Production and Operations Management

Let Clickstream Talk: A Graph Neural Network Approach to Sales Forecasting

Rong Liu
Department of Information Systems and Business Analytics, College of Business, Florida International University, 11200 S.W. 8th Street, RB 206A, Miami, FL 33199, USA. roliu@fiu.edu. Phone: +1 (305) 348-4356.

Zihan Chen
School of Business, Stevens Institute of Technology, 1 Castle Point Terrace, Hoboken, NJ 07030, USA. zchen61@stevens.edu. Phone: +1 (201) 680-1738.

Denghui Zhang
School of Business, Stevens Institute of Technology, 1 Castle Point Terrace, Hoboken, NJ 07030, USA. dzhang42@stevens.edu. Phone: +1 (201) 216-3499.

Feng Mai
Department of Business Analytics, Tippie College of Business, University of Iowa, 21 East Market Street, Iowa City, IA 52242, USA. feng-mai@uiowa.edu. Phone: +1 (319) 335-0858.

Xuying Zhao
Department of Information and Operations Management, Mays Business School, Texas A&M University, Wehner Building, 210 Olsen Boulevard, 4113 TAMU, College Station, TX 77843, USA. xzhao@mays.tamu.edu. Phone: +1 (979) 845-4248. Corresponding author.

Abstract: This paper proposes ForecastClickGraph, a deep learning framework that extracts cross-product relationships and demand information from clickstream data for probabilistic sales forecasting. ForecastClickGraph models stages in consumer shopping journeys as a dynamic graph, where nodes represent individual product-stage combinations (such as product A viewed or added to cart), and directed edges capture the transitions between stages. A customized graph neural network learns product representations that incorporate both product-level temporal patterns and cross-product associations. Another graph-based module further detects demand spikes using early signals from related products. ForecastClickGraph cross-learns demand patterns across large-scale product catalogs and estimates demand distributions through quantile regression. Extensive experiments on a real-world dataset show that ForecastClickGraph outperforms state-of-the-art benchmark models by 10-28% in forecast accuracy, with particularly strong performance in predicting promotional sales bursts. It also yields superior probabilistic forecasts with substantially lower quantile loss and improved calibration relative to benchmarks.

Key words: sales forecasting, graph neural network, clickstream, product relationship, sales burst
History: Received: November 2022; accepted: August 2026 by Subodha Kumar after three revisions.

DOI: 10.1177/10591478261490953

Let Clickstream Talk: A Graph Neural Network Approach to Sales Forecasting

Abstract

This paper proposes ForecastClickGraph, a deep learning framework that extracts cross-product relationships and demand information from clickstream data for probabilistic sales forecasting. ForecastClickGraph models stages in consumer shopping journeys as a dynamic graph, where nodes represent individual product-stage combinations, such as product A viewed or added to cart, and directed edges capture the transitions between stages. A customized graph neural network learns product representations that incorporate both product-level temporal patterns and cross-product associations. Another graph-based module further detects demand spikes using early signals from related products. ForecastClickGraph cross-learns demand patterns across large-scale products and estimates demand distributions through quantile regression. Extensive experiments on a real-world dataset show that ForecastClickGraph outperforms state-of-the-art benchmark models by 10-28% in forecast accuracy, with particularly strong performance in predicting promotional sales bursts. It also yields superior probabilistic forecasts with substantially lower quantile loss and improved calibration relative to benchmarks.

Keywords: sales forecasting, graph neural network, clickstream, product relationship, sales burst

1. Introduction

The rapid growth of e-commerce has made accurate sales forecasting increasingly critical for effective inventory management and order fulfillment. According to data from the Federal Reserve Bank of St. Louis, e-commerce sales in the United States grew from $80 billion in 2015 to $289 billion in 2024, while e-commerce retail sales of total sales rose from 4.2% in 2010 to 15.9% in 2024 (FRED 2024). The COVID-19 pandemic has further accelerated the shift to online shopping, prompting many brick-and-mortar retailers to expand their online channels (Torry 2020). A key advantage of online platforms is the rich clickstream data they generate, providing granular information on consumer browsing behavior prior to purchase. The literature has increasingly recognized the operational value of clickstream data (Huang and Van Mieghem 2013, 2014; Martínez-de-Albéniz et al. 2020). Haruvy et al. (2014) further show that clickstream data contains extensive information about the search process and the consideration set of a consumer, which can improve the prediction of online consumer behavior.

However, existing clickstream-based forecasting methods primarily focus on individual product-level patterns (Huang and Van Mieghem 2013, 2014; Martínez-de-Albéniz et al. 2020), overlooking the rich cross-product relationships that shape consumer purchase decisions. During online shopping, consumers often browse multiple products, compare substitutes, and seek complementary items before

making a purchase (Shocker et al. 2004). The order and duration of product views, therefore, provide valuable signals about consumer preferences. For example, frequent co-views may indicate substitutability, while frequent views of one product after purchasing another may suggest complementarity. Leveraging such latent product associations can improve forecast accuracy and efficiency, particularly for retailers with large and diverse product catalogs.

Capturing cross-product relationships is especially important during promotional periods, such as Black Friday/Cyber Monday and other holidays, and during significant external events that alter consumer purchasing patterns. Catalysts such as price discounts, product recommendations, or shifting consumer behaviors can drive sudden spikes in demand and induce cross-product spillover effects not present in historical data. For example, a steep discount on gaming consoles may stimulate demand for associated games and accessories. These dynamics are difficult to forecast because promotion events are sparse and irregular, and they produce sales bursts that deviate significantly from historical norms. This challenge is particularly consequential because such intensive periods often account for a disproportionate share of annual sales and profits for online retailers (CNBC 2023). Failure to anticipate and incorporate these dynamics can lead to substantial forecast errors and sub-optimal inventory decisions.

To address these challenges, we propose ForecastClickGraph, a novel graph-construction approach that uses clickstream data to generate more accurate, granular demand forecasts. The core innovation of our approach is modeling aggregated consumer shopping journeys as a dynamic graph. Each node represents a product at a specific shopping stage, such as viewed or purchased, with time-series attributes capturing both quantity and burstiness patterns. Directed edges capture the transitions between product-stage nodes, and edge weights represent transition frequencies that reveal nuanced relationships beyond simple co-occurrence. Customized graph neural networks (GNNs) learn product representations that incorporate both product-level temporal patterns and cross-product associations. To scale to large catalogs, ForecastClickGraph tailors graph attention to product-stage transitions leading to purchase and incorporates a novel burst-prediction module that detects sudden demand spikes using early signals from related products. Importantly, this graph structure evolves daily to reflect real-time changes in consumer behavior, particularly during promotions.

ForecastClickGraph offers distinct advantages over traditional time series and machine learning models in the operations management (OM) literature. While many models forecast using only individual product features (Bastani et al. 2022), ForecastClickGraph learns from interconnected products and uncovers implicit relationships that influence demand. It also scales better than multivariate time series models, which often struggle with the high dimensionality and granularity of clickstream data (X. Zhang and Zhao 2010). By allowing each product to attend only to relevant neighbors and encoding their

information into low-dimensional feature vectors, ForecastClickGraph remains effective and tractable for large product catalogs with extensive cross-sectional and temporal data.

Moreover, ForecastClickGraph addresses several key limitations of existing machine learning methods in sales forecasting. To tackle the issue of large-scale product catalogs and insufficient sales history, ForecastClickGraph enriches cross-learning (Salinas et al. 2020) with advance demand signals and burstiness features learned across multiple products. Coupled with quantile regression, it provides probabilistic forecasts that capture demand uncertainty – a crucial feature for inventory management. It also explicitly models the nonlinear, multi-stage nature of online shopping journeys through dynamic intra- and inter-product transitions. Finally, its burst prediction module anticipates demand spikes during promotional events by aggregating early signals from related products, addressing a major shortcoming of traditional methods that struggle with high dimensionality of promotion-related variables.

Using real-world data from a major e-commerce platform, we demonstrate ForecastClickGraph outperforms a comprehensive set of state-of-the-art time series forecasting benchmarks, including DeepAR (Salinas et al. 2020), Informer (Zhou et al. 2021), NHITS (Challu et al. 2023), and dynamic graph neural network models like T-GCN (Zhao et al. 2020). ForecastClickGraph improves MSE by 10-28%, with especially strong gains during promotional periods. Ablation studies confirm the importance of the graph modeling and burst prediction components in driving ForecastClickGraph's superior performance. Furthermore, an evaluation of the probabilistic forecasts shows superior calibration, which provides a more reliable foundation for inventory decisions.

In summary, this paper makes the following contributions. First, we propose ForecastClickGraph, a novel graph-construction approach for cross-learning demand forecasts from clickstream data. The main contributions are a product-stage graph and a burst-prediction module, which together model relevant product associations and adapt to changing demand patterns. Second, our work enriches the literature on clickstream analysis by introducing novel uses of clickstream: capturing dynamic product relationships and detecting early signals for sales bursts. While our approach is inspired by established techniques such as GNN, it is specifically tailored to the challenges of probabilistic sales forecasting in dynamic e-commerce environments (Abbasi et al. 2024). Third, through comprehensive benchmarking on a large-scale e-commerce dataset, we demonstrate ForecastClickGraph's superior performance, particularly in capturing bursty dynamics during promotions. Finally, our framework is particularly suited for marketplace and intermediary platforms, where access to sellers' proprietary pricing and promotion information is limited. By leveraging real-time behavioral signals and observed sales dynamics from clickstream data, which implicitly reflect heterogeneous and often unobservable seller actions, our approach provides a robust and practical solution for demand forecasting under information constraints.

2. Background and Literature Review

Our study draws upon several domains in the literature, including sales forecasting on e-commerce platforms, the analysis of clickstream data and online shopping journeys, promotional effects, and advanced forecasting models. In this section, we introduce basic concepts in each domain and survey related work in the literature.

2.1. Sales Forecasting for E-commerce Platforms and Cross Learning

E-commerce platforms can be categorized into three business models: *reseller platforms*, *intermediary marketplaces*, and *hybrid platforms* (Tian et al. 2018). Reseller platforms, such as Bestbuy.com and Macys.com, purchase products from suppliers and resell them to consumers. Intermediary marketplaces (or simply *marketplaces*), like Tmall and eBay, allow suppliers to sell their products directly to consumers. Hybrid platforms, such as Amazon.com, combine both models. Resellers typically have full control over sales decisions and can aggregate sales data across products for forecasting. In contrast, on marketplaces and hybrid platforms, these decisions are managed by individual suppliers. Strategic information, such as pricing policies and promotion schedules, is often considered proprietary and not shared across suppliers, limiting collective forecasting capabilities. Our study specifically addresses sales forecasting under such information access constraints.

Sales forecasting in OM generally relies on two types of information: (1) historical sales data often represented as time series, and (2) contextual information relevant to consumers' purchase behaviors (Abolghasemi, Hurley, et al. 2020). Previous studies have examined a variety of contextual information, including product complementarity and substitution relationships (Z.-Y. Chen et al. 2023), clickstream data (Huang and Van Mieghem 2014), marketing campaigns (Fildes et al. 2022), intra- or inter-category sales promotions (Abolghasemi, Hurley, et al. 2020), product reviews (G. Chen et al. 2023), online search traffic of competitive product pre-release (Schaer et al. 2022), and even weather (Steinker et al. 2017). While these contextual factors have been studied in isolation, accurate sales forecasting often requires integrating multiple factors and understanding their joint effects (e.g., the spillover effects of promotions on multiple related products) (Qi et al. 2019).

In practice, individual sales forecasting models, such as regression and Prophet (Taylor and Letham 2018), are often created for each stock-keeping unit (SKU) using its own sales history and features. Such individual models can perform well when products have long, regular, and stable sales histories. Their accuracy often deteriorates, however, for newly launched products, seasonal products with short lifecycles (Cohen et al. 2022), or products affected by promotions which create non-stationary and volatile sales patterns (Abolghasemi, Hurley, et al. 2020; Fildes et al. 2022). In addition, training separate models for large numbers of SKUs is also computationally costly (Spiliotis et al. 2022).

Cross-learning addresses these limitations by pooling sales data across products to train a single forecast model for multiple SKUs. With enhanced data availability, cross-learning models can capture common patterns across time series, adapt to structural changes (e.g., due to promotions), and thereby improve forecasting accuracy (Fildes et al. 2022). For example, Salinas et al. (2020) demonstrated the effectiveness of cross-learning across thousands of SKU time series.

In practice, cross-learning models often estimate *probabilistic demand distributions* in two approaches. Unlike a point forecast, which offers a single demand expectation, probabilistic forecasts capture a range of possible demands and help decision makers evaluate downstream risks in replenishment and inventory management (Petropoulos et al. 2022). The first approach assumes a parametric probability distribution for forecasting and directly estimates the distribution parameters (e.g., mean and variance). For instance, DeepAR (Salinas et al. 2020) uses a Gaussian distribution for real-valued data and negative-binomial for count data. However, a fixed distributional assumption may be restrictive across thousands of products with heterogeneous demand patterns. Quantile regression offers a flexible alternative by directly forecasting selected demand quantiles without imposing a parametric distribution (Koenker and Bassett Jr 1978). Studies have shown that integrating quantile regression with deep learning is particularly effective for demand forecasting (Petropoulos et al. 2022; Wen et al. 2017).

