National College of Ireland

National College of Ireland

Project Submission Sheet

Student Name:	Group G - Lin Myat Zaw Latt / Zin Win Phyo / Zuu Zuu Kyaw Shwe
Student ID:	x24121614 / x23402849 / x24106585
Programme:	MSCDAD_C **Year:** 2024 - 2025
Module:	Deep Learning and Generative AI
Lecturer:	Athanasios Staikopoulos
Submission Due Date:	7th Apr 2025
Project Title:	Deep Learning Models for Customer Time-to-Repurchase Prediction
Word Count:	4753

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	*Lin / Zin Win Phyo / Zuu*
Date:	7th Apr 2025

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. **Late submissions will incur penalties.**
5. All projects must be submitted and passed in order to successfully complete the year. **Any project/assignment not submitted will be marked as a fail.**

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI Acknowledgement Supplement

Deep Learning and Generative AI

Project

Your Name/Student Number	Course	Date
Lin Myat Zaw Latt / 24121614	Deep Learning and Generative AI	7th Apr 2025
Zin Win Phyo / 23402849	Deep Learning and Generative AI	7th Apr 2025
Zuu Zuu Kyaw Shwe / 24106585	Deep Learning and Generative AI	7th Apr 2025

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click here.

AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

Tool Name	Brief Description	Link to tool
ChatGPT	Clarify concepts and provide additional insights on complex topics	https://chatgpt.com/

Description of AI Usage

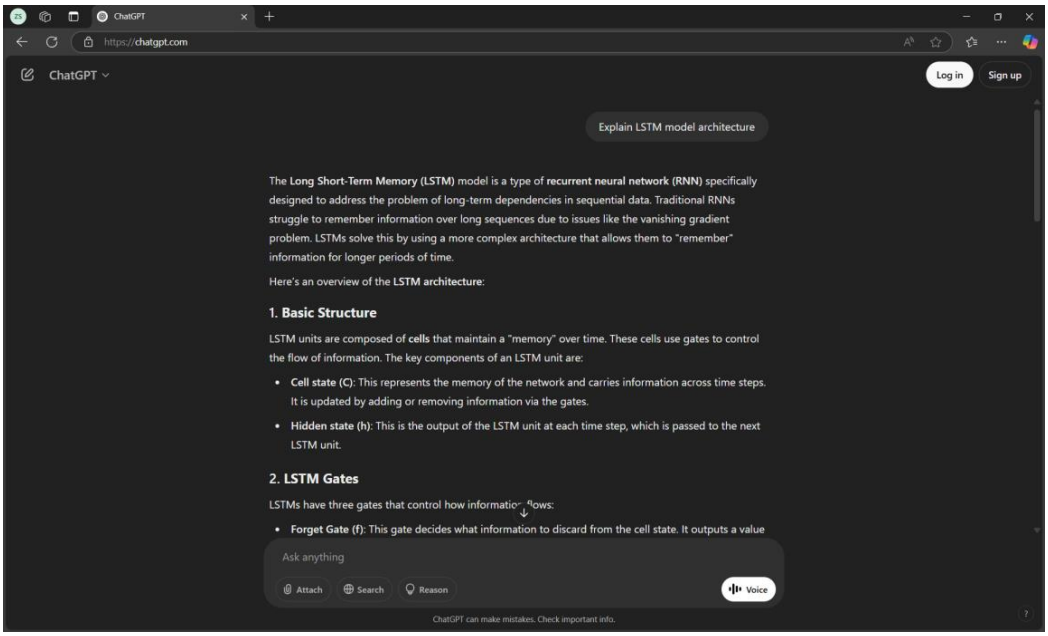
This section provides a more detailed description of how the AI tools were used in the assignment. It includes information about the prompts given to the AI tool, the responses received, and how these responses were utilized or modified in the assignment. **One table should be used for each tool used**.

ChatGPT	
For some of the questions and concepts explored, we sought help from OpenAI's ChatGPT, which helped provide additional understanding.	
Explain LSTM model architecture	1. The Long Short-Term Memory (LSTM) model is a type of recurrent neural network (RNN) specifically designed to address the problem of long-term dependencies in sequential data. Traditional RNNs struggle to remember information over long sequences due to issues like the vanishing gradient problem. LSTMs solve this by using a more complex architecture that allows them to "remember" information for longer periods of time.

Evidence of AI Usage

This section includes evidence of significant prompts and responses used or generated through the AI tool. It should provide a clear understanding of the extent to which the AI tool was used in the assignment. Evidence may be attached via screenshots or text.

