GALAXY INTERNATIONAL INTERDISCIPLINARY RESEARCH JOURNAL (GIIRJ)
ISSN (E): 2347-6915
Vol. 11, Issue 10, Oct. (2023)

REGULATION OF THE TRANSPORT SYSTEM BASED ON THE KOHONEN ALGORITHM

Djurayev Sherzod Sobirjonovich
Namangan Institute of Engineering and Technology
sherzoddjurayev1989@gmail.com

Fayzullayev Dedaxuja Zokirjon o`g`li
Namangan Institute of Engineering and Technology
fdedaxuja@gmail.com

ABSTRACT

Efficient regulation of transport systems is essential for minimizing congestion and ensuring smooth traffic flow. Traditional traffic regulation methods often rely on fixed signal timings, which may not adapt well to changing traffic conditions. In recent years, intelligent algorithms, such as the Kohonen algorithm (self-organizing map or SOM), have emerged as promising solutions for dynamically regulating transport systems. This article provides an overview of the Kohonen algorithm's mathematical expression and explores its potential applications in transport system regulation.

Keywords: Kohonen Algorithm, Self-Organizing Map (SOM), Unsupervised Learning, Convergence Speed, Decaying Learning Rate, Decreasing Neighborhood Radius, Initialization Bias, K-Means Initialization, Optimal Parameters, Grid Search, Cross-Validation, Dimensionality Reduction, Principal Component Analysis (PCA)

INTRODUCTION

The efficient regulation of transport systems is crucial for ensuring smooth traffic flow and minimizing congestion. Traditional approaches to traffic regulation often rely on fixed signal timings or pre-defined control strategies, which may not adapt well to changing traffic conditions. In recent years, intelligent algorithms have been explored to improve traffic regulation. One such algorithm is the Kohonen algorithm, which offers a promising solution for dynamically regulating the transport system. This article aims to provide an overview of the mathematical expression of the Kohonen algorithm and its potential applications in transport system regulation.

The Kohonen Algorithm: The Kohonen algorithm, also known as the self-organizing map (SOM), is a type of artificial neural network that can be used to cluster and classify data. It was developed by Teuvo Kohonen in the 1980s and has since found applications in various fields, including pattern recognition and data visualization. The algorithm is based on unsupervised learning, where the network learns to identify patterns and relationships in the input data without explicit labels or targets.

Mathematical Expression: The Kohonen algorithm involves a competitive learning process, where neurons in the network compete to become the "winning" neuron for a given input. The winning neuron, also known as the best matching unit (BMU), is selected based on its proximity to the input vector. The mathematical expression of the Kohonen algorithm can be summarized as follows:

1. Initialization:

- Initialize the network with random weights for each neuron.
- Define the learning rate (α) and the neighborhood radius (σ).

2. Training:

- Select an input vector from the training dataset.
- Calculate the Euclidean distance between the input vector and the weights of each neuron.
- Identify the BMU, which is the neuron with the smallest distance to the input vector.
- Update the weights of the BMU and its neighboring neurons based on the following formula: $\Delta W = \alpha * h * (X - W)$

ΔW: Weight update
α: Learning rate
h: Influence of the neighborhood function
X: Input vector
W: Weights of the neuron

3. Repeat step 2 for a specified number of iterations or until convergence is achieved.

Applications in Transport System Regulation: The Kohonen algorithm can be applied to regulate the transport system by dynamically adjusting traffic signal timings based on real-time traffic conditions. By analyzing data such as vehicle flow, speed, and occupancy, the algorithm can identify traffic patterns and adapt signal timings accordingly. This approach allows for more efficient traffic flow, reduced congestion, and improved travel times.

The Kohonen algorithm offers a promising solution for the regulation of transport systems. By utilizing unsupervised learning and competitive processes, the algorithm can adaptively adjust traffic signal timings based on real-time traffic conditions. This can lead to improved traffic flow, reduced congestion, and enhanced overall efficiency of the transport system. Further research and implementation of the Kohonen algorithm in real-world scenarios can help revolutionize traffic regulation and contribute to the development of smart and sustainable transport systems.

While the Kohonen algorithm, with the expression $DW = a * h * (X - W)$, is a powerful tool for clustering and classification tasks, it may encounter certain challenges or limitations. Here, we propose a solution to address some of these problems:

1. Convergence Speed: The standard Kohonen algorithm may require a large number of iterations to converge, especially for complex datasets. To mitigate this issue, one can introduce a decaying learning rate (α) and a decreasing neighborhood radius (σ) over time. This allows the algorithm to converge faster by gradually reducing the influence of neighboring neurons and fine-tuning the weights.