For operational planning, it is also desirable to generate *multi-step forecasts* over a future horizon H based on sales history up to time T. Such forecasts can help estimate demand fluctuations across an entire promotion period. The literature distinguishes between recursive and direct forecasting strategies. Recursive approaches generate one forecast at a time and feed each prediction back into the model, which can accumulate errors over the forecast horizon (Petropoulos et al. 2022). A direct strategy can mitigate error propagation by producing multi-step forecasts simultaneously (Petropoulos et al. 2022).

Overall, the key challenge in cross-learning is to extract rich features that capture diverse patterns in massive product sales time series (Semenoglou et al. 2021). Meanwhile, these features should remain sensitive to shifts in individual product demand over a multi-step forecasting horizon.

2.2. Clickstream Data and Online Shopping Journeys

Clickstream data not only provides advance demand information (Huang and Van Mieghem 2013) but also allows firms to learn about consumer behaviors and preferences (Ding et al. 2015; Ferreira et al. 2022; Martínez-de-Albéniz et al. 2020). An online shopping journey typically involves three stages: *viewing* products,[1] creating shopping carts (referred to as *cart* for simplicity), and making *purchases*

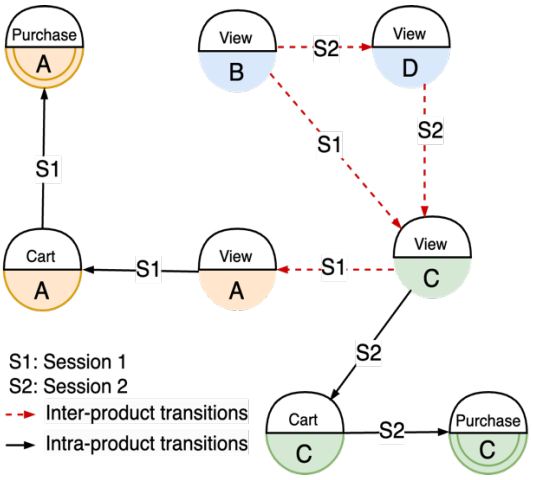
[1] View events can be triggered when a product enters a user's viewpoint during scrolling or when the user interacts with a product, such as by clicking its link. Thus, multiple view events may occur within a short time window when several products appear on the same page. View events may also co-occur with other event types; for example, a one-click purchase may generate a view event immediately followed by a purchase event.

(Sahni et al. 2019). However, the journey is not necessarily linear. Customers may move back and forth across stages, or terminate their sessions at any point.

Figure 1(a) shows two sessions from a user's online shopping journey. In Session S1, the customer viewed Apple-branded smartphones and headphones but purchased a Samsung smartphone, while remaining interested in headphones as a possible accessory. In Session S2, the customer browsed JBL headphones but purchased Apple headphones. These stage transitions reveal hidden preferences and product relationships. Figure 1(b) represents these two sessions as a clickstream graph, where each *SKU-stage is shown as a node and each transition as an edge*. The graph captures both intra-product transitions, such as view-to-cart-to-purchase for the same SKU, and inter-product transitions surrounding SKU C, which may indicate substitute or complementary products. Thus, product relationships naturally emerge from clickstream graphs.

Time	Event	Product	Category	Brand
User Session S1				
		……		
11/14/19 16:39:26	view	B	smartphone	Apple
11/14/19 16:39:53	view	C	headphone	Apple
11/14/19 16:40:09	view	C	headphone	Apple
11/14/19 16:40:21	view	A	smartphone	Samsung
11/14/19 16:40:33	cart	A	smartphone	Samsung
11/14/19 16:40:51	purchase	A	smartphone	Samsung
User Session S2				
		……		
11/16/19 14:50:19	view	B	smartphone	Apple
11/16/19 14:52:22	view	D	headphone	JBL
11/16/19 14:53:02	view	C	headphone	Apple
11/16/19 14:53:04	cart	C	headphone	Apple
11/16/19 14:53:40	purchase	C	headphone	Apple

(a) User Session Excerpts



(b) Clickstream Graph

Figure 1: Examples of Online Shopping Journeys and Clickstream Graphs

A closely related stream is recommender systems, which also leverage clickstream data but pursue a different objective: recommending or ranking products to influence consumer choices and improve engagement or conversion. In OM, recommender systems are studied both as demand-shaping technologies (e.g., Adamopoulos and Todri 2025; Wan et al. 2025) and as decision or optimization problems (e.g., Demirezen and Kumar 2016; Ding et al. 2015; Ferreira et al. 2022). Recommender systems typically predict an *individual* user's next click or purchase, while our focus is predicting *aggregate* SKU-level demand distributions over a future horizon for operational planning. Appendix 1 provides a structured survey of clickstream graph models (Section A1-1) and a detailed comparison between recommendation and our forecasting task (Section A1-2).

Prior OM research has emphasized the value of clickstream data for demand modeling. As Figure 1(a) suggests, the time lag between a consumer's first product view and eventual purchase may span several days, giving retailers advance signals of potential demand shifts. Such information is obtained "without asking" customers (Huang and Van Mieghem 2014). Martínez-de-Albéniz et al. (2020) develop a hierarchical model of sequential shopping decisions using clickstream data and show its effectiveness in forecasting volatile demand during flash sales. Huang and Van Mieghem (2013, 2014) demonstrate that clickstream data improves forecast accuracy and reduces inventory holding and backorder costs by about 5%; they further show that clickstream data from non-transactional websites helps predict the propensity, amount, and timing of offline purchase orders.

To summarize, recommendation systems focus on individual consumer behaviors within a user session, while sales forecasting targets aggregate demand across consumers over a multi-step forecast horizon that may span multiple user sessions. Existing OM studies on clickstream-based forecasting primarily emphasize intra-product transitions while overlooking inter-product transitions. By contrast, our study highlights inter-product transitions, which provide valuable demand signals during promotions, as promotional activities often generate substantial spillover effects across related products.

2.3. Holiday Seasons, Promotions and Bursting Sales

The holiday season features numerous promotional events initiated either jointly by platforms or independently by each party (Cheng et al. 2023). Platform-led promotions, such as Black Friday, are typically organized by platforms, with sellers deciding whether to participate. In contrast, sellers on marketplaces or hybrid platforms can independently launch promotions within a brand or category (e.g., buy one get one free) or across categories (e.g., clearance sale) (Cheng et al. 2023). These seller-led strategies are often proprietary and may not be disclosed to platforms or other sellers in advance (Hagiu and Wright 2015).

Promotions can generate complex spillover effects across products. Promoting one product may increase demand for its complements while negatively affecting its substitutes due to brand switching (Fildes et al. 2022). These effects are particularly prominent on e-commerce platforms where users can easily compare alternatives (Z.-Y. Chen et al. 2023). Promotion events may also extend to sellers that do not offer promotions themselves (Hagiu and Wright 2015; Tian et al. 2018; D. J. Zhang et al. 2020). For example, Zhang et al. (2020) find that shopping cart promotions increase the proportion of products added to carts, but reduce the prices customers subsequently pay, regardless of whether the products are promoted. Moreover, spillover effects may also be shaped by factors beyond promotions, including intrinsic product characteristics, brand identity, perceived quality, price elasticity, and consumption context (Voleti et al. 2015). As a result, identifying which products are affected by spillover effects is particularly challenging on large platforms with millions of products and rapid turnover (Tian et al. 2018).

Promotions and other exogenous shocks, such as viral product fads, can dramatically change demand for both target and related products, posing substantial forecast and operational challenges. First, promotions often cause sudden sales spikes, whose timing and scale are difficult to predict. A grand promotion can increase sales up to ten times (Chi et al. 2024), placing pressure on warehousing and distribution systems. This issue is particularly acute in marketplaces or hybrid platform settings, where sellers independently schedule promotions (Cheng et al. 2023), making it nearly impossible for other sellers to anticipate the timing and impact of potential spillover effects. Second, promotion strategies are often sparse, irregular, and non-repetitive, making it difficult to learn underlying demand patterns (Q. Feng and Shanthikumar 2018), particularly for products with limited sales history, such as new products (Baardman et al. 2018) or seasonal items (Wolters and Huchzermeier 2021). Third, consumer stockpiling during promotions can reduce post-promotion demand (Hewage et al. 2022). Finally, promotions may trigger "herd behavior" where consumers are influenced by high purchase volume rather than discounts themselves (Chi et al. 2024). Together, these effects can substantially reduce forecast accuracy.

Existing forecast methods attempt to incorporate promotion-related variables, such as promotion schedules and discount schemes, for target and related products (Abolghasemi, Hurley, et al. 2020; Huang et al. 2014; Ma et al. 2016). However, such variables can be difficult to obtain, because seller-led promotions are often proprietary (Hagiu and Wright 2015). Even when available, modeling cross-product promotional effects substantially increases dimensionality and raises the risk of overfitting. Prior studies address this issue with continuous shrinkage methods to identify influential predictors (Huang et al. 2014; Ma et al. 2016). In practice, however, many retailers rely on simple statistical models supplemented by human judgmental adjustments because of model complexity and data collection cost (Fildes et al. 2019). These adjustments are often subject to human biases (Fildes et al. 2022). Moreover, most methods overlook post-promotional demand effects (Hewage et al. 2022).

In summary, existing promotion studies suffer from difficulties in identifying product relationships and spillover effects at scale, often rely on proprietary promotion information, and are susceptible to bias from manual interventions. These challenges necessitate a dynamic, automated approach capable of capturing the effects of promotions on related products and inferring the timing of sales bursts from real-time clickstream data.

2.4. Forecast Models and Deep Learning

Traditional demand forecasting methods range from univariate time series analysis, such as ARIMA and Prophet (Taylor and Letham 2018), to multivariate time series regression like vector autoregression (Abolghasemi, Beh, et al. 2020; Spiliotis et al. 2022; Wolters and Huchzermeier 2021). Researchers have found that simple time series techniques generally can perform well during periods

without promotions, while advanced models are needed to incorporate promotional information for better accuracy during promotions (Fildes et al. 2022). Next, we review relevant forecast models.

Traditional machine learning and deep learning. Machine learning methods, including Support Vector Regression (SVR) and multilayer perceptron networks (MLP), have been used for forecasting with historical sales and contextual factors as inputs (Fildes et al. 2022; Spiliotis et al. 2022). These methods can capture heterogeneous nonlinear relationships among features. Stacked MLPs further extract complex time series characteristics. For instance, NHITS (Challu et al. 2023) uses a residual architecture, where each stack learns a signal (e.g., trend) and passes unexplained residuals to the next stack. However, these methods typically treat each product time series as an observation and each time point as an independent feature, limiting their ability to capture deep temporal patterns or cross-product relationships.

Sequence learning. Sequence learning models capture temporal dependencies using memory or attention mechanisms. Through encoder-decoder architectures, they transform historical time series into multi-step sales forecasts. Recurrent neural networks (RNNs) maintain hidden states that are updated throughout the sequence. For instance, DeepAR employs an auto-regressive RNN with lagged target values and covariates as inputs (Salinas et al. 2020). Transformer models (Devlin et al. 2019) and variants like Informer (Zhou et al. 2021) instead use attention to selectively focus on relevant parts of the sequence. These methods capture temporal patterns but do not directly model cross-product relationships.

Graph neural networks. Graph neural networks (GNNs), which model relational dependencies through graph structures, have been widely adopted to capture shopping journeys, product relationships, and user-product interactions (Wang et al. 2021). For example, GNNs have been used to infer product relationships from clickstream data (Hao et al. 2020; Yan et al. 2022; Yilmaz and ÖĞüdücü 2021). These models treat products as nodes and specific product interactions, such as co-views and co-purchases, as edges. Each node selectively aggregates neighbors' information through attention or MLPs to generate product-relation embeddings. A survey of GNNs for clickstream can be found in Appendix 1. However, existing studies focus on a narrow set of product relationships, such as co-view and co-purchase, while more nuanced relationships, e.g., purchase-after-view (e.g., one-click buy; Unal and Park 2023) and purchase-after-purchase, remain unexplored. These relationships are often learned in offline settings and may fail to capture real-time product dynamics driven by ongoing promotions.

Dynamic graph neural networks. *Dynamic* GNNs, which combine sequence learning and GNNs, are a natural choice for sales forecasting, because they can capture both temporal dynamics and inter-product relationships. A variety of models have been proposed with different integration structures. For example, T-GCN (Zhao et al. 2020) applies graph convolution to extract neighborhood features at each snapshot and feeds them into an RNN for sequence learning. GConvGRU and GConvLSTM (Seo et al. 2018) instead use RNN hidden states as node features for graph convolution. GC-LSTM (J. Chen et al.

