***
End
End

3.1.2 K-Helly Property:

A hypergraph $H_G(V,E)$ is said to be K-helly iff $\forall$ vertices of set U with $|U| = k + 1$, the intersection between the hyperedges E_i of the hypergraph with $|E_j \cap U| \geq k \neq \emptyset$ i.e., the intersection of the vertices between the hyperedges should be a non-empty core.

A hypergraph $H_G(V,E)$ is said to be K-helly $\forall L \subset \{1,2,\ldots, z\}$, the following conditions should hold, $J \subset L$, $|J| < k \rightarrow \cap_{j \in J} \neq \emptyset$ & $\cap_{l \in L} \neq \emptyset$.

3.2 Mutual Information (MI):

MI [41] is a significant quantity of Information Theory, which measures the dependency among two random variables and is generally calculated using Eqn. 1,

$$I(D_y; X) = H(D_y) - H(D_y \mid X) \tag{1}$$

Where, $H(D_y)$ is the uncertainty (entropy) of the variable D_y and is computed using Eqn.2,

$$H(D_y) = -\sum_{d_y} p(d_y) \log\left(p(d_y)\right) \tag{2}$$

Also, the conditional entropy given X is determined by Eqn. 3,

$$H(D_y \mid X) = -\sum_{x}\sum_{d_y} p(x, d_y) \log\left(p(d_y|x)\right) \tag{3}$$

Inference:

(i) If $I(D_y, X) = 0$, X & D_y are independent random variables
(ii) If $I(D_y, X) > 0$, X & D_y are dependent among each other

MI can also be computed based on the distance measure Eqn. 4,

$$KL(F \mid\mid G) = \int F(d_y) \log\left(\frac{F(d_y)}{G(d_y)}\right) = E_F\left[\log\left(\frac{F(d_y)}{G(d_y)}\right)\right] \quad (4)$$

Where, KL denotes the Kullback–Leibler divergence among two entities. F & G represents the probability distribution. E_F indicates the Expectation with respect to F. Mutual Information is equivalent to Kullback–Leibler divergence between joint probability distribution and Marginal distribution. Thus MI can be written as Eqn. 5,

$$I(D_y; X) = \sum_x \sum_{d_y} p(x, d_y) \log\left(\frac{p(x, d_y)}{p(x).p(\, d_y)}\right) \quad (5)$$

Among the existing MI estimation techniques, three of them which are relevant to the context are discussed briefly in Appendix A.

3.3 Partial Information Decomposition

Partial Information Decomposition explains how a single variable provides information about another variable by means of Synergistic, Unique and Redundant information as in Fig. 3 [33]. Let X_1, X_2, X_3 are the predictor variables (neuron activations) from which the target variable Y (input data) has been predicted. Mutual Information $I(Y;X_1)$ (Eqn.6) $I(Y;X_2)$ (Eqn.7) & $I(Y;X_3)$ (Eqn.8) denotes the information which are individually provided by X_1, X_2 & X_3 respectively.

$$I(Y;X_1) \rightarrow Ue(Y;X_1) + Ry(Y;X_1,X_3) + Ry(Y;X_1,X_2) + Ry(Y;X_1,X_2,X_3) \quad (6)$$

$$I(Y;X_2) \rightarrow Ue(Y;X_2) + Ry(Y;X_2,X_3) + Ry(Y;X_1,X_2) + Ry(Y;X_1,X_2,X_3) \quad (7)$$

$$I(Y;X_3) \rightarrow Ue(Y;X_3) + Ry(Y;X_1,X_3) + Ry(Y;X_2,X_3) + Ry(Y;X_1,X_2,X_3) \quad (8)$$

The Unique, Redundant and Synergistic information are denoted by $Ue(Y;X)$ (Eqn.9), $Ry(Y;X)$ (Eqn.10) and $Sy(Y;X)$ (Eqn.11) respectively.

$$Ue(Y;X) \rightarrow Ue(Y;X_1) + Ue(Y;X_2) + Ue(Y;X_3) \quad (9)$$

$$Ry(Y;X) \rightarrow Ry(Y;X_1,X_2) + Ry(Y;X_1,X_3) + Ry(Y;X_2,X_3) + Ry(Y;X_1,X_2,X_3) \quad (10)$$

$$Sy(Y;X) \rightarrow Sy(Y;X_1,X_2) + Sy(Y;X_1,X_3) + Sy(Y;X_2,X_3) + Sy(Y;X_1,X_2,X_3) \quad (11)$$

Similarly, joint MI $I(Y;X_1, X_2)$, $I(Y;X_2, X_3)$, & $I(Y;X_1, X_3)$ measures the information jointly provided by (X_1, X_2) Eqn. 12, (X_2, X_3) Eqn. 13, and (X_2, X_3) Eqn. 14 together towards the target variable Y.

$$I(Y;X_1, X_2) \rightarrow Ue(Y;X_1) + Ue(Y;X_2) + Ry(Y;X) + Sy(Y;X_1,X_2) \quad (12)$$

$$I(Y;X_2, X_3) \rightarrow Ue(Y;X_2) + Ue(Y;X_3) + Ry(Y;X) + Sy(Y;X_2,X_3) \quad (13)$$

$$I(Y;X_1, X_3) \rightarrow Ue(Y;X_1) + Ue(Y;X_3) + Ry(Y;X) + Sy(Y;X_1,X_3) \quad (14)$$

Eqn. 15 sums up the overall information.

$$I(Y;X_1, X_2, X_3) \rightarrow Ue(Y;X) + Re(Y;X) + Sy(Y;X) \quad (15)$$

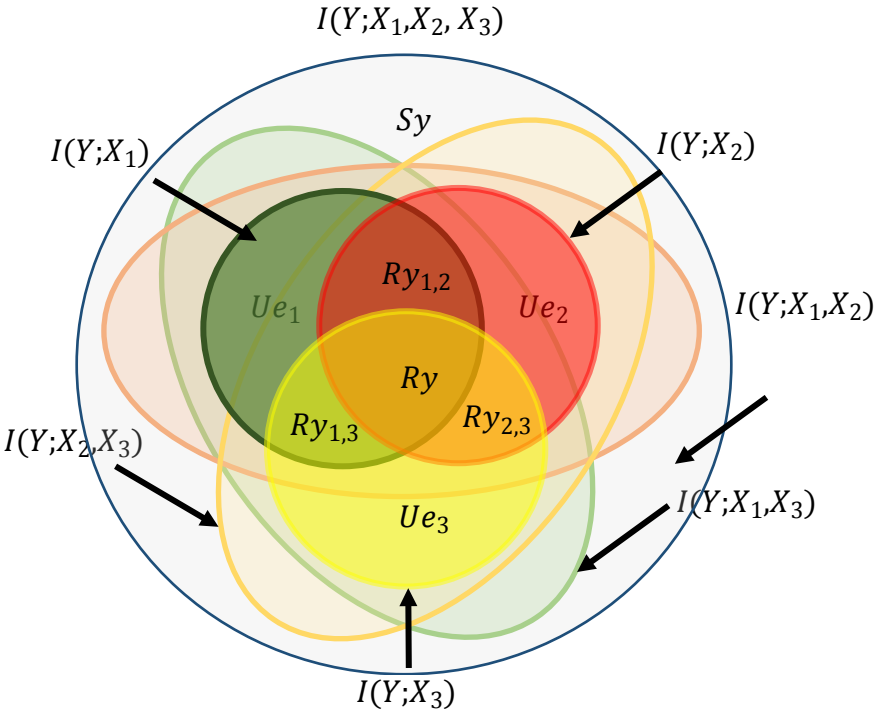


Fig. 3. depicts the erection of multivariate information which contains four variables. $Sy \rightarrow Sy(Y;X_1,X_2,X_3)$ denotes the synergy; $Ry \rightarrow Ry(Y;X_1,X_2,X_3)$, $Ry_{1,2} \rightarrow Ry(Y;X_1,X_2)$, $Ry_{2,3} \rightarrow Ry(Y;X_2,X_3)$, $Ry_{1,3} \rightarrow Ry(Y;X_1,X_3)$ represents the redundancy; $Ue_1 \rightarrow Ue(Y;X_1)$, $Ue_2 \rightarrow Ue(Y;X_2)$, $Ue_3 \rightarrow Ue(Y;X_3)$ indicates the unique information.

