sustainability

MDPI

Article

Intricate Supply Chain Demand Forecasting Based on Graph Convolution Network

Tianyu Niu, Heng Zhang *, Xingyou Yan and Qiang Miao

College of Electrical Engineering, Sichuan University, Chengdu 610065, China; niutianyu@stu.scu.edu.cn (T.N.); yan_xingyou@foxmail.com (X.Y.)
* Correspondence: hengzhang27@scu.edu.cn

Abstract: Globalization has contributed to the increasing complexity of supply chain structures. In this regard, precise demand forecasting for the intricate supply chain holds paramount importance in effective supply chain management. Traditional statistical models and deep learning methods often face challenges in efficiently discerning correlations within a myriad of interconnected demands. To tackle this issue, this paper proposes an intricate supply chain demand forecasting method based on graph convolution networks adept at handling non-Euclidean data. First, the companies within the supply chain are treated as nodes in the graph structure, and the relationships between them are treated as edges, with demand data serving as the features of these edges. Then, a graph convolutional network is constructed to aggregate node and edge information. Through a multi-layer network, the relationships among nodes, edges, and historical demand are established to facilitate the prediction of supply chain demands. In this process, the graph convolutional network incorporates supply chain connectivity information into demand time series analysis. This integration of surface-level topological features and deeper latent correlation attributes across the supply chain's nodes refines the demand forecasting precision across the entire supply chain. The validation experiment in this paper is grounded in sales data of a singular product from multiple sales nodes of an electronics company. The results demonstrate that the proposed method surpasses four other traditional demand forecasting algorithms significantly in terms of accuracy, providing substantial evidence for the superior performance of graph networks in the analysis of intricate supply chain relationships.

Keywords: demand forecasting; supply chain management; graph neural network; deep learning

check for updates

Citation: Niu, T.; Zhang, H.; Yan, X.; Miao, Q. Intricate Supply Chain Demand Forecasting Based on Graph Convolution Network. *Sustainability* **2024**, *16*, 9608. https://doi.org/10.3390/su16219608

Academic Editor: Alessio Ishizaka

Received: 28 August 2024
Revised: 14 October 2024
Accepted: 2 November 2024
Published: 4 November 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Demand forecasting entails predicting future customer demand for a specific product, holding significant importance in the broader context of supply chain planning, encompassing production planning [1] and inventory management [2]. Accurate demand forecasting provides critical support for production planning and inventory management, effectively reducing operational costs and enhancing profitability [3]. Through demand forecasting, firms can optimize production scheduling and avoid overproduction or inventory buildup, thereby minimizing resource waste and storage expenses. Furthermore, precise demand forecasting increases a company's flexibility in responding to market fluctuations. Consequently, demand forecasting plays a vital role in supply chain management, significantly improving operational efficiency and competitive advantage [4]. Demand forecasting can be viewed as a time series forecasting task, typically relying on historical data and its temporal features [5]. However, the supply chain encompasses not only time-related features but also complex topological relationships, where the supply and demand interactions among different entities form a complex supply chain network. These relationships often exhibit intricate interactions. The demand of each entity is influenced not only by its own demand and associated features but also by the competitive and cooperative dynamics with related entities [6]. When the available supply chain data are limited to historical demand

data, traditional time series forecasting methods fail to capture the potential interactions between different entities and struggle to achieve effective predictions due to the lack of feature data. Considering the connection information among different entities embedded in the historical demand data, applying graph theory can transform these entities and their relationships into graph data. By employing graph analytical methods, it becomes possible to extract the underlying interactions among different entities from the limited historical demand data, thereby having the potential to significantly improve the accuracy of demand forecasting.

Deep learning methods have been widely applied to time series forecasting tasks, where traditional deep learning typically utilizes Euclidean space data. However, graph data, such as social networks, e-commerce [7], and citation networks [8], represent irregular non-Euclidean structures, posing challenges for traditional deep learning methods to effectively extract features from such data. Consequently, graph neural networks have been developed to address this issue. Their distinctive structure and mechanisms empower them to adeptly capture the intricate features of graph data, overcoming the limitations encountered by traditional deep learning methods when confronted with irregular graph structures [9]. Thus, graph neural networks (GNNs) offer a more potent and adaptable framework for the analysis of graph data [10]. Currently, research on graph-based supply chain demand forecasting is limited. Exploring how to structure supply chain demand data in graph form and how to utilize graph neural networks for demand prediction are important areas for investigation.

This article endeavors to reconstruct supply chain relationships by conceptualizing entities within the supply chain as nodes and the relationships between them as edges. The approach involves amalgamating historical demand data to formulate a time-series graph representing supply chain demand, providing a more effective representation of the supply chain's topological information, which facilitates the optimal utilization of interaction features between supply chain nodes. Subsequently, a graph convolutional network model is established for extracting and processing the features of internode correlations within the supply chain as well as the features of historical demand data and is employed for forecasting the demand across the entire supply chain. The primary contributions of this study are outlined as follows:

- Transforming an intricate supply chain network comprising numerous entities into a graph data structure enriches the depiction of interenterprise relationships.
- A multi-layer GCN model efficiently processes graph data to predict retailer demand for distributors, leveraging the topology and historical demand.
- Case studies and comparative experiments are executed to highlight the superiority of GCN for forecasting in complex supply chain contexts.

The subsequent sections of this paper are structured as follows: Section 2 reviews the pertinent literature related to the research work. Section 3 presents an elucidation of the graph neural network model formulated in this paper. In Section 4, a practical case is introduced for the analysis and demonstration of the proposed model and algorithm. Section 5 provides a summary of the research study and outlines potential future directions.

2. Related Work

In recent years, research on demand forecasting has been broadly categorized into two main methodological approaches as follows: statistical methods and machine learning methods. This section will elaborate on the relevant research content within these two categories.

