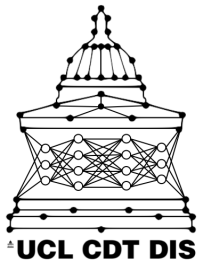

UCL CDT DIS Note

UCLCDTDIS-2022-XX

24th August 2022

Optimized Structure-based Graph Neural Networks to Predict Returns in Fashion Retail

Zheng Cong[a], Mason Cusack[b], Charlie Donaldson[b], Jamie McGowan[a], Gabriel Facini[a], Ofer Lahav[a], and Fabon Dzogang[b]

[a]University College London
[b]ASOS

To address the challenge of predicting customer returns in the fashion retail domain, a graph neural network (GNN) based on an optimized structure is proposed. Based on the graph-represented return event dataset, strategies such as skip connections and max pooling are introduced to improve the model structure and embedding aggregation of the original graph neural network. Benchmark experiments show that the F1 score of the GNN with this optimized structure can reach up to 0.793, which is significantly improved compared to the optimal F1 score of the original GNN of 0.775. Furthermore, GNNs based on the optimized structure have higher F1 scores for most individual countries (8/9) and brands (7/10). In conclusion, the GNN based on the optimized structure has more robust prediction performance than the original GNN model and baseline models that do not utilize graph structure (such as XGBoost, Random Forest). In the future, optimizing GNN models will have the potential to provide more informative predictive results for formulating targeted marketing strategies.

Contents

1 Declaration

I, Zheng Cong, confirm that the work presented in this dissertation is my own. Where information has been derived from other sources, I confirm that this has been indicated in the dissertation.

My code has been submitted to the CongZheng branch in this repository.

2 Introduction

2.1 Overview of background information

ASOS is a British online fashion and cosmetics sales company. The company was established in 2000 and currently employs more than 4,000 people. With sales centers in the UK, US and Europe, ASOS serves 196 countries and sells over 850 brands. In the first half of fiscal 2021, ASOS' revenue has grown to £1.98bn [1, 2].

An important element of the unique online selling experience offered by ASOS is the option of free returns. However, unnecessary duplication of transport increases environmental and economic costs. With the continuous development of e-commerce systems, existing data can be used to model and predict user preferences, thereby optimizing customers' corresponding product recommendation services. Therefore, achieving robust and reliable return rate forecasting is critical to improving customer experience and economics [15, 12].

2.2 Overview of current challenges and past contributions

One approach to address this challenge is a graph representation learning [18] based approach. With the development of high computing power, graph representation learning has been applied to many different scenarios, and the understanding of graph data structure has entered a new era [23].

The graph Neural Network (GNN) is a neural network that can be directly applied to graph-structured data, and has achieved great success in different fields in recent years [27, 24]. In the case of the ASOS data considered here, the connections between customers and products naturally lead to a graph structure, which provides the possibility to use GNNs to train data based on graph representation learning. Another advantage of using a GNN is its ability to make predictions on new instances that have never been seen before. This is important because when many other models make predictions on new instances, the quality of the predictions will be greatly reduced, which is known as the cold start problem [22]. Therefore, the introduction of GNNs is a particularly attractive advantage for industry environments where new products and new customers are added in real time.

However, degradation and over-smoothing issues affect the performance of GNNs for mining information from very large graphs [11]. The degradation problem means that as the depth of the GNN model deepens, the prediction performance of the model will decrease because the redundant layers in the depth cannot learn more information [4]. Over-smoothing means that the information learned by different nodes is more similar and lacks identification, which will

further affect the prediction performance of the model. Therefore, how to modify the deep model structure and aggregation strategy is very important to improve the prediction performance of the GNN model [31].

2.3 Overview of the innovations

In this work, we optimize the original GNN using skip connections, max pooling, and mini-batch training [6, 28, 26]. Among them, customers and products are described as nodes in the graph, the optimized GNN is used to generate embeddings of nodes such as customers and products, and the edges connecting the nodes are marked according to whether they are returned or not. A multilayer perceptron (MLP) classifier is used to complete the downstream task of predicting whether a customer will return a product using the embeddings. The GNN was trained and tested on ASOS data in October and November 2021, and achieved the highest F1 score of 0.793. Compared with the original GNN model and the best baseline XGBoost model, the F1 score has been greatly improved, and achieved significant reduction in classification error.

