International Journal of Innovation and Multidisciplinary Research
DOI.ORG/10.67505/IJIAMR.v6.i1.61007
Vol. 6 (1), 2026
E-ISSN: 2583-4452

# Quiver Representation of Neural Networks for Multi-Class Classification: A Traffic Sign Recognition Case Study

Mansi[1*] and Ritika Chopra[2]
[1] Department of Mathematics, Shaheed Rajguru College of Applied Sciences for Women, University of Delhi, Delhi
[*]Corresponding Author ✉ mansibhatt794@gmail.com

## ABSTRACT

Traffic management in cities has become extremely complex due to rapid urbanization, higher number of vehicles, and increased network of roads. Traditional methods that control traffic using fixed algorithms and scheduling models fail to adapt to changes in traffic. Although neural networks can predict and take decisions within intelligent traffic management systems, their unpredictability makes it difficult for them to be employed in situations when safety plays a critical role. This paper proposes the mathematical model for modeling neural networks based on quivers, where layers would be the vertices while connections would be the directed edges of the quiver, with a weight. This mathematical model provides a clear picture regarding how information flows through neural networks. The current research demonstrates the application of our method for classification of data and its stability. The reason for this is that in order to prove it, this research uses a feature contribution analysis of the inputs to the output as well as the stability analysis using the spectral norm to find out whether the model reacts to any perturbation in the input and parameters. The proposed method provides the researchers with a mathematical representation of the functioning of neural networks. The proposed framework is evaluated on the German Traffic Sign Recognition Benchmark (GTSRB). In the future, it is planned to generalize our framework for other network structures and learning algorithms.

***Keywords:*** *Stability Analysis, Feature Contribution Analysis, Linear Transformations, Vector Spaces*

## 1 Introduction

The efficient traffic management of the increasing number of vehicles and ever-changing nature of road situations has emerged as a major issue nowadays owing to the exponential rise in the number of vehicles and increasing rate of urbanization and growth in population. The increasing demand for transport services in cities causes traffic congestion, longer travelling time, high amount of fuel consumption, and a greater likelihood of causing road accidents. Apart from these, traffic congestion is also harmful to the environment because it results in increased air pollution and carbon emission, thus damaging both public health and environment.

Traffic behavior depends upon various variables including time of day, weather conditions, road constructions, accidents, and other unexpected situations; hence, managing traffic according to the pre-fixed set of criteria and rules is quite difficult in reality. The existing traffic management systems depend on fixed timings for signaling and programmed methods for regulating traffic and do not have the capacity to adapt to dynamic conditions of traffic. Since these systems lack the ability to adapt, they cannot play their role of controlling traffic. The development of intelligent systems through the use of machine learning techniques such as neural networks has been extensively examined to increase the efficiency of traffic control mechanisms. Neural networks have shown effectiveness in many applications including the identification of traffic signs, vehicle detection, and prediction of traffic flow. Learning from large amounts of data can be done by neural networks, making them a suitable technology for use in traffic control. Studies conducted by Rumelhart et al. (1986) marked the beginning of research related to neural networks with the introduction of the backpropagation method for training multi-layered networks through data learning. Subsequently, Hornik (1991) demonstrated that feed-forward neural networks could approximate all continuous functions.

The use of neural networks to analyze learning systems mathematically was further explored by Wood and Shawe-Taylor (1996) with the help of representation theory. This was followed up by the development of the representation theory of associative algebras, which was done by Assem et al. (2006). As computational power improved and larger data sets could be utilized, neural networks started to show good performance on practical problems. Convolutional neural networks were used to recognize traffic signs by Sermanet and LeCun (2011),

Received on: 10 Jun 2026 | Accepted on: 26 Jun 2026 | Published Online on: 25 July 2026

whereas deep neural networks were proven efficient at image classification by Ciresan et al. (2012). At around the same time, Stallkamp et al. (2012) developed the German Traffic Sign Recognition Benchmark dataset.

Further advances in the field were made by Bengio et al. (2013), who detailed how deep learning algorithms learn hierarchical representations of features, and Zeiler & Fergus (2014), who offered visualization techniques for understanding convolutional networks. Deep learning was also defined as layered representation learning by LeCun et al. (2015). Notable performance gains have been demonstrated by He et al. (2016), who developed residual networks capable of training very deep architectures. The period also saw Goodfellow et al. (2016) provide an extensive review of deep learning techniques. Li et al. (2016) revealed that convolutional neural networks work well on real-life traffic sign recognition tasks, while Redmon et al. (2016) presented the YOLO system for fast real-time object detection.

With increased complexity of neural networks, there is a need to understand the basis for decisions made by the network. Zeiler and Fergus (2014), as well as Selvaraju et al. (2017), developed techniques of visualization such as Grad-CAM that allow highlighting the regions of an image that affect the prediction of the network. These approaches provide the ability to comprehend the operations performed by convolutional neural networks on visual data. Rudin (2019) argued that black-box models should not be used when making critical decisions because they lack interpretability. Recently, Armenta and Jodoin (2021) studied neural networks through algebraic and representation approaches.

Despite the achievements mentioned above, neural networks do not have sufficient interpretability when used in safety-critical domains like traffic systems. In a mathematical sense, neural networks can be considered as directed graphs where neurons serve as nodes and edges represent the information flows. Thus, quiver representation theory can also be applied to neural networks, where nodes represent vector spaces and edges are linear transformations between these spaces. However, it has not been used effectively enough to investigate how neural networks handle information in traffic-related tasks.

To overcome these problems more efficiently, a new approach using quivers to model neural networks for traffic sign recognition tasks has been suggested. It aims at gaining insight into how inputs, such as images of traffic signs, affect outputs in a traffic recognition application. In other words, modeling of neural networks by using quivers makes it possible to analyze features, classification process, and system stability. All these aspects are significant when working with intelligent traffic systems.