2.1. Traditional Time Series Forecasting Methods

Traditional time series forecasting methods, such as exponential smoothing [11,12] and autoregressive integrated moving average [13,14], are commonly employed to predict future trends and periodic features in time series data. However, relying solely on historical data for predictions makes it challenging to achieve accurate results [15]. To address this,

researchers have integrated multiple forecasting models and employed data preprocessing techniques to enhance prediction accuracy. For example, Garcia-Ascanio et al. [16] applied Vector Autoregressive (VAR) and iMLP (interval MultiLayer Perceptron) models to interval time series, achieving monthly electricity demand forecasting for Spain over the next two years at an hourly resolution. Es et al. [17] developed a novel gray seasonal forecasting model for predicting monthly natural gas demand in Turkey, demonstrating superior predictive accuracy compared to SGM and SARIMA models. Petropoulos et al. [18] analyzed the predictive performance of different method and time combinations in the context of intermittent demand, proving that appropriate combinations can enhance predictive performance and simplify the forecasting process. Guo et al. [19] addressed issues with the lack of sophistication in demand forecasting methods for repairable aircraft spare parts and inconsistencies between basic forecast data and actual consumption. They proposed a dual-layer combination forecasting method based on relevant data, which achieves more accurate predictions consistent with actual demand. Hu et al. [20] addressed the challenge of energy forecasting by integrating neural networks with gray prediction models, thereby developing interval gray prediction models that function without statistical assumptions. The proposed models performed well compared to other interval gray prediction models. Li et al. [21] introduced an uncertain time series model that leverages annual data to forecast urban household water demand, demonstrating higher accuracy and ease of use compared to traditional prediction models. Although the improved methods can enhance prediction accuracy to some extent, they are limited in their ability to address the complexities of a networked supply chain comprising multiple entities. These methods can only predict linear supply chain relationships and fail to capture the intricate interactions between different entities.

Traditional time series forecasting methods often require additional modeling efforts to establish the relationships between entities, which not only increases complexity but may also limit the model's flexibility and adaptability. In contrast, machine learning approaches possess the ability to automatically extract and construct relational features between different entities from data, thereby simplifying the modeling process and enhancing efficiency. This characteristic enables machine learning methods to more effectively identify potential non-linear features and interactions. Moreover, machine learning methods outperform traditional time series methods in data processing capabilities, allowing them to handle more complex and diverse data [22]. Therefore, machine learning-based methods have become one of the important directions in demand forecasting research.

2.2. Machine Learning Methods

Machine learning methods are broadly categorized into two main types, namely, traditional machine learning methods and deep learning methods. Traditional machine learning methods are support vector regression (SVR), Decision Trees, and Multilayer Perceptron (MLP). For example, Xu et al. [23] proposed a SARIMA-SVR hybrid model for aviation demand forecasting, highlighting the optimal accuracy of SARIMA-SVR3 through experiments and proving that incorporating Gaussian white noise is able to increase forecasting accuracy. Kumar et al. [24] integrated fuzzy reasoning and artificial neural networks to construct a framework for a big data-driven fuzzy classifier, surpassing baseline time series data prediction methods in terms of optimality, efficiency, and other statistical indicators. Lu et al. [25] combined variable selection methods and support vector regression to establish a hybrid forecasting model for computer product sales, with the proposed solution showing lower error than comparative models and identifying crucial predictive variables. Spiliotis et al. [26] compared seven machine learning methods trained both in a series-by-series and a cross-learning fashion, revealing that some methods achieve superior results in terms of accuracy and bias in predictions. Traditional machine learning methods rely on prior knowledge and expert experience, which limits their adaptability in complex systems. This makes it challenging to capture the dynamic data features and

complex relationships within supply chain systems, thereby restricting their predictive capabilities and accuracy in practical applications.

On the other hand, deep learning methods encompass various network structures and their variants, such as convolutional neural networks [27,28] and recurrent neural networks [29,30]. These methods learn abstract representations of data through multi-layered neural network models, offering stronger feature learning and representation capabilities suitable for handling large-scale, high-dimensional data. Abbasimehr et al. [31] achieved demand data forecasting for a furniture company using a multi-layer LSTM network approach, automatically selecting the optimal prediction model through grid search, outperforming other comparative time series forecasting methods in performance indicators. Bi et al. [32] proposed a computer vision-based deep learning model encoding tourism demand data into images for prediction, demonstrating the model's performance compared to seven benchmark models. Bandara et al. [33] associated individual time series predictions with relevant time series, uniformly representing associated information using a global training long short-term memory network, validating the superior performance of the proposed method using real online market data from Walmart.com. Rodrigues et al. [34] combined text information with time series data, introducing a network model that integrates natural language processing and time series prediction. By fusing two complementary cross-modal information sources, they significantly reduced errors in predictions. These studies demonstrate the advantages of deep learning in extracting features from Euclidean data. However, when confronted with non-Euclidean data from complex supply chains with irregular topological structures, such methods fail to fully capture the potential associative features between different supply relationships, thereby limiting their adaptability in demand forecasting. Kosasih et al. [35] introduced graph neural networks into automotive supply chain analysis, achieving better predictive results than existing algorithms. While this study did not extend to demand forecasting, it confirmed the efficacy of graph neural networks in addressing supply chain complexities.

In summary, limited research exists on applying graph neural network methods to supply chain demand forecasting. Existing deep learning research has not fully considered associative features and model adaptability. To address this issue, this paper focuses on the topological structure of supply chains, transforming supply chain relationships into graphs. Then, utilizing graph neural networks to model and analyze the supply chain by capturing more associative features in the supply chain, thereby improving the accuracy of demand forecasting.

3. Methodology

This section first introduces the fundamental theory of GCN and then provides a detailed explanation of the proposed method from four aspects as follows: data processing, hyperparameter settings, model construction, and model training evaluation.

3.1. Graph Convolutional Network

The graph convolutional network (GCN) [36] is a deep learning model explicitly crafted for processing graph-structured data. In contrast to traditional convolutional neural networks, which are tailored for regular grid data, GCN is adept at handling graph data. This characteristic renders it particularly suitable for the analysis and prediction of supply chain structures. Its inherent compatibility with the topology of supply chains enables its effective application in analyzing and forecasting the dynamics within supply chain systems.

