Learnable Global Layerwise Nonlinearities Without Activation Functions

Justin Diamond
University of Basel
Basel, Switzerland
justin.diamond@unibas.ch

***Abstract*—In machine learning and neural networks, non-linear transformations have been pivotal in capturing intricate patterns within data. These transformations are traditionally instantiated via activation functions such as Rectified Linear Unit (ReLU), Sigmoid, and Hyperbolic Tangent (Tanh). In this work, we introduce DiagonalizeGNN, an approach that changes the introduction of non-linearities in Graph Neural Networks (GNNs). Unlike traditional methods that rely on pointwise activation functions, DiagonalizeGNN employs Singular Value Decomposition (SVD) to incorporate global, non-piecewise non-linearities across an entire graph's feature matrix. We provide the formalism of this method and empirical validation on a synthetic dataset, we demonstrate that our method not only achieves comparable performance to existing models but also offers additional benefits such as higher stability and potential for capturing more complex relationships. This novel approach opens up new avenues for research and offers significant implications for the future of non-linear transformations in machine learning.**

***Index Terms*—activation functions, Neural Networks, Nonlinearity, SVD**

I. Introduction

In machine learning, particularly in the realm of neural networks, non-linear transformations play a critical role in capturing complex relationships in data. Traditionally, these non-linearities are introduced through activation functions like the Rectified Linear Unit (ReLU), Sigmoid, or Hyperbolic Tangent (Tanh). These functions operate in a pointwise fashion, altering the network's output based on individual elements. However, such local changes might not capture more global, complex interactions effectively.

In this paper, we introduce DiagonalizeGNN, a novel framework for Graph Neural Networks (GNNs) that employs a different paradigm for non-linearity. Our method uses the Singular Value Decomposition (SVD) to capture global non-linearities across the entire feature space of the graph nodes. This SVD-based non-linear transformation is not piecewise, providing a fundamentally new approach to introduce non-linearities in neural networks.

The closest relative project we encountered is a work that focuses on aggregating features to define global activation functions [FMP21]. However, their method still fundamentally relies on conventional, piecewise activation functions. Unlike their approach, our method is truly global and non-piecewise, using mathematical operations that holistically consider the entire graph structure and feature matrix. This paper will delve into the mathematical foundations of our method, providing proofs to affirm the non-linearity of our operations, and present experimental results to validate its effectiveness.

One of the unique aspects of our approach stems from principles loosely inspired by quantum field theory (QFT), a framework in theoretical physics that describes how fields and particles interact. In QFT, the equations governing fields are often linear when considered in isolation. However, interaction terms added to the equations introduce non-linearities, which result in a wide range of complex behaviors that simple linear equations could not capture, [CGP07].

The SVD-based interaction term in our DiagonalizeGNN method serves a similar role. In its absence, the graph neural network would be analogous to a system of non-interacting fields, wherein the mapping from input features to output features could potentially remain linear or piecewise non-linear, depending on the activation functions used. The introduction of our SVD-based interaction term is akin to the interaction terms in QFT, breaking the linearity and introducing rich, global non-linearities into the system.

II. Relevance to Physical Sciences

The machine learning frameworks commonly used in physical sciences often require an intricate balance between computational complexity and representational power. Diffusion models, [HJA20], often employed for understanding molecular interactions or fluid dynamics, are a prime example of this. These models usually rely on solving complex partial differential equations (PDEs) or simulating Markov Chain Monte Carlo (MCMC) steps, both of which can be computationally expensive.

Our formalism could serve as an intuitive and robust foundation for developing machine learning methods targeted at physical systems. Its emphasis on capturing global interactions could better the way we model complex systems in the physical sciences [Che+18; Hoo+23; DL23], particularly for diffusion models of molecular structures and interactions.

A. Conventional Non-linearities

In standard neural networks, the output h is computed as:

$$h = f(Wx + b)$$

where f is an element-wise activation function.

B. DiagonalizeGNN's Non-linearities

In DiagonalizeGNN, the output h is computed through Singular Value Decomposition:

$$h = \text{SVD}(A \cdot X \cdot W^T - A \cdot X \cdot V^T)$$

with A an adjacency matrix, X a feature vector for each node of the graph, and W and V parameters of the neural network.