Additional Evidence: ChatGPT



Deep Learning Models for Customer Time-to-Repurchase Prediction

Lin Myat Zaw Latt
School of Computing (MSCDAD-C)
National College of Ireland
Dublin, Ireland
x24121614@student.ncirl.ie

Zin Win Phyo
School of Computing (MSCDAD-C)
National College of Ireland
Dublin, Ireland
x23402849@student.ncirl.ie

Zuu Zuu Kyaw Shwe
School of Computing (MSCDAD-C)
National College of Ireland
Dublin, Ireland
x24106585@student.ncirl.ie

***Abstract*—This research looks into deep learning architectures for predicting the number of days till a customer makes the next purchase using the Online Retail II dataset. LSTM, CNN, TCN, CNN+LSTM, and CNN+TCN model were all progressively trained on the sequential purchase data. Convolutional Neural Network (CNN) model performed best in terms of MAE and MedianAE, while TCN had the lowest RMSE. Results show that short-term purchase behavior can be effectively captured using lightweight CNN-based models, offering practical solutions for time-to-event forecasting in retail.**

***Index Terms*—Time-to-event prediction, deep learning, LSTM, CNN, TCN, retail analytics**

I. Introduction

In the context of modern retail, predicting and understanding customer purchasing patterns is important in increasing customer loyalty, managing stock levels, and ensuring marketing campaigns are conducted on time. One key task in this domain is forecasting which is also a challenge is predicting the time of the customer's next purchase, widely known as time-to-event prediction. Unlike traditional sales forecasting, this problem focuses on the individual customer level, requiring models that can learn from behavioral patterns embedded in historical transactions.

As sequential purchase data becomes more accessible, deep learning has proven to be an invaluable asset in capturing temporal dependencies for customer behavior. A variety of mechanisms offered by Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and more recently, Temporal Convolutional Networks (TCNs), provide options for learning from sequential data. However, the relative efficiency of these models, with respect to predicting time-to-next-purchase, is still a largely open question.

We focus on the application of deep learning for predicting time-to-purchase using the Online Retail II dataset. Our analysis centers on comparing the predictive accuracy of five models: LSTM, CNN, TCN, CNN+LSTM, and CNN+TCN, to determine which architecture most effectively captures short-term and long-term purchasing trends.

The remainder of this paper is structured as follows: Section II reviews related work on deep learning for time series and retail prediction tasks. Section III outlines the methodology, including dataset preparation, feature engineering, and models development. Section IV presents the evaluation methodology and experimental results. Section V concludes the paper by summarizing key findings and discussing potential directions for future work.

II. Related Work

Over the past few years, deep learning has proved to be a highly effective approach to time series modeling and sequential behavior prediction, particularly in the domain of e-commerce. With The availability of detailed transaction data, deep learning models could learn from historical patterns to predict the future customer actions. One of the promising architectures includes the Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNNs) and Temporal Convolutional Networks (TCNs). When these models are applied individually or in hybrid combinations like CNN+TCN and CNN+LSTM, they have shown promising results on various forecasting and behavior prediction tasks.

The following literature review explores the relevant aspects of research on these deep learning architectures, focusing on their applications on their usage in retail and customer behavior forecasting. These studies provide the fundamental basis upon which our modeling approach is built and adapted and extended from existing architectures in order to predict time-to-next-purchase at the individual customer level, which is an area still underexplored in the literature.

Diamantaras et al. [1] developed an LSTM-based model to predict shopping intent (browsing, cart abandonment, or purchase) using short user sessions without relying on user profiles. Their forward-looking session step labeling approach allows for real-time intent prediction. With intent prediction accuracy reaching 98%, the model proves that LSTM can usefully be applied to noisy sequential e-commerce data. However, the study focuses on classification, not regression, limiting its direct applicability to tasks like predicting time until next purchase. Regardless, it advocates the use of LSTM for customer behavior modeling and analysis.

Bandara et al. [2] developed a global LSTM model to forecast e-commerce product-level sales demand and capturing cross-series dependencies using both static and dynamic features. They overcome data sparsity through clustering and sequence windowing which is more effective than traditional

models like ARIMA and Prophet on Walmart datasets. While the focus is on aggregate product forecasting, not individual customer behavior, the study validates LSTM's strength in modeling noisy, real-world retail time series. This supports our use of LSTM for customer-level purchase interval prediction.

Pan et al. [3] have focused on solving the problem of recommending a next product and introduced a large scale LSTM based recommendation model (BTRNN). The model combines Word2Vec embeddings with sequential learning to build a purchase path prediction model based on previously made purchases and remakes it to the product purchase order. The model is trained on over 400,000 items which enabled it to surpass Co-Ocurrence and Word2Vec baselines in both offline accuracy and real-world engagement. Although the focus is on item recommendation rather than time prediction, the study reinforces LSTM's effectiveness for modeling ordered purchase behavior. This supports our use of sequence learning for customer-level purchase interval forecastings.

Zhao et al. [4] propose a Bi-LSTM model with self-attention to enhance Sequential Item Recommendation by considering both the temporal order and the importance of interactions. With the improved Item2Vec embeddings, their model surpasses baselines such as GRU in Recall@20 and MRR@20. While concentrating on item recommendation, the architecture demonstrates the potential of Bi-LSTM with attention for enhancing behavioral modeling. This informs potential extensions to our LSTM-based time-to-next-purchase prediction model.