## 2 Prelimanries

**Definition 1** *(Assem et al., 2006) The quiver can be mathematically defined as an ordered pair of sets together with two maps that define a source and target for each arrow . More precisely, it is written as*

$$Q = (Q_0, Q_1, s, t),$$

*where $Q_0$ denotes the set of vertices, $Q_1$ the set of arrows, and the maps $s, t : Q_1 \to Q_0$ specify where each arrow originates and where it terminates.*

**Definition 2** *(Assem et al., 2006) In a quiver representation, each vertex $v \in V$ gets assigned a vector space $W_v$. Each arrow $e \in E$ is given a linear map $W_e : W_{s(e)} \to W_{t(e)}$.*

**Definition 3** *(Armenta and Jodoin, 2021) The activation at node v with respect to $a(W, f)_v(x)$ is given as:*

$$a(W, f)_v(x) = f_v\left(\sum_{\alpha \in E_v} W_\alpha \cdot a(W, f)_{s(\alpha)}(x)\right) \tag{1}$$

**Definition 4** *(Armenta and Jodoin, 2021) A neural network may be considered as a function ( $f : \mathbb{R}^n \to \mathbb{R}^k$ ) that is expressed as a sequence of functions associated with layers. Normally, each layer implements a linear mapping followed by the application of an activation function. For a given input ( x ), each layer computes*

$$h = \sigma(Wx + b),$$

*where ( W ) and (b) are a weight matrix and bias vector, respectively, while ( $\sigma$ ) is an activation function.*

## 3 Quiver Representation of Neural Networks

The proposed framework combines neural networks with quiver representations to model traffic sign recognition mathematically. It aims to show how info moves through the network and shapes classification decisions.

Inputs, shown as feature vectors $x \in \mathbb{R}^n$, pass through a neural network that outputs $y$. This includes different layers that have weights, biases, and activation functions.

In order to provide a more mathematical structure to the neural network, we will consider the neural network to be a quiver $Q = (V, E)$. In the quiver, nodes will be considered as layers and edges will represent information flow. Each node has a vector space and the mapping between two nodes will be a linear map.

Hence, we will consider the neural network to be vector spaces with linear transformations between them.

### 3.1 Neural Network Architecture

A neural network is basically a series of affine transformations along with nonlinear activation functions. For traffic sign recognition, it takes images, analyzes key features, and matches them to a list of known traffic signs.

As illustrated in Fig. 1, the neural network consist of three layers, namely input, hidden, and output layer. Let $V_0 = \mathrm{R}^n$, $V_1 = \mathrm{R}^m$, and $V_2 = \mathrm{R}^k$ be the input, hidden, and output spaces. These spaces show how data moves through the network. Each layer, thus, does a specific kind of data transformation.

- **Input Layer** : The input layer uses $V_0 = \mathrm{R}^n$, where each vector $x \in \mathrm{R}^n$ comes from traffic sign images and shows details like color, shape, and text.

  Next up is an affine map that shifts the data into $V_1$:

  $$T_1(x) = W^{(1)}x + b^{(1)}, \tag{2}$$

  with $W^{(1)} \in \mathrm{R}^{m\times n}$ and $b^{(1)} \in \mathrm{R}^m$.

- **Hidden Layer** : The hidden layer works in the space $V_1 = \mathrm{R}^m$ and helps the network figure out patterns in the input data. To make the model more flexible, a nonlinear activation function $\sigma : V_1 \to V_1$ is used, applied individually to each component.

  Here, the ReLU function, short for Rectified Linear Unit, is choosen for this role. ReLU is defined as $\sigma(z) = \max(0, z)$.

  So, the hidden representation is calculated as $h = \sigma(W^{(1)}x + b^{(1)})$. This does a great work at bringing forward key features from the input while reducing noise and other unimportant stuff.

- **Output Layer** : The output layer is represented by the space $V_2 = \mathrm{R}^k$, which stands for the different traffic sign classes. It transforms the learned hidden representations into final classification results.

  This transformation is shown by

  $$T_2 : V_1 \to V_2,\ T_2(h) = W^{(2)}h + b^{(2)} \tag{3}$$

  where

  $$W^{(2)} \in \mathrm{R}^{k\times m},\ b^{(2)} \in \mathrm{R}^k.$$

  The output vector is

  $$y = T_2(h), \tag{4}$$

  with each part of $y$ telling us how well the input matches a certain traffic sign class.

In the end, this work produces a vector that gives scores for each class, which is what the model uses right before making its final decision.

### 3.2 Quiver Representation of the Model

The quiver can be mathematically defined as an ordered pair as defined in defination (Definition 1), illustrated in Fig. 2. Although the quiver is not meant to reflect the actual traffic in reality, it acts as an abstraction of what happens in the flow of information. The transformation associated with each arrow is responsible for the changes to the incoming information and thus fulfills a function equivalent to that of weight values in neural networks. The information transformation aspect is thereby easily incorporated into the quiver representation.

According to the defination (Definition 2) in a quiver representation, each vertex $v \in V$ gets assigned a vector space $W_v$. According to Armenta and Jodoin (2021), as defined in the defination 3.

The following describes the notation employed in the equation 1. $a(W, f)_v(x)$ corresponds to the activation (output representation) for the vertex $v$, whereas $a(W, f)_{s(\alpha)}(x)$ is the activation for the source vertex of edge $\alpha$. By $W_\alpha$ it means the learned linear map that takes the information from $s(\alpha)$ to $t(\alpha)$. The tuple $(W, f)$ stands for the definition of a neural network that operates over the quiver $Q$. it also have $f = (f_v)_{v\in Q0}$ standing for the nonlinear activation functions for vertices. The notation $x \in \mathrm{R}^n$ means the vectorized input. Finally, $E_v$ corresponds to the set of all edges whose target vertex is $v$.