2022) adopts a similar structure, while also utilizing the adjacency matrix in hidden state updates. Dynamic GNNs, particularly, a subclass named spatio-temporal graph models, have been widely applied in traffic forecasting and human motion modeling, where graph structures, such as road networks, are predefined and relatively stable (Z. Feng et al. 2025). In contrast, product graphs on e-commerce platforms are highly dynamic due to frequent product entry and exit as well as nuanced interactions like co-cart or sequential promotions (e.g., buy-A-then-B). Moreover, most dynamic GNNs often assume stable temporal patterns and lack mechanisms to infer abrupt demand changes when promotion signals are unknown or unavailable. Appendix 2 provides further discussion on dynamic GNNs.

To summarize, our literature review identifies four key challenges or design requirements (R1-R4) in sales forecasting for e-commerce platforms, as shown in Table 1. (R1) Because platforms contain massive products with high demand variability, probabilistic cross-learning models are preferred but require rich features to capture diverse demand patterns. (R2) Existing methods do not effectively model intra-/inter product transitions for aggregate sales forecasting. (R3) Forecasting models need to discover dynamic, nuanced product relationships and assess promotional spillover effects at scale. (R4) Since promotions may not be disclosed in advance, forecast models need to detect burstiness signals and infer the timing and magnitude of sales bursts in real time without prior knowledge of promotions.

Table 1: Summary of Design Requirements, Related Work, and Design Components

A. Design Requirements	**B. Existing Methods and Limitations**	**C. Design Components**
(R1) Large scale of products, limited sales history, and high demand variability	- Individual SKU-level forecast models are computationally expensive and inaccurate [1] - Cross learning pools thousands time series across products [2]; required to capture diverse patterns	Enriching cross learning with expressive features learned from related products to capture advance demand and burstiness features
(R2) Nonlinear online shopping journeys with dynamic intra-/inter-product transitions	- Recommendation systems focus on individual consumer behaviors within a user session [3], while sales forecasting targets aggregated demand across consumers and sessions - Aggregated clicks used as covariates without fully considering intra-/inter-product transitions [4, 5]	Unique graph construction that explicitly models shopping journey stages and intra-/inter- stage transitions; with aggregated time series data embedded at each node to capture temporal dynamics
(R3) Complex product relationships and promotional effects on related products	- Product relationships are manually captured [6]; difficult to scale on e-commerce platforms - A narrow set of inter-product relationships (e.g., co-view, co-purchase) are extracted from clickstream and represented as static graphs [7,8]	Automatically capturing a broad range of nuanced product relationships (e.g., co-cart, one click purchases) from intra-/inter-product transitions; learning advance demand from related products
(R4) Changed sales patterns and dramatic sales bursts due to promotions	- Assuming promotions are disclosed in advance, modeling them as features but suffering from high dimensionality [9,10,11] - Human adjustment of promotional effects but suffering from bias [6, 12]	Using a GNN to aggregate burstiness signals from related products and inferring the timing and scale of sales bursts in real time, without requiring prior knowledge of promotions
[1] Spiliotis et al. 2022; [2] Semenoglou et al. 2021; [3] Wu et al. 2023; [4] Huang and Van Mieghem 2013; [5] Huang and Van Mieghem 2014; [6] Fildes et al. 2022; [7] Hao et al. 2020; [8] Yilmaz and ÖĞüdücü 2021; [9] Abolghasemi, Hurley, et al. 2020; [10] Huang et al. 2014; [11] Ma et al. 2016; [12] Fildes et al. 2019		

3. Design Requirements and Forecasting Architecture

We propose a customized deep learning model, *ForecastClickGraph*, as shown in Figure 2(A), to respond to requirements R1-R4 summarized in Column C of Table 1. To address R1, ForecastClickGraph combines cross-learning with quantile regression to generate multi-step probabilistic forecasts for each SKU. The final forecasts aggregate three effects learned jointly. First, similar to existing methods (Salinas et al. 2020), an LSTM layer extracts time-dependent features from each SKU's historical sales to produce a base forecast. Second, to address R2 and R3, a GNN layer captures dynamic product interactions from aggregated customer shopping journeys. The model encodes neighbors leading to a product's purchase as features for sales forecasting. Because purchase nodes may be highly correlated with their neighbors, such as view nodes of the same SKU, we introduce a customized gate to regulate neighbor effects and avoid inflating sales forecasts. Lastly, to address R4, a dedicated burst prediction component detects advance demand signals from neighbors and adjusts forecasts automatically. Next, we formally define the forecasting problem, introduce the notation, and describe each component in detail.

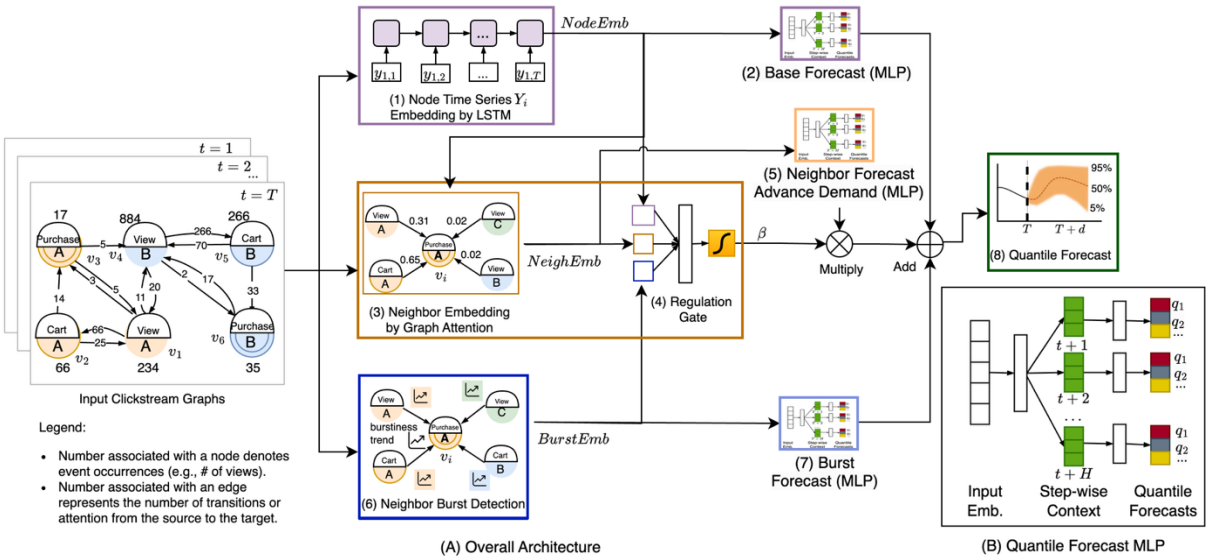


Figure 2: ForecastClickGraph: A Cross-learning Framework for Multi-step Probabilistic Forecasts

3.1. Problem Definition and Clickstream Data

Since we forecast daily SKU sales, we aggregate user sessions by SKU-stage and day and then model the aggregated data as a graph, as illustrated in Figure 3(A). *Each SKU-stage is treated as an event and represented as a node, whose attribute records the number of event occurrences*. In graph g_1, SKU A has been viewed (node v_1) 146 times, added into carts (v_2) 40 times, and purchased (v_3) 9 times. A *transition between two SKU-stages is modeled as an edge, whose weight denotes the number of such transitions*. For instance, SKU A's 146 views lead to 40 intra-product transitions to "SKU A Cart" and 30 inter-product transitions to "SKU B View," which indicate some users subsequently viewed SKU B.

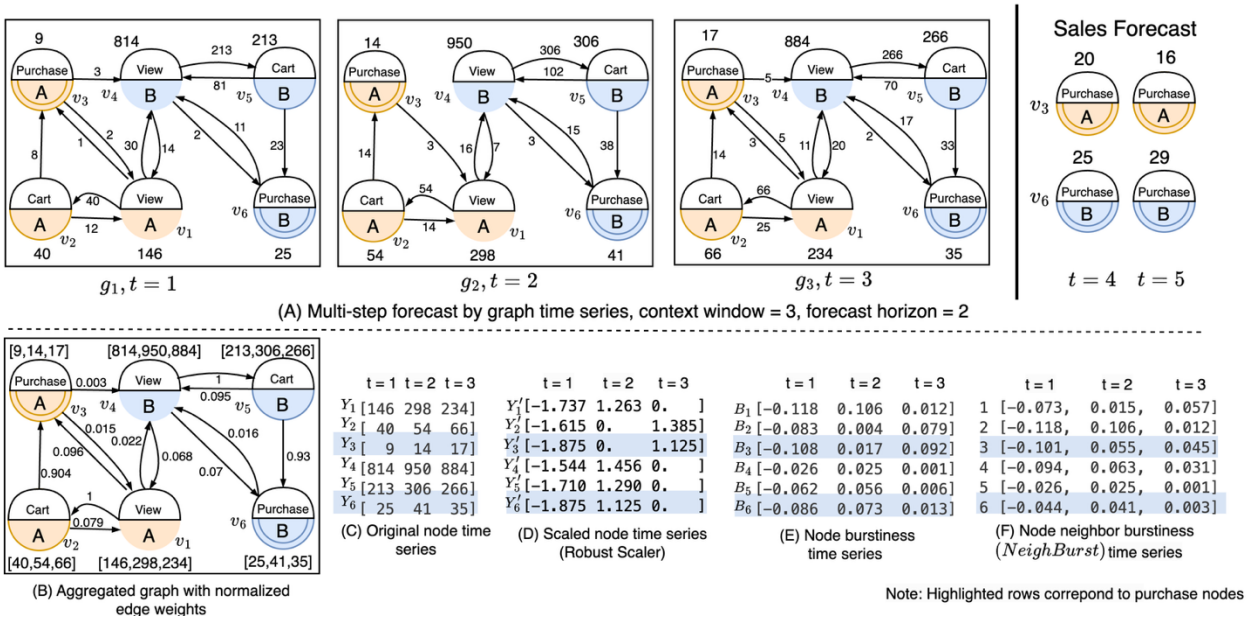


Figure 3: An Illustrative ForecastClickGraph Example

Our model takes a time series of graphs as input. Appendix 3 lists the notation. Formally, let $\mathbb{V} = (v_1, v_2, \dots, v_M)$ denote the set of all nodes. Over a context window T, the clickstream data is represented as a sequence of timestamped graphs, $\mathbb{G} = (G_1, G_2, \dots, G_T)$, with graph $G_t = (V_t, E_t)$ at time t. The node set is $V_t = \{(v_i, y_{i,t})\}_{i=1}^{M}$, where $y_{i,t}$ denotes occurrences of node v_i. The edge set is $E_t = \{(v_i, v_j, w_{i,j,t}) \mid w_{i,j,t} > 0\}_{i,j=1}^{M}$, where $w_{i,j,t}$ indicates the weight of the edge from v_i to v_j. Each node v_i has a *node time series* $Y_i = \{(y_{i,t})\}_{t=1}^{T}$, and $W_{i,j} = \{(w_{i,j,t})\}_{t=1}^{T}$ represents the *weight time series* of the edge from v_i to v_j. Figure 3(C) lists all node time series. For instance, SKU A purchase (node v_3) has a time series $Y_3 = (9,14,17)$.

Because node time series differ in scale, we apply the Robust Median Scaler (Rousseeuw and Croux 1993) to normalize them. This scaler normalizes $Y = \{y_1, y_2, \dots, y_n\}$ into $Y' = \{z_1, z_2, \dots, z_n\}$, where $z_i = \frac{y_i - Median(Y)}{MAD(Y)}$, $MAD(Y) = \frac{1}{n}\sum_{j=1}^{n} |y_j - Median(Y)|$. For example, for Y_3, with median 14 and MAD 8/3, $Y_3' = (-1.875, 0, 1.125)$. Figure 3(D) shows the scaled time series on a comparable scale across nodes.

To stabilize node transitions, we construct an aggregated graph by averaging edge weights across the context window, shown in Figure 3(B). To account for scale differences, each averaged edge weight is normalized by the average quantity of its target node. We use this graph to identify the upstream and downstream neighbors of each node v_i, denoted as $N_i^{Up} = \{v_j \mid w_{j,i,t} > 0\}$ and $N_i^{Down} = \{v_j \mid w_{i,j,t} > 0\}$, respectively. For instance, in Figure 3(B), for node v_3, $N_3^{Up} = (v_1, v_2)$ and $N_3^{Down} = (v_1, v_4)$.

Finally, we *define the forecasting task as a multi-step quantile forecasting problem* (Wen et al. 2017). Given $\mathbb{G} = (G_1, G_2, \dots, G_T)$ and a node v_i in the graphs, our goal is to provide a probabilistic estimate of v_i's time series in the next H steps (Equation 1), i.e.,

Multi-step probabilistic forecast: $\mathcal{P}(\, y_{i,T+1}, y_{i,T+2,} \cdots, y_{i,T+H} \mid \mathbb{G})$ (1)

Given a set of predefined quantiles Q, (e.g., $10\%, 50\%, 90\%$), for any $q \in Q$ and $1 \le h \le H$, we predict the conditional quantile $y_{i,T+h}^{(q)}$ of the target distribution, i.e.,

q-Quantile forecast: $\mathcal{P}\big(y_{i,T+h} \le y_{i,T+h}^{(q)} \mid \mathbb{G}\big) = q$ (2)

For example, as illustrated in Figure 3(A), the target is to compute the probabilistic forecasts for the purchase of SKU A (v_3) and SKU B (v_6) over the forecast horizon $[4, 5]$. Next, we describe the design components in our ForecastClickGraph framework shown in Figure 2.