Ala'raj et al. [5] design a behavioral scoring model using Bidirectional LSTM to estimate the probability of a user defaulting on payments for a credit card loan based on temporal and non-temporal customer information. With the inclusion of attention and compared to basic classifiers SVM, RF, and logistic regression, their model exceeds benchmark accuracy, AUC, and Brier scores. The study showcases the ability of LSTM to capture complex sequences of financial actions, particularly among high-risk clients. Regardless of tackling the credit scoring domain, the work underlines the applicability of Bi-LSTM for predicting user actions in advance, which helps in justifying the decision to employ an LSTM to estimate the time-to-next-purchase in retail scenarios.

Using 1D convolutional layers, the raw transaction logs serve as inputs for which Zhao and Wang [6] were the first to pioneer using CNNs to forecast e-commerce sales. Their architecture performed better than the traditional methods such as ARIMA and gradient boosting. Without manual feature extraction, this work shows CNNs' ability to learn local sales patterns. Nevertheless, the model was restricted to aggregate product level forecasts and did not consider time to event prediction at the customer level.

In retail analytics, Verma [7] furthered the role of CNNs and TCNs by introducing a classification framework to determine whether a customer will re-purchase a product within a given time frame. It was found that CNNs and TCNs were able to learn temporal sequences from transaction data well. While this was not a regression-based work, it did show that CNNs and TCNs are able to model sequential purchase behavior and thus provide empirical support for the architectures used in this study.

In a hybrid CNN-TCN model purposed by Navarro et al. [8], they used it for the advanced e-commerce sales forecasting. The architecture captured short term temporal features through stacking CNN layers and long-range temporal features through TCN blocks after passing the outputs to TCN blocks. Both CNN and CNN-TCN models were found to generate accurate and stable forecasts across a variety of product categories and it was shown that they both have the potential to model the purchasing behaviors of the complex. A hybrid structure is introduced to enable such strong foundation for inherent time series prediction tasks that rely on multiscale temporal dynamics.

To estimate consumer's behavior in the e-commerce platforms, Zhang and Khan [9] developed an AI based predictive analytics model that integrates the use of convolution layers. In their architecture, they were using CNNs to detect behavioral trends and use them to make a predictive system that predicted future buying. Although their model did not explicitly include TCNs or model purchase intervals, this further demonstrates that CNNs have utility within structured multivariate retail environments.

Bhattarai and Poudel [10] a CNN-based sales forecasting system that can work in a noisy e-commerce context while reducing prediction errors. The prediction of daily sales was done using several layers of convolutional and dense layers. The study verifies the use of CNNs as scalable, interpretable models for sales prediction and provides guidance on training practices for CNN as well as CNN-TCN architecture.

Li et al. developed [11] a multivariate CNN-LSTM model which forecasted PM2.5 concentrations through the use of CNNs for short-term feature extraction and LSTMs for modeling long-term dependencies. The research demonstrated that the CNN-LSTM hybrid produced better accuracy and convergence than standalone CNNs and LSTMs and GRUs. The application of CNN-LSTM to customer purchase interval modeling requires patterns that contain both regular cycles and unpredictable events.

The research conducted by Yoon et al. [12] showed CNN-LSTM models can effectively predict multivariate energy consumption data including electricity and heating loads. The model generated results with lower prediction error than both traditional statistical models ARIMA and basic deep learning models. The results show the architecture's ability to generalize across different time-dependent behaviors which aligns with how consumer purchase cycles behave.

Yu et al. [13]made use of CNN-LSTM models for the last application involving urban passenger traffic flow prediction. The generated model was superior in error mitigation and managing pattern variation relative to LSTM and ARIMA methods. There is strong evidence to suggest that the CNN-LSTM architecture is useful for predicting sequential actions where the user behavior is random, but prior actions can be leveraged as a basis for learning.

Across the reviewed literature, it is evident that deep learning models have been widely used for a range of predictive tasks in the e-commerce and retail space. LSTM based models are leaders of sequential modeling research landscape because they can remember long-term dependency in the noisy behavior data. However, CNNs and TCNs achieve very good performance on local and global temporal pattern identification without relying in recurrent mechanisms, which frequently faster in training and scalability. Hybrid approaches attempt to merge the strengths of both classes of models to deliver better performance in forecasting scenarios.

However, most of the existing works tend to focus only on classification tasks like product recommendation, intent prediction, or customer segmentation, while relatively fewer studies have investigated the regression of the purchase intervals or time to next purchase estimation as a regression problem. This is an opportunity for further exploration of this gap. By implementing and comparing LSTM, CNN, TCN, CNN+TCN and CNN+LSTM models in particular for customer level time to event prediction, our project contributes to this space. We thus aim to bridge the gap between item-level forecasting and personalized behavioral modeling by adapting these known architectures to a regression setting where we can gain new insights into customer purchase cycles and their temporal dynamics.