The non-linearity within this particular model comes into existence because of the activation function $f_v$ that has been designed by using the ReLU function. The ReLU function converts all negative values into zeroes but keeps the positive values the same. This is very important for the model to be able to learn complex mappings.

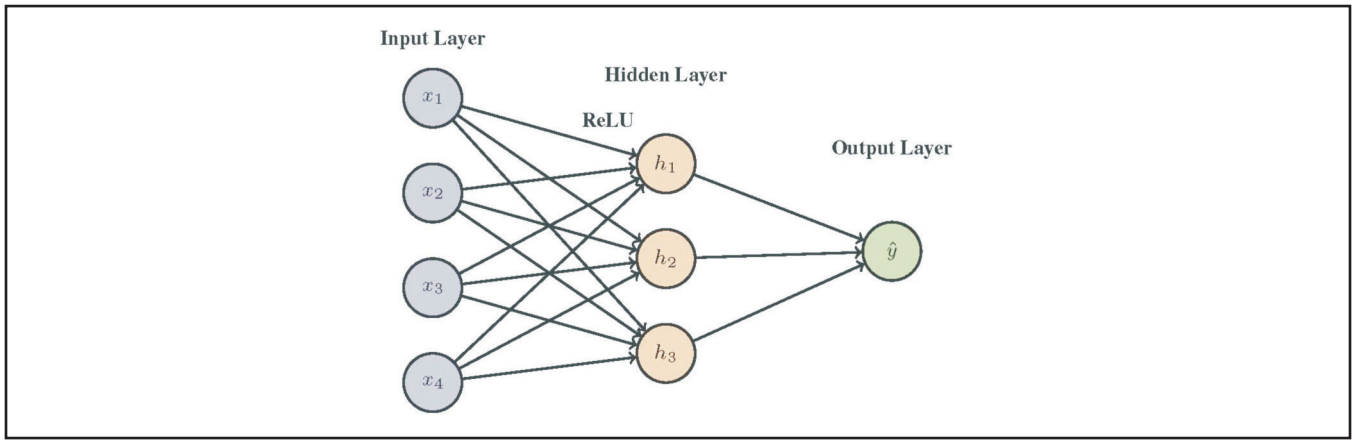


***Fig. 1:*** *Neural network architecture with ReLU activation*

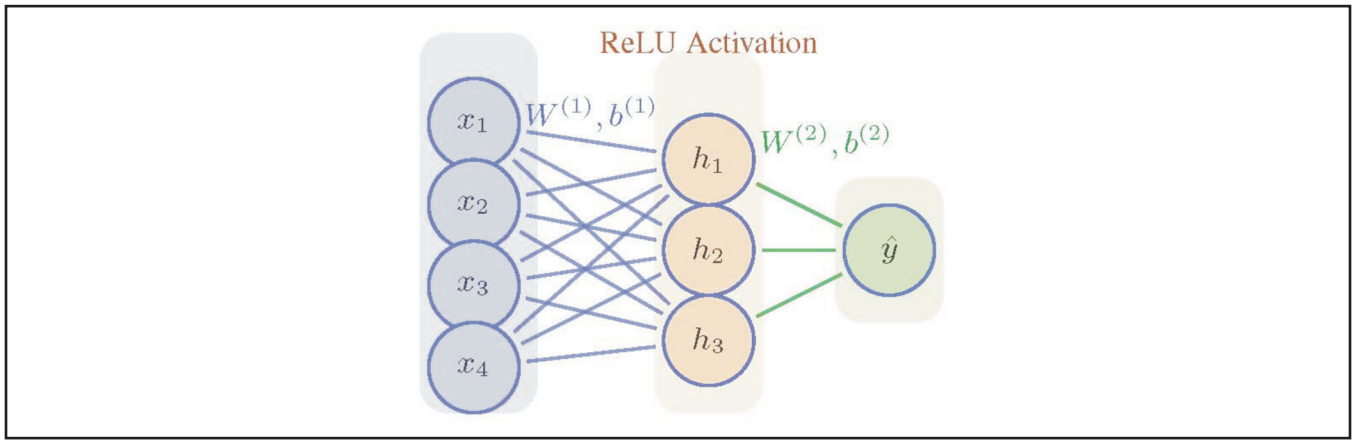


***Fig. 2:*** *Quiver-based representation of the neural network model*

### 3.3 Integration with Traffic Management System

This proposed system combines traffic sign recognition with traffic flow control to simplify decision-making in smart traffic setups. It begins with feature representations $x \in V_0$, where $V_0$ holds the traffic sign features.

Then, a neural network transforms these features into distinct categories using the mapping $f : V_0 \to V_2$.

Along the way, a hidden layer $h \in V_1$ appears to help the learning process from the initial inputs.

It's called a simple layered structure:

$$V_0 \xrightarrow{T_1} V_1 \xrightarrow{T_2} V_2.$$

At the end, the system gives $y \in V_2$, which offers scores for various traffic signs such as STOP, speed limit, yield, and directional signs. The vehicle uses the information provided here in determining what the expected actions should be from the vehicle. For example, in case of seeing a STOP sign, the vehicle will stop at an intersection. A yield sign can help determine which vehicle will pass through the intersection first. The sight of a speed limit sign slows down the vehicle, while directional signs make turns and changes in lanes easier. Here, we use the quiver representation to model the traffic structure. Here, the nodes represent the traffic states or locations, while edges denote movement between these locations. These nodes have feature representations extracted using a neural network.

In this work, vector spaces and linear transformations are used to describe how information flows in a system. Each traffic sign updates its vertex, showing changes in the graph. This represents data moving through the structure, influencing different parts as it goes.