4. Proposed Methodology

This section discusses the intuition behind the fusion of Information Theory and Hypergraph concepts to pre-train the DL model for optimizing and regularizing the parameters. Among the existing pre-training techniques, layer-wise pre-training is more effective as it freezes the parameters of the hidden layers by dividing the entire training process into layer-wise training and aggregates the local optimum solutions to obtain the global best solution. From Information Theoretic perspective different layers are characterized by their own Information Theoretic measures and becomes incomparable with other layers. Further each neurons in a particular layer has distinct role for effective attainment of optimum solution. Hence in the proposed *unsupervised bin-wise pre-training* model, at every layer of DNN depending on MI & PID, hypergraph has been constructed and the bin-wise freezing takes place. In optimization perspective, the proposed pre-training model involves updation of initial model parameters to reduce the tortuous exploration and helps to start fine-tuning phase from the region closer to the

sensed optimal region. In regularization perspective, it maintains the variance thereby improving the generalization of the deep learning model. The working of the proposed *unsupervised bin-wise pre-training model* consists of four predominant phases namely (i) Estimation of Mutual Information (MI), (ii) Selection of number of Hyperedges (bins) using Partial Information Decomposition (PID), (iii) Construction of Hypergraph (H_G) based on MI & PID, and (iv) Updating the parameters using a novel weight update rule during pre-training. The workflow and the algorithm (Algorithm 2) of the proposed pre-training model has been depicted in Fig. 4.

4.1 Parameter Initialization:

Initialization of model parameters is an important step, as it influences the effectiveness and correctness of DNN. Moreover, Initializing parameters without much care shall cause Vanishing or Exploding gradient problem. In this work, Uniform Random Initialization of parameters has been employed. The initial model parameters are calculated using Eqn. 16,

$$\sigma = \frac{1}{\sqrt{V^i}} \tag{16}$$

Where, V^i denotes the number of neurons in the i^{th} layer. W^i & B^i are the weights and biases of the i^{th} layer respectively. $W^i \in U(-\sigma,\sigma)$ & $B^i \in U(-\sigma,\sigma)$, where U is the uniform distribution.

This bare minimum method is preferred over other improved Weight Initialization methods [42–44] as it helps us to estimate the actual amount of improvement obtained by the proposed pre-training model in means of both Optimization and Regularization.

4.2 Estimation of Mutual Information:

Estimation of Mutual Information is a computationally challenging task and it is essential to choose a simple & effective estimation method. In this work, Histogram-Based Estimation (Appendix A) method has been exploited which results in a considerably small amount of Estimation Error that can be safely neglected. Usually, pre-training is performed using larger amount of data. Whereas, in this work the traditional practice has been contradicted and the model is pre-trained using relatively smaller amount of data (Validation dataset $(V_D = \{x_1, x_2, x_3, \ldots, x_v\})$), assuming it would be the representative of the entire dataset. The intuition behind is that, the main purpose of the proposed work is to capture more amount of information which in turn improve the better selection of unique features in fine-tuning phase. Moreover using larger general data may obscure the Mutual Information obtained and reduces the efficiency of the proposed model. As the proposed method is unsupervised, the decision class labels were removed from V_D.

Let $S_j^i(x_k) = \mathfrak{H}(\sum_t W^i{}_{t,j}^{-1}.S^{i-1}_t(x_k) + B_j^i)$ where, $S_j^i(x_k)$ denotes the activation of j^{th} neuron in i^{th} layer when x_k (k^{th} input sample) is given as input, $W^i{}_{t,j}^{-1}$ denotes the weight connecting from t^{th} neuron in $i-1^{th}$ layer to j^{th} neuron in i^{th} layer, B_j^i denotes the bias of j^{th} neuron in i^{th} layer. It is computationally infeasible to compute MI between the

input data and the individual neuron activations as the dimensions are not equal. Thus, in the proposed work the MI is estimated between the each dimension of the input data (V_D), against the neuron activations $S_j^i(V_D)$ and the summation of MI in each dimension is equal to the *actual MI* between the input data and the neuron activations.

Mathematically, the MI between the two entities is not equal to the summation of MI between each dimension d of the single entity with the other entity (Eqn. 17) as the dependency is not taken into account.

$$I(X,Y) \neq \sum_{i=1}^{d} I(X_i,Y) \tag{17}$$

However, if there is no interdependency among the dimensions of the entity the above mentioned is true.

For instance,

(i) If $X \rightarrow [X_1, X_2]$ & $X_2 = X_1$, then MI between X & Y is $I(X_1,Y)$ and not $I(X_1,Y) + I(X_2,Y) = 2I(X_1,Y)$

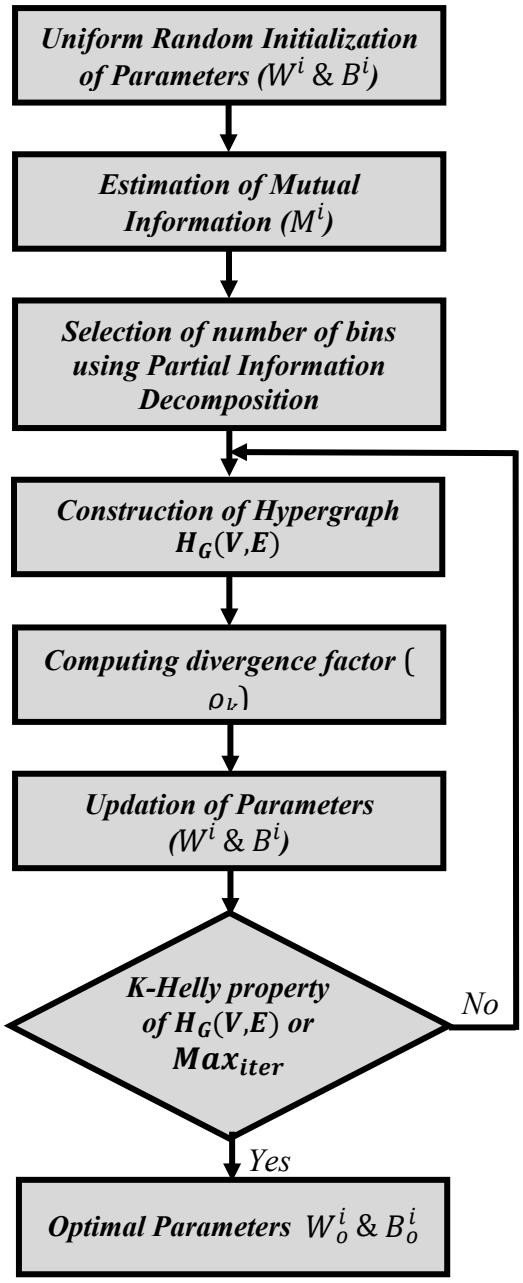
(ii) If $X \rightarrow [X_1, X_2]$ & X_2 is independent on X_1, then MI between X & Y is $\sum_i I(X_i,Y)$ i.e., $I(X_1,Y) + I(X_2,Y)$