The structure of the paper is as follows. Section 3 describes raw data and graph-structured data. The prediction results achieved by the baseline model and the original GNN model in phase 1 are discussed in Section 4. The specific principles of optimization strategies for the original graph neural network, such as skip connections, will be discussed in Section 5. Section 6 discusses the benchmark experimental results of the structure-optimized GNN model. Section 7 includes a discussion of specific error analysis and an outlook on future directions.

3 Dataset

3.1 Description of the dataset

This dataset contains records of purchases and returns from ASOS for the period September 2021 to November 2021. The data from September to October is randomly divided into training and validation datasets in a ratio of approximately 9:1, while all data from October to November is used as the test dataset. Each event record, and its corresponding customer and product characteristics, are included. The specific structure of this dataset is shown in Fig. 1.

The training and test datasets each contain $\sim 1,000,000$ purchase events, each item in this event dataset is based on either a purchase or a return as a label and contains the user to whom the event corresponds and the product it deals with. In Fig. 1, this dataset is represented as an edge connecting the customer and product nodes, with blue and red representing whether the event is labelled as a purchase or a return, respectively.

To enhance the impact of user and product features in a purchase or return event, the dataset also includes $\sim 660,000$ customers and $\sim 440,000$ sets of features for the corresponding products. As shown in Fig. 1, for each orange customer node, its features are recorded, such as the age and nationality of that customer, the average historical return rate of that customer, and the rate of different return reasons, among other features. In addition, for the cyan product node, characteristics such as price, discount rate, and average historical return rate are also recorded. A more detailed data table is shown in Table 14.

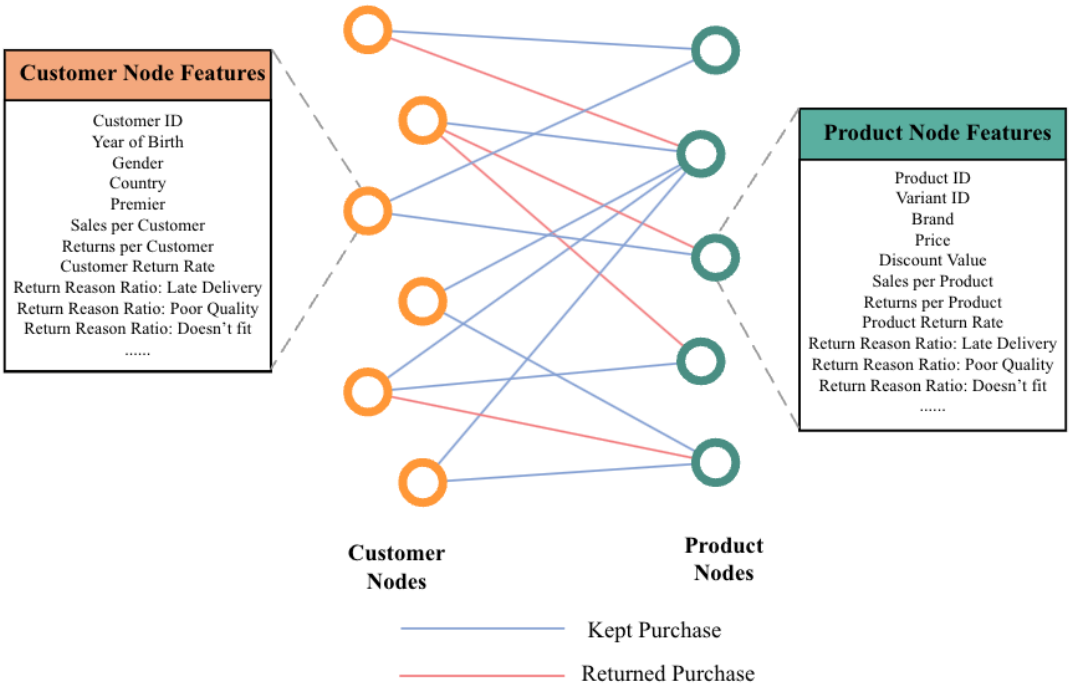

Figure 1: Schematic diagram of the structure of the dataset

To ensure that there are no isolated nodes in the graph structure used for training, each customer in the user dataset is ensured to have existed in the event dataset, i.e. it has purchased or returned at least once, and similarly, each product has been processed at least once, thus ensuring that all nodes in the graph structure have at least one connection.