Consider a graph data set comprising N nodes, each endowed with D-dimensional feature information. Let the $N \times D$-dimensional matrix denoted by the node features be X, while the relationships between nodes are captured by the adjacency matrix denoted as A. The elements of the adjacency matrix represent the existence of connections between nodes and represent the weights of edges. Each row in X represents the feature vector of one node, and the elements of A represents the topological structure of connections between

nodes along with potential weight information. The combination of X and A forms the input for the GCN.

The GCN employs various propagation formulas between layers, outlined as follows:

$$H^{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{1}$$

where $\hat{A} = A + I$, and I is the unit matrix, referred to as the self-connection term; $\hat{D}$ is the diagonal matrix; $W^{(l)}$ is the weight matrix for the *l-th* layer, which is a trainable parameter in the GCN network; $H^{(l)}$ is the node representation for the *l-th* layer, and $H^{(0)}$ represents the input node features X; and σ is the activation function.

The inclusion of a self-connection term I in the adjacency matrix $\hat{A}$ within the propagation formula serves the purpose of enabling each node to take into account its own information. This ensures the preservation of the original information and attributes of the node, contributing to the control of information propagation intensity. This measure helps prevent an undue dependence on information from neighboring nodes in the graph convolutional layer, thereby maintaining a certain reliance on the node's own features. The subsequent step involves normalizing the adjacency matrix $\hat{A}$, and the calculation method is as follows:

$$\hat{A}' = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}} \tag{2}$$

where $\hat{d}_{ii} = \sum_j \hat{a}_{ij}$. Specifically, for node i and its neighboring node j, each having corresponding degrees d_i and d_j, the calculation method for the adjacency matrix element is as follows:

$$\hat{a}'_{ij} = \frac{\hat{a}_{ij}}{\sqrt{d_i \cdot d_j}} \tag{3}$$

Following normalization, the resulting adjacency matrix $\hat{A}'$ becomes a symindicator matrix, with each element representing the normalized connection strength between corresponding nodes. The normalized adjacency matrix, when multiplied by the node representation $H^{(l)}$, facilitates the transmission and aggregation of node features. Subsequently, a linear transformation is executed using the learnable weight matrix $W^{(l)}$, followed by the application of a non-linear activation function, leading to the formation of the new node representation $H^{(l+1)}$. This iterative process, achieved through the stacking of multiple layers, enables nodes to gradually converge and propagate information within the graph structure.

GCN accomplishes the propagation and aggregation of node features by normalizing the adjacency matrix, executing a linear transformation with the combination of node features, and mapping them to a new feature space. The incorporation of a non-linear activation function enhances the ability of the model to capture non-linear relationships. This iterative process stacks multiple layers to integrate and transmit information within the graph structure, enhancing the model's capabilities for various graph-related tasks.

3.2. Proposed Method

This article seeks to develop a model capable of accurately predicting demand between various nodes within a complex supply chain. Leveraging the adaptability of GCN to supply chain data and their demonstrated superior performance, we construct and train a multi-layer GCN model to forecast overall demand data in the supply chain. The schematic diagram of the proposed method is illustrated in Figure 1.

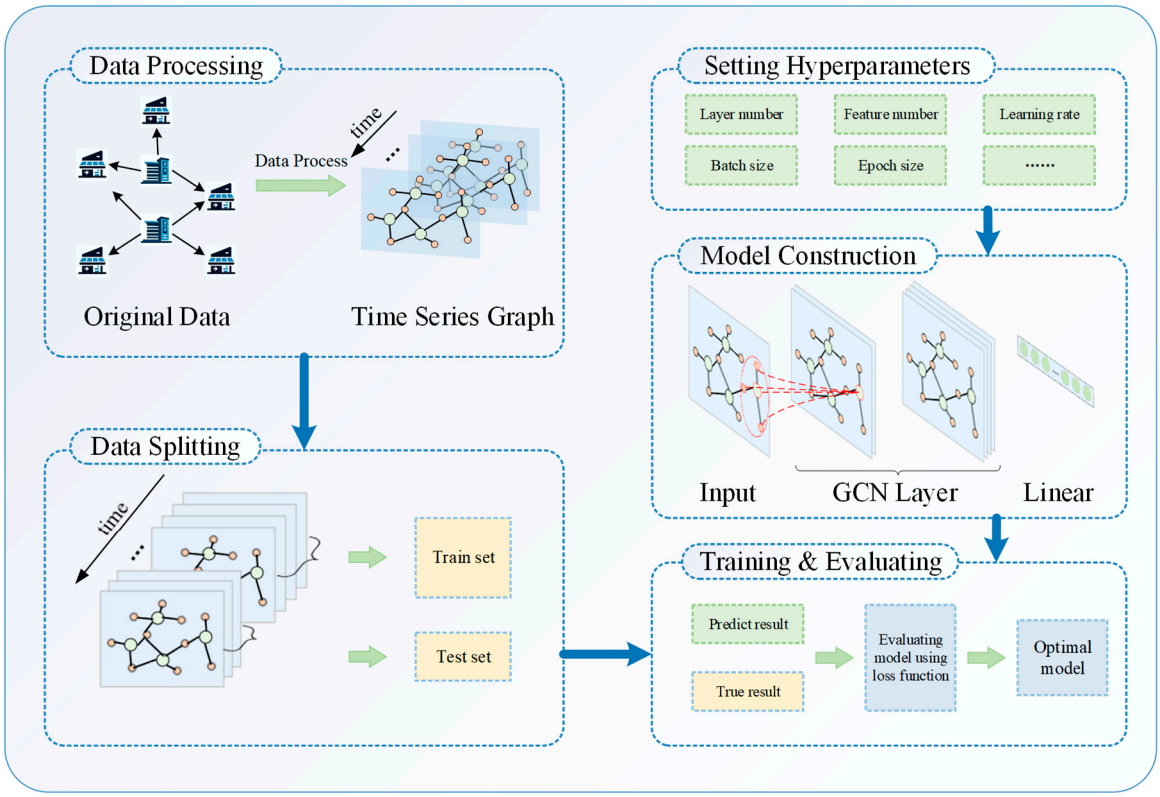


Figure 1. The structure of the proposed method.