2. Initialization Bias: The initial random weights assigned to neurons can impact the convergence and quality of the clustering. To overcome this, one can use a more intelligent initialization strategy. For example, the K-means algorithm can be employed to initialize the weights by clustering the input data beforehand. This ensures a better starting point for the Kohonen algorithm and potentially improves the final clustering results.

3. Determining Optimal Parameters: The performance of the Kohonen algorithm heavily relies on the appropriate selection of parameters such as the learning rate (α) and neighborhood radius (σ). These parameters need to be carefully tuned to achieve optimal results. One approach is to employ a grid search or cross-validation technique to systematically explore different parameter combinations and select the ones that yield the best performance on a validation dataset.

4. Handling High-Dimensional Data: The original Kohonen algorithm may struggle with high-dimensional datasets due to the curse of dimensionality. To address this, dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) can be applied to reduce the dimensionality of the input data while preserving its structure. This allows the Kohonen algorithm to operate on a lower-dimensional space, improving its performance and interpretability.

5. Dealing with Noisy or Outlier Data: The Kohonen algorithm is sensitive to noisy or outlier data, as they can significantly affect the clustering results. One approach to mitigate this issue is to preprocess the data by applying outlier detection or noise reduction techniques. This can involve methods such as median filtering, Gaussian smoothing, or robust statistical measures to identify and remove or correct noisy or outlier data points before feeding them into the algorithm.

By addressing these challenges and incorporating the proposed solutions, the Kohonen algorithm can be enhanced to deliver more robust and accurate results in various clustering and classification tasks. These improvements can contribute to its wider adoption and application in real-world scenarios, including transport system regulation.

To address the problems in the Kohonen algorithm using the expression $DW = a * h * (X - W)$, let's analyze each problem mathematically:

1. Convergence Speed: To improve convergence speed, we can introduce a decaying learning rate (α) and a decreasing neighborhood radius (σ) over time. The updated formulas for α and σ can be:

$\alpha(t) = \alpha\_initial * \exp(-t/\tau)$ $\sigma(t) = \sigma\_initial * \exp(-t/\tau)$

Where t is the current iteration, τ is a time constant, and α_initial and σ_initial are the initial learning rate and neighborhood radius, respectively. By decaying α and σ over time, the algorithm gradually reduces the influence of neighboring neurons and fine-tunes the weights, leading to faster convergence.

2. Initialization Bias: To overcome initialization bias, we can use a more intelligent initialization strategy. One approach is to employ the K-means algorithm to initialize the weights by clustering the input data beforehand. Let's denote the K-means centroids as C and the weights of the neurons as W. The initialization step can be modified as follows:

$W = C$

By initializing the weights with the K-means centroids, we ensure a better starting point for the Kohonen algorithm, reducing the impact of random initialization and potentially improving the final clustering results.

3. Determining Optimal Parameters: To determine the optimal values for the learning rate (α) and neighborhood radius (σ), we can employ a grid search or cross-validation technique. By systematically exploring different parameter combinations and evaluating their performance

on a validation dataset, we can select the values that yield the best results. Let's denote the validation performance as P(α, σ), the set of possible α values as A, and the set of possible σ values as S. The optimal parameters can be found as:

α_optimal, σ_optimal = argmax(P(α, σ)) for α in A, σ in S

By selecting the parameter values that maximize the validation performance, we ensure that the algorithm performs optimally on unseen data.

4. Handling High-Dimensional Data: To handle high-dimensional data, we can apply dimensionality reduction techniques such as Principal Component Analysis (PCA) or t-distributed Stochastic Neighbor Embedding (t-SNE) before feeding the data into the Kohonen algorithm. Let's denote the dimensionality-reduced data as X_reduced. The modified algorithm can be summarized as:

5. Initialize the network with random weights for each neuron.

6. Apply dimensionality reduction to the input data: X_reduced = PCA(X) or X_reduced = t-SNE(X).

7. Training:

- Select an input vector from the dimensionality-reduced training dataset.
- Calculate the Euclidean distance between the input vector and the weights of each neuron.
- Identify the BMU, which is the neuron with the smallest distance to the input vector.
- Update the weights of the BMU and its neighboring neurons based on the formula: ΔW = a * h * (X_reduced - W).