### 3.4 Mathematical Framework For Classification

This work uses a neural network structure to classify traffic signs using images and explains how information from inputs is processed to get the output classification result.

A vector $x \in R^n$ is given to the network. This input is based on the important characteristics of traffic signs, such as shapes and colors, extracted from their images. For us to make computations on this input, it needs to be quantified, meaning all the traffic signs are represented as points in the feature space consisting of n dimensions.

There are multiple transformations performed on the data by the neural network during its computation. The first one is a transformation performed by the weight matrix $W^{(1)} \in R^{m \times n}$, and the bias vector $b^{(1)} \in R^m$, which produce an intermediate result of:

$$z^{(1)} = W^{(1)}x + b^{(1)} \tag{5}$$

The relationship between the inputs begins to be learned by the network as a result of the linear transformation and the blending of the input features in an appropriate proportion. The set of feature detectors is embedded in the weight matrix $W^{(1)}$, and every feature detector in it is capable of responding to a particular pattern within the input data. The use of the bias term $b^{(1)}$ allows the detector to operate independent of the input data.

In addition to the sequence of linear mappings, in this work, a nonlinear activation function is used on each intermediate result, one entry at a time. For that purpose, the most widely used activation function is the so-called Rectified Linear Unit or ReLU given by:

$$\text{ReLU}(z) = \max(0, z)$$

which leaves positive numbers intact and maps negative numbers into zeros.

The reason for the introduction of such nonlinearity lies in the fact that without it, multiple layers would simply combine into one linear mapping. By introducing nonlinearity, the model gets an opportunity to construct complex hierarchical decision surfaces that will allow it to effectively differentiate between the various types of traffic signs. As a consequence of ReLU application, the hidden representation becomes:

$$h = \text{ReLU}(z^{(1)}) = \text{ReLU}(W^{(1)}x + b^{(1)})$$

Then, the result is feed to the next layer, where another linear map takes place by applying matrix $W^{(2)}$ followed by adding the bias $b^{(2)}$:

$$T(x) = W^{(2)}\,\text{ReLU}(W^{(1)}x + b^{(1)}) + b^{(2)} \tag{6}$$

This equation characterizes the full mapping procedure performed by our two-layer network starting from input $R^n$ vector x until reaching its output in $R^k$, where $k$ stands for the number of traffic sign classes.Observe that the mapping is an affine function and behaves similar to the linear map

$$T(x) = W^{(2)}h \tag{7}$$

in terms of classification, feature analysis & stability.

Thus, the resultant is a vector,

$$T(x) = (y_1, y_2, \ldots, y_k)$$

In this vector, the component $y_i$ indicates the score of class $i$. The higher the value, the more related is the input data to this class. Hence, each value in the result vector corresponds to the score of a connection between the input and one specific traffic sign class.

It is possible to focus on individual classes through projecting the result vector:

$$P_i(T(x)) = y_i$$

In this case, each projection will be considered individually. In order to fairly compare the predicted scores, the absolute values are taken into account:

$$\|P_i(T(x))\| = |y_i|$$

Absolute values allow us to compare different scores in equal conditions and choose the highest value among all $k$ classes.

As a result, the predicted class is the following:

$$\text{Class}(x) = \arg\max_i |y_i|$$

At this stage, the network computes the output values for all classes simultaneously and takes the class with the highest absolute value as an answer.This neural network architecture stands out in its ability to demonstrate the synergy between linear and nonlinear transformations, which results in discriminative feature learning that is finally reduced to a single class label.

## 4 Analysis and Interpretation

To understand the behavior of the entire framework, the final perspective on the quiver model need to be consider. Modeling the network via the use of the quiver allows us to analyze how certain elements are connected within this structure, and how information is exchanged among components.

Under such conditions, many important characteristics become evident. Furthermore, understanding the effects of a certain component on the entire output becomes relatively easy. The major benefit of the proposed approach is that the model becomes more open. Each transformation within the system becomes more transparent both independently and in the context of the entire framework.

### 4.1 Feature Contribution Analysis

Here, the analysis focuses on the importance of individual input characteristics for the overall classification procedure. This is particularly important as all features have unequal influence on the prediction process, since some input features contribute significantly to the resulting prediction, whereas other features are relatively insignificant. Therefore, the understanding of those differences plays a crucial role in the interpretation of how the model functions and what level of significance each particular input feature possesses.

Each arrow in the quiver is associated with a matrix that governs the way linear transformation of the input is performed along the edge, hence signaling about the relevance of various characteristics as they flow along the layers of the model. In a mathematical sense, the arrow represents a linear mapping:

$$W_\alpha : \mathrm{R}^n \rightarrow \mathrm{R}^m$$

that shows how a vector x in $\mathrm{R}^n$ results in a vector in $\mathrm{R}^n$ .

To determine how much of an influence each input element has on the output, this work connect each feature to its respective standard basis vector $e_j \in \mathrm{R}^n$, Specifically, $e_j$ refers to the vector consisting of zeroes with one located at position $j$. Then, by multiplying $W_\alpha$ with the above vector, this work obtain:

$$W_\alpha e_j$$

The latter equals the $j$-th column of the $W_\alpha$. In other words, it describes the extent of influence that the $j$-th feature has on the transformed output. If a feature, acting separately from others, can strongly move the output within the space, then it is considered significant because it leads to a certain shift in the generated representation to the next node.

To measure this significance numerically, the following norm of the obtained vector is computed:

$$\mathrm{Contribution}(j) = \|W_\alpha e_j\| \tag{8}$$

Thus, this work obtain a concise quantitative representation of the feature's influence on the transformation performed by the node. The higher the value of the norm, the greater the impact of the feature on the output. In contrast, a lower value indicates that the feature exerts a minor influence.