3.2. Base Forecast by Recurrent Neural Network

Our model first generates a base forecast from each node time series. When a product has stable sales history and no scheduled promotions in the forecast horizon, historical sales alone can provide sufficient information for accurate forecasting (Fildes et al. 2019). To capture the temporal patterns of sales history, we use an LSTM layer to encode each node time series into a *node embedding*. Given time series input Y_i, the LSTM recursively encodes each element $y_{i,t}$ into a vector representation that incorporates information from $y_{i,t}$'s predecessors. The representation of the final element, therefore, summarizes the entire sequence and can serve as an embedding of Y_i. For simplicity, we treat the LSTM layer as a function that maps a node time series Y_i to its embedding, denoted as $NodeEmb_i$ (Equation 3).

Next, the model produces multi-step forecasts based on $NodeEmb_i$. As discussed in Section 2.1, we adopt a direct strategy, which produces multi-step forecasts simultaneously. Following Wen et al. (2017), we implement this strategy using an MLP network, as shown in Figure 2(B). For each forecast step, $NodeEmb_i$ is first projected into a step-specific context vector through a dense layer with parameter $W_{context}$ and ReLU activation function. This context vector is then passed to another dense layer with parameter W_{output} to output $BaseForecast_{\,i} \in \mathbb{R}^{H \times |Q|}$ for forecast horizon H and quantiles Q (Equation 4).

$$NodeEmb_i = LSTM(Y_i) \quad (3)$$

$BaseForecast_{\,i} = MLP(NodeEmb_i)$, where the function MLP can be decomposed as:

$$Z_{context,i} = ReLU(W_{context}\, NodeEmb_i),$$

$$BaseForecast_{\,i} = W_{output}\, Z_{context,i} \quad (4)$$

3.3. Burst Prediction by Graph Neural Network

While the RNN component captures a product's normal sales dynamics and generates baseline forecasts, promotion-driven deviations are difficult to predict from a product's own sales history alone. Clickstream data provides valuable early signals from related products. As illustrated in Figure 4, view and cart events for two related products begin to rise on November 13, followed by purchase bursts two days later. Such upstream bursts can therefore serve as leading indicators of future purchase bursts. Moreover, signals aggregated from neighbors can be more reliable and informative. A burst in a single time series may reflect noise, whereas simultaneous bursts across several related nodes are more likely to

indicate a platform-driven demand shift. Thus, we employ a graph neural network to aggregate burstiness indicators from neighboring nodes and adjust baseline forecasts during promotion periods.

We first calculate a burstiness score for each interval in a node time series Y_i, forming a burstiness time series (B_i) that captures both the scale and timing of deviations from normal behavior. Following Lappas et al. (2009),[2] burstiness is defined as the deviation of observed behavior from an expected behavior baseline. Given a time series $(x_1, x_2, \ldots, x_T)$, Equation 5 computes burstiness of a segment $[r, l] \subseteq [1, T]$ as $\frac{\sum_{t=r}^{l} X_t}{\sum_{i=1}^{T} X_t} - \frac{l-r+1}{T}$, where $\frac{l-r+1}{T}$ expresses the baseline, while $\frac{\sum_{t=r}^{l} X_t}{\sum_{i=1}^{T} X_t}$ represents the observed fraction. For example, in Figure 3(C), for $Y_3 = (9, 14, 17)$, the burstiness score of each interval is $B_3 = (-0.108, 0.017, 0.092)$. The last two elements receive higher scores because their observed fractions exceed the expected baseline of 1/3. Figure 3(E) shows the burstiness time series of all nodes.

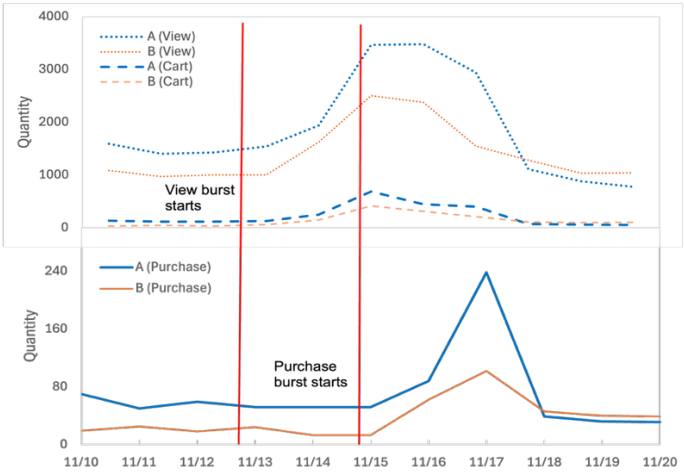


Figure 4: Advance Demand Information from View and Cart Events

Next, for each node i, we average the burstiness time series of its upstream neighbors to construct a neighbor burstiness context ($NeighBurst_i$) (Equation 6). This context captures the timing and magnitude of burstiness in the node's local neighborhood. The intuition is that bursts in upstream nodes (e.g., view and cart) may lead to increased activities in downstream nodes (e.g., purchase). To illustrate, consider Figure 3(B), where the purchase node v_3 has two upstream nodes, v_1 (view) and v_2 (cart). The neighbor burstiness for v_3 is the average of the burstiness time series of these two nodes, i.e., $NeighBurst_3 = \left(\frac{B_1+B_2}{2}\right) = (-0.101, 0.055, 0.045)$. With a burst beginning at $t = 2$, this series suggests that a purchase spike may follow. Figure 3(F) shows the neighbor burstiness time series of all nodes. Note that neighbor burstiness is always computed in the source-to-target direction. It is possible that a node acts as both a source and a target, such as v_3. As a source node, v_3 contributes to the neighbor burstiness of v_1.

[2] Although our method uses burstiness scores as input, it fundamentally differs from Lappas et al. (2009). More detailed comparison can be found in Appendix 4.

Then, $NeighBurst_i$ is transformed through a dense layer to generate a burstiness embedding $BurstEmb_i$ (Equation 7). Using $BurstEmb_i$ as an input, a separate MLP network produces burst quantile forecasts, $BurstForecast_i \in \mathbb{R}^{H\times|Q|}$ for node i (Equation 8), as shown in Figure 2(B). Since the input $NeighBurst_i$ is scale-free, the forecasts represent projected percentage deviations from the current trend. We use a hyperbolic tangent activation (tanh) to constrain the forecasts to the range [-1,1]. The current trend is estimated using exponential moving average (denoted as Emv_i) with a smoothing factor ($\gamma = 0.7$).

Given $Y_i = (y_{i,1}, y_{i,2}, \dots, y_{i,T})$, $B_i = \{b_{i,t}\}_{t=1}^{T}$, where $b_{i,t} = \frac{y_{i,t}}{\sum_{k=1}^{T} y_{i,k}} - \frac{1}{T}$ (5)

$NeighBurst_i = \left\{\frac{1}{|N_i^{UP}|}\sum_{k\in N_i^{UP}} b_{k,t}\right\}_{t=1}^{T}$, where $|\cdot|$ denotes the number of nodes in the set (6)

$\lambda_i = tanh\big(MLP(BurstEmb_i)\big)$, where $BurstEmb_i = ReLU(W_{burst}\ NeighBurst_i)$ (7)

$BurstForecast_i = \lambda_i * Emv_i$ (8)

3.4. Shopping Journey Modeling via Attention-based Neighbor Embeddings

We apply Graph Transformer (Shi et al. 2021), an attention-based GNN to learn the advance demand signals from shopping journeys. In Figure 3, each target node receives information from its upstream neighbors, which may represent either intra-product transitions, such as view-to-cart for the same SKU, or inter-product transitions, such as viewing one SKU before purchasing another. The attention mechanism learns the relative importance of these upstream nodes for predicting the target node.

Formally, for node i with node embedding $NodeEmb_i$ and upstream neighbors N_i^{UP}, the Graph Transformer maps $NodeEmb_i$, each neighbor j's $NodeEmb_j$, and the corresponding edge weight $w_{j,i}$ into an attention score $\alpha_{j,i}$ for $j \in N_i^{UP}$. The scores are normalized across all neighbors, and i's neighbor embedding $NeighEmb_i$ is computed as an attention-weighted aggregation of neighbor information (Equations 9-14).

$query_i = W_{query}\ NodeEmb_i$ (9)

$key_i = W_{key}\ NodeEmb_i$ (10)

$value_i = W_{value}\ NodeEmb_i$ (11)

$edge_{j,i} = W_{edge}\ W_{j,i}$ (12)

$\alpha_{i,j} = \frac{Att(query_i,\ key_j + edge_{j,i})}{\sum_{k\in N_i^{UP}} Att(query_i,\ key_k + edge_{k,i})}$, where $Att(x_1, x_2) = \exp(\frac{x_1^T x_2}{\sqrt{d}})$, d is the dimension of x (13)

$NeighEmb_i = \sum_{k\in N_i^{UP}} \alpha_{i,k}\ value_k$ (14)

Intuitively, attention score $\alpha_{i,j}$ measures the relative influence of upstream node j on target node i. For example, for SKU A purchase node, a high attention score from this product's cart node indicates a strong intra-product conversion signal, while a high attention score from the view node of another product suggests an inter-product relationship, such as substitution or complementarity. Thus, the neighbor embedding $NeighEmb_i$ summarizes both within-product conversion patterns and cross-product spillover effects embedded in customer shopping journeys.

The neighbor embedding is then fed into a separate MLP to generate neighbor-based quantile forecasts, i.e., $NeighForecast_i \in \mathbb{R}^{H\times|Q|}$ (Equation 15).

$$NeighForecast_i = MLP(NeighEmb_i) \quad (15)$$

At the last step, we adjust $NeighForecast_i$ using a gate mechanism to avoid double counting information captured from three sources: the node's own time series, burst patterns, and neighbors. For example, when customers purchase a product immediately after view or cart, the neighbors' information may already be reflected in the target node's node embedding. Similarly, during sales bursts, neighboring nodes may also contain burst-related signals that overlap with the target node's burst embedding. Directly adding neighbor-based forecasts to the base and burst forecasts may therefore inflate the final forecasts.

To reduce this inflation, the gate concatenates three embeddings for node i: $NodeEmb_i$, $BurstEmb_i$, and $NeighEmb_i$, and uses a dense layer (with parameter W_{NeighW}) to collectively learn a vector of coefficients, β_i for each step within the forecast horizon (Equation 16). A sigmoid activation ensures $\beta_i \in \mathbb{R}^{H\times1}$ is in the range of [0,1], indicating the percentages of $NeighForecast_i$ that can be taken into final forecasts (Equation 17). More elaboration of the gate mechanism can be found in Appendix 5.

$$\beta_i = \frac{1}{1+e^{-z_i}}, \text{ where } z_i = W_{NeighW}\ Concat(NodeEmb_i, BurstEmb_i, NeighEmb_i) \quad (16)$$

$$RegulatedNeighForecast_i = \beta_i * \ NeighForecast_i \quad (17)$$

3.5. Final Forecasts and Loss Function

Finally, we sum the three sets of forecasts as the final demand forecasts for node i at Q quantiles over a forecast horizon of H step (Equation 18). The quantile forecasts are compared with actual sales to calculate quantile loss (Wen et al. 2017). For each quantile q, let $Forecast_{i,T+h}^{(q)}$ denote the q-quantile forecast and $y_{i,T+h}$ be the actual sales at time $T+h$. The quantile loss is calculated using Equation (19):

$$Forecast_i = BaseForecast_i + BurstForecast_i + \ RegulatedNeighForecast_i \quad (18)$$

$$Loss_q\left(Forecast_{i,T+h}^{(q)}, y_{i,T+h}\right) = q\left(y_{i,T+h} - Forecast_{i,T+h}^{(q)}\right)_+ + (1-q)\left(Forecast_{i,T+h}^{(q)} - \ y_{i,T+h}\right)_+ \quad (19)$$

where $(x)_+ = \max(0,x)$. For instance, let $q = 0.75, y_{i,T+h} = 10, Forecast_{i,T+h}^{(0.75)} = 8$, then $Loss_q(8,10) = 0.75 * 2 + 0.25 * 0 = 1.5$. Finally, we sum the quantile loss across all quantiles, all forecast steps, and all target products to get the total loss (Equation 20).

$$Loss = \sum_i \sum_{1\le h\le H} \sum_{q\in Q} Loss_q\left(Forecast_{i,T+h}^{(q)}, y_{i,T+h}\right) \quad (20)$$

4. Dataset and Evaluation

4.1 Dataset Description

We evaluate our model using a public Kaggle clickstream dataset from a Middle East e-commerce platform spanning October 2019 to April 2020.[3] Because our model is designed to capture

[3] https://www.kaggle.com/datasets/mkechinov/ecommerce-behavior-data-from-multi-category-store/data.

dynamic product relationships and detect early signals of sales bursts, the promotion-heavy holiday season (October to December) provides a natural and challenging testbed for our approach. It contains 38 million user sessions spanning over 239,718 SKUs. Each session records a series of events in an online shopping journey. Each event includes a timestamp, product id, event type (e.g., view), and other attributes such as selling price, category, and brand (see Figure 1(a)). Due to substantial missing values, we exclude category and brand information from the main analysis while presenting a metadata-enhanced model in Appendix 6. We also exclude pricing variables because detailed pricing schemes and promotion schedules are often unavailable in advance on marketplace or hybrid platforms (Tian et al. 2018).