The method proposed in this article comprises five main components as follows: data processing and dataset splitting, setting hyperparameters, model construction, and model training and evaluation. The specific details of each component are outlined as follows:

3.2.1. Data Processing and Splitting

Given that the original supply chain data lacks a graph structure and is unsuitable for direct utilization in graph neural network models, preprocessing of the raw data becomes imperative. The initial data must undergo transformation into a time series graph format to facilitate effective training and prediction within graph neural network models.

The process of constructing a time series graph involves two distinct steps as follows: supply chain decomposition and time series graph construction. Let $G = (V, E)$ be the supply chain graph, where $V = \{v_1, v_2, \ldots, v_n\}$ represents the set of nodes and $E = \{e_1, e_2, \ldots, e_n\}$ represents the set of edges denoting connections between nodes.

First, the supply chain is decomposed at the topological level. Different entities within the supply chain are abstracted into nodes v_i in the graph. Then, the entity types and other node features are constructed into the node feature vector $X \in \mathbb{R}^{n \times m}$, where n represents the number of nodes and m represents the dimensionality of node features. Second, historical demand data between different nodes a_{ij}^t, combined with edge $e_k = (v_i, v_j)$, are converted into the adjacency matrix A of the supply chain graph at each time step. Finally, the topological information of the graph and the weight information of the edges are amalgamated to obtain a time series graph sequence $T = \{G_1, G_2, \ldots, G_t\}$ containing t time steps, where $G_i = (X, A)$. The construction process of the time series graph is elucidated in Figure 2.

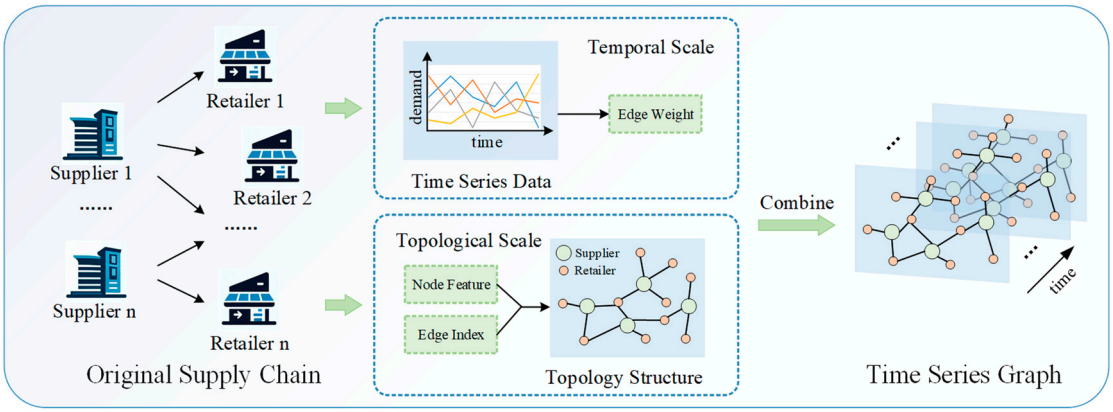


Figure 2. Processing original supply chain data into graph.

Different entities within the supply chain are abstracted into nodes v_i in the graph, and representations of edges $e_k = (v_i, v_j)$ are derived based on the connections between nodes. Subsequently, a node feature vector $F_v \in \mathbb{R}^{n \times m}$ is constructed, encapsulating the information inherent to each node, with m representing the dimensionality of node features. This establishes the topological structure of an individual graph. Next, the original supply chain data are decomposed into demand data w_{ij}^t between distinct nodes at the time scale, constituting the weights of edges within the graph. Finally, the topological information of the graph and the weight information of the edges are amalgamated to obtain a time series graph sequence $T = \{G_1, G_2, \ldots, G_t\}$ containing t time steps, where $G_i = (V, E, W_e)$. The construction process of the time series graph is elucidated in Figure 2.

In the data preprocessing stage, after transforming the original supply chain data, it is important to divide the time series graph data into a training set and a test set. This step ensures the model can generalize well to unseen data after training. Typically, the training set contains most of the data and is organized in chronological order to maintain the temporal sequence. Each data sample comprises an input graph G_i and an output graph G_{i+1} serving as the label.

The main role of the training set is to provide the model with diverse data for effective learning of patterns and features. During training, the model extracts features and makes predictions based on this dataset. After training, the model is validated using the test set, which contains unseen data. This validation assesses the performance of the model and generalization ability. The results on the test set provide an objective measure of predictive effectiveness and validate its practical use in real-world situations. Dividing the data into training and test sets, along with the evaluation process, are essential steps in training and validating machine learning models, ensuring their reliability and robustness.

3.2.2. Setting Hyperparameters

Although graph neural networks are well-suited for supply chain graph data, effective feature extraction and prediction require careful tuning of various hyperparameters. These hyperparameters must be set in advance and cannot be learned during model training, significantly impacting model performance. Therefore, ongoing adjustments before training are essential for achieving optimal results. The key tunable hyperparameters in this process include the following:

1. Number of GCN Layers: Increasing the number of hidden layers may augment the model's capacity to learn complex graph structure features, but it concurrently introduces complexity. However, an excess of GCN layers can lead to diminished model performance. Hence, striking a balance in the model's expressive power based on the graph data's complexity is essential, requiring the selection of an appropriate number of GCN layers.

2. Number of Features: GCN feature extraction denotes the number of dimensions in the features output by each graph convolutional layer. The determination of the number of features is directly linked to the abstraction and capture of feature information within the graph data. Augmenting the number of features may enhance the model's expressive power, but it could also result in underfitting and a notable increase in computational burden.
3. Learning Rate: The learning rate serves as a hyperparameter governing the step size of model parameter updates. A higher learning rate might induce unstable convergence during training, whereas a lower learning rate could lead to excessively slow training.
4. Epoch Size: The number of iterations represents how many times the entire training dataset is cycled through for model learning. An excessive number of iterations may foster overfitting, while an insufficient number may result in underfitting.

The hyperparameter configuration entails assigning different values to each hyperparameter, generating a range of hyperparameter combinations, and experimenting to determine the optimal combination for the final model parameters.

