Enhancing Deep Neural Network Classification Performance through Novel Weight Initialization: t-SNE Supported Walsh Matrix Approach

Muhammed Nur Talha Kilic , Vishu Gupta, Yuwei Mao, Kewei Wang,
Alec Peltekian, Alok Choudhary, Wei-keng Liao, Ankit Agrawal
Northwestern University, Evanston, IL, USA
{talha.kilic, vishugupta2020, yuweimao2019, keweiwang2019, alec.peltekian}@u.northwestern.edu,
{choudhar, wkliao, ankitag}@eecs.northwestern.edu

***Abstract*—Deep Neural Networks, as a subset of AI, outperform in understanding complex relationships. The key to this success lies in the network's ability to adapt to problem-specific nuances. During model training, the network dynamically optimizes its weights by updating them during backpropagation while trying to minimize the value of the loss function. Throughout this process, the shaping of model weights is crucially linked to how they were initialized. In this study, we introduce the auxiliary network model, called Sup-Walsh (Support Walsh), which reorganizes weights to enhance class boundaries. We tested our approach on three publicly available datasets using popular classification models. For instance, when using AlexNet [1] on the MNIST dataset [2], integrating Sup-Walsh led to a significant increase in accuracy after first epoch from 14.61% to 78.99%. Similarly, GoogleNet [3] on the FashionMNIST dataset [4] showed a notable 31.61% accuracy difference between configurations without and with Sup-Walsh after first epoch. Across nearly all experiments, our proposed method consistently outperformed existing approaches, demonstrating its potential to improve classification accuracy.**

***Code availability*: Code is available at Efficient-Weight-Initializer.**

***Index Terms*—Weight Initialization, Deep Learning, Dimension Reduction, t-SNE, Walsh Matrix**

I. Introduction

Deep neural networks, with their nonlinear network structures, offer solutions to a variety of complex problems, including speech recognition [5], computer vision [6]–[8], natural language processing [9], [10], etc. The effectiveness and comprehension of these models are improved by deepening their structures [1]. However, one of the most crucial factors for the effective utilization of these aforementioned structures is the initialization of model weights. During the model training, initial weights are commonly generated in a similar manner across different types of problems. However, the goal in each problem is to identify problem-specific global minima, which may vary significantly [11], and inappropriate weight initialization can lead to convergence issues.

In this study, our objective is to give essential details about the dataset to the model prior to the training phase with t-SNE Supported Walsh matrix approach, rather than directly inputting the dataset into the deep learning model. This method helps the model distinguish between different class clusters more effectively. Compared to conventional weight initialization methods, our proposed methodology accelerates model convergence by providing dataset insights upfront.

Contributions of this study are as follows:

- We propose an auxiliary model designed to expedite the convergence process of the primary model.
- Our analysis demonstrates that models incorporating pre-learned distribution information tend to exhibit lower loss.

II. Problem Definition

We study the process of weight initialization in the context of classification tasks. Specifically, we denote the training dataset as $D_{train} = (x_i, y_i)_{i=1}^{n}$, where n represents the number of data points. The ultimate objective is to determine centroids for each class that are positioned at the farthest possible locations.

Let's denote the set of centroids as $c_1, c_2, ...c_z$ with z being different the number of distinct classes. Let $D(c_i, c_j)$ represent the distance between centroid c_i and c_j, where i and j range from 1 to z and $i \neq j$. The formula for calculating the distance between all pairs of centroids is given by:

$$D(c_i, c_j) = \sqrt{\sum_{k=1}^{d} (c_{ik} - c_{jk})^2} \tag{1}$$

where,

- c_i and c_j are the centroids.
- c_{ik} is the k-th cordinate of centroid c_i.
- c_{jk} is the k-th cordinate of centroid c_j.
- d is the number of dimensions/features.

The ultimate goal is to maximize $\sum D(c_i, c_j)$. Therefore, the objective in the Walsh Vector integration is to relocate centroids to positions where the product of their components equals zero, i.e., $c_i \bullet c_j = \sum_{k=1}^{d} (c_{ik} \bullet c_{jk}) = 0$, ensuring perpendicularity between centroids.

III. Methods

In this section, we offer explanations of the dimensionality reduction techniques and Walsh Vector utilized in this study.