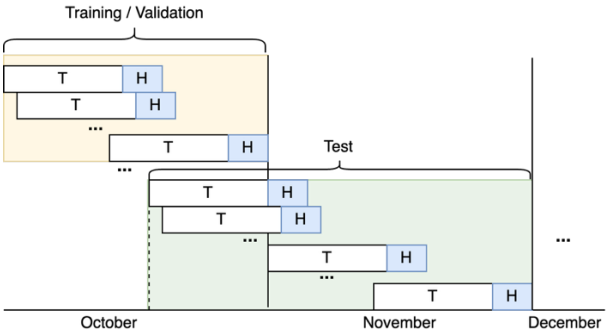
To predict daily SKU sales, we construct one clickstream graph for each day using sessions from that day, as discussed in Section 2.2. Table 2 shows the statistics of these graphs. On average, each graph consists of over 75,600 unique SKUs, 87,490 nodes, and 917,220 unique edges, about 95% of which are inter-product edges. The graphs vary substantially across days due to promotions or frequent SKU entry and exit. The largest graph has 40% more SKUs and six times more edges than the smallest graph.

Table 2: Clickstream Graph Summary Statistics

Variable	Mean	Std	Min	25%	50%	75%	Max
Number of unique SKUs (thousand)	75.60	7.78	61.07	70.54	74.99	80.39	98.59
Number of nodes (thousand)	87.49	12.19	71.15	77.80	83.41	97.04	115.80
Number of unique edges (thousand)	917.22	627.74	377.25	433.25	510.91	1533.05	2365.07
- Intra-product edges	41.52	40.06	10.20	12.27	15.43	83.33	128.86
- Inter-product edges	875.70	589.17	366.45	420.51	498.33	1449.23	2236.22

Clickstream data is highly sparse and we use two strategies to reduce noise and ensure meaningful evaluation. First, we restrict forecasting targets to SKUs with sufficient purchase activity. As shown in Appendix 6, 50% of SKUs have no purchase events within a month, leaving their sales series mostly filled with zeros. Including these products would mainly reward models for predicting zero demand. We therefore forecast only SKUs with non-zero sales on at least 50% of the training days, yielding 3445 SKUs in October and 3251 SKUs in November. Other SKUs remain in the graphs if they connect to the target SKUs. Second, we apply an edge threshold th_{edge} to remove infrequent incidental co-clicks. Not all co-clicks in the same session reflect meaningful product associations. We remove edges that appear in fewer than th_{edge} daily graphs in the context window.

We adopt a periodic model refresh strategy to maintain short-term (e.g., 3-5 days) forecasting accuracy. Short-term demand is sensitive to platform dynamics, such as product launches, promotions, or discontinuations. Thus, at the end of month t, models are fine-tuned with data from month t, and used to forecast sales in month $t + 1$. With the holiday-season data, we implement a two-month sliding train–test cycle as illustrated in Figure 5(a). In practice, the model can be updated more frequently if needed.



(a) Training/Validation/Test Split

Time Series Instances		Use
Context (Input)	Target (Output)	
$Y_1 = (y_{Oct1}, y_{Oct2}, \cdots, y_{Oct7})$ $Y_2 = (y_{Oct2}, y_{Oct3}, \cdots, y_{Oct8})$ … $Y_{22} = (y_{Oct22}, y_{Oct23}, \cdots, y_{Oct28})$	$(y_{oct8}, y_{oct9}, y_{oct10})$ $(y_{oct9}, y_{oct10}, y_{oct11})$ … $(y_{oct29}, y_{oct30}, y_{oct31})$	Training/ validation
$Y_{25} = (y_{Oct25}, y_{Oct26}, \cdots, y_{Oct31})$ $Y_{26} = (y_{Oct26}, y_{Oct27}, \cdots, y_{Nov1})$ … $Y_{52} = (y_{Nov21}, y_{Nov22}, \cdots, y_{Nov27})$	$(y_{Nov1}, y_{Nov2}, y_{Nov3})$ $(y_{Nov2}, y_{Nov3}, y_{Nov4})$ … $(y_{Nov28}, y_{Nov29}, y_{Nov30})$	Test

(b) Training, validation, and test instances in Oct-Nov, T=7, H=3

Figure 5: Time-based nested cross validation

We follow the time-based nested cross validation procedure (Sun et al. 2022) to split each train-test cycle into training, validation, and test subsets. For each cycle, we slice each SKU's sales history into a set of time series of length $T + H$, with a context window $[1, T]$ and prediction horizon $[T + 1, T + H]$. We start with $T = 7, H = 3$ and will perform a sensitivity analysis later. The test set includes instances whose prediction horizons fall into the second month. From the remaining data, 10% of the SKUs are set aside for validation, while the rest are used for training. Figure 5(b) illustrates the instances sliced from an SKU's sales history from October to November. In total, there are 22 instances for training (or validation if the SKU is reserved for that purpose), and 28 test instances that involve November data. In each instance, the first seven values are model inputs and the last three serve as prediction targets.

4.2 Baseline Models, Evaluation Metrics, and Experiment Setup

Based on Section 2.4, we benchmark ForecastClickGraph against 14 widely adopted forecasting models. Clickstream data, which contains view/cart/purchase event sequences within or across SKUs, can be represented as covariates, time series, or graphs, but not all models can support these features. For example, Prophet can only take univariate time series as inputs. Thus, we compare performance under four progressive settings. Table 3 describes these baseline models and their evaluation settings. The implementation of these models is provided in Appendix 7.

(1) *Univariate*: This group of models forecast an SKU's sales only using its past sales data. All non-graph models are evaluated in this setting.

(2) *Covariates*: We add daily view, cart, and purchase counts for each SKU as covariates, creating a multivariate time series per SKU as model inputs. Most univariate models, except Prophet, can be extended to include covariates.

(3) *Graph-based*: Models incorporate both intra-/inter-product transactions for sales forecasting. As illustrated in Figure 2(A), each model first generates a node embedding from the product's time

series through LSTM, and then applies graph convolution to aggregate neighbors' information. This group includes three widely accepted models, GAT, Graph Transformer, and GraphSage.

(4) *Dynamic GNNs*: Unlike the graph-based models that apply graph convolution only once, this group interweaves graph convolution into sequence learning (e.g., GRU or LSTM) by performing it at each time step. We include four models, T-GCN, GConvGRU, GConvLSTM, and GC-LSTM, as they are representative in terms of both model structure and application domain. The detailed model comparison and selection justification can be found in Appendix 2.

Table 3: Benchmarking Models

Model	Description	Setting [1]
Machine learning /*Deep learning* [2]		
Prophet [3]	Prophet integrates three key components: trend, seasonality, and holiday effects learned through maximum likelihood estimation. Predictions are then sampled from the components.	U
MLP [4]	MLP maps an input time series into forecasts through multiple dense layers (see Figure 2(B)).	U, C
NHITS [5]	NHITS iteratively extracts unique information representing a characteristic (e.g., trending) of the input data through sampling and adds it to forecasts.	U, C
Sequence Learning (*Temporal*)		
LSTM [6]	See Section 3.2 for the details of LSTM	U, C, T
DeepAR [7]	DeepAR employs an auto-regressive RNN where the input consists of lagged target values and covariates. It assumes a probability distribution for the target values and estimates the distribution parameters during training.	U, T
Transformer [8]	Transformer embeds each element in a sequence through self-attention (see Section 3.4). It employs an encoder-decoder structure, where the encoder processes input time series and the decode generates forecasts by referencing to encoder embeddings.	U, C, T
Informer [9]	Informer extends from Transformer by ProbSparse attention and distilling operations to focus on dominant information and to improve computational efficiency.	U, C, T
Graph Neural Networks (*Spatial*)		
Graph Transformer [10]	This model employs the self-attention to learn the attention scores between a focal node and its neighbors, and generates a new embedding for the focal node as the attention-weighted sum of the neighbors' information. See Section 3.4. for details.	C, G
GAT [11]	GAT is similar to Graph Transformer except that its attention mechanism is based on the similarity between two node embeddings.	C, G
GraphSage [12]	GraphSage transforms a node's features along with the aggregated information from its neighbors through a dense layer to generate a new node embedding.	C, G
Dynamic GNNs (Spatial + temporal; see Appendix 2 for more details)		
T-GCN [13]	TGCN, originally developed for traffic prediction, first applies graph convolution to extract spatial features at each snapshot, which are then fed into an RNN as sequential inputs.	C, T, G
GConvLSTM GConvGRU [14]	These two models embed graph convolution directly into the hidden state computations by modifying the LSTM/GRU architectures	C, T, G
GC-LSTM [15]	GCLSTM embeds graph convolution directly into the LSTM module, while also utilizing the network adjacency matrix within the LSTM to compute updated hidden states.	C, T, G

[1] Model evaluation settings, **U**: univariate, i.e., SKU purchase data only, **C**: covariate, i.e., SKU view, cart, and purchase as covariates, **T**: considering temporal dynamic in time series; **G**: considering neighborhood features from graphs (i.e., spatial)
[2] We also trained ARIMA and SVR models for each SKU, but excluded them due to their poor performance.
[3] Taylor and Letham 2018; [4] Spiliotis et al. 2022; [5] Challu et al. 2023; [6] Hochreiter and Schmidhuber 1997; [7] Salinas et al. 2020; [8] Devlin et al. 2019; [9] Zhou et al. 2021; [10] Shi et al. 2021; [11] Veličković et al. 2018; [12] Hamilton et al. 2017; [13] Zhao et al. 2020; [14] Seo et al. 2018; [15] Chen et al. 2022

We assess model performance using two widely adopted metrics: mean squared error (MSE) and symmetric mean absolute percentage error (SMAPE). These two metrics are calculated by Equations (21)-(22), where M is the number of target SKUs and $\hat{y}_{i,T+h}, y_{i,T+h}$ are median sales forecast and true sales of SKU i at prediction horizon $T+h$, respectively. MSE emphasizes errors for high-sales SKUs and is therefore suitable when accurate forecasts for popular products are operationally important. In contrast, SMAPE is scale-free, ranges from 0 to 1, and gives more balanced weight across SKUs. However, SMAPE is unreliable for extremely low-sales or zero-sales SKUs, as small forecast errors always produce large SMAPE. For reliability, we calculate SMAPE only for SKUs with $y_{i,T+h} \geq 5$. It is also worth noting that quantile loss optimization (Equations 19-20) drives the models to minimize forecast errors for high-sales SKUs, whereas SMAPE divides errors by sales volume and thus dampens improvements for high sales. As a result, improvement in SMAPE may appear modest. For robustness, we train each model for 30 times using different random seeds and report the mean and standard deviations of both metrics. We assess the significance of model differences using Wilcoxon signed-rank test, as suggested by Steinker et al. (2017).

$$MSE = \frac{1}{M*H}\sum_{i,h}(\hat{y}_{i,T+h} - y_{i,T+h})^2 \qquad (21)$$

$$SMAPE = \frac{1}{M*H}\sum_{i,h}\frac{|\hat{y}_{i,T+h} - y_{i,T+h}|}{(|\hat{y}_{i,T+h}| + |y_{i,T+h}|)} \qquad (22)$$

5. Results

We evaluate ForecastClickGraph in several steps. We first compare it with benchmark models across four scenarios: univariate, covariate, graph-based, and dynamic GNNs, as discussed in Section 4.2. We then examine performance during promotional sales bursts, evaluate new-product forecasting without retraining, and discuss extensions to non-holiday periods. Finally, we conduct ablation analysis.

5.1 Model Performance

Univariate group. As shown in Table 4, within the univariate group, Prophet performs the worst in both MSE and SMAPE, mainly because sparse SKU-level sales histories make individual model fitting difficult. All the cross-learning models outperform Prophet, reducing MSE by over 12% and SMAPE by over 7%. Besides cross learning, MLP and NHITS also benefit from nonlinear layers that can capture complex time series patterns. DeepAR slightly exceeds NHITS in MSE, but only achieves moderate performance because recursive forecasting can propagate errors and its distributional assumption may be restrictive across heterogeneous SKUs. Transformer-based models achieve slightly lower MSEs than DeepAR, likely because attention can extract more complex patterns. LSTM coupled with quantile regression and direct multi-step forecasting performs the best in this group. ForecastClickGraph outperforms all univariate models. Compared to LSTM, it reduces MSE by 15% and SMAPE by 3%. This improvement suggests that relying only on past sales overlooks advance demand signals embedded in view and cart events, as well as interactions with related products.