This approach allows one to analyze the influence of all input features quantitatively to get insight into their impact on the resulting output of the classifier framework. In particular, by evaluating a contribution of every $j \in \{1, 2, \ldots, n\}$ it is possible to produce a comprehensive picture of the feature importance within the whole feature space. Those features that have significant contributions will be considered highly discriminative for the task under consideration, whereas features with low contribution values might not carry useful information at all and can even be redundant. Hence, one can easily identify features of an input image of a traffic sign that influence classification results to the highest degree.

### 4.2 Stability Analysis

The stability aspect entails how slight changes in input affect the behavior of a particular framework. In case the framework exhibits stability, slight changes in input cause proportional changes in its output while in case of instability, slight changes in input will lead to huge changes in output. Stability plays a very vital role in classifying a certain framework since it helps us determine whether the framework produces reliable output when there are slight changes in input, which in our case would mean slight differences in lighting conditions, angles at which the camera takes photos, among others.

The following shows an illustration of the arrows and their associated linear transformations in the quiver framework:

$$\mathbb{R}^n \xrightarrow{T} \mathbb{R}^k$$

The input vector belongs to the input space:

$$x \in \mathrm{R}^n$$

and after the transformation is applied, the output is given by:

$$T(x) \in \mathrm{R}^k$$

Every such linear transformation can be expressed in matrix form as:

$$T(x) = Ax, \quad A \in \mathrm{R}^{k \times n} \tag{9}$$

Here, $A$ is the matrix of the linear mapping T. In other words, the entire information about the transformation behavior is encoded in A, and analyzing the matrix $A$ allows one to obtain valuable information on the behavior of the corresponding arrow in the quiver and, in particular, to judge its stability.

To study the stability, one needs to consider the matrix norm $\|A\|$, which indicates the maximum possible stretching effect of the transformation. The value $\|A\|$ is critical to determine whether the transformation remains stable or becomes amplifying.

The stability condition of the transformation is determined by $\|A\|$. For:

$$\|A\| < 1 \quad \Rightarrow \quad \textbf{Stable transformation}$$

This transformation either retains the magnitude or decreases it. Here, it will either decrease or limit the magnitude of the input such that an input with slight change will produce an output with slight change. Therefore, the system works in a controlled manner and can be trusted in classification even where the input may be slightly noisy.

Conversely, when:

$$\|A\| > 1 \quad \Rightarrow \quad \textbf{Unstable transformation}$$

This property would definitely cause an increase in the amount of data used as input. The slightest change that is introduced to the input data would definitely have a large effect on the output that is produced. In terms of classification, this particular property can be viewed as quite undesirable because any small change in the input image would produce completely different output.

## 5 Result and Discussion

This section presents the experimental results for the model's performance, including stability, feature extraction, and classification. The model checks off several categories, including STOP, speed limit, yield, and directional signs, when tested on a common traffic sign recognition dataset. These results center on classification accuracy, feature representation learning performance, and input variation stability.

### 5.1 Case Study: Traffic Sign Recognition

The primary objective of this case study is to apply the proposed framework to the traffic sign identification problem. The objective is to categorize different types of traffic signs using real-world image data and evaluate how well the model performs in real-world situations.

#### 5.1.1 Dataset

The proposed model is estimated using the German Traffic Sign Recognition Benchmark (GTSRB), which is a benchmark dataset for traffic sign classification. As shown in Fig. 3, the dataset consists of various traffic sign categories. This dataset includes real-world images of traffic signs taken in various conditions like changes in lighting, viewing angles, scale, and background complication.

Every image comes with structured information including details about the traffic sign region, including the image width and height, and the coordinates of the region of interest (ROI). The ROI coordinates tell us where exactly the traffic sign is located in the image. This helps focus on the important bits of the image and makes feature extraction more efficient.

In addition to the labeling, each image also has the type of sign labeled, whether a speed limit, warning, or regulatory sign. For input into the model, the image are represented in terms of features $x \in V_0$. Here, $V_0$ is known as the input feature space, containing visual information such as shape, color, edges, and structure.

All this structure makes the GTSRB great for testing how well multi-class classification models can mark different types of traffic signs. With its multiple set of images, it gives us a good way to measure how the proposed model handles real-life variations in traffic sign appearances.

The dataset is obtained from the Kaggle platform: *https://www.kaggle.com/datasets/meowmeowmeowmeowmeow gtsrb-german-traffic-sign*

| | Width | Height | Roi.X1 | Roi.Y1 | Roi.X2 | Roi.Y2 | ClassId | Path |
|---|---|---|---|---|---|---|---|---|
| 0 | 27 | 26 | 5 | 5 | 22 | 20 | 20 | Train/20/00020_00000_00000.png |
| 1 | 28 | 27 | 5 | 6 | 23 | 22 | 20 | Train/20/00020_00000_00001.png |
| 2 | 29 | 26 | 6 | 5 | 24 | 21 | 20 | Train/20/00020_00000_00002.png |
| 3 | 28 | 27 | 5 | 6 | 23 | 22 | 20 | Train/20/00020_00000_00003.png |
| 4 | 28 | 26 | 5 | 5 | 23 | 21 | 20 | Train/20/00020_00000_00004.png |
| 5 | 31 | 27 | 6 | 5 | 26 | 22 | 20 | Train/20/00020_00000_00005.png |
| 6 | 31 | 28 | 6 | 6 | 26 | 23 | 20 | Train/20/00020_00000_00006.png |
| 7 | 31 | 28 | 6 | 6 | 26 | 23 | 20 | Train/20/00020_00000_00007.png |
| 8 | 31 | 29 | 5 | 6 | 26 | 24 | 20 | Train/20/00020_00000_00008.png |
| 9 | 34 | 32 | 6 | 6 | 29 | 26 | 20 | Train/20/00020_00000_00009.png |
| 10 | 36 | 33 | 5 | 6 | 31 | 28 | 20 | Train/20/00020_00000_00010.png |
| 11 | 37 | 34 | 5 | 6 | 32 | 29 | 20 | Train/20/00020_00000_00011.png |
| 12 | 38 | 34 | 5 | 6 | 32 | 29 | 20 | Train/20/00020_00000_00012.png |
| 13 | 40 | 34 | 6 | 6 | 34 | 29 | 20 | Train/20/00020_00000_00013.png |
| 14 | 39 | 34 | 5 | 5 | 34 | 29 | 20 | Train/20/00020_00000_00014.png |