The input data V_D have d dimensions which are independent to each other. Therefore Eqn. 18 is valid.

$$I(X,Y) = \sum_{i=1}^{d} I(X_i,Y) \tag{18}$$

4.3 Selection of number of Hyperedges (bins) using PID:

Selecting number of bins of a Hypergraph for a particular layer is a crucial step which improves the efficiency of the proposed model. To accomplish this, the concept of Partial Information Decomposition (PID) is found to be a promising mathematical model which captures the exact dependency between the entities in the language of Information Theory [33] (Section 3.3). Construing PID in Deep Learning perspective many interesting observations are noted [20,33]. One of the seminal observations is that the neurons in the shallow layer provide Information which are synergistic and are very less class specific, by making each neuron in the shallow layer act as an information source for encoding general features, whereas deeper layer neurons provide information which are class specific unique & redundant. This statement is also verified in [20], as pruning of neurons in shallow layer brings down the model performance than pruning in deeper layers. The selection of the number of bins is commensurate to the amount of updation to force neurons to acquire more Mutual Information from input data. Hence having a greater number of bins in shallow layer than in deeper layer is desirable. Therefore, a new hyperparameter $\mathscr{b}^i$ has been defined and larger $\mathscr{b}^i$ is assigned for shallow layers as layer deepens, $\mathscr{b}^i$ is reduced accordingly.



Algorithm 2:

$n_h \rightarrow$ *Number of hidden layers*
$\mathfrak{H} \rightarrow$ *Activation function*
$m^i \rightarrow$ *Number of hidden neurons*
$\mathscr{b}^i \rightarrow$ *Number of bins in* i^{th} *layer*
$\varrho_k \rightarrow$ *Divergence Factor of* k^{th} *bin*

For each $h^i, i \rightarrow 1,2,\ldots,n_h$
  Initialize W^i & B^i *based on Section* 4.1
  $S^i \rightarrow \mathfrak{H}(W^{i-1}.S^{i-1} + B^i)$
  For each V_j^i, $j \rightarrow 1,2,\ldots, m^i$
    Calculate MI (M_j^i) *using Eqn.* 18
  Fix $\mathscr{b}^i$ *based on Section* 4.3
  /*Bin-wise pre-training begins*/
  $Iter \rightarrow 1$
  /*Construction of Hypergraph*/
  Create hyperedges (bins) E^i *based on* M^i
  While $Iter < Max_{iter}$ && !K_Helly (E^i)
    For each E_k^i, $k \rightarrow 1, 2, \ldots, \mathscr{b}^i$
      Calculate ϱ_k *based on Section* 4.5
      Update W^i & B^i *based on Eqn.* 19 & 20
    Calculate $\widehat{M}^i$ *using Eqn.*16
    Update bins E^i *based on* $\widehat{M}^i$
    $Iter \rightarrow Iter + 1$
Return W_o^i & B_o^i

Fig. 4. The Proposed Workflow and the Algorithm

4.4 Construction of Hypergraph:

The number of bins $\mathscr{b}^i$ for each hidden layer i has been identified from which Hypergraph is constructed. The neurons for each bin has been decided using the MI (M_j^i) between the *activation neurons* $(S_j^i(V_D))$ and the *input data* (V_D), where $j \rightarrow 1,2,\ldots, m$ denotes the neurons present in the i^{th} hidden layer. In order to construct the Hypergraph $H_G(V_j^i, E_k^i)$, $k \rightarrow 1,2, \ldots, \mathscr{b}^i$; neurons present in the hidden layer is viewed as vertices (V_j^i) of a hypergraph and neurons are grouped together as bins (E_k^i) based on M_j^i (Section 3.2.1). The construction of the initial hypergraph and the movement of neurons are illustrated in Fig. 5. and Fig. 6. respectively. For each bin, based on M_j^i the divergence factor (ϱ_k) has

been computed while updating the parameters, which quantifies the amount of deficient MI which E_k^i has to further acquire.

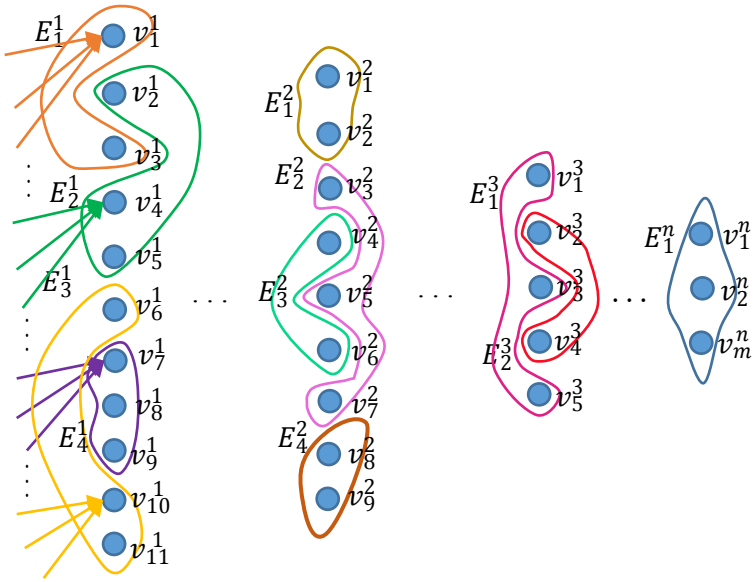


Fig. 5. depicts the formation of hypergraph by considering the neurons of each layer as vertices and the grouping of neurons as hyperedges (bins) based on Mutual Information. For each coloured bins in a particular layer, the parameters (respective coloured arrow heads) corresponding to the particular bin have been updated collectively based on the divergence factor.

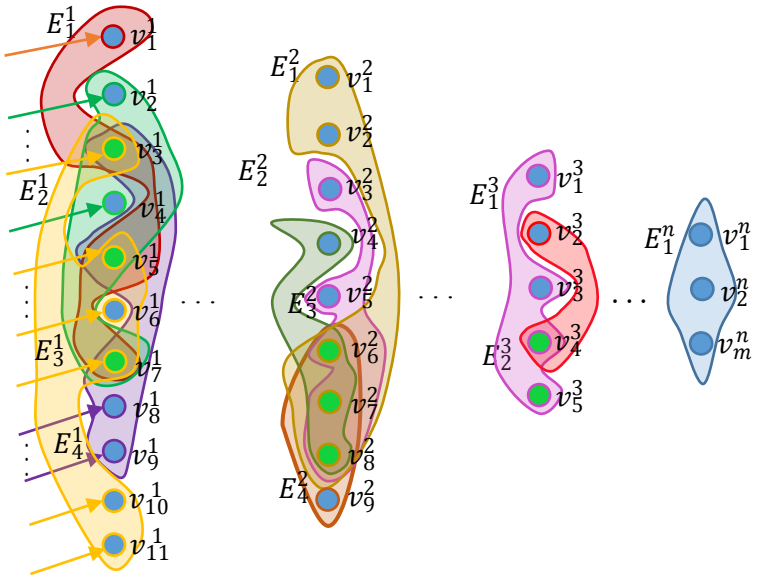


Fig. 6. depicts the intersection of the bins after a specific number of iterations. The intersection happens when a particular neuron from a bin has been improved by updation and moved to a better bin (better average MI). For instance, if the bin E_4^1, has the highest average MI and as the updation proceeds, the neurons v_3^1 , v_5^1,& v_7^1 from E_1^1, E_2^1 , & E_3^1 respectively moves to E_4^1, (green coloured neurons) as, there is an upsurge in their

MI values. The same divergence factor is assigned for v_3^1, v_5^1,v_6^1, v_7^1, v_{10}^1, & v_{11}^1 and the updation proceeds until the termination condition is reached.