3.2 Graphical representation of data

The advantage of graph representation learning is to explore the graph representation structure of the original data set to the greatest extent [25]. In this case, the graph neural network can not only use the own characteristics of the original data set, but also learn its relative relationship from the geometric shape of the graph representation structure of the data, and generate node embeddings more efficiently. The node embedding [13] is a group of continuous vectors of node features in the low-dimensional space, which is used to represent the information of the node and the relative relationship between the nodes in the low-dimensional space.

From Fig. 1 in section 3.1, the purchase dataset naturally has the property of being represented using a graph structure. However, the original dataset only contains two kinds of nodes and one kind of edge, which is not conducive to the learning quality of the message passing link during the training of the subsequent graph neural network, and cannot fully utilize the homophily and structural equivalence for graph representation learning influences.

Furthermore, it is worth noting that the customer node and the product node contain common features. For example, each customer has a nationality feature. If the features are aggregated to get their node embeddings, they can be identified as nationality virtual nodes, and by connecting

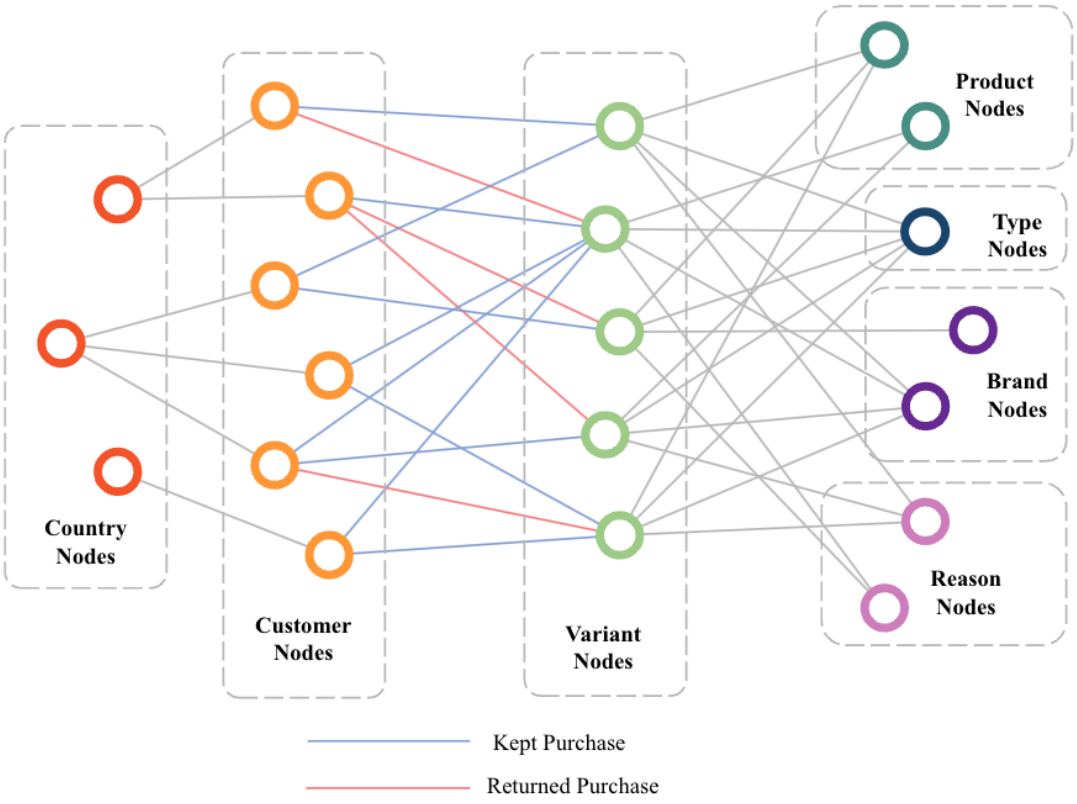

Figure 2: Schematic diagram of the graph representation structure of the dataset

them to each client node, the connectivity and representation complexity of the original graph structure can be greatly improved.