8. Repeat step 3 for a specified number of iterations or until convergence is achieved.

By reducing the dimensionality of the input data, the modified algorithm operates on a lower-dimensional space, improving its performance and interpretability.

9. Dealing with Noisy or Outlier Data: To handle noisy or outlier data, we can preprocess the data by applying outlier detection or noise reduction techniques. Let's denote the preprocessed data as X_processed. The modified algorithm can be summarized as:

10. Initialize the network with random weights for each neuron.

11. Apply outlier detection or noise reduction techniques to the input data: X_processed = preprocess(X).

12. Training:

- Select an input vector from the preprocessed training dataset.
- Calculate the Euclidean distance between the input vector and the weights of each neuron.
- Identify the BMU, which is the neuron with the smallest distance to the input vector.
- Update the weights of the BMU and its neighboring neurons based on the formula: ΔW = a * h * (X_processed - W).

13. Repeat step 3 for a specified number of iterations or until convergence is achieved.

By preprocessing the data to remove or correct noisy or outlier data points, we ensure that the Kohonen algorithm is less sensitive to such data, leading to more accurate clustering results.

By incorporating these modifications into the Kohonen algorithm, we can address the aforementioned problems and enhance its performance in clustering and classification tasks.

CONCLUSION

We have discussed several problems that can arise in the Kohonen algorithm and proposed mathematical formulas as solutions to address these issues. By implementing these modifications, we can improve the convergence speed, overcome initialization bias, determine optimal parameters, handle high-dimensional data, and deal with noisy or outlier data.

The introduction of a decaying learning rate and neighborhood radius allows for faster convergence by gradually reducing the influence of neighboring neurons and fine-tuning the weights. Employing a more intelligent initialization strategy using K-means centroids ensures a better starting point for the algorithm, reducing the impact of random initialization. Determining optimal parameters through techniques like grid search or cross-validation helps select values that maximize performance on unseen data. Applying dimensionality reduction techniques such as PCA or t-SNE allows the algorithm to handle high-dimensional data more effectively. Preprocessing the data to remove or correct noisy or outlier data points improves the algorithm's robustness and accuracy.

By incorporating these modifications into the Kohonen algorithm, we can enhance its performance in clustering and classification tasks, making it more accurate and reliable for various applications. It is important to note that the specific implementation details and parameter values may vary depending on the dataset and problem at hand. Therefore, it is recommended to experiment and fine-tune these modifications based on the specific requirements of each application.

REFERENCES

1. Барский А.Б. Нейронные сети: распознавание, управление, принятие решений. – М.: Финансы и статистика, 2004. – 176 с.
2. Каллан Роберт. Основные концепции нейронных сетей: Пер. с англ. – М.:Издательский дом «Вильямс», 2001.
3. Комарцова Л.Г., Максимов А.В. Нейрокомпьютеры: Учеб. пособие для вузов. – 2-е изд.,перераб. и доп. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2004. – 400 с. Технические науки — от теории к практике № 11 (47), 2015 г. www.sibac.info
4. Мелихова О.А., Чумичев В.С., Джамбинов С.В., Гайдуков А.Б. Некоторые аспекты криптографического взлома и повышения надежности алгоритмов шифрования// Молодой ученный. – Казань, № 11(91), 2015. –С. 392–394.
5. Мелихова О.А. Приложение матлогики к проблемам моделирования// Известия ЮФУ. Технические науки. – Таганрог: Изд-во ТТИ ЮФУ, 2014. № 7(156). – С. 204–214.
6. Madaliyev , X. «Creation of interface through app design of matlab software for automatic determination of loads on roller machine worker shaft». Interpretation and Researches, т. 1, вып. 10, июнь 2023 г.,
7. Fayzullayev Dedakhuja Zokirjon oglu. (2022). Mexbios development studio software package for developing control programs and modeling electric drive systems. Web of Scientist: International Scientific Research Journal, 3(5), 1964–1967. https://doi.org/10.17605/OSF.IO/G9BF8

8. Ismonovich K. A., Abdusamatugli I. M. Modeling the Method of Linear Approximation of Signals in SPLC (Sensor Programmable Logic Controller) //International Journal on Orange Technologies. – 2021. – T. 3. – №. 10. – C. 55-59.
9. Ismonovich K. A., Muhammadziyo I. Mathematical Modeling of Heat Flux Distribution in Raw Cotton Stored in Bunt //Engineering. – 2020. – T. 12. – №. 8. – C. 591-599.