4.5 Parameter Updation

Parameter updation is performed to maximize the Mutual Information (Objective function) between the Input data (V_D) and Neuron activations ($S_j^i(V_D)$). Let each bin ($E^i{}_1$, $E^i{}_2$, $E^i{}_3$, ... ,$E^i{}_{\beta^i}$) in the i^{th} layer have average MI ($\mathcal{M}_1^i$, $\mathcal{M}_2^i$, $\mathcal{M}_3^i$,....,$\mathcal{M}_{\beta^i}^i$) which are contributed by their respective member neurons. Parameters of the particular bins are updated collectively with common ϱ_k to obtain improved MI $\widehat{M_j^i}$ using Eqn.19 and Eqn.20,

$$W_{k,j}^{new} \rightarrow W_{k,j}^{old} - \varrho_k^i * \Lambda * \left(\frac{1}{iter+1}\right) \tag{19}$$

$$B_{k,j}^{new} \rightarrow B_{k,j}^{old} - \varrho_k^i * \Lambda * \left(\frac{1}{iter+1}\right) \tag{20}$$

Where $W_{k,j}$ denote the weights that influencing the activation of j^{th} neuron in k^{th} bin and $B_{k,j}$ denotes the bias of the j^{th} neuron in k^{th} bin. Divergence factor (ϱ_k^i) of the bin (E_k^i) quantifies the distance between the bin with the best average MI $(E_{best}^{\ i})$ and the respective bin (E_k^i). This divergence metric can be effectively estimated using KL-Divergence (Section 3.2) and hence, $\varrho_k^i = KL(E_k^i \,||\, E_{best}^{\ i})$. The factor $(\Lambda * \left(\frac{1}{iter+1}\right))$ decays the amount of updation to ensure better convergence. Hyperparameter Λ can be set depending on the particular layer and the network architecture. The bin-wise pre-training on the particular layer has been continued until K-helly property of hypergraph is satisfied (Section 3.1.2). In other words, once the K neurons with M_j^i from lower average MI bins' move to the best average MI, pre-training of the specific layer has been terminated. Selecting suitable K, Max_{iter} and Λ during the parameter updation effectively balances both optimization and regularization, leading to a better model.

5. Experimental Setup and Discussions:

5.1 Experimental Setup:

In order to examine the effectiveness of the proposed *unsupervised bin wise pre-training model* in terms of both as the Optimizer and Regularizer, the MNIST benchmark dataset was used. The MNIST dataset consists of handwritten digits, which are widely preferred by the researchers to evaluate their proposed models. The dataset comprises of 70,000 handwritten digit images, out of which 60,000 and 10,000 images were used for training (TR_D) and testing (TE_D) respectively. The training dataset (TR_D) was sub-divided into training dataset (T_D) and validation dataset (V_D) in the ratio 80:20 (48,000:12,000). DNN was configured with an input layer consisting of 784 (28 x 28 pixels) input neurons, five hidden layers comprising 1024, 200, 20, 20, 20 neurons in each layer respectively and an output layer with 10 output neurons which corresponds the class labels of the MNIST dataset. Using less number of architectural entities is desirable as it makes the evaluation process simple and efficient. Softmax was used as the activation function for the output layer. Cross entropy loss was employed as the objective function and Adam optimizer was

applied. The models were implemented in Python3 using PyTorch library. In addition, NumPy and Matplotlib were used to perform matrix operations and plotting respectively. All the experimentations were carried out in the machine with 6 GB of Primary Memory, Intel i3 dual core CPU with 1.7 GHz and 64-bit Debian operating system.

5.2 Discussions:

The goal of this research article is to develop a parochial pre-training model which initializes near optimal parameters by forming bins based on the MI between the input data and the neuron activations. Further, the number of bins was selected using partial information decomposition and the parameters were updated using the novel weight update rule during pre-training. This way of pre-training ensures the high possibility of reaching the near global optimal position. To show the supremacy of the proposed model in terms of stability, convergence rate, the comparisons were made with state-of-the-art weight initialization methods and pre-training techniques.

5.2.1 Comparison with other Weight Initialization methods:

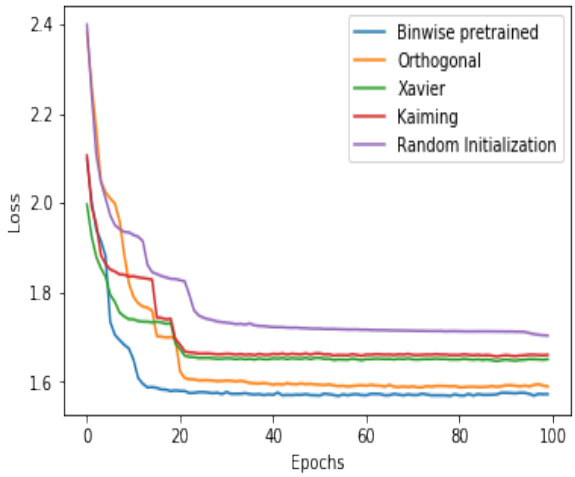


Fig. 7. (a)

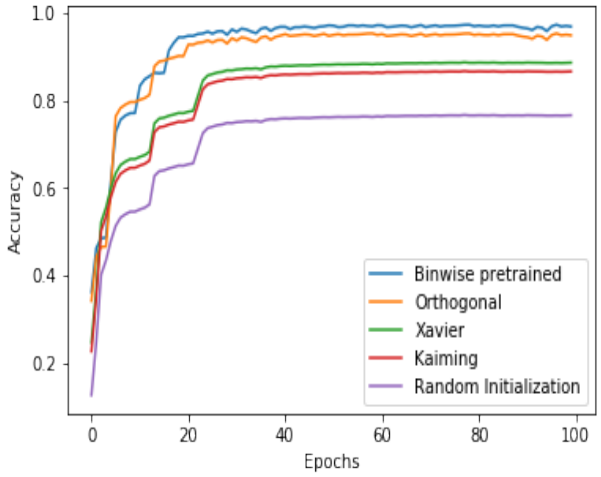


Fig. 7. (b)

Fig. 7. (a) shows the validation loss of various Weight Initialization methods and Fig. 7. (b) presents the Validation accuracy, when activation function was assigned as Sigmoid.

To check how effective the parameters obtained from *unsupervised bin-wise pre-training model* can effectively locate the model near to the potential local minima and results to faster convergence, the proposed work was compared with various Weight Initialization methods. Well-known and effective Weight initialization methods including Xavier [43], Kaiming [44] and Orthogonal methods [42] were taken into account for comparison. The evaluation process was carried out with Sigmoid and ReLU activation functions separately, due to disparity in their characteristic property on Neuron's output and deviant behaviour when estimating Information Theoretic quantities. Fig. 7. & Fig. 8. elucidates the performance of various Weight Initialization methods when Sigmoid and ReLU were configured as activation functions respectively. Irrespective of the activation functions it was cogent that, the proposed model clearly outperforms the other weight initialization methods and converges faster.