Based on the above considerations, the graph representation structure shown in Fig. 2 is designed. And the types of nodes and edges contained in the modified heterogeneous data are recorded in the table.

From Fig. 2, compared to the original dataset, this new representation structure extracts node embeddings for country, brand, variant, product type (shoes, shirts, luxury goods, etc.) and reason for return (doesn't fit, too big, etc.) from the characteristics of the customer and product nodes, as virtual nodes and connects their counterparts. This enhances the full connectivity and representation diversity of the graph structure. The different nodes have been labelled using different corresponding colours.

We have introduced more specific variant nodes than the product nodes in the original dataset. Specifically, the same product has several individual variants due to different colours, sizes, etc. Therefore, we used multiple variant nodes instead of connecting individual product nodes to customer nodes.

Increasing the complexity and connectivity of the graph representation structure helps graph neural networks to learn homophily and structural equivalence in topology [30]. Specifically, homophily of nodes refers to multiple nodes that have more of the same neighbours or are closer

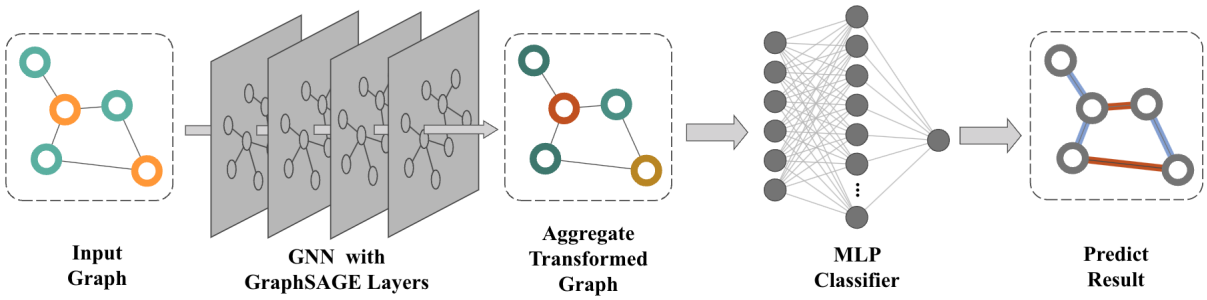

Figure 3: Schematic diagram of learning process of graph neural network in phase 1

together and have more similar embeddings [20]. For example, two product variants that are returned by most customers for the same return reason will have more similar node embeddings because they share most of their neighbouring nodes with that return reason node as a connection. In addition, structural equivalence refers to the fact that embedding of structurally similar nodes is closer [14]. For example, two customers who do not belong to the same nationality but buy the same product will have more similar embedding representations, a property that makes graph neural networks trained to focus more on the structural similarity of different nodes rather than on whether the nodes are adjacent to each other.

In summary, by modifying the dataset to a graph structure, the expressiveness of the data in terms of geometric features is increased, which in turn increases the sensitivity of the graph neural network to the structure of the nodes when it is trained.

4 Summary of Phase 1 Results

4.1 Methodology for phase 1

In the field of fashion recommendation systems, the information in the dataset can be used to predict whether a customer will return an item. Due to the graph-structured nature of this dataset, in phase 1, a simple graph neural network and several baseline models are constructed, thereby highlighting the superiority of graph neural networks using inductive representation learning in several machine learning models.

In order to explore the influence of the graph structure of the data on the improvement of the learning quality of model training, in phase 1, several groups of models based on the baseline method were constructed. Among them, baseline methods are defined as methods that do not learn or exploit any latent representation embedding of the data, including: logistic regression [29], random forest [3], XGboost [7], and multilayer perceptron (MLP) [8].

After the baseline is established, a simple graph neural network is also established to implement inductive representation learning with neural message passing to build the graph structure. The model architecture is shown in Fig. 3.