III. Novelty

The novel aspects are:

1) Interaction terms $A \cdot X \cdot W^T - A \cdot X \cdot V^T$ that introduce a new form of non-linearity after SVD.
2) The global context is considered during the diagonalization, unlike localized, element-wise activation functions.
3) Non-linearity is introduced not by a function but through a series of transformations (SVD).

GCN, [KW16]: $$X^v_{\text{new}} = \sigma\left(\sum_{u \in \mathcal{N}(v)} A_{uv} \cdot W \cdot X^u\right)$$

DiagonalizeGNN: $$Z^v = \sum_{u \in \mathcal{N}(v)} A_{uv} \cdot W \cdot X^u - A_{uv} \cdot V \cdot X^u$$
$$[U, S, V] = \text{SVD}(Z^v)$$
$$X^v_{\text{new}} = U \cdot S$$

IV. Intuition Behind Diagonalize Non-Linear Layer

The Diagonalize Non-Linear Layer introduces non-linearity by considering relationships between nodes in a graph. Below are some key points:

1) **Capturing Dependencies:** The interaction term $Z - AXV^T$ serves as a form of contrast between two different views (W and V) of the graph. This helps the model identify and learn subtle differences and relationships between nodes.
2) **Global Context:** The Singular Value Decomposition (SVD) operation projects the interaction term into a lower-dimensional space where the most significant variations lie. In doing so, the relationships between different nodes are embedded into a space that emphasizes the most meaningful dependencies.
3) **Learnable Parameters:** The weight matrices W and V are learnable parameters. These matrices adapt during training to capture relevant features and correlations in the graph data.
4) **Non-Linearity through Contrast:** The interaction term itself serves as a non-linear transformation by capturing how two linear transformations (AWX^T and AXV^T) differ. The contrasting nature of this calculation serves as a form of non-linearity.
5) **Dynamic Non-Linearity:** Unlike traditional activation functions like ReLU or Sigmoid, the non-linearity here is dynamic. It evolves based on the learning of W and V, allowing the model to adapt its form of non-linearity to better capture complex and evolving correlations.

V. Diagonalize Non-Linear Layer

Let A be the adjacency matrix of shape $[N, N]$, and X be the feature matrix of shape $[N, F]$.

Initialize weight matrices W, V of shape $[H, F]$.

1) **Standard Operation:**
$$Z = AXW^T$$
2) **Interaction Term:**
$$\text{interaction} = Z - AXV^T$$
3) **Diagonalization:**
$$\text{interaction} = U\Sigma V^T$$
4) **Feature Update:**
$$\text{new}_X = U\Sigma$$

Non-linearities in DiagonalizeGNN:

1. Interaction Term: The difference $Z - A \cdot X \cdot V^T$ creates an interaction term.
$$\text{interaction} = Z - A \cdot X \cdot V^T$$

2. Singular Value Decomposition: Singular value decomposition (SVD) is a form of matrix factorization and introduces a unique form of non-linearity in the model.
$$\text{interaction} = U\Sigma V^T$$

3. Feature Update: Multiplying U and Σ to get the new feature $\text{new}_X = U\Sigma$ acts as a non-linear transformation of the original feature space.

VI. Nonlinearity of the SVD-based Operation

A. Preliminaries

Let A be the adjacency matrix of shape $[N, N]$, and X be the feature matrix of shape $[N, F]$.

Initialize weight matrices W, V of shape $[H, F]$.

The sequence of operations for any feature matrix X is defined as:

$$\begin{aligned} Z &= AXW^T, \\ \text{interaction} &= Z - AXV^T, \\ \text{interaction} &= U\Sigma V^T, \\ \text{new}_X &= U\Sigma. \end{aligned}$$

A function $f : \mathbb{R}^{N \times F} \to \mathbb{R}^{N \times H}$ is said to be linear if it satisfies:

- Additivity: $f(X_1 + X_2) = f(X_1) + f(X_2)$
- Homogeneity: $f(\alpha X) = \alpha f(X)$

VII. Non-linearity of Eigenvalues in SVD

Let A, B be arbitrary matrices. Consider the SVD decompositions $A = U_A \Sigma_A V_A^T$ and $B = U_B \Sigma_B V_B^T$.