III. Methodology

This project follows the CRISP-DM (Cross Industry Standard Process for Data Mining) framework to guide the development of a predictive model for estimating the number of days until a customer's next purchase. This structured approach guided our work through six key stages: business understanding, data understanding, data preparation, modeling, evaluation and deployment.

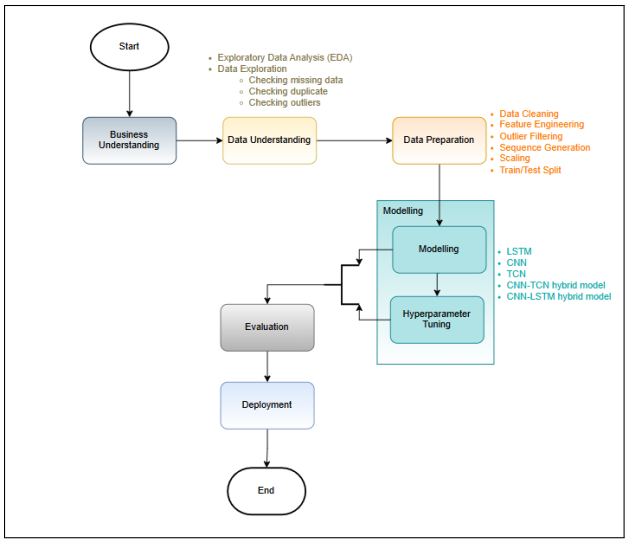


Fig. 1: CRISP-DM Framework for Implementation

A. *Business Understanding*

The primary business objective of this project is predicting the time span until a given customer makes a subsequent purchase. The ability to predict this return interval with precision enables marketing steps to be taken in a more customized and timely manner, enhances the accuracy of demand forecasting, and aids in customer retention strategies. As described, this problem was formulated as a regression problem where the purchasing gap will be the target variable.

B. *Data Understanding*

We used the Online Retail II dataset, which contains over 500,000 historical e-commerce transactions from a UK-based retailer spanning the years 2009 to 2011. The dataset includes key attributes such as `InvoiceDate`, `Customer ID`, `Quantity`, `Price`, and `Country`. The dataset presents a real-world scenario with noisy and incomplete entries, which shaped our preprocessing strategies.

Column Name	Description	Data Type
`Invoice`	Unique identifier for each transaction (invoice number)	Object
`StockCode`	Unique product/item identifier	Object
`Description`	Text description of the product	Object
`Quantity`	Number of items purchased per line item	Numeric (Integer)
`InvoiceDate`	Date and time of the transaction	DateTime
`UnitPrice`	Price per unit of the item	Numeric (Float)
`Customer ID`	Unique identifier for the customer	Numeric (Float/ID)
`Country`	Country where the customer resides or placed the order	Categorical

TABLE I: Summary of Columns

Exploratory Data Analysis (EDA)

EDA provided critical insights into data quality, customer behavior, and temporal purchase trends. Key findings and visualizations include:

- **Transaction Volume**: A boxplot indicating the number of transactions each customer made showed that the majority of the customers made only a few purchases, while a small subset had significantly higher repeat activity. This led us to set a filter where only customers with six or more purchases would be retained.

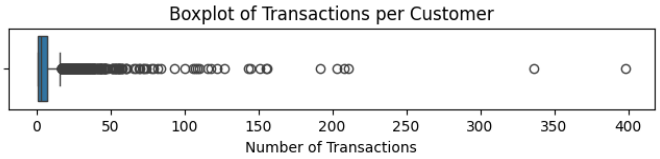


Fig. 2: Boxplot of Transactions per Customer

- **Time Between Purchases**: A new feature, `DaysUntilNext`, was created by grouping transactions by customer and calculating the number of days between

consecutive purchases. This feature exhibited a heavy right skew, which was visualized using boxplots and used to identify and clip outliers beyond the 99th percentile (51 days).

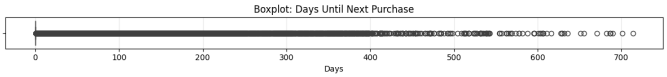


Fig. 3: Boxplot of Days Until Next Purchase

- **Sales Trends**: By resampling data over a period of time, monthly total sales were computed alongside seasonal spikes and business cycles. A time-series line plot revealed signifcant sales increases during the holiday season.

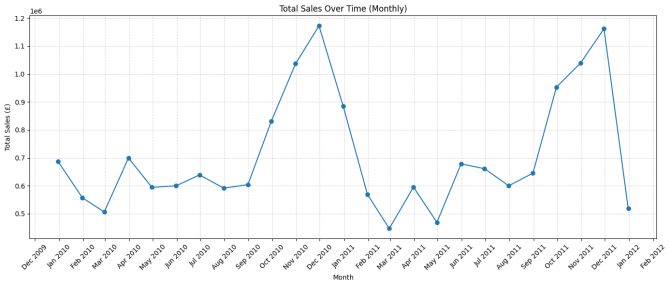


Fig. 4: Total Sales Over Time (Monthly)

- **Invoice Metrics**: Generated histograms of invoice values and quantities, after applying log transformation, confirmed the lognormal distribution which validated our decision to transform the target variable.