Observe that Fig. 3 shows that a GNN encoder based on the GraphSAGE layer [19, 16, 10] is used to generate embeddings for each node, and that node information such as customers and variants, products, etc. are represented using embeddings. Based on this data structure, an

MLP-based decoder is also built to binary classify whether a connection between a customer and a variant node is a return, which in turn enables a prediction of whether a user will return an item.

4.2 Experimental results of baseline models

In phase 1, in order to provide a comparison to the capability of GNNs, a number of baseline methods applicable to the task of predicting customer returns using graph-structured data were constructed. Specifically, logistic regression, random forest, 2-layer MLP and XGBoost models were constructed, trained on, validated and tested using graph-structured datasets. The hyper-parameters and architecture of each baseline model are summarised in Table 10. Any parameters not listed here are retained as the default values provided by the package.

The results of the performance of the above models in tests are summarised in Table 1. In the experiments, cross-entropy loss [9], F1 score, precision and recall were selected as metrics to judge the predictive performance of the models, and the rules for calculating the above metrics are shown in the section A .

Model	Test Scores			
	Precision	Recall	F1-score	CE Loss
Logistic Regression	0.723	0.726	0.725	0.6023
Random Forest	0.783	0.716	0.748	0.6293
MLP	0.778	0.723	0.749	**0.5196**
XGBoost	**0.805**	**0.745**	**0.774**	0.5616

Table 1: Results for the baseline models in phase 1 evaluated on the full test data

The best-performing models under each of the judged metrics are highlighted in bold in the results in Table 1. Clearly, the XGBoost model has the best performance in all three metrics, Precision, Recall and F1 score. In the F1 score, XGBoost achieves 0.774, significantly higher than logistic regression's 0.725, while for Precision, XGBoost's 0.805 is still much higher than logistic regression's 0.723. Random Forest and MLP perform relatively similarly in all three metrics. It is worth noting that MLP performs best in terms of cross-entropy loss, achieving a loss as low as 0.5196, which is even lower than the XGBoost model by about 0.05.

To show the baseline results in aggregate, the receiver operating characteristic curves (ROC) and their area under the curve (AUC) values for all models are also shown in Fig. 4. Clearly, XGBoost performs best at all classification thresholds and obtains an AUC value of 0.83, followed by the MLP, random forest and logistic regression models. In summary, XGBoost is the best performing baseline model in the combined evaluation metrics.

4.3 Experimental results of original GNN

In phase 1, the results of this simple graph neural network are summarised in detail by controlling for variables. Specifically, the size of the training data set (10k, 50k and full) was first controlled to explore the effect of data volume on graph neural networks with different number of layers.

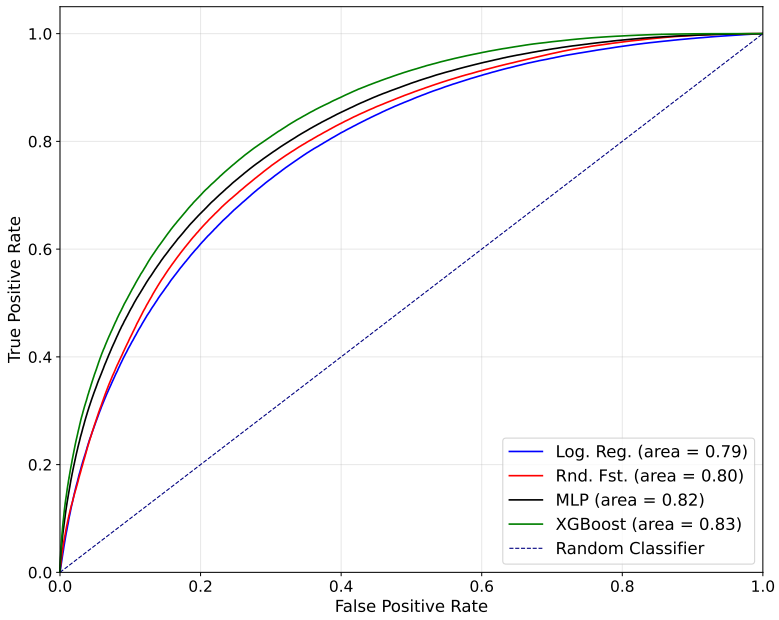

Figure 4: ROC curve for baseline methods evaluated on the full test data.