Now, let $C = A + B$. Its SVD decomposition is $C = U_C \Sigma_C V_C^T$.

A. Singular Values and Eigenvalues

The singular values σ_A and σ_B are the square roots of the eigenvalues of A^TA and B^TB, respectively.

$$\lambda_A = \text{eig}(A^TA), \quad \lambda_B = \text{eig}(B^TB)$$

$$\sigma_A = \sqrt{\lambda_A}, \quad \sigma_B = \sqrt{\lambda_B}$$

B. Eigenvalues of C

The singular values σ_C are derived from the eigenvalues of C^TC:

$$C^TC = (A+B)^T(A+B) = A^TA + B^TB + A^TB + B^TA$$

$$\lambda_C = \text{eig}(C^TC), \quad \sigma_C = \sqrt{\lambda_C}$$

C. Proof of Non-linearity in Eigenvalues

We need to show that $\sigma_C \neq \sigma_A + \sigma_B$.

Firstly, λ_C is an eigenvalue of $A^TA+B^TB+A^TB+B^TA$, which contains terms that are not simply eigenvalues of A^TA or B^TB.

Hence, λ_C is generally not equal to $\lambda_A + \lambda_B$. As a consequence, σ_C is also not $\sigma_A + \sigma_B$.

VIII. Non-linearity of Orthogonal Matrices in SVD

To prove that the unitary (or orthogonal in the real case) matrices obtained from the Singular Value Decomposition (SVD) operation are non-linear, we first recall the definition of linearity. A function $f(x)$ is linear if it satisfies:

$$f(x_1 + x_2) = f(x_1) + f(x_2),$$
$$f(ax) = af(x).$$

For SVD, let's consider square matrices A, B of the same dimension. Their SVDs are given by:

$$A = U_A\Sigma_AV_A^T,$$
$$B = U_B\Sigma_BV_B^T.$$

If $C = A + B$, the SVD of C is $C = U_C\Sigma_CV_C^T$.

Proof Points:

1) *Uniqueness and Non-additivity*: The unitary matrices U_C and V_C are uniquely determined by C, but it is not implied that $U_C = U_A + U_B$.
2) *Orthogonal Constraints*: The unitary matrices satisfy $U^TU = I$, and $(U_A+U_B)^T(U_A+U_B)$ is not necessarily I.
3) *Decomposition Equation*: The SVD of C in terms of A and B is

$$U_C\Sigma_CV_C^T = U_A\Sigma_AV_A^T + U_B\Sigma_BV_B^T.$$

Due to the complexities introduced by the diagonal and unitary matrices, U_C and V_C cannot be simply expressed as $U_A + U_B$ or $V_A + V_B$.

4) *Explicit Counter-example*: If Q_1 and Q_2 are orthogonal, $Q_1 + Q_2$ is generally not orthogonal. To show this, we check that $(Q_1 + Q_2)^T(Q_1 + Q_2) \neq I$.

Based on these points, U_C and V_C resulting from the SVD operation are not linear functions of U_A, U_B or V_A, V_B. Hence, the unitary matrices are non-linear when derived from the SVD operation.

IX. Non-Linearity in Operations and Implications for Neural Networks

A. Breakdown of Operations

1) **Standard Operation:**

$$Z = AXW^T$$

This operation is linear with respect to X.

2) **Interaction Term:**

$$\text{interaction} = Z - AXV^T$$

This is essentially a linear combination of linear terms, so still linear with respect to X.

3) **Diagonalization:**

$$\text{interaction} = U\Sigma V^T$$

Here's where non-linearity comes into play. The singular value decomposition is inherently non-linear with respect to the matrix it decomposes. Therefore, the matrices U, Σ, V are non-linear functions of interaction, which itself is a function of X.

4) **Feature Update:**

$$\text{new}_X = U\Sigma$$

Again, U and Σ are derived in a non-linear manner from X, making new_X also non-linear with respect to X.