3.2.3. Model Construction

The model proposed in this paper comprises multiple layers of GCN for the extraction of graph features, along with a linear layer for regression prediction. In the initial GCN layer, the node features X of the input graph G_i serve as $H^{(1)}$, and the adjacency matrix A is calculated following equation (1), yielding the node representations $H^{(2)}$ forwarded to the subsequent layer. Post-processing by the initial GCN layer, the output node representations $H^{(2)}$ encapsulate comprehensive features amalgamating supply chain topology information and demand data. This $H^{(2)}$ is then fed into the successive GCN layer, where multiple layers of the network engage in profound extraction of node representations, culminating in the ultimate representation $H^{(l)}$ of the input graph data G_i by the GCN network, with l denoting the number of GCN layers. Ultimately, a linear layer is employed to regress $H^{(l)}$, yielding demand prediction results for the subsequent time step, specifically the predicted adjacency matrix $\hat{A}$ for G_{i+1}. This framework achieves demand prediction for diverse connections within the supply chain network.

3.2.4. Model Training and Evaluating

After constructing the model, it is trained with the training set and then evaluated with the test set. During training, all samples in the training set must be processed iteratively. For each sample, the model predicts the demand matrix $\overline{A}$, and the disparity between the predicted matrix $\overline{A}$ and the actual demand matrix A is quantified using the defined loss function. An optimization algorithm is then iteratively applied to adjust the parameters $W^{(l)}$ of each GCN layer in the model, with the objective of minimizing the loss. This iterative optimization process is meticulously designed to ensure that the model adeptly captures the intricate relationships between nodes within the supply chain network, consequently enhancing the precision of demand prediction.

Following the completion of training, the model is evaluated with the test set. This assesses its performance on unseen data using various indicators. The evaluation validates the model's effectiveness with new data, ensuring its practicality and robustness in real supply chain settings. It involves analyzing demand prediction accuracy and generalization ability, providing insights into the model's performance beyond the training data.

4. Case Study

In the assessment and analysis of the proposed algorithm's performance, this paper undertakes a comparative study involving various algorithm models. The comparison includes traditional statistical methods such as ARIMA, machine learning-based models like MLP and SVR, and deep learning-based LSTM. ARIMA models were executed and optimized using the Statsmodels package in Python, while the SVR models were imple-

mented and optimized using the scikit-learn package in Python. The remaining models were implemented using PyTorch 1.31.1 in Python 3.10.

4.1. Data Description

The experimental analysis uses data from a home appliance sales company, including 7 distributors and 150 retail stores. Each distributor supplies multiple retail stores, creating a networked supply network. The data are sampled daily, with each record containing the demand timestamp, distributor, retail store, and demand quantity. Covering three years from 30 December 2019 to 30 December 2022, this dataset reflects the company's operational dynamics.

During preprocessing, the temporal aspect is emphasized by dividing the data into daily segments. Daily demand data are combined with the supply chain topology and then organized into time series graphs, resulting in 1097 graphs. For model training and evaluation, the processed data are split chronologically into a training set (80% of the total data) and a test set (20%).

4.2. Evaluation Indicators

The experiment utilizes root mean square error (RMSE) and mean absolute percentage error (MAPE) to comprehensively analyze the performance of all models. RMSE quantifies the magnitude of deviation between predicted values and actual values, providing a measure of the overall accuracy. Meanwhile, MAPE is employed to assess the percentage deviation of predicted values relative to actual values, offering insights into the precision of the predictions. The definitions of these two evaluation indicators are outlined as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \hat{y}_t)^2} \tag{4}$$

$$\mathrm{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100\% \tag{5}$$

where n represents the sample size, y_i denotes the actual demand vector among various nodes at time t, and $\hat{y}_t$ signifies the predicted demand vector among various nodes at time t.

4.3. Experimental Parameter Settings

To identify the optimal model parameters, this study meticulously designs a series of hyperparameter sequences for systematic exploration of the model's hyperparameter configurations. Through iterative model training processes aligned with the hyperparameters presented in Table 1, our objective is to comprehensively assess the model's performance across various configurations. The ultimate aim is to pinpoint the hyperparameter combination that yields the best performance in demand prediction tasks.

Table 1. Values specified for each hyperparameter.

Hyperparameter	Values
The number of GCN layer	[1, 2, 3, 4]
The number of feature	[8, 16, 32, 64, 128]
Learning rate	[0.01, 0.001, 0.0001, 0.00001]
Epoch size	[50, 100, 200, 300, 400]

Following training with various hyperparameter combinations, the optimal model parameters are identified as a 2-layer GCN with 32 feature parameters per layer, utilizing a learning rate of 0.0001 for 200 epochs. Tables 2 and 3 display the model performance indicators obtained by individually tuning a single hyperparameter based on an optimal parameter set, with the optimal parameters highlighted in bold. This set of hyperparameters

employs the Adam optimizer, with RMSE serving as the loss function. The trained model is then applied to predict the test set.

Table 2. Comparison of hyperparameter experimentation 1.

The Number of GCN Layer	RMSE	MAPE	Learning Rate	RMSE	MAPE
1	6.662	12.037	0.01	14.671	41.93
2	3.298	8.562	0.001	6.514	17.933
3	6.535	21.868	0.0001	3.298	8.562
4	6.944	21.631	0.00001	7.167	21.196

Table 3. Comparison of hyperparameter experimentation 2.

The Number of Features	RMSE	MAPE	Epoch Size	RMSE	MAPE
8	6.326	19.292	50	10.057	25.073
16	3.719	9.336	100	6.630	16.582
32	3.298	8.562	200	3.298	8.562
64	6.044	15.586	300	5.761	15.636
128	5.904	15.757	400	6.308	15.867

Given the daily volatility in actual supply chain demand, the 7-day moving average prediction results demonstrate heightened utility. The application of a 7-day moving average effectively suppresses short-term fluctuations, rendering it more adaptable to demand dynamics in the real supply chain. This, in turn, furnishes more reliable information for supply chain management. Furthermore, the consideration of a seven-day cycle aligns closely with actual business operations, enhancing the accuracy of data support for decision-making in inventory management and production planning. Therefore, Figure 3 presents the original demand prediction results between specific nodes alongside the prediction results incorporating the 7-day moving average. Simultaneously, the original prediction indicators for all nodes of the model are 3.298 RMSE and 8.562 MAPE, while the 7-day moving average prediction indicators are 1.334 RMSE and 3.370 MAPE.