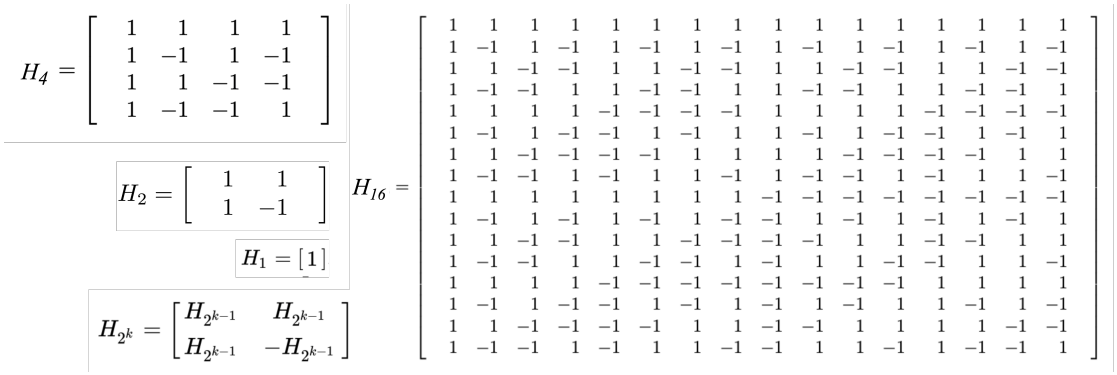

Fig. 1. A specific formula and initial version of the Hadamard Matrix, represented as H_2. Matrices undergo extensions in powers of two, exemplified by R^{1x1}, R^{2x2},R^{4x4},R^{8x8} and R^{16x16}.

A. Dimensionality Reduction

Dimensionality reduction involves transforming high-dimensional data into a lower-dimensional representation while minimizing information loss. In this study, the purpose of using dimension reduction is based on extracting the most significant features and centroids associated with class distinctions from the input data obtained. The goal is to emphasize unique characteristics while potentially eliminating noise. Considering the distribution of classes, t-SNE [12], which demonstrates the broadest dispersion in space, surpasses one hot encoding, PCA [13], and UMAP [14], forming the foundational component of the support network in this study.

The process of dimensionality reduction using t-SNE can be described as follows: Let X be a dataset, $X \in \mathbb{R}^{nxD}$, with n data points x_i ($i \in$ 1, 2,..., n), where each data point possesses D dimensions, resulting in a matrix of size $n \times D$. We aim to obtain a lower-dimensional representation denoted by Y, where Y is a matrix of low-dimensional embeddings $Y \in \mathbb{R}^{nxd}$, with d being significantly smaller than D ($d << D$). If two data points x_i and x_j are close to each other in the input space X (i.e., the original dataset), their corresponding lower-dimensional embeddings y_i and y_j should also be close. This is achieved through conditional probability, where the probability of the existence of point j given point i in the original space is utilized for mapping the points into the lower-dimensional space [12].

$$p_{j|i} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2\sigma_i^2}\right)} \tag{2}$$

where,

- x_i and x_j are the original high-dimensional data points.
- y_i and y_j are the reduced low-dimensional data points.
- σ_i is the variance of the Gaussian distribution centered at point i.
- $\| \ . \ \|$ represent the Euclidean distance

B. Walsh Vector

The Walsh matrix, initially developed by L. Walsh in 1923 as a variant of the Hadamard matrix, has been widely utilized in communication systems [15]. Its effectiveness lies in its capacity to enhance predictive accuracy even in the presence of noise, achieved by amplifying the distinctions between transmitted data [16]. When analyzing the Walsh Matrix as depicted in Fig. 1, it is observed that the greatest separation between vectors arises from their orthogonality. Taking inspiration from this situation, instead of one-hot encoding, we opt for Walsh vectors to more precisely describe the distributional centers of classes within the model, which is expressed as follows [17]:

$$W_k(x) = (-1)^{\sum_{j=0}^{\infty} k_j x_{j+1}} \tag{3}$$

where,

- $W_k : [0,1] \rightarrow \{-1, 1\}, k \in \mathbb{N} - 1$.
- $W_0(x) = 1$ everywhere on the interval.
- k_j is the j_{th} bit in the binary. representation of k, starting with k_0.
- x_j is the j_{th} bit in the fractional binary representation of x, starting with x_1.