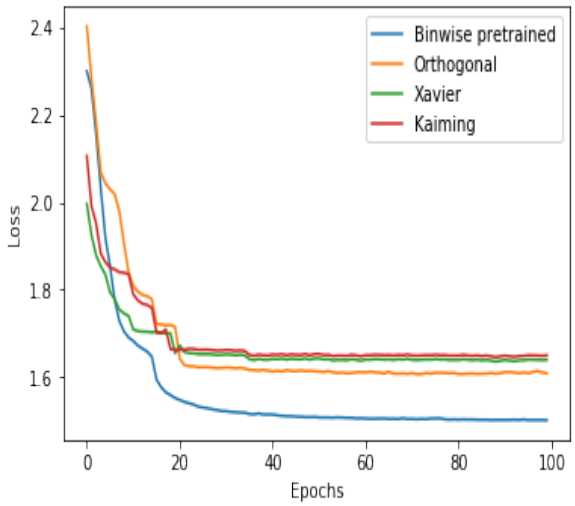


Fig. 8. (a)

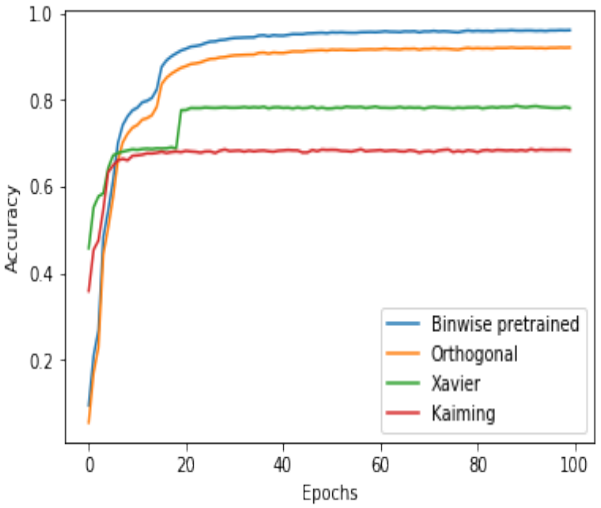


Fig. 8. (b)

Fig. 8. demonstrates the performance of the proposed model in terms of (a) Validation loss and (b) Validation accuracy when Weight Initialization methods are set with ReLU as activation function.

5.2.2 Comparison with other Pre-training techniques:

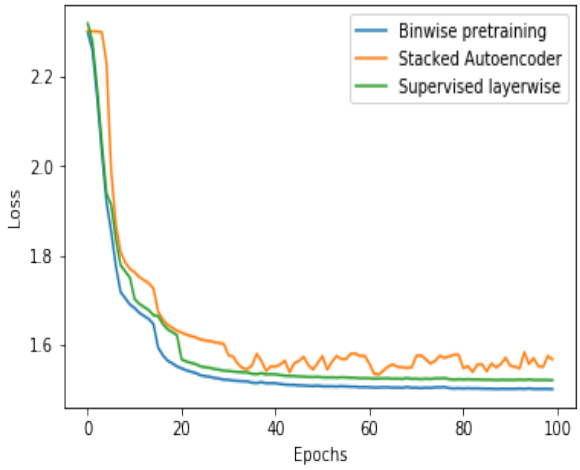


Fig. 9. (a)

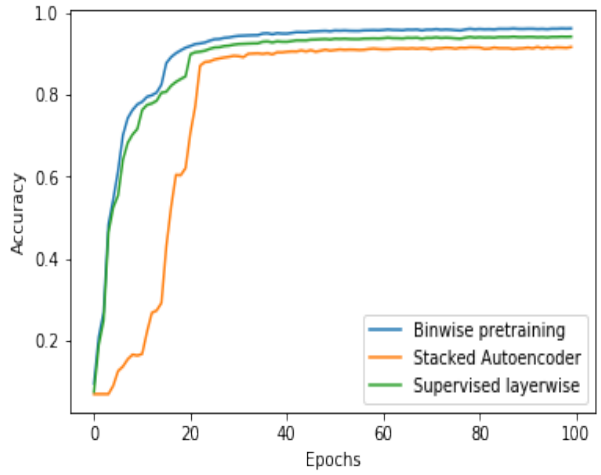


Fig. 9. (b)

Fig. 9. shows the performance comparison of the proposed model in terms of (a) Validation loss and (b) Validation accuracy with the existing pre-training models

The proposed bin-wise pre-training model was compared with the existing pre-training models: Stacked Autoencoders and Supervised Greedy layer-wise pre-training model. Stacked Autoencoder was configured with three stacks of encoder-decoder, whereas Supervised Greedy layer-wise pre-training model was shaped with six layers added sequentially. From Fig. 9. (a) & (b) It was observed that the proposed unsupervised bin-wise pre-training model dominates the other two techniques. Although, there was a minuscule difference inferred between the proposed and the existing techniques, an ample difference would be observed when the non-linearity of data increases, in which language of Information Theory is quite dominant.

5.2.3 Regularization Capability:

To evaluate the performance of the proposed work in Regularization perspective, the distribution of randomly initialized weights and weights after bin-wise pre-training were recorded and the change imbibed on them due to bin-wise pre-training were analysed. It was inferred that the weight boundaries were maintained within $\pm$ 0.04 and the variance of both the distribution remains the same throughout the pre-training process which confirms that the proposed model behaves as a good regularizer. Fig. 10. shows the comparison of distribution of randomly initialized weights and pre-trained weights. Fig. 11. depicts the pattern of weight updation during pre-training process and it was inferred that the new weights obtained had the desirable distribution, thus making the Deep Learning model simple with better Generalization behaviour along with less susceptible to overfitting and Vanishing or Exploding gradient problems.