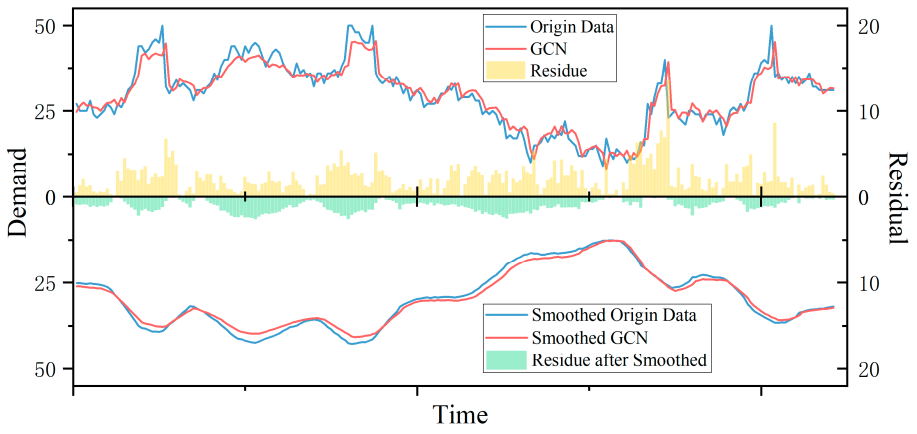


Figure 3. Prediction result of GCN.

The prediction results reveal that the original GCN model effectively aligns with the overall trend of demand data at the daily scale. Although there is a certain degree of lag in predicting demand changes at abrupt points, the results demonstrate the ability to track short-term fluctuations in demand. Upon adopting a 7-day average demand, the disparity between demand predictions and actual values notably diminishes, and the prediction residuals show a substantial reduction. This signifies a significant enhancement in the model's predictive performance at the weekly time scale. In conclusion, the GCN model

adeptly anticipates demand changes at the daily scale and accurately tracks demand at the weekly scale, achieving more precise forecasting.

4.4. Comparison with the Widely-Used Algorithm

To demonstrate the superiority of the proposed method, this section selects four methods for comparative analysis as follows: ARIMA, SVR, MLP, and LSTM, which include traditional time series forecasting methods, machine learning methods, and deep learning methods.

4.4.1. Experimental Settings for Comparison

It is important to note that the chosen comparative method is specifically designed for processing and forecasting a single time series. Consequently, the data employed in the comparative experiments exhibit variations. ARIMA independently trains and predicts demand sequences for each node, whereas other algorithms utilize datasets constructed using sliding windows with a size of 7 days. These datasets incorporate demand data from all nodes, and the sliding window is applied to predict data for the subsequent day. The parameter configurations for the comparative method are outlined in Table 4.

Table 4. Parameter settings for the comparative method.

Method	Parameters
ARIMA	Configured by Auto-ARIMA function
SVR	Kernel: Radial Basis Function (RBF) Penalty parameter (C): 1.0 Epsilon: 0.1
MLP	Number of Layers: 2 Number of units in the hidden layer (n): 256 Activation function: ReLU Learning rate: 0.0001
LSTM	Number of Layers: 2 Number of units in the hidden layer (n): 64 Learning rate: 0.0001

The comparative experiments employed corresponding datasets and performed training and prediction based on the specified experimental parameters. The training set-to-test set ratio was established at 8:2. Subsequently, the prediction results, denoted as the demand predictions $\hat{y}_i$ from each node, were amalgamated to construct a matrix $y_{pred} = [\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n]^T$ comprising n rows of predicted data. Evaluation indicators were subsequently computed by comparing this matrix with the actual values represented by the ground truth matrix $y_{true} = [y_1, y_2, \ldots, y_n]^T$.

4.4.2. Analysis of Experimental Results

Figure 4 presents a comparative analysis of various algorithms forecasting the demand between the same nodes discussed before, alongside a compilation of the aggregate residuals associated with each method. The comparative experiments computed indicators at both daily and weekly scales. The results are presented in Table 5, with optimal values highlighted in bold.

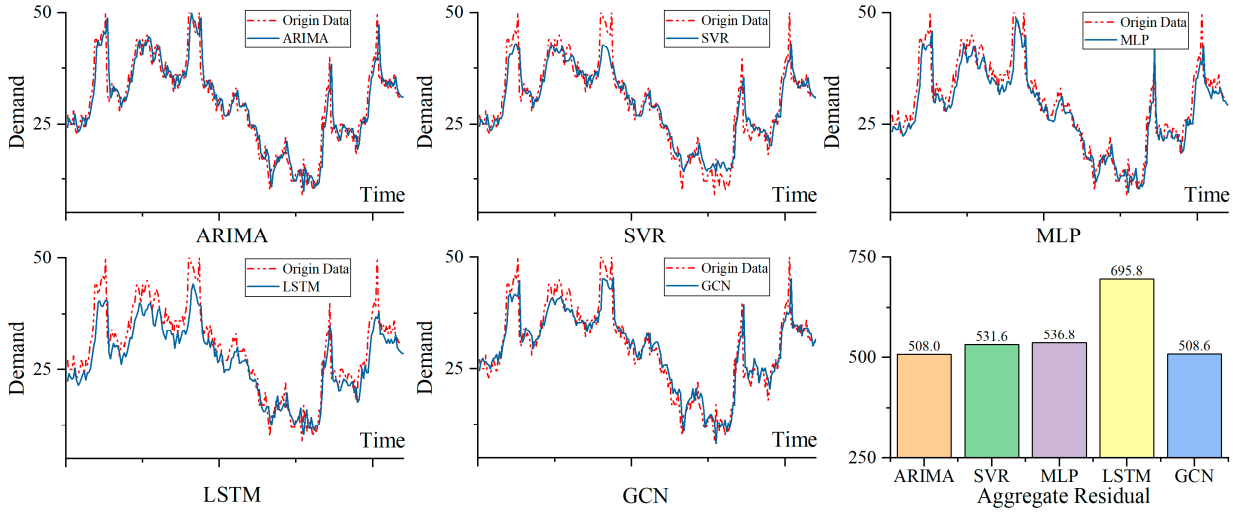


Figure 4. Comparative prediction for the single node and aggregate residuals of various methods.