C. The Proposed Architecture

In this section, we will provide a visual representation of the method's flow in Fig. 2. Initially, the training dataset classes are identified. Random selection of one sample from these classes is performed for each batch (size of 200). Following this, a data reduction method is applied to project these randomly sampled classes onto a two-dimensional plane. The objective behind implementing this approach is to convey class characteristics that the main model will subsequently capture during training. Subsequently, we opt for the Walsh matrix due to the broader dispersion observed when applying t-SNE to randomly selected samples from higher dimensions onto a two-dimensional plane. Each class is then randomly assigned a vector from the Walsh matrix. After selecting the Walsh vector, we combine our primary model with the Sup-Walsh model to form a comprehensive pipeline. The learning process concludes as the concatenated model converges during training. This process fine-tunes the model's weights.

IV. Experiments

This section presents the experimental results and datasets.

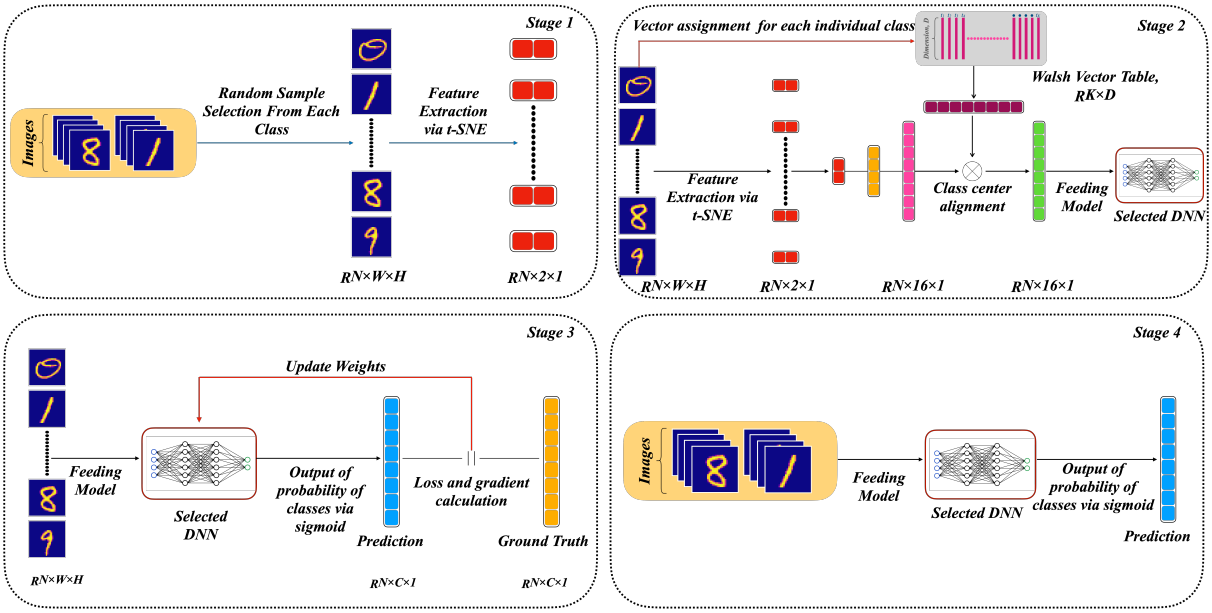

Fig. 2. The complete pipeline of four stages, starting from Stage 1 and progressing to Stage 4.

A. Dataset

The performance of the proposed structure is assessed using three widely recognized datasets that are publicly available. These datasets include CIFAR-10 [18], which comprises a total of 60,000 images. Additionally, MNIST [2] consists of a total of 70,000 images. Fashion-MNIST [4], a variant of MNIST, contains 70,000 images organized into distinct classes.