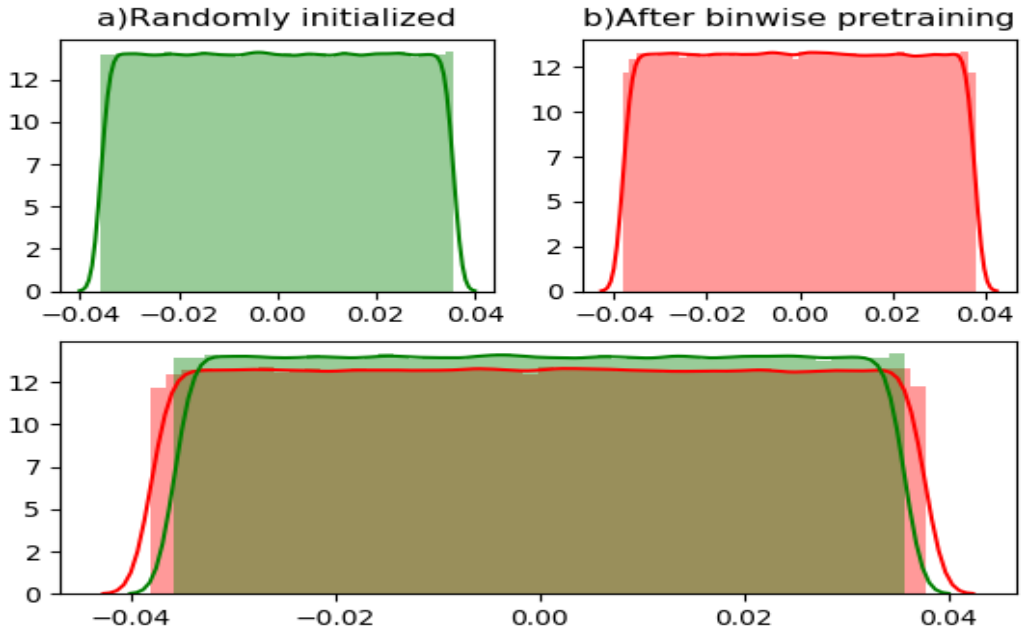


Fig. 10. Weight Distribution Plot

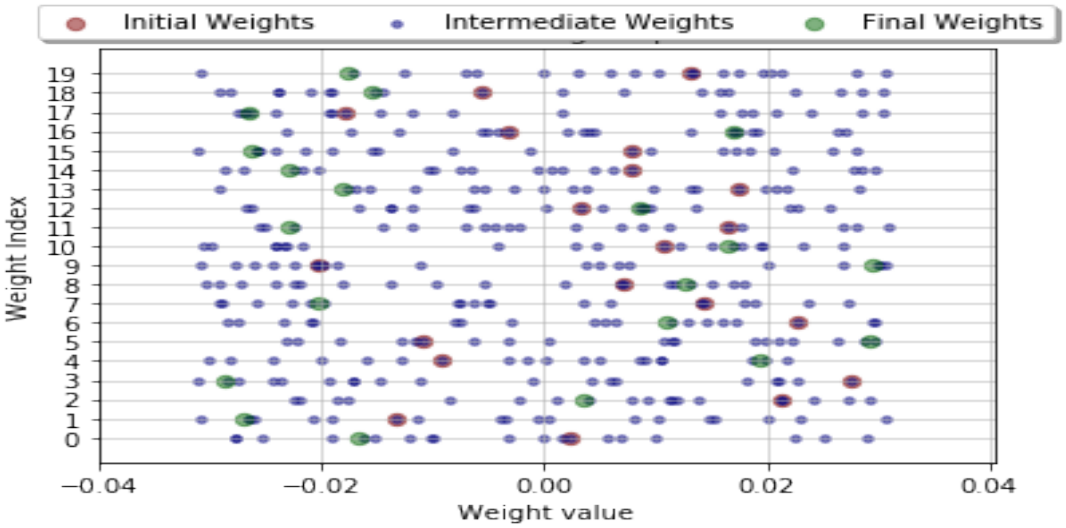


Fig. 11. Weight updation pattern of randomly selected twenty weights for twenty iterations during pre-training process.

5.2.4 Stability:

Quantifying the stability mirrors the robustness of the proposed pre-training model. To evaluate the stability of the proposed work, the model was assessed for twenty times and the variance of them was turned to be 0.3942. Thus, the proposed work provides consistent outcomes and maintains the efficacy.

5.2.5 Convergence Rate:

Rate of convergence is an important metric that measures the ability of the model to converge in better minima with reasonable amount of time. From Fig. 7, 8 & 9 it has been inferred that the rate of convergence for all the existing approaches were mostly similar at the beginning and as the iteration proceeded the proposed *unsupervised bin-wise pre-training model* converges earlier than others. Selecting appropriate number of bins using Partial Information Decomposition alters the degree of change and directly influences the convergence rate. Fig. 12. shows the increase in Mutual Information between j^{th} Neuron's output (S_j^i) and Input data (V_D) as the iterations were proceeded.

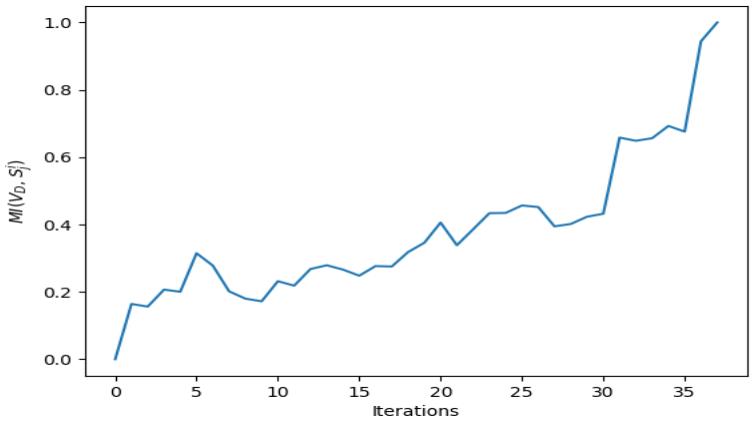


Fig. 12. Mutual Information Vs Number of iterations

5.2.6 Initialization Time:

Initialization time for the proposed bin-wise pre-training was observed to be more than the existing pre-training techniques. The main reason for such an increased initialization time is due to the estimation of Mutual Information between Input data with large dimension and Neuron output. However, the computational burden can be optimized using Graphical Processing Unit (GPU) parallelization with the help of library function like Numba. Although the usage of hardware acceleration helps in mitigating the computational time, yet the model becomes more infrastructure dependent. To make the model simple and consume less computational time, in the proposed work Fast-histogram library was employed, which uses native CPU and C extensions to compute histograms for estimating Mutual Information [45]. This reduces the computational time than Numba. It was also observed to be 15 times faster than Numpy's 1D histogram and 25 times faster than Numpy's 2D histogram. Therefore without using any sophisticated

hardware, the difference in initialization time of the proposed work and other pre-training techniques were considerably small.

6 Conclusions:

The recent surge in the field of deep learning paves the way for the evolution of *pre-training*. In order to minimize the training time and computational overhead of the deep learning model, pre-training is indispensable. Predominantly used pre-training techniques often fail to extract important feature representations and endure stability problem. Hence in this work, an attempt was made to mitigate the aforementioned issues using a novel *unsupervised bin-wise pre-training model* which fuses Information Theory and Hypergraph concepts. The performance graphs edify the predominance of the proposed *unsupervised bin-wise pre-training model* over the other existing approaches and found to be simple, stable and computationally attractive. However the proposed work may take longer initialization time when the dimension of input data is very large, for which sophisticated hardware shall be used to ameliorate the computational time. As a future direction, the work can be further extended by utilizing the concepts of Information Bottleneck and effective ways to identify the information accumulation in network. Further the proposed pre-training model can be used along with the other variants of Deep Neural Networks for various applications such as pattern recognition, computer vision, natural language processing etc.

Acknowledgements

This work was supported by The IBM Shared University Research Grant 2017

Conflict of Interest

All the authors declare that they do not have any conflict of interest.

Appendix A

Histogram-Based Estimation (HBE):

Though MI is a predominant Information Theoretic quantity, calculation of MI is crucial as it involves complex computation. Among the estimation methods, HBE is one of the most straightforward and extensively preferred approaches [46]. Let x_i & y_i,$i \rightarrow 1, 2, \ldots, Z$ are the random variables of instantaneous quantities from a collection of Z. Let us consider an origin o and a width of bars of the histogram as b_h for the random variable x which lies between the intervals $[o + mb_h, o + (m + 1)b_h]$, $m \rightarrow 1, 2, \ldots, M$., thus the data are segregated into M discrete bins b_i and the number of quantities is represented as k_i which lies within b_i. The probabilities of relative frequencies of the event are calculated using Eqn. 21,

$$p(b_i) = \frac{k_i}{Z} \tag{21}$$

And the MI $I(X,Y)$ can be computed using Eqn. 22,

$$I(X,Y) = \log Z + \frac{1}{Z}\sum_{i=1}^{M_B}\sum_{j=1}^{M_C} k_{ij}\log\frac{k_{ij}}{k_i k_j} \quad (22)$$

Where, k_{ij} represents the number of quantities where x & y lies in b_i & c_j respectively. Among various estimation methods, HBE is the simplest and fastest method. One of the crucial step involving in HBE is the selection of number of bars for quantization, though [20] the number of bars is important, different quantization resolutions are strongly correlated. Hence selection of smaller quantization resolution brings faster and reasonably accurate estimation.