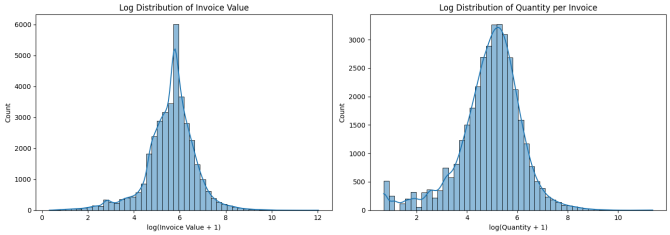


Fig. 5: Invoice Metrics

- **Country-Level Sales**: Through bar plots, the top ten countries by transaction volume as well as their percent share of total sales were highlighted, reinforcing the United Kingdom's dominant dealer market.

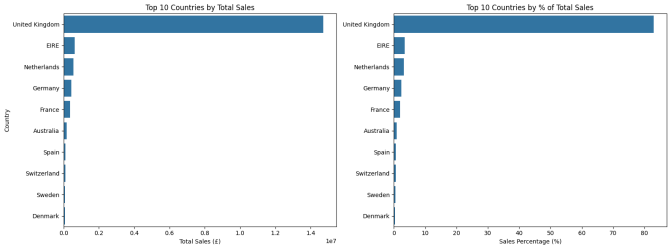


Fig. 6: Country-Level Sales Analysis

C. *Data Preparation*

We applied a range of cleaning and transformation steps to convert the dataset into a form suitable for sequence-based time-series modeling.

1) **Data Cleaning**: The dataset was from a real-world retail environment, and hence it contained various forms of noise and inconsistencies. First, we removed all rows with missing values in the `Customer ID` or `Description` fields to maintain quality as these rows could not be tied to a user or product respectively.

   Secondly, we excluded all transactions with non-positive values for `Quantity` or `UnitPrice`, because entries such as these are not expected to represent valid purchases with the unclear underlying reasons. We removed all the invoices whose identifiers started with letter 'C' to remove the canceled transactions. In addition, `StockCode` values beginning with 'TEST' were removed, as they likely belong to internal test records rather than the actual sales.

2) **Feature Engineering**: With the objective of understanding total sales, we created a feature called `TotalSales`, which is defined by the multiplication of `Quantity` and `Price`. We also ensured the temporal ordering of data by arranging transactions in ascending order of `Customer ID` and `InvoiceDate`.
   To create the target variable `DaysUntilNext`, a forward shift operation was performed to calculate the difference in time between a customer's purchases. Due to the presence of multiple rows per invoice (same timestamp), many transactions had a time gap of 0 days. However, these values were retained to reflect frequent purchasing patterns.

3) **Outlier Filtering**: The `DaysUntilNext` variable had a heavy right skewed distribution that was driven by customers with ragged or long purchase holes. To reduce this effect and improve model training, we applied a 99th percentile cutoff. First, removal of transactions with gap greater than 51 days removes extreme outliers and reduce the target range to be a more learnable interval.

4) **Sequence Generation**: Input data of fixed length are required for sequence-based modeling architectures. We then chose users having at least 6 transactions so that we could have 5 step historical sequences, 6th values as the target. The resulted number of approximately 580000 training samples include samples from all the

qualifying customers.

5) **Target Transformation and Scaling**: To handle the skewness in the target variable, we used the `log1p()` function, that's particularly good for right skewness while preserving zero value.

   Thereafter, the input features and the log transformed targets were scaled using `MinMaxScaler` to make their values in the range of `[0,1]`. Secondly, in order to prevent data leakage as well as be consistent, the scaler was fit to the flattened training inputs, as well as the flattened training targets. This preprocessing step is crucial for stabilizing the training process of neural network models and improving convergence.

6) **Train/Test Split**: Since the time series data needs to be prepared before its evaluation with the model, we decided to split the dataset into a training and a test set with an 80/20 ratio. Since the data is time dependent and sequence dependent, the temporal contiguity of the sequences is enforced by disabling shuffling during the split. It maintains the order of the events chronologically and does not allow data leakage from future transactions into the same training set which is essential in time-series modeling.

The above preparation steps made the dataset reshaped to structured, clean, and normalized form for deep learning. The processed data made it possible to have a consistent and fair training across all the chosen model architectures, since all have clear advantages from the uniform sequence structure and normalized input-target input distribution.