***Fig. 3:*** *Sample numerical data from the GTSRB (German Traffic Sign Recognition Benchmark) dataset*

### 5.1.2 Model Implementation and Results

This model uses vectors to represent information from traffic signs, where each characteristic is represented numerically. This makes it possible for us to work on the data in an organized manner.

Each traffic sign will have its features categorized into two groups depending on the nature of the features.

The first group is made up of basic image properties such as width and height, which may be represented as:

$$x_1 = \begin{bmatrix} \text{Width} \\ \text{Height} \end{bmatrix} \in \mathbb{R}^2$$

The second contains the ROI coordinates indicating the exact location of the traffic sign within the image.

$$x_2 = \begin{bmatrix} \text{Roi.X1} \\ \text{Roi.Y1} \\ \text{Roi.X2} \\ \text{Roi.Y2} \end{bmatrix} \in \mathbb{R}^4$$

These two vectors are combined into a single input vector:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2 \oplus \mathbb{R}^4 = \mathbb{R}^6$$

This vector entirely represents a traffic sign example containing the sign's size and position within the image.To understand the way the data is being processed by the model, it is useful to consider various vector spaces:

$$V_1 = \mathrm{R}^2,\ V_2 = \mathrm{R}^4,\ V_3 = \mathrm{R}^6,\ V_4 = \mathrm{R}^{43}$$

In this case, $V_1$ and $V_2$ are the input feature spaces, $V_3$ is the feature space that contains both vectors, and $V_4$ represents the output feature space. The elements of $V_4$ are related to traffic sign classes.

Using a vector space representation, a neural network can be modeled with a quiver. As shown in Fig. 4, this shows how different vector spaces relate through linear transformations. The mappings between them are described by:

$$W_{\alpha 1} : R^2 \rightarrow R^6,\ W_{\alpha 2} : R^4 \rightarrow R^6$$

Since the two paths produce outputs belonging to $R^6$, it means that they are able to mix information about different feature sets. Their sum, processed by a nonlinear activation function (ReLU) is used as the hidden representation:

$$v_3 = \text{ReLU}(W_{\alpha 1} x_1 + W_{\alpha 2} x_2) \in R^6 \quad (10)$$

In this way, the hidden vector makes the processing more effective by combining the information about the object's size and position, thus improving its representation.

Finally, the hidden representation is converted to the output space by another transformation:

$$W_{\alpha 3} : R^6 \rightarrow R^{43}$$

The resulting output belongs to $R^{43}$, having each dimension correspond to some traffic sign classes. The maximal value will correspond to the predicted class. The quiver diagram will help visualize the process with transformations depicted by arrows and model layers by nodes.

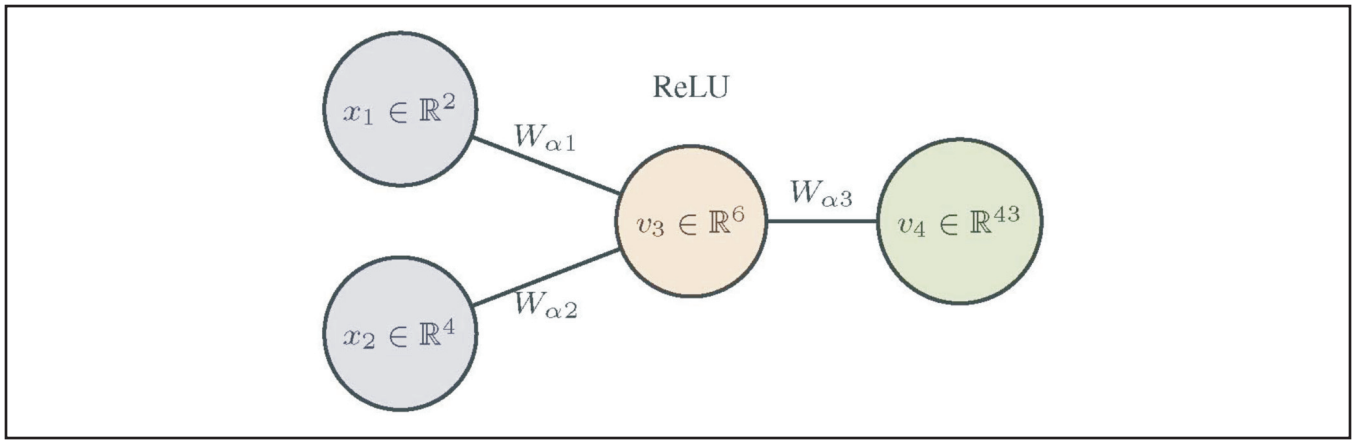


***Fig. 4:*** *Quiver representation of neural network showing feature flow across vector spaces*

In this arrangement, the inputs pass through their respective layers and combine at the hidden layer to give rise to the ultimate decision. Any slight change in the value of the inputs or weight matrices will not result in a significant difference in output and will not affect the predicted class either. This is an indicator of the stability of the model.