To explore the diversity and complexity of the graph structure, the type of virtual nodes added was also controlled to investigate how the predictive performance of the model changed as the complexity of the graph structure increased. Finally, a comparison and discussion of the effect of graph neural network depth on model performance was achieved by varying the number of layers in the graph neural network, where models A, B and C represent GNN models containing 1, 2 and 3 GraphSAGE layers respectively, with the specific parameters shown in Table 11.

Combining the above variables, the final best results achieved in the graph neural network in phase 1 were achieved by using a dataset of size full and incorporating the largest number of virtual nodes (variants, product types, brands and countries) and training the GNN containing 2 layers of GraphSAGE layers.

Model	Test Scores			
	Precision	Recall	F1-score	CE Loss
A	0.826	**0.728**	0.774	0.5025
B	**0.831**	0.726	**0.775**	**0.4998**
C	0.830	0.726	0.774	0.5137

Table 2: Results for the original GNN models evaluated on the full test data.

From Table 2, we can see that the GNN with 2 GraphSAGE layers has the best overall performance. In addition, its cross-entropy loss is only 0.4998, which is also slightly lower than the 0.5025 and 0.5137 of the other two models, demonstrating that when the aggregated neighbourhood order is 2, the GNN is able to learn the most effective information through the message passing mechanism of the graph structure, thus achieving the relatively best prediction performance.

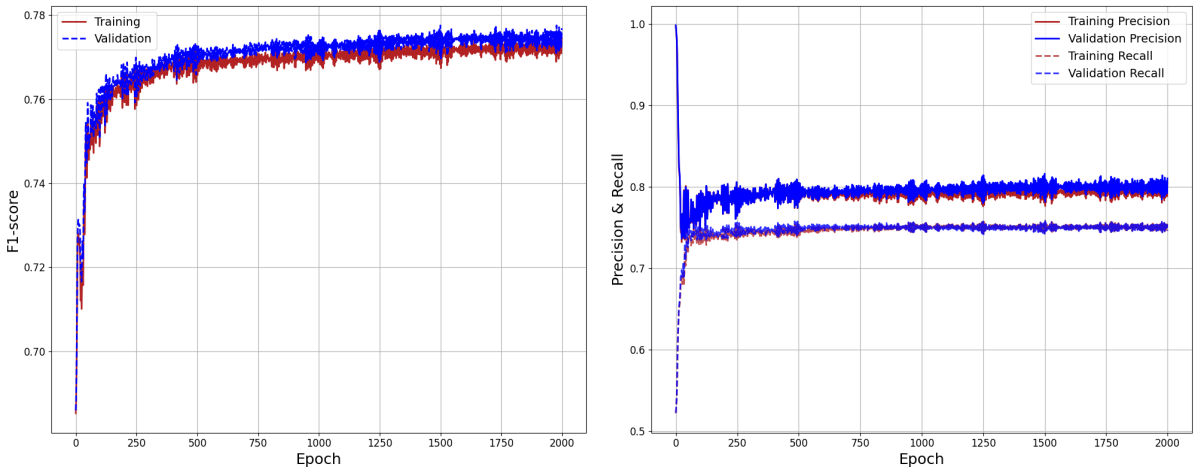

Figure 5: F1-score (left) and precision-recall (right) training and validation metrics across the training period for the original GNN.

By comparing the experimental results in Table 1 and Table 2, it is easy to see that the GNN outperforms the best baseline model, XGBoost, in terms of Precision, F1 score and cross-entropy loss metrics. The best GNN model B achieves a Precision of 0.831, which is higher than the best baseline model XGBoost at 0.805. Its cross-entropy loss is also the lowest among all models. However, its Recall is only 0.726, much lower than XGBoost's 0.745, which results in a final F1 score of 0.775 compared to the XGBoost model's 0.774.

Specifically, the GNN predicted a higher proportion of all events that were in fact labelled as returns (Precision), however, GNN predicted a lower proportion of all events that were in fact labelled as returns (Recall) as true returns. This demonstrates that GNN has a tendency to predict events as returns compared to other baseline models.