B. Implications for Neural Networks

- **Contextual Understanding**: The SVD operation captures the essential 'modes' of variation in the data. In a neural network, this could serve as a form of "global context," helping the model to understand overarching relationships and dependencies that simpler, local operations might miss.
- **Parameter Efficiency**: In a neural network, the matrices W and V could be learned parameters, allowing the network to adaptively learn the "best" way to introduce non-linearity into the system. This could potentially lead to more expressive yet parameter-efficient models.
- **Hierarchical Features**: The singular values Σ and the corresponding U and V could capture different levels of abstraction in the data, potentially aiding in the hierarchical representation learning that deep networks are known for.

- **End-to-End Learning**: Importantly, all these parameters can be learned in an end-to-end fashion, allowing for better integration of this global context information with the local features learned by the rest of the network.
- **Complexity of SVD**: The Singular Value Decomposition (SVD) of a matrix $A \in \mathbb{R}^{m \times n}$ typically has a computational complexity of $O(\min(m^2n, mn^2))$. In our specific context, where SVD is applied to the interaction term, this adds significant computational overhead. This makes the method less scalable for large-scale problems or real-time applications, where computational efficiency is a prime concern. However, it's worth noting that in the case of small graphs, the computational complexity may not present a noticeable difference, making it a feasible approach for such specific cases.
- **Possible SVD simplifications**:To mitigate the computational burden, one possible avenue for future research could be developing linear approximations of the SVD operation. While this would potentially lose some of the fine-grained information captured by SVD, it could dramatically speed up the computations, making it feasible for applications requiring quick decision-making. Another promising direction is the investigation of sparse versions of SVD. Sparsity can be introduced either in the input matrices or in the matrices resulting from the decomposition (U, Σ, V). Sparse versions would reduce the computational complexity and memory requirements, albeit at the cost of some loss of information.

X. Graph Convolutional Network (GCN)

The Graph Convolutional Network (GCN),[KW16], generally involves linear transformations followed by non-linear activation functions.

Mathematical Representation:

1. Linear Transformation:

$$H^{(l+1)} = A \cdot H^{(l)} \cdot W^{(l)}$$

2. Non-linear Activation:

$$H^{(l+1)} = \sigma(A \cdot H^{(l)} \cdot W^{(l)})$$

Here, σ represents the activation function, commonly ReLU, Sigmoid, or Tanh.

XI. Comparison

- **Linear Transformations:** Both models employ linear transformations but use them differently. GCNs aggregate neighbor information, whereas DiagonalizeGNN also considers interaction terms.
- **Non-linearities:** GCNs use standard activation functions like ReLU, Tanh, etc., for introducing non-linearities. DiagonalizeGNN uses a singular value decomposition step to introduce non-linearities.
- **Activation Functions:** GCNs employ activation functions directly on the aggregated feature, while DiagonalizeGNN does not use any activation function after the diagonalization step.

2

XII. In-depth Explanation

A. GCN

The activation function, such as ReLU, used in GCN introduces non-linearity by essentially partitioning the feature space into regions separated by hyperplanes. This enables the model to learn more complex relations in the data.

B. DiagonalizeGNN

The singular value decomposition (SVD) step is itself a non-linear operation in terms of how it decomposes the interaction term. It captures the essence of the original matrix (interaction term) in a reduced form by approximating the matrix as a product of U, Σ, and V^T. This provides the model with a capability to capture complex patterns in the adjacency and feature matrices.

XIII. Synthetic Dataset and Baselines

A. Synthetic Dataset

To evaluate the performance and robustness of our proposed method, we generated a synthetic dataset using Python's PyTorch library. The dataset consists of 100 samples, where each sample represents a graph characterized by an adjacency matrix A and a node feature matrix X. Each graph has 10 nodes, and each node has 10 features. The labels are randomly generated and set to either 0 or 1.

The adjacency matrix A is created as follows: for each graph, a random symmetric matrix with elements in $[0, 1]$ is generated. The matrix is further thresholded to make it sparse, with elements less than 0.5 set to zero and those greater than or equal to 0.5 set to one. The diagonal elements of A are set to the value of the label to introduce a rudimentary form of class separation.

The feature matrix X is constructed such that each node feature is the sum of its connections in A. Each node feature vector is then replicated to fill the 10 feature dimensions.