D. Modeling

This project is framed as a supervised time-series forecasting task, where the goal is to predict the number of days until a customer's next purchase using a sequence of prior inter-purchase intervals. For this purpose, we experimented with five deep learning models: LSTM, CNN, TCN, CNN+TCN, and CNN+LSTM, each of which has unique strengths in sequence modeling. We performed all of the modeling in TensorFlow, ensuring the same hyperparameters during training to maintain comparability.

LSTM (Long Short-Term Memory) is a recurrent neural network (RNN) variant designed to capture long-range dependencies in sequential data. It is particularly effective in cases where temporal relationship span variable time intervals, as is common in customer purchase behavior. The LSTM model serves as a strong sequntial baseline.

Layer (Type)	Output Shape	Param #
LSTM (64 units)	(None, 5, 64)	16,896
LSTM (32 units)	(None, 32)	12,416
Dropout (rate = 0.3)	(None, 32)	0
Dense (1 unit)	(None, 1)	33
Total Parameters		**29,345**
Trainable Parameters		**29,345**
Non-trainable Parameters		**0**

TABLE II: LSTM Model Architecture Summary

CNN (Convolutional Neural Network) is useful for detecting local temporal patterns with sliding window filters, capturing local temporal dependencies in features. Despite lacking recurrence capability, CNNs compensate with lower computational cost, making them efficient for time-sensitive tasks.

Layer (Type)	Output Shape	Param #
Conv1D (64 filters, kernel=3)	(None, 3, 64)	256
MaxPooling1D	(None, 1, 64)	0
Flatten	(None, 64)	0
Dense (64 units)	(None, 64)	4,160
Dropout (rate = 0.3)	(None, 64)	0
Dense (1 unit)	(None, 1)	65
Total Parameters		**4,481**
Trainable Parameters		**4,481**
Non-trainable Parameters		**0**

TABLE III: CNN Model Architecture Summary

TCN (Temporal Convolutional Network) takes CNNs further with the use of dilated and causal convolutions to incorporate long-term dependencies without recurrence. TCNs preserve the sequence order and are often faster to train than RNNs, while achieving comparable performance in many time-series applications.

Layer (Type)	Output Shape	Param #
TCN (64 filters)	(None, 64)	136,256
Dense (1 unit)	(None, 1)	65
Total Parameters		**136,321**
Trainable Parameters		**136,321**
Non-trainable Parameters		**0**

TABLE IV: TCN Model Architecture Summary

CNN + TCN Hybrid combines both architectures to leverage local pattern recognition from CNNs and long-range modeling from TCNs. The parallel-branch design makes it possible for the model to process disparate multiple temporal resolutions simultaneously, which is well-suited for sophisticated customer behavior patterns in the data.

Layer (Type)	Output Shape	Param #
InputLayer	(None, 5, 1)	0
Conv1D (32 filters)	(None, 3, 32)	128
GlobalAveragePooling1D	(None, 32)	0
TCN (32 filters)	(None, 32)	15,712
Concatenate	(None, 64)	0
Dense (32 units)	(None, 32)	2,080
Dense (1 unit)	(None, 1)	33
Total Parameters		**17,953**
Trainable Parameters		**17,953**
Non-trainable Parameters		**0**

TABLE V: CNN + TCN Hybrid Model Architecture Summary

CNN + LSTM Hybrid uses convolutional layers for early-stage feature extraction followed by an LSTM layer to learn temporal relationships. This hybrid takes advantage of the diverse feature learning abilities of CNNs, and the sequential modeling capabilities of LSTMs. It balances computational efficiency with strong predictive performance.

Layer (Type)	Output Shape	Param #
Conv1D (32 filters)	(None, 4, 32)	96
MaxPooling1D	(None, 2, 32)	0
LSTM (50 units)	(None, 50)	16,600
Dense (1 unit)	(None, 1)	51
Total Parameters		**16,747**
Trainable Parameters		**16,747**
Non-trainable Parameters		**0**

TABLE VI: CNN + LSTM Hybrid Model Architecture Summary

E. Development

This project was performed within the scope of an academic setting however the deployment phase was designed with the aim of executing with real world scenarios. The primary goal was to illustrate how the developed deep learning models can be integrated into a production environment for forecasting the customer purchase behavior.

- Reusable Codebase: Modular Python scripts and Jupyter notebooks were developed in reusable codebase for preprocessing, training and evaluation. We each defined each model as a function to easily retrain and inference with new data.
- Evaluation Pipeline: To relatively evaluate the model performance, we implemented a unified evaluation framework to calculate MAE, RMSE, and MedianAE. To perform the model comparison, these metrics were collected and visualized during deployment to help with the decision-making.
- Scalability Consideration: The focus of deployment consideration was given to scalable and lightweight models like CNN and hybrid models with efficient architectures due to their balance of performance and computational cost.
- Prediction Output: We can use the best performing model to do predictions on unseen customer data that could feed into downstream applications such as personalized marketing, retention strategies or inventory planning systems.
- Future Integration: To be used in real-world scenarios, the model can be packaged in a REST API using frameworks like Flask or FastAPI. It can be deployed as a microservice that is connected to a customer database and triggered in real-time or batch mode depending on the business need.