The validation metrics F1-score and Precision-recall for each epoch of GNN model B training are shown in Fig. 5, respectively. Clearly, as training proceeds, the metrics largely converge to values similar to the results in the Table 2, but accompanied by more intense turbulence.

Combining the above results, it is clear that at phase 1, a simple combination of GNN and MLP is built, and this model achieves higher F1 scores and lower cross-entropy losses compared to the other baseline models. However, the magnitude of the improvement achieved by this model is more limited, and therefore, it is necessary to further improve the efficiency and the way in which the neural network acquires information about the graph structure. For example, by trying to increase the model depth to increase the GNN's ability to mine graph-structured data, which will exceed the baseline model, or by introducing skip connections to allow more direct connections between clients and variant nodes by bypassing non-essential virtual nodes, or by modifying the aggregation calculation of neighbourhood node embeddings.

5 Graph Representation Learning

In the summary of phase 1 in Section 4.3, it is easy to realise that the simple GraphSAGE-based GNN model used in phase 1 achieved better but limited prediction performance relative

to the baseline model. In phase 2, several optimisation methods were employed to improve the combined predictive performance of the original GNN model over the dataset.

Specifically, three new directions of model optimisation were chosen. Firstly, skip connections were implemented so that messages during model training could be passed over virtual nodes, thus establishing direct connections between nodes (e.g. between variant nodes). Secondly, mini-batch training of the model is achieved by splitting the large graph into multiple sub-graphs. Finally, the aggregation method of the GraphSAGE layer was modified to max pooling [28] in order to change the message aggregation strategy during its embedding generation, to fix the degradation and over-smoothing problems that existed in the original GNN in phase 1..

5.1 Skip connection

The advantage of deep neural networks is that they can learn more complex functions than shallow models in general. However, as shown in Table 2, model C does not perform as well as models A and B combined, which means that when increasing the depth of the neural network, the performance of the model degrades as the number of layers of the architecture increases.

For graph neural networks, the performance degradation caused by increasing the depth of the model does not simply mean that the deeper layers cannot obtain useful information from distant nodes. More seriously, this means that nodes will encounter more nodes when computing the embedding, which makes the set of neighbor nodes of different nodes more similar. The consequence is that node embeddings within the same connected component will tend to converge to the same value, which will be very unfavorable for binary classification prediction, known as the over-smoothing problem. In addition, problems such as overfitting, vanishing gradients and exploding gradients are also likely to arise as the model size increases.

To solve this problem, Skip connection [5] was introduced. Skip connections, which refer to skipping certain layers to propagate gradients further throughout the neural network, so that shallower and deeper layers can be influenced and updated more equally by backpropagation. The figure shows the architecture of a model based on skip connections. The GNN model architecture based on the skip connection strategy is shown in Fig. 6.

Compared to the original model architecture shown in Fig. 3, the biggest difference in this architecture is the introduction of a skip connection mechanism between GraphSAGE layers. From a graph structure perspective, this operation will help to skip certain virtual nodes when computing embeddings, thus reducing the similarity between sets of neighbours of different nodes and thus further fixing the over-smoothing problem. From a gradient calculation perspective, skipping connections helps the vanishing gradient or gradient explosion problem and minimises the detrimental effect of redundant layers on the model.

5.2 Mini-batching

To further improve the graph neural network's ability to exploit node information, the overall embedding aggregation order of the model during training was also improved. Fig. 7 is the process by which GNN uses mini-batching to generate a particular customer node (orange) embeded.