B. Baseline Methods

- **Graph Convolutional Network (GCN)**: We use the standard GCN as a baseline to compare the performance of our proposed method. This serves as a traditional approach for learning on graph-structured data.
- **SVD-based Method (Our Method)**: The novel method proposed in this paper uses Singular Value Decomposition to introduce a global non-linear function, aiming to replace the traditional activation functions in graph neural networks.

XIV. Implementation Details

A. Normalization

Normalization is applied to both the Z matrix (result of AXW^T) and the interaction term interaction $= Z - AXV^T$. For each of these matrices, the last dimension is normalized by its Frobenius norm to avoid numerical instabilities and to

scale the features to a similar range. Mathematically, this can be expressed as:

$$Z = \frac{Z}{\|Z\|_F + \epsilon}$$

$$\text{interaction} = \frac{\text{interaction}}{\|\text{interaction}\|_F + \epsilon}$$

where ϵ is a small constant to avoid division by zero.

B. Regularization

A regularization term is added to the output features new_X to prevent overfitting. This term is a function of the Frobenius norm of the weight matrices W and V as well as the norms of the bias terms $b1$ and $b2$. The regularization term reg_term is defined as:

$$\text{reg\_term} = \lambda \left(\|W\|_F + \|V\|_F + \|b1\| + \|b2\| \right)$$

Here, λ is a regularization parameter.

C. Bias Terms

Bias terms $b1$ and $b2$ are added to Z and interaction, respectively. These bias terms allow for a shift in the feature space and provide the model with greater expressive power. They are initialized to zero and learnable during the training process. Mathematically, this can be described as:

$$Z = Z + b1$$

$$\text{interaction} = \text{interaction} + b2$$

XV. Linear Version and Baseline Model

A. Linear Version

The linear version of our proposed model is a simplified form of the primary model that excludes the SVD-based diagonalization, bias terms, normalization, and regularization. It performs a simple linear transformation on the input features, defined by a weight matrix W. The forward pass is simplified to:

$$Z = AXW^T$$

Unlike the nonlinear version, this simplified model directly uses Z as the updated node features.

B. Baseline: Graph Convolutional Network (GCN)

The baseline model for comparison is a standard Graph Convolutional Network (GCN). Each GCN layer consists of a linear transformation followed by a ReLU activation function. In mathematical terms, each layer is defined as:

$$X' = \text{ReLU}(AXW)$$

Here, W is the learnable weight matrix for the linear transformation. The GCN model uses global average pooling for the output features, similar to our proposed model.

XVI. Experimental Results

A. Convergence Analysis

Figure 1 illustrates the training loss of our proposed SVD-based model, the standard GCN model, and the simplified Linear version of our model. Both the SVD-based model and the GCN model exhibit convergence to nearly the same accuracy on the training set. However, the Linear version experiences high variability in loss values between epochs, ranging from 1 to 500, indicating a lack of stable convergence.

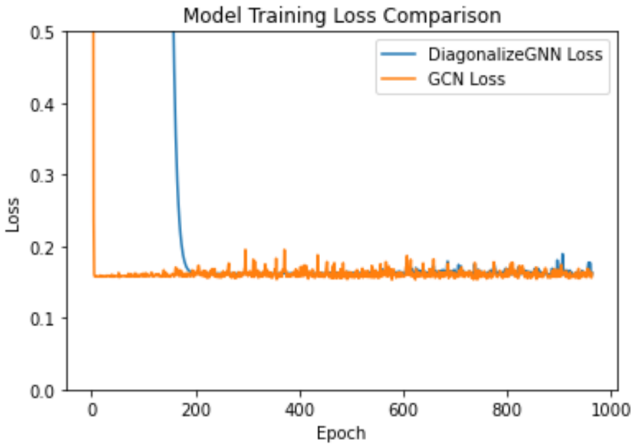

Fig. 1. Training loss comparison of the SVD-based model, GCN model, and Linear version.

This convergence behavior corroborates the mathematical understanding that the SVD-based diagonalization introduces beneficial non-linearities, enabling the model to learn complex patterns similar to the GCN model. On the other hand, the Linear model lacks this capacity, making it more volatile and less effective in capturing the underlying graph structure.