Kernel Density Estimation (KDE):

In 1995 Moon et al. [47] proposed KDE for calculating MI which is an alternate to HBE. The aim of this method is to improve the probability density of the Kullback entropy $K\left(p_d|p_d^0\right)$ Eqn. 23,

$$K\left(p_d|p_d^0\right) = \sum_{i=1}^{M_B}\sum_{j=1}^{M_C} p(b_i,c_j)\log\left(\frac{p(b_i,c_j)}{p(b_i)p(c_j))}\right) \quad (23)$$

Which has been derived from the Eqn. 24,

$$K\left(p_d|p_d^0\right) = \sum p_i \log\left(\frac{p_i}{p_i^0}\right) \quad (24)$$

In this context, Kullback entropy represents the Mutual Information $I(D_y;X)$. According to the KDE approach, there is no appropriate cause to consider histogram as bars. Other shapes may still yield the better estimate of the probability density (PD). Thus, for a generalised kernel function $f(x)$, where $f(x)$ should be a normalized PD and the KDE $\hat{f}(x)$ can be computed using Eqn. 25,

$$\hat{f}(x) = \frac{1}{N\aleph}\sum_{i=1}^{N} f\left(\frac{x - x_i}{\aleph}\right) \quad (25)$$

Where $\aleph$ represents the smoothing parameter. One of the main problems of KDE is that inconsistent estimates for a fixed $\aleph$.

Kraskov Estimation (KE):

Based on the nearest neighbour algorithm, in 2004 Kraskov et al. [48] proposed a method for calculating MI among continuous random variables. KE is extensively used non-parametric method which calculates the nearest distance among the neighbouring samples and estimates the entropy of it. This estimator is applied to find the entropy of the unknown representations U_r. Let Z be a random variable which is independent of X & $U_r \rightarrow h + Z$. then Eqn. 26,

$$I(U_r,X) = H(U_r) - H(U_r|X) \quad (26)$$

$$= H(U_r) - H(Z)$$

$$= H(U_r) - c \quad (27)$$

Where $H(Z)$ is a constant and it is enough to compute $H(U_r)$ in Eqn. 27, and is given by Eqn. 28,

$$I(U_r,X) = \frac{r}{s}\sum_{i=1}^{s}\log(d_i + \in) + \frac{r}{2}\log(\pi) - log\gamma\left(\frac{r}{2}+1\right) + \varphi(s) - \varphi(k) \quad (28)$$

Where r denotes the dimension of the unknown representation, s represents the number of samples, d_i indicates the distance from the k^{th} nearest sample of the current sample i, $\in$ is introduced as a small numerical constant to maintain the stability, $\gamma(.)$ denotes the gamma function & $\varphi(.)$ represents the digamma function.

In this context, the mapping between the unknown representation (neurons present in the hidden layer) and the input data is deterministic, thus MI becomes infinite. Additional assumptions should be made in order to estimate MI.

7 References:

[1] P.L. Qingchen Zhanga, Laurence T. Yang, Zhikui Chen, A survey on deep learning for big data, Inf. Fusion. 42 (2018) 146–157. doi:http://dx.doi.org/10.1016/j.inffus.2017.10.006.

[2] T. Bouwmans, S. Javed, M. Sultana, S.K. Jung, Deep neural network concepts for background subtraction: A systematic review and comparative evaluation, Neural Networks. 117 (2019) 8–66. doi:10.1016/j.neunet.2019.04.024.

[3] J.Y.F. Yam, T.W.S. Chow, A weight initialization method for improving training speed in feedforward neural network, Neurocomputing. 30 (2000) 219–232. doi:10.1016/S0925-2312(99)00127-7.

[4] J.Y.F. Yam, T.W.S. Chow, Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients, IEEE Trans. Neural Networks. 12 (2001) 430–434. doi:10.1109/72.914538.

[5] T. Talaška, M. Kolasa, R. Długosz, P.A. Farine, An efficient initialization mechanism of neurons for Winner Takes All Neural Network implemented in the CMOS technology, Appl. Math. Comput. 267 (2015) 119–138. doi:10.1016/j.amc.2015.04.123.

[6] M. Kolasa, R. Długosz, T. Talaśka, W. Pedrycz, Efficient methods of initializing neuron weights in self-organizing networks implemented in hardware, Appl. Math. Comput. 319 (2018) 31–47. doi:10.1016/j.amc.2017.01.043.

[7] Q. Song, Robust initialization of a Jordan network with recurrent constrained learning, IEEE Trans. Neural Networks. 22 (2011) 2460–2473. doi:10.1109/TNN.2011.2168423.

[8] S.P. Adam, D.A. Karras, G.D. Magoulas, M.N. Vrahatis, Solving the linear interval tolerance problem for weight initialization of neural networks, Neural Networks. 54 (2014) 17–37. doi:10.1016/j.neunet.2014.02.006.

[9] N. Jiang, J. Xu, S. Zhang, Neural network control of networked redundant manipulator system with weight initialization method, Neurocomputing. 307 (2018) 117–129. doi:10.1016/j.neucom.2018.04.039.

[10] G. Thimm, E. Fiesler, High-Order and Multilayer Perceptron Initialization, IEEE Trans. Neural Networks. 8 (1997) 349–359. doi:10.1109/72.557673.

[11] N. Pinchaud, Unsupervised pre-training helps to conserve views from input distribution, Int. Conf. Mach. Learn. Edinburgh, Scotland, UK, 2012. (2012). http://arxiv.org/abs/1905.12889.

[12] Y. Bengio, A. Courville, P. Vincent, Why Does Unsupervised Pre-training Help Deep Learning? Dumitru, J. Mach. Learn. Res. 11 (2012) 625–660. doi:10.1145/1756006.1756025.

[13] C. Lee, P. Panda, G. Srinivasan, K. Roy, Training deep spiking convolutional Neural Networks with STDP-based unsupervised pre-training followed by supervised fine-tuning, Front. Neurosci. 12 (2018). doi:10.3389/fnins.2018.00435.

[14] Y. Furusho, T. Kubo, K. Ikeda, Roles of pre-training in deep neural networks from information theoretical perspective, Neurocomputing. 248 (2017) 76–79. doi:10.1016/j.neucom.2016.12.083.

[15] A. Saxe, Y. Bansal, J. Dapello, M. Advani, The information bottleneck, Iclr 2018. 24 (2018) 3–6. doi:10.1108/eb040537.

[16] S. Yu, K. Wickstrøm, R. Jenssen, J.C. Principe, Understanding Convolutional Neural Networks with Information Theory: An Initial Exploration, Mach. Learn. Cornell Univ. (2018) 1–13. http://arxiv.org/abs/1804.06537.

[17] N. Tishby, N. Zaslavsky, Deep learning and the information bottleneck principle, 2015 IEEE Inf. Theory Work. ITW 2015. (2015). doi:10.1109/ITW.2015.7133169.