B. Experimental Results

Based on the four different stages depicted in Fig. 2, we opted for well-known models widely available in the literature and commonly used for comparative analysis. These models include AlexNet [1], ResNet50 [19], VGG19 [20], GoogleNet [3], SqueezeNet [21], and Nvidia-MIT [22]. Furthermore, we introduce a relatively straightforward network architecture, denoted as the *base*, comprising only three feed-forward layers. During the initial epoch of training, AlexNet on the MNIST dataset showed an accuracy gap of 64.38%, with a performance rising from 14.61% to 78.99% after using Sup-Walsh. Similarly, GoogleNet on the FashionMNIST dataset had a remarkable 31.61% increase in accuracy when using Sup-Walsh. GoogleNet also demonstrated a 15.37% increase in accuracy on the Cifar-10 dataset. Our proposed solutions outperform nearly in all 21 scenarios, except Base and SqueezeNet on Cifar-10. This includes 7 models across 3 datasets, yielding better results in the final 10th epoch.

TABLE I

THE ACCURACY RATES OF MODELS ON MNIST DATASET AT THE INITIAL AND FINAL EPOCHS. *'With'* REPRESENTS TRAINING AS WITH SUP-WALSH, *'Without'* CORRESPONDS TO SITUATIONS WHERE THESE MODELS ARE DIRECTLY TRAINED WITHOUT BEING CONNECTED TO THE PROPOSED MODEL. THE CONTRIBUTION (ACCURACY INCREASE) PROVIDED BY THE PROPOSED MODEL IS ILLUSTRATED PROPORTIONALLY WITH GREEN-RED ARROWS.

Models	Epoch	Dataset		Accuracy Change
With Sup-Walsh: "With"		MNIST [2]		(%)
Without Sup-Walsh: "Without"		Without (%)	With (%)	
Base	1^{st}	**55.55**	68.01	12.45 ↑
	10^{th}	88.93	91.65	2.71 ↑
VGG-19 [20]	1^{st}	54.17	83.85	29.68 ↑
	10^{th}	96.80	98.53	1.73 ↑
ResNet-50 [19]	1^{st}	13.13	27.14	14.01 ↑
	10^{th}	51.83	85.32	33.49 ↑
AlexNet [1]	1^{st}	14.61	78.99	64.38 ↑
	10^{th}	89.06	96.64	7.57 ↑
GoogleNet [3]	1^{st}	44.79	**89.34**	44.54 ↑
	10^{th}	**97.95**	**98.84**	0.88 ↑
SqueezeNet [21]	1^{st}	16.07	42.41	26.34 ↑
	10^{th}	51.10	93..74	42.64 ↑
Nvidia-Mit [22]	1^{st}	25.53	56.89	31.36 ↑
	10^{th}	94.21	95.67	1.45 ↑

Note: Bold values indicate the best values of 1^{st}, 10^{th} epochs both for "*With*" and "*Without*" Sup-Walsh.

V. CONCLUSION

In this research, we integrated the Walsh Matrix into the initialization process of various deep learning models. This was achieved by introducing the Sup-Walsh network, aiming to improve the distinction between different classes. We demonstrated the validity of our opinions on this matter through the experiments we conducted with different datasets and models.

In future work, our intention is to shift towards a model where weights are determined according to dataset specifications, eliminating the necessity for a support network.

ACKNOWLEDGMENT

This work was performed under the following financial assistance award 70NANB19H005 from U.S. Department of Commerce, National Institute of Standards and Technology as

TABLE II

THE ACCURACY RATES OF MODELS ON FASHION-MNIST DATASET AT THE INITIAL AND FINAL EPOCHS. *'With'* REPRESENTS TRAINING AS WITH SUP-WALSH, *'Without'* CORRESPONDS TO SITUATIONS WHERE THESE MODELS ARE DIRECTLY TRAINED WITHOUT BEING CONNECTED TO THE PROPOSED MODEL. THE CONTRIBUTION (ACCURACY INCREASE) PROVIDED BY THE PROPOSED MODEL IS ILLUSTRATED PROPORTIONALLY WITH GREEN-RED ARROWS.