B. Test Dataset Analysis

Figure 2 shows the test loss of our SVD-based model, the standard GCN model, and the Linear version. The SVD-based model takes longer to converge compared to the GCN model but eventually achieves slightly better performance. Additionally, the SVD-based model's performance appears to be more stable, as indicated by the relatively smoother loss curve. Conversely, the green lines representing the Linear version demonstrate erratic behavior, emphasizing the model's instability and ineffectiveness.

These results suggest that although the SVD-based approach may be slower to converge, it potentially provides a more stable and accurate model for graph-based learning tasks. The stability in test loss indicates the robustness of the model, an aspect not found in the baseline GCN or the Linear version.

XVII. Conclusion

We introduced DiagonalizeGNN, a novel graph neural network architecture that employs singular value decomposition (SVD) to introduce global non-linearities into the model. This approach, inspired by quantum field theory, stands in contrast

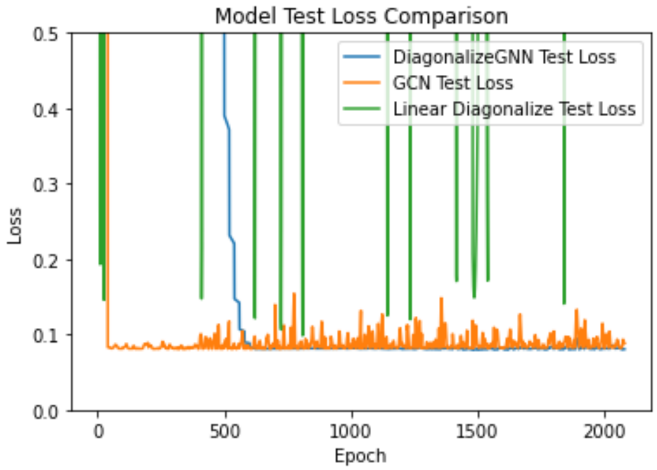

Fig. 2. Test loss comparison of the SVD-based model, GCN model, and Linear version.

to conventional methods that rely on piecewise activation functions.

Our experimental results establish the DiagonalizeGNN as a viable alternative to traditional Graph Convolutional Networks (GCNs). While the training losses of the two models are comparable, our SVD-based model demonstrates superior stability and generalization in the test dataset, albeit at the cost of slower convergence.

The Linear version of our model serves as a valuable point of comparison. Its erratic behavior in terms of training and test loss highlights the necessity of effective non-linearities for learning complex graph-structured data.

However, one crucial aspect to consider is the scalability of our proposed model. Singular value decomposition is computationally expensive, which could be a bottleneck for larger graphs. To address this, future work could explore the use of sparse graph structures or approximated linear techniques that could capture the essence of the non-linear interactions, but with less computational overhead.

These findings not only validate the efficacy of DiagonalizeGNN but also open up new avenues for future research. The stability and robustness of the SVD-based model suggest its applicability across a broad range of complex systems, making it a promising subject for interdisciplinary investigations that span machine learning and theoretical physics.

References

[CGP07] Jean-François Colombeau, Andre Gsponer, and Bernard Perrot. “Nonlinear generalized functions and the Heisenberg-Pauli foundations of Quantum Field Theory”. In: *arXiv preprint arXiv:0705.2396* (2007).

[KW16] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).

[Che+18] Ricky T. Q. Chen et al. “Neural Ordinary Differential Equations”. In: *arXiv preprint arXiv:1806.07366* (2018). URL: https://arxiv.org/abs/1806.07366.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising Diffusion Probabilistic Models”. In: *arXiv preprint arXiv:2006.11239* (2020). URL: https://arxiv.org/abs/2006.11239.

[FMP21] Tiago AE Ferreira, Marios Mattheakis, and Pavlos Protopapas. “A new artificial neuron proposal with trainable simultaneous local and global activation function”. In: *arXiv preprint arXiv:2101.06100* (2021).

[DL23] Justin Diamond and Markus Lill. “Geometric Constraints in Probabilistic Manifolds: A Bridge from Molecular Dynamics to Structured Diffusion Processes”. In: *arXiv e-prints* (2023), arXiv–2307.

[Hoo+23] Emiel Hoogeboom et al. “Equivariant Diffusion for Molecule Generation in 3D”. In: *International Conference on Machine Learning*. 2023, pp. 8867–8887. URL: https://arxiv.org/pdf/2203.17003.pdf.