[18] S. Vasudevan, Dynamic learning rate using Mutual Information, Mach. Learn. Cornell Univ. (2018). http://arxiv.org/abs/1805.07249.

[19] K. Torkkola, Nonlinear feature transforms using maximum mutual information, in: Int. Jt. Conf. Neural Networks, 2002: pp. 2756–2761. doi:10.1109/ijcnn.2001.938809.

[20] R.A. Amjad, K. Liu, B.C. Geiger, Understanding Individual Neuron Importance Using Information Theory, Eur. Conf. Mach. Learn. (2019). http://arxiv.org/abs/1804.06679.

[21] Y.F. Yam, W.S. Chow, C.T. Leung, A new method in determining initial weights of feedforward neural networks for training enhancement, Neurocomputing. 16 (1997) 23–32. doi:https://doi.org/10.1016/S0925-2312(96)00058-6.

[22] D. Erdogmus, O. Fontenla-Romero, J.C. Principe, A. Alonso-Betanzos, E. Castillo, R. Jenssen, Accurate initialization of neural network weights by backpropagation of the desired response, Int. Jt. Conf. Neural Networks. (2004) 2005–2010. doi:10.1109/ijcnn.2003.1223715.

[23] D. Erdogmus, O. Fontenla-Romero, J.C. Principe, A. Alonso-Betanzos, E. Castillo, Linear-least-squares initialization of multilayer perceptrons through backpropagation of the desired response, IEEE Trans. Neural Networks. 16 (2005) 325–337. doi:10.1109/TNN.2004.841777.

[24] Y. Liu, C.F. Zhou, Y.W. Chen, Weight initialization of feedforward neural networks

by means of partial least squares, Proc. 2006 Int. Conf. Mach. Learn. Cybern. 2006 (2006) 3119–3122. doi:10.1109/ICMLC.2006.258402.

[25] S. Timotheou, A novel weight initialization method for the random neural network, Neurocomputing. 73 (2009) 160–168. doi:10.1016/j.neucom.2009.02.023.

[26] D. Gian Paolo, S. Ridella, Statistically Controlled Activation Weight Initialization, IEEE Trans. Neural Networks. 3 (1992) 627–631. doi:10.1109/72.143378.

[27] S. Yang, S. Siu, C. Ho, Analysis of the Initial Values in Split-Complex Backpropagation Algorithm, IEEE Trans. Neural Networks. 19 (2008) 1564–1573. doi:10.1109/TNN.2008.2000805.

[28] A. Dubey, M. Sachan, J. Wieczorek, Summary and discussion of : " Why Does Unsupervised Pre-training Help Deep Learning ?," Stat. J. Club. (2014) 1–22.

[29] M.M. Hassan, M.G.R. Alam, M.Z. Uddin, S. Huda, A. Almogren, G. Fortino, Human emotion recognition using deep belief network architecture, Inf. Fusion. 51 (2019) 10–18. doi:10.1016/j.inffus.2018.10.009.

[30] P. Vincent, Stacked Denoising Autoencoders : Learning Useful Representations in a Deep Network with a Local Denoising Criterion, J. Mach. Learn. Res. 11 (2010) 3371–3408. https://dl.acm.org/citation.cfm?id=1953039.

[31] M.I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, R.D. Hjelm, MINE: Mutual Information Neural Estimation, Proc. 35th Int. Conf. Mach. Learn. Stock. Sweden, PMLR 80, 2018. (2018). http://arxiv.org/abs/1801.04062.

[32] M. Gabrié, A. Manoel, C. Luneau, J. Barbier, N. Macris, F. Krzakala, L. Zdeborová, Entropy and mutual information in models of deep neural networks, 32nd Conf. Neural Inf. Process. Syst. (NeurIPS 2018). (2018). http://arxiv.org/abs/1805.09785.

[33] P.L. Williams, R.D. Beer, Nonnegative Decomposition of Multivariate Information, ArXiv. (2010) 1–14. http://arxiv.org/abs/1004.2515.

[34] D. Chicharro, S. Panzeri, Synergy and redundancy in dual decompositions of mutual information gain and information loss, Entropy. 19 (2017) 1–29. doi:10.3390/e19020071.

[35] N. Somu, M.R.G. Raman, K. Kirthivasan, V.S.S. Sriram, Hypergraph Based Feature Selection Technique for Medical Diagnosis, J. Med. Syst. 40 (2016) 239. doi:10.1007/s10916-016-0600-8.

[36] M.R. Gauthama Raman, N. Somu, K. Kirthivasan, R. Liscano, V.S. Shankar Sriram, An efficient intrusion detection system based on hypergraph - Genetic algorithm for parameter optimization and feature selection in support vector machine, Knowledge-Based Syst. 134 (2017) 1–12. doi:10.1016/j.knosys.2017.07.005.

[37] M.R.G. Raman, N. Somu, K. Kirthivasan, V.S.S. Sriram, A Hypergraph and Arithmetic Residue-based Probabilistic Neural Network for classification in Intrusion Detection Systems, Neural Networks. 92 (2017) 89–97. doi:10.1016/j.neunet.2017.01.012.

[38] N. Somu, G.R.M. R, K. Kirthivasan, S.S. V S, A trust centric optimal service ranking

approach for cloud service selection, Futur. Gener. Comput. Syst. 86 (2018) 234–252. doi:https://doi.org/10.1016/j.future.2018.04.033 0167-739X/©.

[39] A. Bretto, Hypergraphs : First Properties - Hypergraph Theory, Mathematical Engineering, Springer International Publishing Switzerland, 2013. doi:10.1007/978-3-319-00080-0.

[40] M.C. Dourado, F. Protti, J.L. Szwarcfiter, Complexity aspects of the Helly property: Graphs and hypergraphs, Electron. J. Comb. (2009) 1–53. https://www.combinatorics.org/ojs/index.php/eljc/article/view/DS17.

[41] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Electr. Eng. 40 (2014) 16–28. doi:dx.doi.org/10.1016/j.compeleceng.2013.11.024.

[42] A.M. Saxe, J.L. McClelland, S. Ganguli, Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, in: Int. Conf. Learn. Represent., 2013: pp. 1–22. http://arxiv.org/abs/1312.6120.

[43] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proc. 13th Int. Conf. Artif. Intell. Stat. 9 (2010) 249–256.

[44] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, Proc. IEEE Int. Conf. Comput. Vis. 2015 Inter (2015) 1026–1034. doi:10.1109/ICCV.2015.123.

[45] T.P. Robitaille, Fast-histogram v0.7: fast simple 1D and 2D histograms in Python, Zenodo. (2019). https://zenodo.org/record/3268560/export/hx#.XTbGyegzZPZ.

[46] J. Kurths, C.O. Daub, J. Weise, J. Selbig, Steuer, The mutual information: detecting and evaluating dependencies between variables., Bioinformatics. 18 (2002) S231-40. doi:10.1093/bioinformatics/18.suppl_2.S231.

[47] Y. Moon, U. Lall, Estimation of Mutual Information Using Kernel Density Estimators, Phys. Rev. E. 52 (1995) 2318–2321. doi:10.1103/PhysRevE.52.2318.

[48] A. Kraskov, H. Stögbauer, P. Grassberger, Estimating mutual information, Phys. Rev. E - Stat. Physics, Plasmas, Fluids, Relat. Interdiscip. Top. 69 (2004) 16. doi:10.1103/PhysRevE.69.066138.