Table 5. Comparative method performance for the overall demand.

Algorithm	Origin Data		Smoothed Data	
	RMSE	MAPE	RMSE	MAPE
ARIMA	4.172	10.681	1.324	3.430
SVR	3.854	10.719	1.863	5.155
MLP	4.221	10.868	1.880	4.873
LSTM	4.594	12.053	3.003	7.849
Proposed method	3.298	8.562	1.334	3.370

From the prediction results for the demand from a single node, it is shown that ARIMA has the smallest residuals, and its prediction curve is closest to the actual curve. However, further analysis reveals that ARIMA essentially uses the actual values from the previous day as its predictions. While this approach can get better indicators, it holds no practical value in actual supply chain management.

The aggregate residuals of the other three comparative methods are significantly higher, and the prediction curves indicate that the performance of the other three methods is worse than GCN. Among them, LSTM shows the worst predictive performance, primarily due to the relatively weak temporal correlations in the purely historical demand data. In the absence of other temporal features, LSTM struggles to establish effective features to achieve effective prediction.

The experimental findings confirm that the proposed method consistently outperforms other algorithms in forecasting overall supply chain demand, both daily and weekly, as measured by RMSE and MAPE. Notably, the GCN stands out by leveraging historical demand data and incorporating features from various demand links through a topologically enriched adjacency matrix. This ability effectively builds demand relationships among different nodes in the supply chain, leading to improved accuracy in predicting demand fluctuations between these nodes.

Traditional algorithms focus on modeling individual time series, leading to isolated demand features across different supply chain nodes. These methods do not capture the relationships between adjacent nodes, limiting their accuracy in demand predictions. By ignoring the interdependencies among various entities, these algorithms struggle to improve forecasts based solely on individual node data. In comparison, the experiments demonstrate the superior effectiveness of GCN in demand forecasting for intricate supply chains. GCN effectively uses the topological information within the supply chain, allowing them to identify the relationships between interconnected nodes. This capability enhances

the accuracy of overall demand forecasts by incorporating both historical demand data and the relational features of supply chain entities. Consequently, GCN provides more accurate and reliable predictions, thereby improving decision-making in complex supply chain environments.

5. Conclusions

This paper introduces graph theory into supply chain modeling and prediction tasks, proposing a GCN-based method for predicting demand in complex supply chains. By transforming the entities and their relationships within the supply chain into a graph composed of nodes and edges, the method effectively leverages the topological information of the supply chain. The GCN model is then employed to extract and analyze features from the graph data, capturing the relational features among different nodes. Finally, a fully connected layer is used to achieve regression predictions. This approach integrates supply chain topological features with historical demand data, allowing it to enhance prediction accuracy through the relational features among nodes, thereby providing more precise decision support for supply chain management.

In this study, four widely used algorithms—ARIMA, SVR, MLP, and LSTM—were selected for comparative analysis. The performance evaluation of all algorithmic models utilized two distinct indicators, namely, RMSE and MAPE. The experimental outcomes unequivocally demonstrate the superior performance of the proposed algorithm over the four comparative methods. This substantiates the assertion that graph neural networks exhibit greater adaptability in addressing the complexities of supply chain forecasting, surpassing the efficacy of traditional single time series analysis and prediction algorithms. It is worth noting that the data used in this study exhibits some limitations in terms of node features and interactions between nodes. Future research endeavors could potentially enhance the model's predictive capabilities by incorporating more comprehensive features, such as node features like storage, costs, and promotional information, as well as edge features like transportation costs and transportation priority. Node features can be directly included in the feature matrix. However, the proposed GCN cannot handle additional relationship features between nodes beyond demand quantity. Therefore, it is necessary to improve the GCN network to better represent the feature information between edges.

The proposed method enhances demand forecasting in intricate supply chains, enabling optimized inventory management without increasing stock levels. Accurate demand prediction reduces resource waste from overstocking and helps avoid shortages, lowering operational costs. In complex networked supply chains, this improves efficiency across all stages and minimizes unnecessary resource use. Furthermore, it optimizes resource allocation, reducing carbon emissions and energy waste during transportation. Thus, the method not only supports inventory optimization and cost reduction but also contributes to sustainability by minimizing environmental impact.