Finally, this phase demonstrates that the proposed solution is not only academically good but also practical and generalizable to be used in industry scenarios.

IV. Evaluation and Results

A. Evaluation Methodology

In order to evaluate the performance of proposed models on predicting the time to next purchase a customer would make, we applied a standard, regression-based evaluation approach. The target variable, `DaysUntilNext`, was defined as the number of days between two consecutive purchases for each customer. To reduce the influence of extreme outliers identified during exploratory data analysis, values above the 99th percentile were clipped.

A temporal train-test split was used, in which 80% of the sequences were allocated for training, and 20% for testing, ensuring that the order of events was preserved to avoid information leakage. The target variable was log transformed using `log1p` to reduce variance, while input and output data were scaled using `MinMaxScaler`. All predictions were shifted back to the original scale before evaluation to allow interpretation on the day-scale.

In terms of model training, the Adam optimizer was employed alongside a batch size of 128, a maximum of 20 epochs, and early stopping after 3 epochs to mitigate overfitting. Evaluation was performed using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Median Absolute Error (MedianAE), so that the model could be assessed using both average and extreme outlier performance.

B. Parameterization and Design Choices

All models shared common training parameters but differed in architecture. The CNN model begins with one-dimension convolutional processing before executing max pooling operation. The design incorporates multiple dense layers with an aim to find short-term temporal patterns. The LSTM model achieved its purposes by stacking recurrent layers to analyze sequential patterns.

To utilize different architectural benefits, two hybrid models were developed. The CNN+LSTM model used convolutional feature extraction paired with temporal sequence modeling, while the CNN+TCN model integrated local and long-range representations into a dual-branch structure. Due to resource limitations, there was no extensive hyperparameter optimization; however, some parameter choices like filter sizes, number of units, and dropout rates aligned with empirically determined optimal values.

C. *Quantitative Results*

The test set output results are summarized through Table VII. Among the examined models, CNN achieved the minimum MAE value of 0.5824. The predicted values determined through MedianAE (0.0580) demonstrate the highest level of accuracy together with MAE (0.0580). The CNN+LSTM and CNN+TCN hybrids also performed competitively, with marginally higher errors.

the TCN model also exhibited the smallest RMSE (3.9503), indicating predictive stability against larger deviations, but also had the highest MAE and MedianAE, meaning it was less accurate in its typical predictions. The LSTM model performed consistently well but was outperformed slightly by the simpler CNN-based alternatives.

TABLE VII: Model performance comparison

Model	MAE	RMSE	MedianAE
CNN	**0.5824**	3.9522	**0.0580**
CNN + LSTM	0.5832	3.9520	0.0587
CNN + TCN	0.5849	3.9515	0.0585
LSTM	0.5856	3.9518	0.0614
TCN	0.5950	**3.9503**	0.0707

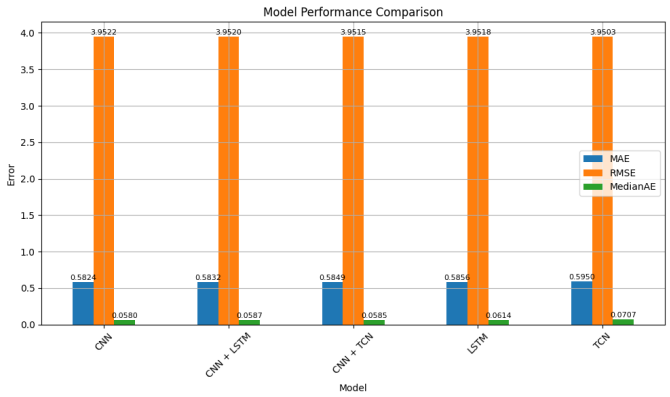


Fig. 7: Model Performance Comparison

D. *Discussion and Implications*

Out of the five models, the results showed that all of them performed with the same level of accuracy regarding predictive capabilities, with only minimal discrepancies across model evaluation metrics. The CNN model outperformed the rest with the lowest MAE and MedianAE values. This suggests that short inter-purchase delays are very predictable and can be compressed to simple cut points without the need for sophisticated temporal constructs invoking intricate frameworks reliant on memory retrieval.

The hybrid models such as CNN+LSTM and CNN+TCN did not show much improvement when compared to less complicated models. A reason for this could be the relatively small input sequence length, which is five time steps, restricting benefits from long-range temporal dependency advantages. While the TCN model had the best performance in RMSE, its higher MAE and MedianAE suggest that it was too lenient on large errors, which negatively impacted the results.