Covariate models. We evaluate three top-performing univariate models with daily view, cart, and purchase counts as covariates. All covariate models improve over their univariate counterparts. In particular, NHITS reduces MSE by 6%, reflecting its ability to extract incremental signals from covariates. ForecastClickGraph still outperforms this group by 12%-15% in MSE. One explanation is that covariates only capture intra-product transitions but miss inter-product signals, such as co-purchase.

Graph-based models. We continue to evaluate three GNNs baselines that incorporate both intra- and inter-product transitions. As shown in Table 4, these models exhibit similar MSE and SMAPE performance, and improve over the univariate and covariate baselines, confirming the value of capturing both types of transitions using graph structures. With the same inputs, ForecastClickGraph still outperforms these baselines by about 11% in MSE and 3% in SMAPE. This gain highlights the advantage of ForecastClickGraph: standard weighted aggregation may overemphasize signals from highly correlated view/cart/purchase events, whereas the gate mechanism reduces the overlapping information.

Table 4: Model Performance Benchmarking

Model	MSE			SMAPE (%) [4]		
	Mean	SD	Diff (%) [1]	Mean	SD	Diff (%) [1]
A. Machine learning, deep learning, and sequence learning (*Univariate*)						
Prophet [2]	596.60	-	28.25	35.68	-	10.09
MLP	528.87	12.86	19.06***	33.21	0.31	3.40***
NHITS	525.83	11.26	18.59***	33.38	0.50	3.89***
DeepAR	514.48	1.86	16.79***	33.36	0.29	3.84***
Informer	508.24	3.94	15.77***	33.22	0.73	3.43***
Transformer	506.14	1.69	15.42***	33.50	0.45	4.24***
LSTM	505.93	0.91	15.39***	33.09	0.11	3.05***
B. deep learning and sequence learning (*covariates;* with *SKUs' view and cart event as features*) [3]						
Informer	504.61	4.46	15.16***	32.81	0.77	2.22***
LSTM	497.38	2.37	13.93***	32.35	0.32	0.83**
NHITS	487.22	11.24	12.14***	32.23	0.65	0.47
C. Graph Neural Networks (*GNNs*) (w*ith aggregated neighbor event time series as features*)						
Graph Transformer	484.39	3.25	11.62***	33.38	0.31	3.89***
SAGE	482.94	2.48	11.36***	32.85	0.21	2.34***
GAT	480.60	4.18	10.93***	33.18	0.25	3.32***
D. Dynamic GNNs (*with aggregated neighbor events as features*)						
GC-LSTM	494.90	2.33	13.50***	32.63	0.18	1.69***
GConvGRU	492.63	1.86	13.11***	32.60	0.17	1.60***
GConvLSTM	493.76	2.21	13.31***	32.62	0.26	1.66***
T-GCN	473.84	1.96	9.67***	33.47	0.47	4.15***
Our Model	**428.09**	10.55	-	**32.08**	0.35	-

[1] *: p<0.10, **: p<0.05, ***: p<0.01; one-sided Wilcoxon signed-rank test on the performance differences.
[2] The Prophet model fitting is deterministic. Thus, its standard deviation is omitted.
[3] Only top-5 univariate models are considered for the covariate setup. We skip DeepAR because it requires future view and purchase events as covariates for recursive forecasting. Transformer is ignored due to its similarity to Informer.
[4] Cross-learning models show similar SMAPE because quantile loss optimization emphasizes high-sales SKUs, for which SMAPE improvements tend to appear small given the large sales denominator in Equation (22).

Dynamic GNNs. Finally, we evaluate dynamic GNN models, which combine graph convolution with sequence learning by aggregating neighbor information at each time step. As shown in Table 4, these models perform slightly better than the graph-based models in SMAPE but still fall short of ForecastClickGraph. ForecastClickGraph reduces MSE by 9–13% and improves SMAPE by up to 4% relative to these baselines. The results suggest that dynamic GNNs capture time-varying neighbor patterns but remain susceptible to noise and redundant signals. In contrast, ForecastClickGraph uses aggregated graphs and burstiness signals to reduce noise and a gate mechanism to mitigate redundancy.

Overall, ForecastClickGraph consistently outperforms all baselines in terms of MSE and SMAPE. The comparison across four scenarios shows that intra- and inter-product transitions provide useful advance demand information, but the high correlation among view/cart/purchase events necessitates selective extraction of these signals.

5.2 Burst Period Breakdown

Next, we analyze the model performance during active promotion periods. After sorting total sales volume by date, we identified two periods (11/14-20 and 12/15-28), during which most SKUs experienced unusually high sales volume perhaps due to platform-wide promotions. Figure 6 displays the sales time series (grey lines) of two representative SKUs, with the burst periods highlighted. Both series exhibit sharp spikes. For instance, SKU A's sales reach 94 on December 16, more than doubling the previous day's sales. The spike was followed by a sharp decline, consistent with reduced post-promotion demand due to consumer stockpiling (Hewage et al. 2022).

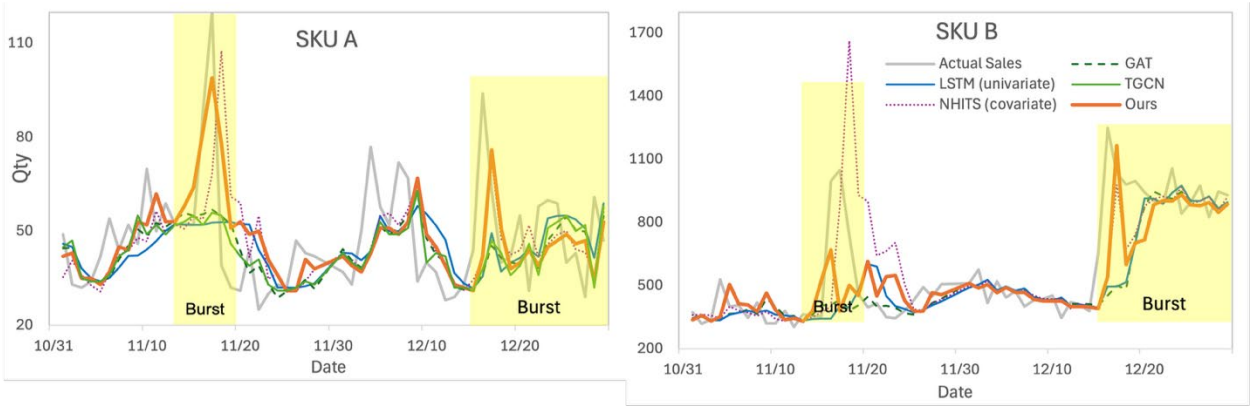


Figure 6: Actual and Forecasts of Example SKUs

During a holiday season, the success of an SKU largely hinges on the burst periods. As shown in Figures 7(A)–(B), these periods cover only 21 of the 61 days in the season but account for 52% of total sales. Accurate SKU-level sales forecasting during burst periods is therefore critical for promotion planning, inventory allocation, and replenishment. However, as shown in Figure 6, promotions create short-lived demand spikes and drops, often lasting only 1-2 days, making forecasting difficult.

(A) (B) (C)

Figure 7: Sales and MSE in Burst and Normal Periods

Next, we compare model performance during burst and normal periods using the four top-performing models from Table 4. As shown in Figure 7 (C), the models have similar MSEs of around 60 during stable normal periods, where even a univariate LSTM can forecast accurately from SKU sales history. In contrast, during burst periods, MSEs exceed 1400 because demand rises and falls sharply. ForecastClickGraph achieves an MSE of 1466, while the best baseline, GAT, reaches 1673, 14% higher.

Figure 6 further shows that ForecastClickGraph detects demand changes more quickly and adjusts its forecasts in line with actual sales. Other models either miss the demand surge or respond with a delay. Such latency is costly because sharp increases are often followed by sharp declines. For example, the NHITS covariate model first underestimates the sales on the peak day (11/17) and then overestimates sales on the following day, leading to a large MSE of around 1700 in the burst periods.

Overall, ForecastClickGraph is more responsive to dramatic demand changes caused by grand promotions than the state-of-the-art baseline models. Because burst periods account for more than half of holiday-season sales, this responsiveness is crucial for inventory and replenishment planning. Appendix 8 provides additional evidence that our graph-based modules are critical in forecasting during promotions.

5.3 Forecasting New Product Demand (Cold Start)

By leveraging generalized patterns learned from pooled product sales data, cross-learning models are well suited for forecasting demand for new products on e-commerce platforms with frequent product launches. To evaluate this cold-start capability, we select the top-performing models from each group in Table 4, train them on November data, and apply them directly to products launched in December.

New products have lower sales volume and more severe data sparsity than established ones. Their average daily purchase event count is 1.41, compared with 4.96 for the full dataset (see Appendix 6 Table A6-1). Their purchase nodes have an average incoming degree of only 1.07 in the daily clickstream graphs, indicating that new products have few neighbors and limited graph-based features. To ensure sufficient activity for evaluation, we focus on new products with at least three days of non-zero view, cart, and purchase events in the context window of length T=7. In total, 4769 new products are evaluated.

As shown in Table 5, ForecastClickGraph outperforms the benchmarks by up to 16% in MSE and 12% in SMAPE, with similar gains during burst periods. Among the benchmarks, the covariate NHITS exceeds the univariate model with a lower MSE, again highlighting the value of cart and view signals. However, GAT performs worse than NHITS, likely because sparse graph connectivity hinders neighbor information extraction. The dynamic graph baseline, T-GCN, has a higher MSE than ForecastClickGraph (6.16 vs. 5.22, +15.3%), indicating that coupling recurrence with graph convolution alone does not fully address cold-start sparsity. In contrast, ForecastClickGraph mitigates weak interconnectivity through a regulation gate that flexibly adjusts its neighbors reliance. Finally, compared with established products in Table 4, new products have low initial sales volumes and thus smaller MSEs (5 vs. 428).

Table 5: Model Performance for New Product Sales Forecasting

Model [1]	**Overall** [2] [3] [4]				**Burst Periods** [2] [3] [4]			
	MSE	**Diff (%)**	**SMAPE (%)**	**Diff (%)**	**MSE**	**Diff (%)**	**SMAPE (%)**	**Diff (%)**
LSTM (Univariate)	6.22 (0.08)	16.08***	44.47 (0.05)	9.17***	6.65 (0.08)	17.59***	44.60 (0.05)	8.95***
NHITS (Covariate)	5.33 (0.14)	2.06	46.07 (2.59)	12.33***	5.57 (0.15)	1.62	47.93 (2.64)	15.27***
GAT (GNN)	6.13 (0.23)	14.85***	42.05 (1.8)	3.95**	6.51 (0.25)	15.82***	42.23 (1.78)	3.84**
T-GCN (Dynamic GNN)	6.16 (0.04)	15.26***	40.47 (1.03)	0.21	6.58	16.75***	40.88 (1.06)	0.65
Our Model	5.22 (0.36)	-	40.39 (0.77)	-	5.48 (0.41)	-	40.61 (0.75)	-

[1] Four top-performing models across groups in the previous Table 4 were chosen for this task. The models, trained on November sales, predict sales of products launched in December without retraining.
[2] The standard deviations are shown in parentheses.
[3] *: p<0.10, **: p<0.05, ***: p<0.01; one-sided Wilcoxon signed-rank test on the performance differences.
[4] SMAPE is calculated on SKUs with sales greater than 2.

5.4 Ablation Study

Having demonstrated the effectiveness of ForecastClickGraph, we now perform an ablation study to assess the impact of its two task-specific graph-based components: burst prediction (Section 3.3) and neighbor embedding (Section 3.4.). We remove each component one at a time to estimate its contribution.

The results in Table 6 underscore the importance of both components. Removing the burst prediction module increases MSE by 1.5% and SMAPE by 2.62%, while disabling the neighbor embedding module raises MSE by 6.62% and SMAPE by 1.5%. Similar patterns appear during burst periods. Because both modules rely on neighbor information, one module may partially compensate when the other is removed. However, removing both modules results in a substantial MSE increase of over 18%, confirming the joint value of incorporating neighbor-based demand signals through graph structures. We also examine the sensitivity of the key parameters, including prediction horizon, context window length, neighbor tiers, and other model settings, with results reported in Appendix 9.