This neural network first takes the input features through a series of operations until it generates the output vectors within $R^{43}$. In other words, the classifier generates 43 numbers for an input data sample where each number represents the relationship between the input and one of the 43 classes of traffic sign images. Rather than reducing the inputs directly into a label, it calculates a value representing its association with each of the 43 classes simultaneously. This means that during classification, the algorithm considers the relationships with all possible competing classes simultaneously since the classes of traffic sign have very much resemblance to each other.

These generated values represent their contribution scores, meaning higher absolute values suggest stronger relationships. For example, the classification depends on how the 43 output values compare to each other, and the rule assigns the input to whichever class has the highest score despite how large or small each value is in absolute terms. The transformation from input to output is represented as:

$$T(x) \in R^{43}$$

It will be the class whose corresponding element in the vector is of the highest magnitude. That means this work select the index of the largest projection out of the 43 classes.

$$\text{Class}(x) = \arg\max_i \|P_i(T(x))\| \quad (11)$$

To illustrate this, consider the output vector returned by the neural network when fed with an input:

[. . . , -0 00019 ,0 000003 ,0 000155 ,0 000516 , . . .]

Despite being insignificant, the slight variations among them become meaningful when making a selection. The fourth element, 0.000516, clearly represents the highest value from the entire 43 elements, thus forming the basis of the classification decision. It is not essential for the framework to generate large absolute figures; rather, the need is for a single element to have a significant value compared to others. The margin between the highest value 0.000516 and the second-highest 0.000155 may seem small in absolute figures, but it provides enough information for making a definite decision. This will give us the index at which the highest number is present in the output vector. Here, each element in the output vector indicates the score for the class. This score decides what class is assigned by the classifier. Here, the element with the index 21 has the highest number in comparison to all other elements; Thus, the predicted class becomes:

$$\text{Class}(x) = 21$$

This proves that the framework works independent of the magnitude of the output values. Regardless of the fact that the 43 elements lie close to zero in absolute terms, the hierarchy among them remains unaltered, and it predicts the right class without fail. Such an ability is significant in traffic sign classification, since different classes resemble one another and consequently the output elements cluster around each other. However, in such scenarios, the framework chooses the class which stands out among the remaining 43 as having a slightly greater element than the others. For instance, the output elements are differentiated from each other by as little as 0.000516 - 0.000155 = 0.000361

The model uses two feature vectors that describe different parts of a traffic sign. The first one, $x_1$, contains data on the object size and consists of the dimensions – width and height of the sign. The second, $x_2$, contains information related to location – the Region of Interest's coordinates. These feature sets go through separate weight matrices before meeting up in the hidden layer. This lets the model handle and weigh both the size and position information effectively, so it works better overall.

Regarding the feature contribution, the results are as follows:

$$[0.0191, 0.0187]$$

It can be seen that there is no substantial difference between width and height – each of them provides a similar amount of importance to the input layer of the hidden layer. Thus, the combined contribution from these two features is relatively small in comparison with the ROI features, implying that this type of information is rather supplementary than decisive for the classification.

On the other hand, the coordinates of the ROI affect the same hidden neuron $v_3$ in the following way:

$$[0.0197, 0.0224, 0.0248, 0.0307]$$

While still below the contributions of the size features, the contributions increase progressively from one coordinate to the next. While 0.0197 is already on equal footing with the size features in terms of importance, 0.0307 is by far the highest contribution (the contribution of ROI.Y2) and triggers the most dramatic change compared to any other input variable. From this increasing progression from 0.0197 to 0.0307 within the ROI coordinates, this work can deduce that vertical information is more discriminative than horizontal information.

In terms of the head-to-head comparison between the two sets of inputs, the prioritization scheme is obvious and consistent throughout the framework. When compared against the features related to the size, the ROI features have a tendency to provide higher contributions to the output at all four positions. In terms of the variations, the width and height of a road sign could change greatly depending on the distance between the observer and the road sign and the perspective from which the observer observes the road sign. However, the location of the road sign within the predefined region of interest could be considered as more reliable and consistent cues about the structure of the road sign and the class to which the road sign belongs.

After passing through the hidden layer, the hidden neurons themselves create the contributions to the output neuron $v_4$. The contribution generated by each hidden neuron will be:

$$[0.0551, 0.0636, 0.0704, 0.0539, 0.0668, 0.0784]$$

These six hidden neurons do not contribute equally towards the final classification output. They contribute anywhere from 0.0539 to 0.0784, which shows significant variations as far as their contribution is concerned. It means that while Neuron 6 has the highest contribution of 0.0784, contributing the maximum push for final output in the output layer, Neuron 4 has the minimum contribution of 0.0539. However, despite the minimum value of its contribution, its input cannot be considered insignificant either. The other four neurons have values of 0.0551, 0.0636,0.0704, and 0.0668 respectively, forming a gradient in between.

The difference between the maximum contribution (Neuron 6) and the minimum contribution (Neuron 4) is just 0.0245. It means a difference of 45% between these two neurons. Although the exact numerical difference may not appear very significant, it shows a definite difference between the two neurons.

In fact, such an analysis reveals a very logical pattern that includes the presence of the same priority criterion at each level of the hierarchical model under consideration. Specifically, the positional attributes play a more important role than size attributes in the process of processing input data; the ROI.Y2 attribute with its value of 0.0307 is the one that plays the biggest role in terms of inputs. In case of the hidden layer, it is hidden node 6 with its weight 0.0784 that has the greatest influence on the output data. This shows a certain hierarchical nature of specialization in the weight matrices.

One could say that a neural network is essentially an organized linear transformation pipeline, going from one vector space to the other. In the context of quiver algebra, each linear transformation becomes the arrow representing a clear process through which information passes from the input layer all the way to the output.