Models	Epoch	Dataset		Accuracy Change
With Sup-Walsh: "With"		Fashion-MNIST [4]		(%)
Without Sup-Walsh: "Without"		Without (%)	With (%)	
Base	1^{st}	**56.68**	62.83	6.14 ↑
	10^{th}	78.44	81.43	2.99 ↑
VGG-19 [20]	1^{st}	55.38	71.33	15.95 ↑
	10^{th}	84.82	88.90	4.16 ↑
ResNet-50 [19]	1^{st}	13.22	26.23	12.98 ↑
	10^{th}	45.82	72.64	26.82 ↑
AlexNet [1]	1^{st}	17.61	70.91	53.29 ↑
	10^{th}	69.77	85.24	15.46 ↑
GoogleNet [3]	1^{st}	44.76	**76.37**	31.61 ↑
	10^{th}	**87.67**	**92.87**	5.19 ↑
SqueezeNet [21]	1^{st}	27.16	48.93	21.77 ↑
	10^{th}	77.40	79.04	1.64 ↑
Nvidia-Mit [22]	1^{st}	41.96	54.99	13.03 ↑
	10^{th}	81.88	82.75	0.87 ↑

Note: Bold values indicate the best values of 1^{st}, 10^{th} epochs both for "*With*" and "*Without*" Sup-Walsh.

TABLE III

THE ACCURACY RATES OF MODELS ON CIFAR-10 DATASET AT THE INITIAL AND FINAL EPOCHS. *'With'* REPRESENTS TRAINING AS WITH SUP-WALSH, *'Without'* CORRESPONDS TO SITUATIONS WHERE THESE MODELS ARE DIRECTLY TRAINED WITHOUT BEING CONNECTED TO THE PROPOSED MODEL. THE CONTRIBUTION (ACCURACY INCREASE) PROVIDED BY THE PROPOSED MODEL IS ILLUSTRATED PROPORTIONALLY WITH GREEN-RED ARROWS.

Models	Epoch	Dataset		Accuracy Change
With Sup-Walsh: "With"		CIFAR-10 [18]		(%)
Without Sup-Walsh: "Without"		Without (%)	With (%)	
Base	1^{st}	13.05	10.72	0.21 ↓
	10^{th}	23.44	21.72	0.97 ↓
VGG-19 [20]	1^{st}	13.11	22.95	9.83 ↑
	10^{th}	39.60	50.13	10.52 ↑
ResNet-50 [19]	1^{st}	10.62	11.47	0.84 ↑
	10^{th}	15.85	21.81	5.95 ↑
AlexNet [1]	1^{st}	9.88	10.70	0.82 ↓
	10^{th}	9.96	14.63	4.66 ↑
GoogleNet [3]	1^{st}	**16.35**	**31.72**	15.37 ↑
	10^{th}	**48.56**	**59.54**	10.98 ↑
SqueezeNet [21]	1^{st}	10.84	9.97	0.13 ↓
	10^{th}	10.00	11.86	1.86 ↓
Nvidia-Mit [22]	1^{st}	12.61	14.12	1.50 ↓
	10^{th}	36.59	37.28	0.68 ↑

Note: Bold values indicate the best values of 1^{st}, 10^{th} epochs both for "*With*" and "*Without*" Sup-Walsh.

part of the Center for Hierarchical Materials Design (CHiMaD). Partial support is also acknowledged from NSF awards CMMI-2053929, OAC-2331329, DOE award DE-SC0021399, and Northwestern Center for Nanocombinatorics.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[2] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2014.

[4] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[5] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE access*, vol. 7, pp. 19 143–19 165, 2019.

[6] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2722–2730.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[10] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *International conference on machine learning*. PMLR, 2017, pp. 933–941.

[11] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," *Advances in neural information processing systems*, vol. 31, 2018.

[12] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[13] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (pca)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[14] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I. W. Kwok, L. G. Ng, F. Ginhoux, and E. W. Newell, "Dimensionality reduction for visualizing single-cell data using umap," *Nature biotechnology*, vol. 37, no. 1, pp. 38–44, 2019.

[15] J. L. Walsh, "A closed set of normal orthogonal functions," *American Journal of Mathematics*, vol. 45, no. 1, pp. 5–24, 1923.

[16] M. N. T. Kılıç and T. Ölmez, "Detection of covid-19 in chest x-ray image by using convolutional network trained with walsh functions."

[17] A. Albrecht, P. Howlett, and G. Verma, "Optimal splitting of parseval frames using walsh matrices," 2020. [Online]. Available: https://arxiv.org/abs/2007.13026

[18] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do cifar-10 classifiers generalize to cifar-10?" *arXiv preprint arXiv:1806.00451*, 2018.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[21] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and ¡0.5mb model size," 2016.

[22] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *CoRR*, vol. abs/2105.15203, 2021. [Online]. Available: https://arxiv.org/abs/2105.15203