Author Contributions: All authors contributed to this study conception and design. Material preparation, data collection, and analysis were performed by T.N., X.Y. and H.Z. The first draft of the manuscript was written by T.N. and H.Z. and Q.M. commented on previous versions of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: The research was supported by the National Key R&D Program of China (No. 2021YFB3300800 and No. 2021YFB3300801).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated during and analyzed during the current study are not publicly available due to the sensitivity of business data, but are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Silver, E.A.; Pyke, D.F.; Peterson, R. *Inventory Management and Production Planning and Scheduling*, 1st ed.; Wiley: New York, NY, USA, 1998; Volume 3, p. 30.
2. Donohue, K.L. Efficient supply contracts for fashion goods with forecast updating and two production modes. *Manag. Sci.* **2000**, *46*, 1397–1411. [CrossRef]
3. Netto, C.F.S.; Bahrami, M.; Brei, V.A.; Bozkaya, B.; Balcisoy, S.; Pentland, A.P. Disaggregating sales prediction: A gravitational approach. *Expert Syst. Appl.* **2023**, *217*, 119565. [CrossRef]
4. Tsoumakas, G. A survey of machine learning techniques for food sales prediction. *Artif. Intell. Rev.* **2019**, *52*, 441–447. [CrossRef]
5. Villegas, M.A.; Pedregal, D.J.; Trapero, J.R. A support vector machine for model selection in demand forecasting applications. *Comput. Ind. Eng.* **2018**, *121*, 1–7. [CrossRef]
6. Bankvall, L.; Bygballe, L.E.; Dubois, A.; Jahre, M. Interdependence in supply chains and projects in construction. *Supply Chain Manag. Int. J.* **2010**, *15*, 385–393. [CrossRef]
7. Liu, W.; Zhang, Y.; Wang, J.; He, Y.; Caverlee, J.; Chan, P.P.; Yeung, D.S.; Heng, P.A. Item relationship graph neural networks for e-commerce. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 4785–4799. [CrossRef]
8. Liu, J.; Xia, F.; Feng, X.; Ren, J.; Liu, H. Deep graph learning for anomalous citation detection. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 2543–2557. [CrossRef]
9. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef]
10. Gupta, A.; Matta, P.; Pant, B. Graph neural network: Current state of Art, challenges and applications. *Mater. Today Proc.* **2021**, *46*, 10927–10932. [CrossRef]
11. Teunter, R.H.; Syntetos, A.A.; Zied Babai, M. Intermittent demand: Linking forecasting to inventory obsolescence. *Eur. J. Oper. Res.* **2011**, *214*, 606–615. [CrossRef]
12. Yang, Y.; Ding, C.; Lee, S.; Yu, L.; Ma, F. A modified Teunter-Syntetos-Babai method for intermittent demand forecasting. *J. Manag. Sci. Eng.* **2021**, *6*, 53–63. [CrossRef]
13. Rostami-Tabar, B.; Goltsols, T.E.; Wang, S. Forecasting for lead-time period by temporal aggregation: Whether to combine and how. *Comput. Ind.* **2023**, *145*, 103803. [CrossRef]
14. Van Calster, T.; Baesens, B.; Lemahieu, W. ProfARIMA: A profit-driven order identification algorithm for ARIMA models in sales forecasting. *Appl. Soft Comput.* **2017**, *60*, 775–785. [CrossRef]
15. Sareminia, S.; Amini, F. A reliable and ensemble forecasting model for slow-moving and repairable spare parts: Data mining approach. *Comput. Ind.* **2023**, *145*, 103827. [CrossRef]
16. Garcia-Ascanio, C.; Maté, C. Electric power demand forecasting using interval time series: A comparison between VAR and iMLP. *Energy Policy* **2010**, *38*, 715–725. [CrossRef]
17. Es, H.A. Monthly natural gas demand forecasting by adjusted seasonal grey forecasting model. *Energy Sources Part A Recovery Util. Environ. Eff.* **2021**, *43*, 54–69. [CrossRef]
18. Petropoulos, F.; Kourentzes, N. Forecast combinations for intermittent demand. *J. Oper. Res. Soc.* **2015**, *66*, 914–924. [CrossRef]
19. Guo, F.; Diao, J.; Zhao, Q.; Wang, D.; Sun, Q. A double-level combination approach for demand forecasting of repairable airplane spare parts based on turnover data. *Comput. Ind. Eng.* **2017**, *110*, 92–108. [CrossRef]
20. Hu, Y.C.; Wang, W.B. Nonlinear interval regression analysis with neural networks and grey prediction for energy demand forecasting. *Soft Comput.* **2022**, *26*, 6529–6545. [CrossRef]
21. Li, W.; Wang, X. Analysis and prediction of urban household water demand with uncertain time series. *Soft Comput.* **2024**, *28*, 6199–6206. [CrossRef]
22. Parmezan, A.R.S.; Souza, V.M.; Batista, G.E. Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. *Inf. Sci.* **2019**, *484*, 302–337. [CrossRef]
23. Xu, S.; Chan, H.K.; Zhang, T. Forecasting the demand of the aviation industry using hybrid time series SARIMA-SVR approach. *Transp. Res. Part E Logist. Transp. Rev.* **2019**, *122*, 169–180. [CrossRef]
24. Kumar, A.; Shankar, R.; Aljohani, N.R. A big data driven framework for demand-driven forecasting with effects of marketing-mix variables. *Ind. Mark. Manag.* **2020**, *90*, 493–507. [CrossRef]
25. Lu, C.J. Sales forecasting of computer products based on variable selection scheme and support vector regression. *Neurocomputing* **2014**, *128*, 491–499. [CrossRef]
26. Spiliotis, E.; Makridakis, S.; Semenoglou, A.A.; Assimakopoulos, V. Comparison of statistical and machine learning methods for daily SKU demand forecasting. *Oper. Res.* **2022**, *22*, 3037–3061. [CrossRef]
27. Pan, H.; Zhou, H. Study on convolutional neural network and its application in data mining and sales forecasting for E-commerce. *Electron. Commer. Res.* **2020**, *20*, 297–320. [CrossRef]
28. Ma, S.; Fildes, R. Retail sales forecasting with meta-learning. *Eur. J. Oper. Res.* **2021**, *288*, 111–128. [CrossRef]
29. Kantasa-Ard, A.; Nouiri, M.; Bekrar, A.; Ait el Cadi, A.; Sallez, Y. Machine learning for demand forecasting in the physical internet: A case study of agricultural products in Thailand. *Int. J. Prod. Res.* **2021**, *59*, 7491–7515. [CrossRef]
30. Haque, M.S. Retail Demand Forecasting Using Neural Networks and Macroeconomic Variables. *J. Math. Stat. Stud.* **2023**, *4*, 1–6. [CrossRef]

31. Abbasimehr, H.; Shabani, M.; Yousefi, M. An optimized model using LSTM network for demand forecasting. *Comput. Ind. Eng.* **2020**, *143*, 106435. [CrossRef]
32. Bi, J.W.; Li, H.; Fan, Z.P. Tourism demand forecasting with time series imaging: A deep learning model. *Ann. Tour. Res.* **2021**, *90*, 103255. [CrossRef]
33. Bandara, K.; Shi, P.; Bergmeir, C.; Hewamalage, H.; Tran, Q.; Seaman, B. Sales demand forecast in e-commerce using a long short-term memory neural network methodology. In Proceedings of the Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, 12–15 December 2019; Part III 26. pp. 462–474.
34. Rodrigues, F.; Markou, I.; Pereira, F.C. Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach. *Inf. Fusion* **2019**, *49*, 120–129. [CrossRef]
35. Kosasih, E.E.; Brintrup, A. A machine learning approach for predicting hidden links in supply chain with graph neural networks. *Int. J. Prod. Res.* **2022**, *60*, 5380–5393. [CrossRef]
36. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.