All the input data collected in advance forms a 6-dimensional space, and further linear transformations take the data into 43 dimensions – each dimension for each of the traffic sign classes:

$$T : \mathrm{R}^6 \rightarrow \mathrm{R}^{43}$$

Thus, the input to the function above is the 6-dimensional vector, consisting of the sizes of both traffic sign and ROI, described above. As for the output of the linear transformation T, this is a 43-dimensional vector, corresponding to each of the traffic sign classes. This expansion from the input space to a substantially greater output

space is determined by the nature of the problem itself: a small amount of information needs to be represented as a large vector, describing the strength of association with any of 43 categories.

In order to simplify the process, all the mappings in the network are combined into a single composite mapping matrix. This input mapping matrix is first constructed by putting together the weight matrices from the two input features in parallel:

$$W_{in} = [W_{\alpha 1}\ W_{\alpha 2}]$$

This figure depicts the combination of the two input vectors as one input vector x consisting of size features x1 and ROI coordinates $x_2$. Input-to-hidden interactions can be represented by the matrix $W_{in}$, thus combining all input-to-hidden mappings to one linear operator. It will make our analysis more simple because now it has one term instead of several arrows coming from the input.

The complete input-to-output mapping can be found by multiplying $W_{in}$ by $W_{\alpha 3}$:

$$A = W_{\alpha 3} W_{in},\ A \in \mathrm{R}^{43\times 6} \tag{12}$$

where $A$ belongs to the real numbers set of dimension 43 × 6. Thus, $A$ is a matrix of dimension 43 × 6 that transforms any point from 6-dimensional input to 43-dimensional output without any intermediate steps. By reducing the two transformations to one, it can simplify the stability analysis and treat it for one linear operator

instead of several arrows.

To verify whether the structure satisfies the condition of stability, the spectral norm of A will be examined:

$$\|A\| = \sqrt{\lambda_{\max}(A^T A)} \tag{13}$$

If the system works with the traffic signs data, the result obtained:

$$\|A\| = 0.0026$$

This is an extremely low value. Stating differently, a spectral norm of 0.0026 is equal to A contracting all vectors no more than 0.26%. All in all, the condition of stability is satisfied not only by a small amount but by a rather large one because the magnitude of the output will always be much smaller than the magnitude of the input.

As

$$\|A\| = 0.0026 < 1$$

it is clear that the system is definitely stable. That is, any change of an input value cannot influence the output in any significant way.

In other words, the effect of a slight disturbance in the input on the output is insignificant; the transformation process reduces the variation before reaching the output layer. This makes the neural network extremely suitable for classification, because the output vector remains almost unchanged even with some disturbance or noise in the input. The stability margin, which is calculated based on the distance between the value of $\|A\|$ and the critical point 1, amounts to 0.9974 – a truly impressive value. Also, the stability of the framework proves the continuity of the transformation described in the weight matrices, as the spectral norm is considerably lower than 1, meaning that the transformation from input to output does not have any discontinuous jumps.

## 6 Limitations

This research is mostly theoretical and gives a mathematical model for the description of data flow with the help of quiver representations and affine transformations. This model is verified by testing on the German Traffic Sign Recognition Benchmark, but this study is still primarily concerned with the mathematical model rather than experimentation and comparison of results with other benchmark models.

The testing is done using only one dataset; therefore, the applicability of this algorithm on other traffic sign datasets or in other conditions is not addressed. Moreover, in the practical scenario, traffic signs may be affected by some noise and blur.

## 7 Conclusion

In this paper, neural networks have been analyzed as directed graphs for gaining better insights. Using the concepts of quiver representation theory, the nodes have been regarded as vector spaces and edges as linear transformations. This made it possible to give a detailed mathematical insight into the working of a neural network. For practical use, this approach has been used in traffic sign recognition along with an evaluation of feature contributions and stability. In conclusion, combining mathematical concept with neural networks is beneficial for better insights and analysis. In this study the theoretical concepts and real-world applications are combined effectively and can be applied to solve more complicated problems in the future.

## Conflict of Interest Statement

The authors declare no conflict of interest.

## Ethical Declarations

The authors declare that this research does not involve human participants, animals, or any procedures requiring ethical approval. All data used in this study were obtained and processed in accordance with applicable ethical & legal guidelines.

## References


1. Armenta, M., and Jodoin, P.-M. (2021). The representation theory of neural networks. *Mathematics, 9*(24), 3216.
2. Assem, I., Simson, D., and Skowroński, A. (2006). *Elements of the representation theory of associative algebras*. Cambridge: Cambridge University Press.
3. Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 1798–1828.
4. Cires¸an, D. C., Meier, U., and Schmidhuber, J. (2012). Multi-column deep neural networks for image classi-fication. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3642–3649.
5. Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. Cambridge: MIT Press.
6. He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.
7. Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks, 4*(2), 251–257.
8. LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444.
9. Li, J., Wang, Z., and Liu, C. (2016). Traffic sign detection using convolutional neural networks. *Neurocom-puting, 214*, 723–734.
10. Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 779–788.
11. Rudin, C. (2019). Stop explaining black box machine learning models for high-stakes decisions and use interpretable models instead. *Nature Machine Intelligence, 1*(5), 206–215.
12. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*(6088), 533–536.
13. Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, 618–626.
14. Sermanet, P., & LeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. In *Pro-ceedings of the International Joint Conference on Neural Networks*, 2809–2813.
15. Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). The German traffic sign recognition benchmark: A multi-class classification competition. *Neural Networks, 32*, 323–332.
16. Wood, J., & Shawe-Taylor, J. (1996). Representation theory and invariant neural networks. *Discrete Applied Mathematics, 69*(1–2), 33–60.
17. Zeiler, M. D., and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision*, 818–833.


❖ ❖ ❖