Important findings support the superiority of models based on CNN for regression tasks involving prediction of time-to-event, particularly with respect to interval estimation of customer returns. These models were shown to be precise, efficient, and reliable without introducing unwarranted complexity, thus proving useful for retail forecasting applications

V. Conclusions & Future Work

This research looked into predicting the purchase recurrence interval for customers using deep learning models with data obtained from Online Retail II transactions. The records were considered as an event and the purchase recurrence interval as the event-timer in a time-to-event regression problem. To this end, we explored five deep learning models: LSTM, CNN, TCN, CNN+LSTM, and CNN+TCN. All models were trained on consistent, preprocessed input sequences and assessed using MAE, RMSE, and MedianAE to provide a balanced view of accuracy and robustness.

Our findings reveal the CNN model yielded the best overall results for MAE and MedianAE, highlighting strong performance accuracy in predictions. The LSTM model also performed well, though slightly less-than-CNN accuracy. The TCN model had the lowest RMSE, which suggests it handled occasional large errors well, but its higher MAE and MedianAE show it was less accurate on average. Both CNN+LSTM and CNN+TCN were competitive models, but did not significantly outperform simpler ones. In these findings, short term purchasing behavior can also be effectively captured by lightweight architectures such as CNN, with minimal benefit gained from incorporating additional temporal complexity in this setting.

The research advances the customer behavior modeling domain by showing how deep learning can be implemented in predicting the time to the next purchase—in this case, using retail data. Also, it offers some empirical assessment identifying model type competition under the same experiment conditions, providing insights into design or architecture strategy for retail purposes.

The project had a scope limitation by time and feature availability. For future work, we would investigate the addition of other features like product categories, seasonality effects, and customer demographics. Other improvements along both model and hyperparameter dimensions, such as longer sequence lengths, attention mechanisms, hyperparameter optimization, can also be explored for further improving model performance. Finally, these models can be applied to online recommendation or real time targeting scenario to validate the practical utility.

References

[1] K. Diamantaras, M. Salampasis, A. Katsalis, and K. Christantonis, "Predicting shopping intent of e-commerce users using LSTM recurrent neural networks," in *Proc. 10th Int. Conf. on Data Sci., Technol. and Appl. (DATA)*, 2021, pp. 252–259.

[2] K. Bandara, P. Shi, C. Bergmeir, H. Hewamalage, Q. Tran, and B. Seaman, "Sales demand forecast in e-commerce using a long short-term memory neural network," in *Proc. Int. Conf. on Data Sci., Technol. and Appl. (DATA)*, 2021, pp. 252–265.

[3] J. Pan, W. Sheng, and S. Dey, "Order matters at Fanatics: Recommending sequentially ordered products by LSTM embedded with Word2Vec," in *Proc. KDD Workshop on Deep Learning on Graphs*, Anchorage, AK, USA, 2019.

[4] C. Zhao, J. You, X. Wen, and X. Li, "Deep Bi-LSTM networks for sequential recommendation," *Entropy*, vol. 22, no. 8, p. 870, 2020.

[5] M. Ala'raj and D. Abu-Errub, "Modelling customers' credit card behaviour using bidirectional LSTM neural networks," *J. Big Data*, vol. 8, no. 1, pp. 1–17, 2021.

[6] K. Zhao and C. Wang, "Sales forecast in e-commerce using convolutional neural network," *arXiv preprint arXiv:1708.07946*, 2017, doi: 10.48550/arXiv.1708.07946.

[7] A. Verma, "Consumer behaviour in retail: Next logical purchase using deep neural network," *arXiv preprint arXiv:2010.06952*, 2020, doi: 10.48550/arXiv.2010.06952.

[8] J. Navarro, A. Oliver, and D. Martí, "Convolutional neural networks for advanced sales forecasting in e-commerce," *Stud. Comput. Intell.*, vol. 14, no. 2, pp. 203–214, 2024, doi: 10.19139/soic-2310-5070-2143.

[9] L. Zhang and A. Khan, "AI-driven predictive analytics model for e-commerce consumer behavior forecasting," in *Proc. 2024 IEEE Int. Conf. on Softw. Eng. for Business and Sustainable Development (SEB4SDG)*, 2024, pp. 58–63, doi: 10.1109/SEB4SDG60871.2024.10630338.

[10] A. Bhattarai and P. Poudel, "Sales forecasting using convolutional neural networks in e-commerce," *Int. J. Intell. Syst. Serv.*, vol. 14, no. 2, pp. 120–126, 2024, doi: 10.51983/ijiss-2024.14.2.20.

[11] T. Li, M. Hua, and X. Wu, "A hybrid CNN-LSTM model for forecasting particulate matter (PM2.5)," *IEEE Access*, vol. 8, pp. [insert page numbers], 2020.

[12] N. Yoon *et al.*, "Energy consumption prediction using CNN-LSTM models: A time series big data analysis," in *Proc. IEEE ICCE-Asia*, 2024.

[13] W. Yu *et al.*, "Prediction of passenger flow based on CNN-LSTM hybrid model," in *Proc. IEEE ISCID*, 2019.