Table 6: Ablation Study

Setup	Overall [1] [2]				Burst Periods [1] [2]			
	MSE	Diff (%)	SMAPE (%)	Diff (%)	MSE	Diff (%)	SMAPE (%)	Diff (%)
Full	428.09 (10.55)	-	32.08 (0.35)	-	1465.68 (42.31)	-	42.59 (0.50)	-
Full – Burst Prediction	434.49 (9.13)	1.50**	32.92 (0.18)	2.62***	1580.38 (46.83)	7.83***	43.54 (0.23)	2.23***
Full – Neighbor Embedding	456.42 (6.62)	6.62***	32.56 (0.36)	1.50***	1496.52 (37.66)	2.10***	43.32 (0.52)	1.71***
Full – Burst Prediction – Neighbor Embedding	505.93 (0.91)	18.18***	33.09 (3.15)	3.15***	1757.63 (2.58)	19.92***	44.65 (0.13)	4.84***

[1] *: p<0.10, **: p<0.05, ***: p<0.01; one-sided Wilcoxon signed-rank test on the performance differences.
[2] The standard deviations are shown in parentheses.

5.5 Extension to Non-promotional Periods

Although ForecastClickGraph is designed for promotional holiday seasons, it can also be extended to non-holiday periods. Holiday seasons feature sudden sales bursts, making burst detection central to our model. In contrast, post-holiday periods have fewer large-scale joint promotions and more stable sales patterns. Figure 8 illustrates this difference using two representative SKUs. Compared with the holiday season (October - December), sales variability is substantially lower from January to April, and burst events occur far less frequently, with only one noticeable burst mainly around Valentine's Day. This pattern suggests that explicit burst modeling is not consistently required during non-holiday periods.

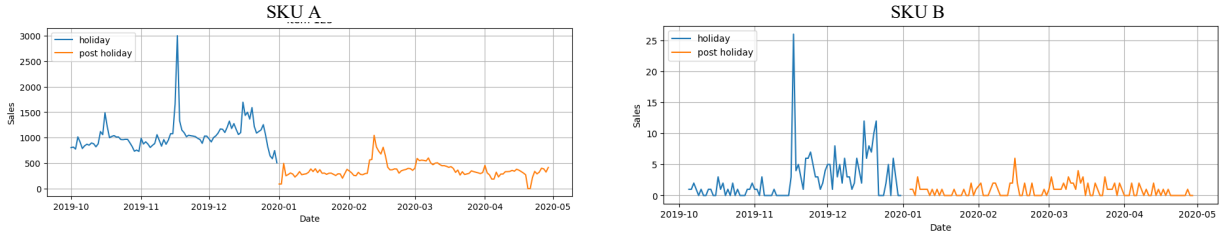


Figure 8: Sales of Two Example SKUs During Holiday and Post-holiday Periods

To adapt ForecastClickGraph to this setting, we introduce a gating mechanism that regulates the contribution of the burst detection module, as specified as in Equations (23-24). Specifically, the concatenated node, burst, and neighbor embeddings jointly learn a coefficient γ_i ($0 < \gamma_i < 1$) through a sigmoid activation function. The burst forecast is then multiplied by γ_i. When no burst behavior is detected, γ_i approaches to 0 and disables the burst detection module; while burst signals are present (e.g., Valentine's Day promotion), γ_i moves to 1 and reactivates it.

$$\gamma_i = \frac{1}{1+e^{-z_i}}, \text{ where } z_i = W_\gamma \, Concat(NodeEmb_i, BurstEmb_i, NeighEmb_i) \quad (23)$$

$$BurstForecast_i = \gamma_i * Emv_i * tanh(MLP(BurstEmb_i)) \quad (24)$$

We evaluate the modified model using sales data from January to April following the same retraining strategy, and benchmark it against the top-performing baseline models from each group in Table 4. Table 7 reports the MSE and SMAPE averaged over the four-month period. Our model outperforms the baselines in MSE and exceeds the best dynamic GNN model by a clear margin. Notably, it captures promotion-driven bursts around Valentine's Day, yielding lower MSE. Since promotions are sparse and sales patterns are relatively stable during this period, all the models show comparable SMAPE.

Table 7: Model Performance Benchmarking During Non-Holiday Seasons

Model	MSE	Diff (%)	SMAPE (%)	Diff (%)
LSTM (Univariate)	362.44 (2.24)	14.41***	30.27 (0.21)	0.17
NHITS (Covariate)	336.86 (16.31)	7.91***	31.02 (0.98)	2.58
GAT (GNN)	338.03 (2.88)	8.22***	30.97 (0.52)	2.42***
T-GCN (Dynamic GNN)	335.26 (2.44)	7.47***	33.94 (0.43)	10.96***
Our Model	310.23 (3.39)	-	30.22 (0.14)	-

[1] Four top-performing models across groups in Table 4 was chosen for this task. The burst detection module in our model was extended with a regulation gate as described in Equations (23-24).
[2] The standard deviations are shown in parentheses.
[3] *: p<0.10, **: p<0.05, ***: p<0.01; one-sided Wilcoxon signed-rank test on the performance differences.

5.6 Evaluation of Quantile Forecasts

While the preceding analyses focus on point forecast accuracy (MSE and SMAPE), our model also produces calibrated quantile forecasts that help retailers achieve target service levels. In inventory management, service level, or fill rate, refers to the probability of satisfying demand without stockouts (Liang et al. 2023). We first evaluate quantile forecast performance using the quantile loss function in Equations (19)-(20). This loss function penalizes under-forecasting (over-forecasting) more heavily at higher (lower) quantiles, which aligns with asymmetric costs in inventory management.

Table 8 reports our model's percentage improvements in quantile loss over top-performing baseline models at the 50th, 75th, and 90th percentiles. We focus on the top three product categories (electronics, computers, and automotive) with sales observed over a 30-day period, as these categories account for most sales records. Consistent with the importance of high service levels in the retail industry, our model delivers the largest gains at the 90th percentile: 13.65% over LSTM, 12.65% over T-GCN, and 7.03% over GAT. It also achieves 4-13% improvements at the 50^{th} and 75^{th} percentiles. Averaged across these percentiles, improvements range from 6.20% to 10.81%, showing that our model supports better inventory and fulfillment decisions across the mid-to-high demand range.

Beyond quantile loss, we assess forecast calibration by comparing target service levels with achieved levels. For a perfectly calibrated model, inventory setting at the q-quantile forecast should

satisfy demand with probability q. For example, a 90th percentile forecast should meet demand on 90% of days and limit stockouts to no more than 10% of days during the experimental period. Figure 9 plots the nominal quantile against the empirical coverage, where the perfect calibration follows the diagonal line. While all benchmarks exhibit some miscalibration, our model tracks the diagonal more closely across quantiles, especially at high service levels (75–90%), which are critical for safety stock planning and service guarantees. For example, at 75th percentile, our model achieves a service level of 68.83%, 8.17% higher than the best benchmarking model. Overall, these results demonstrate that our model improves not only point forecast accuracy but also quantile forecast calibration.

Table 8: Quantile Loss Improvement (%) by our Model at Each Percentile (Higher is Better)

Model	50% Quantile		75% Quantile		90% Quantile		Average Imp.
	Loss	Imp.	Loss	Imp.	Loss	Imp.	
Ours	22.55	---	24.40	---	21.45	---	---
LSTM	24.31	7.25	27.58	11.53	24.83	13.65	10.81
T-GCN	23.73	5.00	28.18	13.41	24.55	12.65	10.35
GAT	23.53	4.16	26.35	7.40	23.07	7.03	6.20

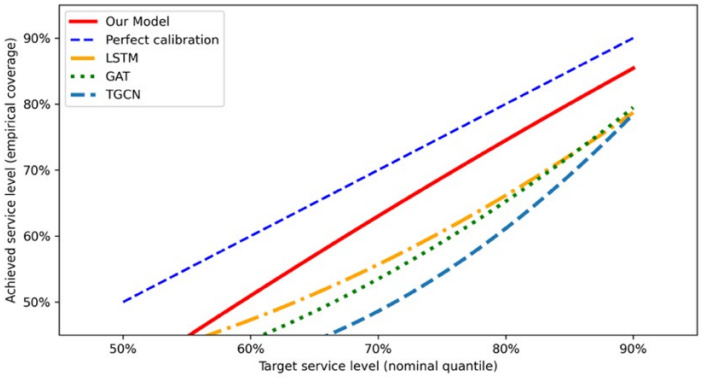


Figure 9: Forecast Calibration Analysis

6. Discussion and Conclusion

6.1 Research and Managerial Implications

Our findings have implications for both OM research and managerial practice, particularly for modeling clickstream data, using product connectivity to guide operational decisions, and forecasting demand when pricing and promotion information is not fully observable.

Novel clickstream graph models: We introduce two graph models for clickstream data, which together constitute the dual-graph design of our framework. The first captures the collective flow of user attention across product stages. Unlike approaches that assume static complementarity or substitution, it constructs directional graphs of dynamic dependencies across products (co-view, co-purchase, view-after-purchase) along daily consumer purchase journeys. Embedding these product-stage graphs into graph neural networks translates granular behavioral interactions into structured forecasting signals. The second model extends graph-based learning to sharp and irregular demand spikes, forecasting sales bursts from

aggregated burstiness patterns within the product-stage graphs, whereas traditional time series models treat such bursts as outliers and fail to predict them. Together, the two models transform clickstream data from unstructured behavioral traces into a dynamic dependency structure suitable for sales forecasting.

Prioritizing product operations using graph connectivity: Connectivity patterns derived from clickstream graphs give managers a practical way to interpret cross-product demand dynamics. We draw on the distinction between search and experience products (P. Huang et al. 2009; Nelson 1970). Search products (e.g., electronics, appliances) involve substantial pre-purchase comparison and generate dense connectivity. As a result, they often exert downstream influence on related products' demand. In our data, their highly connected neighboring nodes (e.g., SKU C in Figure 1) carry advance demand information that improves forecast accuracy. Experience products such as personal care or fashion items depend more on post-consumption learning and brand loyalty, display lower connectivity, and have more isolated demand dynamics, limiting their spillover potential. Our approach converts observed search behavior into a dependency structure, which managers can use to identify leading products, refine operations, and prioritize promotions on highly connected products to stimulate broader demand spillovers.

Forecasting without full pricing transparency: Marketplace and intermediary platforms often lack full access to sellers' proprietary pricing and promotion details, such as discount depth and promotion timing. Our framework instead relies on an alternative and complementary source of demand signals: browsing, comparison, and purchase transitions implicitly reflect the aggregate impact of heterogeneous and often unobserved promotion strategies. Leveraging these behavioral signals captures the realized effects of seller actions without requiring explicit knowledge of pricing rules or future promotion schedules, which is particularly valuable for platforms that do not control sellers' pricing decisions and therefore cannot condition forecasts on future promotion calendars.

6.2 Boundary Conditions, Limitations, and Future Research

The dual-graph design implies that our method performs best when product interactions are rich enough to form highly connected graphs and when promotion events induce bursty sales patterns. It may therefore offer limited advantages over baseline models under three corresponding boundary conditions.

Sparse connectivity. Sparse graph connectivity reduces the value of graph-based relationship modeling. We find that when products are split into high- and low-connectivity groups by node degree (see Appendix 10), our model outperforms both LSTM, a pure time-series benchmark, and GAT by margins above 12% for high-connectivity products, while gains for low-connectivity products fall below 3% because limited neighborhood features constrain the dual-graph design. Rich product interaction is therefore a key boundary condition. It also limits applicability to highly intermittent products, whose infrequent sales cannot support a sufficiently connected graph over time. Our claims of operational applicability apply primarily to active products with sufficient sales histories and graph connectivity.

Stable sales patterns without bursts. The burst-detection component is designed for promotion-driven spikes during holiday or major campaign periods, so it may offer little advantage when demand is stable. Section 5.2 and Figure 7 show that our model substantially outperforms the baselines during active promotion periods but performs comparably outside them. The gating mechanism in Section 5.5 addresses this by dynamically regulating the contribution of the burst-detection module.

Pricing and metadata availability. Our model is built for intermediary or hybrid platforms where proprietary pricing and promotional information is largely inaccessible. Where such information is available, as on reseller platforms, anticipating demand surges directly from these explicit features may be more efficient and accurate than learning them from interactions.

Path to online validation. Online controlled experiments are the standard for quantifying the business value of deployed algorithmic changes, but our anonymized public dataset precludes such tests. Appendix 11 sets out what a production A/B test would randomize, which downstream KPIs it would measure, why substitution and shared capacity across products make clustered or switchback designs necessary (Kohavi et al. 2020), and which of our offline results anchor future validation.

Limitations and future research. Several other limitations point to further work. Incorporating strategic pricing and promotion schedules, where accessible, could improve accuracy beyond what behavioral signals alone deliver. Severe missing values in category and brand data prevented us from using product metadata, and a dataset with fuller metadata would allow a more extensive exploration. We forecast aggregated daily demand, leaving individual purchase behavior and other targets, such as cart abandonment (Ding et al. 2015), for future work. Finally, e-commerce platforms carry large numbers of seasonal and intermittent products, and cross-learning methods that borrow information from related products could predict the onset, timing, and cessation of demand for products with sparse sales histories.

Acknowledgement: The authors extend their gratitude to Haohang Li for his efforts in model implementation and involvement in an early version of the paper.

References

Abbasi, A., J. Parsons, G. Pant, O. R. L. Sheng, S. Sarker. 2024. Pathways for Design Research on Artificial Intelligence. *Information Systems Research* **35**(2): 441–459.

Abolghasemi, M., E. Beh, G. Tarr, R. Gerlach. 2020. Demand forecasting in supply chain: The impact of demand volatility in the presence of promotion. *Computers & Industrial Engineering* **142**: 106380.

Abolghasemi, M., J. Hurley, A. Eshragh, B. Fahimnia. 2020. Demand forecasting in the presence of systematic events: Cases in capturing sales promotions. *International Journal of Production Economics* **230**: 107892.

Adamopoulos, P., V. Todri. 2025. Consumer Social Connectedness and Persuasiveness of Collaborative-Filtering Recommender Systems: Evidence from an Online-to-Offline Recommendation App. *Production and Operations Management* **34**(12): 4015–4038.

Baardman, L., I. Levin, G. Perakis, D. Singhvi. 2018. Leveraging Comparables for New Product Sales Forecasting. *Production and Operations Management* **27**(12): 2340–2343.

Bastani, H., D. J. Zhang, H. Zhang. 2022. Applied Machine Learning in Operations Management. *Springer Series in Supply Chain Management*. Springer Nature, 189–222.

Challu, C., K. G. Olivares, B. N. Oreshkin, F. G. Ramirez, M. M. Canseco, A. Dubrawski. 2023. NHITS: Neural Hierarchical Interpolation for Time Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* **37**(6): 6989–6997.

Chen, G., L. Huang, S. Xiao, C. Zhang, H. Zhao. 2023. Attending to Customer Attention: A Novel Deep Learning Method for Leveraging Multimodal Online Reviews to Enhance Sales Prediction. *Information Systems Research* isre.2021.0292.

Chen, J., X. Wang, X. Xu. 2022. GC-LSTM: graph convolution embedded LSTM for dynamic network link prediction. *Appl Intell* **52**(7): 7513–7528.

Chen, Z.-Y., Z.-P. Fan, M. Sun. 2023. Machine Learning Methods for Data-Driven Demand Estimation and Assortment Planning Considering Cross-Selling and Substitutions. *INFORMS Journal on Computing* **35**(1): 158–177.

Cheng, X., S. Deng, X. Jiang, Y. Li. 2023. Optimal promotion strategies of online marketplaces. *European Journal of Operational Research* **306**(3): 1264–1278.

Chi, Y., D. Lei, L. Zheng, Z.-J. M. Shen. 2024. Demand Forecasting During Grand Promotion for Online Retailing. *Available at SSRN 4777632*.

CNBC. 2023, December 3. Three reasons a strong Black Friday weekend may not mean a blowout holiday season for retailers. CNBC. Available at https://www.cnbc.com/2023/12/03/what-strong-black-friday-retail-sales-mean-for-holiday-spending.html (accessed date June 4, 2024).

Cohen, M. C., R. Zhang, K. Jiao. 2022. Data Aggregation and Demand Prediction. *Operations Research* **70**(5): 2597–2618.

Demirezen, E. M., S. Kumar. 2016. Optimization of Recommender Systems Based on Inventory. *Production and Operations Management* **25**(4): 593–608.

Devlin, J., M.-W. Chang, K. Lee, K. Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. J. Burstein, C. Doran, & T. Solorio, eds. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4171–4186.

Ding, A. W., S. Li, P. Chatterjee. 2015. Learning User Real-Time Intent for Optimal Dynamic Web Page Transformation. *Information Systems Research* **26**(2): 339–359.

Feng, Q., J. G. Shanthikumar. 2018. How Research in Production and Operations Management May Evolve in the Era of Big Data. *Production and Operations Management* **27**(9): 1670–1684.

Feng, Z., R. Wang, T. Wang, M. Song, S. Wu, S. He. 2025. A comprehensive survey of dynamic graph neural networks: Models, frameworks, benchmarks, experiments and challenges. *IEEE Transactions on Knowledge and Data Engineering* **38**(1): 26–46.

Ferreira, K. J., S. Parthasarathy, S. Sekar. 2022. Learning to Rank an Assortment of Products. *Management Science* **68**(3): 1828–1848.

Fildes, R., P. Goodwin, D. Önkal. 2019. Use and misuse of information in supply chain forecasting of promotion effects. *International Journal of Forecasting* **35**(1): 144–156.

Fildes, R., S. Ma, S. Kolassa. 2022. Retail forecasting: Research and practice. *International Journal of Forecasting* **38**(4): 1283–1318.

FRED. 2024, January 1. E-Commerce Retail Sales. FRED, Federal Reserve Bank of St. Louis. Available at https://fred.stlouisfed.org/series/ECOMSA (accessed date June 4, 2024).

Hagiu, A., J. Wright. 2015. Marketplace or Reseller? *Management Science* **61**(1): 184–203.

Hamilton, W., Z. Ying, J. Leskovec. 2017. Inductive Representation Learning on Large Graphs. *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc.

Hao, J., T. Zhao, J. Li, X. L. Dong, C. Faloutsos, Y. Sun, W. Wang. 2020. P-Companion: A Principled Framework for Diversified Complementary Product Recommendation. *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2517–2524.

Haruvy, E., P. T. L. Popkowski Leszczyc, Y. Ma. 2014. Does Higher Transparency Lead to More Search in Online Auctions? *Production and Operations Management* **23**(2): 197–209.

Hewage, H. C., H. N. Perera, S. De Baets. 2022. Forecast adjustments during post-promotional periods. *European Journal of Operational Research* **300**(2): 461–472.

Hochreiter, S., J. Schmidhuber. 1997. Long Short-term Memory. *Neural Computation* **9**(8): 1735–1780.

Huang, P., N. H. Lurie, S. Mitra. 2009. Searching for Experience on the Web: An Empirical Examination of Consumer Behavior for Search and Experience Goods. *Journal of Marketing* **73**(2): 55–69.

Huang, T., R. Fildes, D. Soopramanien. 2014. The value of competitive information in forecasting FMCG retail product sales and the variable selection problem. *European Journal of Operational Research* **237**(2): 738–748.

Huang, T., J. A. Van Mieghem. 2013. The Promise of Strategic Customer Behavior: On the Value of Click Tracking. *Production and Operations Management* **22**(3): 489–502.

Huang, Tingliang, J. A. Van Mieghem. 2014. Clickstream Data and Inventory Management: Model and Empirical Analysis. *Production and Operations Management* **23**(3): 333–347.

Koenker, R., G. Bassett Jr. 1978. Regression quantiles. *Econometrica: Journal of the Econometric Society* 33–50.

Kohavi, R., D. Tang, Y. Xu. 2020. *Trustworthy online controlled experiments: A practical guide to a/b testing*. Cambridge University Press.

Lappas, T., B. Arai, M. Platakis, D. Kotsakos, D. Gunopulos. 2009. On burstiness-aware search for document sequences. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. Paris France, ACM, 477–486.

Liang, P., H. Cavusoglu, N. Hu. 2023. Customers' managerial expectations and suppliers' asymmetric cost management. *Production and Operations Management* **32**(6): 1975–1993.

Ma, S., R. Fildes, T. Huang. 2016. Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra-and inter-category promotional information. *European Journal of Operational Research* **249**(1): 245–257.

Martínez-de-Albéniz, V., A. Planas, S. Nasini. 2020. Using Clickstream Data to Improve Flash Sales Effectiveness. *Production and Operations Management* **29**(11): 2508–2531.

Nelson, P. 1970. Information and Consumer Behavior. *Journal of Political Economy* **78**(2): 311–329.

Petropoulos, F., D. Apiletti, V. Assimakopoulos, M. Z. Babai, D. K. Barrow, S. B. Taieb, C. Bergmeir, R. J. Bessa, J. Bijak, J. E. Boylan. 2022. Forecasting: theory and practice. *International Journal of Forecasting* **38**(3): 705–871.

Qi, Y., C. Li, H. Deng, M. Cai, Y. Qi, Y. Deng. 2019. A Deep Neural Framework for Sales Forecasting in E-Commerce. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 299–308.

Rousseeuw, P. J., C. Croux. 1993. Alternatives to the Median Absolute Deviation. *Journal of the American Statistical Association* **88**(424): 1273–1283.

Sahni, N. S., S. Narayanan, K. Kalyanam. 2019. An Experimental Investigation of the Effects of Retargeted Advertising: The Role of Frequency and Timing. *Journal of Marketing Research* **56**(3): 401–418.

Salinas, D., V. Flunkert, J. Gasthaus, T. Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* **36**(3): 1181–1191.

Schaer, O., N. Kourentzes, R. Fildes. 2022. Predictive competitive intelligence with prerelease online search traffic. *Production and Operations Management* **31**(10): 3823–3839.

Semenoglou, A., E. Spiliotis, S. Makridakis, V. Assimakopoulos. 2021. Investigating the accuracy of cross-learning time series forecasting methods. *International Journal of Forecasting* **37**(3): 1072–1084.

Seo, Y., M. Defferrard, P. Vandergheynst, X. Bresson. 2018. Structured Sequence Modeling with Graph Convolutional Recurrent Networks. *Neural Information Processing* (Vol. 11301). 362–373.

Shi, Y., Z. Huang, S. Feng, H. Zhong, W. Wang, Y. Sun. 2021. Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI 2021)*. 1548–1554.

Shocker, A. D., B. L. Bayus, N. Kim. 2004. Product Complements and Substitutes in the Real World: The Relevance of "Other Products." *Journal of Marketing* **68**(1): 28–40.

Spiliotis, E., S. Makridakis, A.-A. Semenoglou, V. Assimakopoulos. 2022. Comparison of statistical and machine learning methods for daily SKU demand forecasting. *Operational Research* **22**(3): 3037–3061.

Steinker, S., K. Hoberg, U. W. Thonemann. 2017. The Value of Weather Information for E-Commerce Operations. *Production and Operations Management* **26**(10): 1854–1874.

Sun, C., P. Adamopoulos, A. Ghose, X. Luo. 2022. Predicting Stages in Omnichannel Path to Purchase: A Deep Learning Model. *Information Systems Research* **33**(2): 429–445.

Taylor, S. J., B. Letham. 2018. Forecasting at Scale. *The American Statistician* **72**(1): 37–45.

Tian, L., A. J. Vakharia, Y. (Ricky) Tan, Y. Xu. 2018. Marketplace, Reseller, or Hybrid: Strategic Analysis of an Emerging E-Commerce Model. *Production and Operations Management* **27**(8): 1595–1610.

Torry, H. 2020, April 1. Coronavirus Pandemic Widens Divide Between Online, Traditional Businesses. *Wall Street Journal*.

Unal, M., Y.-H. Park. 2023. Fewer Clicks, More Purchases. *Management Science* **69**(12): 7317–7334.

Veličković, P., G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio. 2018. Graph Attention Networks. Presented at the International Conference on Learning Representations.

Voleti, S., P. K. Kopalle, P. Ghosh. 2015. An Interproduct Competition Model Incorporating Branding Hierarchy and Product Similarities Using Store-Level Data. *Management Science* **61**(11): 2720–2738.

Wan, X. (Shawn), A. Kumar, X. Li. 2025. Is It Beneficial to Recommend Differently Priced Products? Experimental Evidence From Online Recommender Systems. *Production and Operations Management* 10591478251392331.

Wang, S., L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, P. S. Yu. 2021, May 13. Graph Learning based Recommender Systems: A Review. arXiv.

Wen, R., K. Torkkola, B. Narayanaswamy, D. Madeka. 2017. A multi-horizon quantile recurrent forecaster. *arXiv Preprint arXiv:1711.11053*.

Wolters, J., A. Huchzermeier. 2021. Joint in-season and out-of-season promotion demand forecasting in a retail environment. *Journal of Retailing* **97**(4): 726–745.

Wu, S., F. Sun, W. Zhang, X. Xie, B. Cui. 2023. Graph Neural Networks in Recommender Systems: A Survey. *ACM Comput. Surv.* **55**(5): 1–37.

Yan, A., C. Dong, Y. Gao, J. Fu, T. Zhao, Y. Sun, J. Mcauley. 2022. Personalized Complementary Product Recommendation. *Companion Proceedings of the Web Conference 2022*. 146–151.

Yilmaz, Y., Ş. G. ÖĞüdücü. 2021. Enriching demand prediction with product relationship information using graph neural networks. *Proceedings of the 2021 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. Virtual Event Netherlands, ACM, 561–568.

Zhang, D. J., H. Dai, L. Dong, F. Qi, N. Zhang, X. Liu, Z. Liu, J. Yang. 2020. The Long-term and Spillover Effects of Price Promotions on Retailing Platforms: Evidence from a Large Randomized Experiment on Alibaba. *Management Science* **66**(6): 2589–2609.

Zhang, X., Y. Zhao. 2010. The Impact of External Demand Information on Parallel Supply Chains with Interacting Demand. *Production and Operations Management* **19**(4): 463–479.

Zhao, L., Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li. 2020. T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems* **21**(9): 3848–3858.

Zhou, H., S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, W. Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 35). 11106–11115.