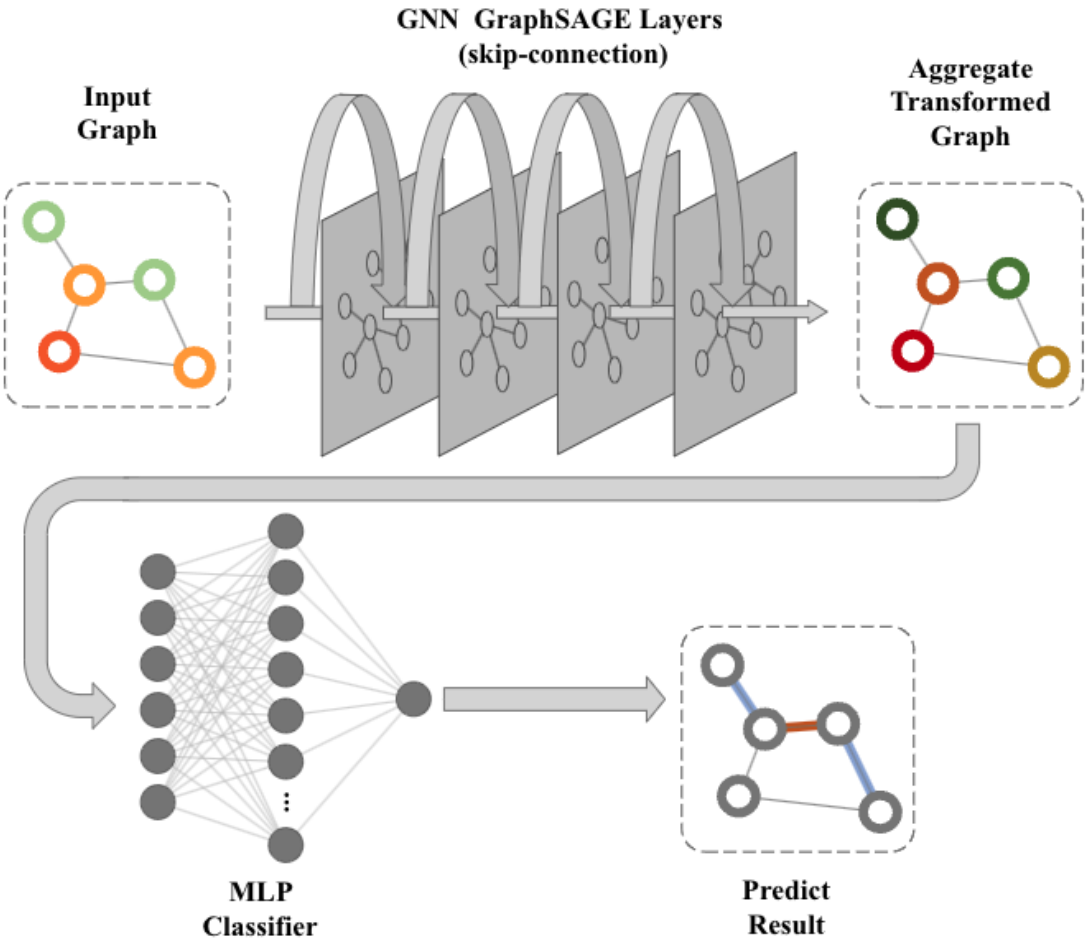

Figure 6: Schematic diagram of learning process of graph neural network based on skip connection

From Fig. 7, the exact process of generating the target node is as follows. First, the number of neighbours sampled for one batch is determined based on the batch size. For example, the number of first-order neighbours sampled is 2 and the number of second-order neighbours sampled is 5. After that, the node features of the second-order neighbours (the five selected nodes in the outer circle of k=2) are aggregated to generate the node embedding of the first-order neighbours (the two selected green nodes in the inner circle of k=1), and then the embedding of the first-order neighbours is aggregated to generate the embedding of the target node. And nodes that are not connected by gray arrows in the graph (such as the red nodes in the circle k=1), which are not selected in this batch, will be selected in the next batch and the embedding will be calculated. Finally, the embeddings of the customer nodes and variant nodes are used as input to the MLP classifier to predict the labels of the target nodes.

The more specific pseudo-code of the algorithm is shown in algorithm 1, and lines 4-5 are the core code. The fourth line represents the aggregation of the embedding of the neighbours sampled at the $k-1^{th}$ layer connected to node v to obtain the k^{th} layer neighbour aggregation feature . The fifth line represents the k^{th} layer neighbour aggregation feature spliced with the $k-1^{th}$ layer embedding of node v to get the embedding of node v at the k^{th} layer.

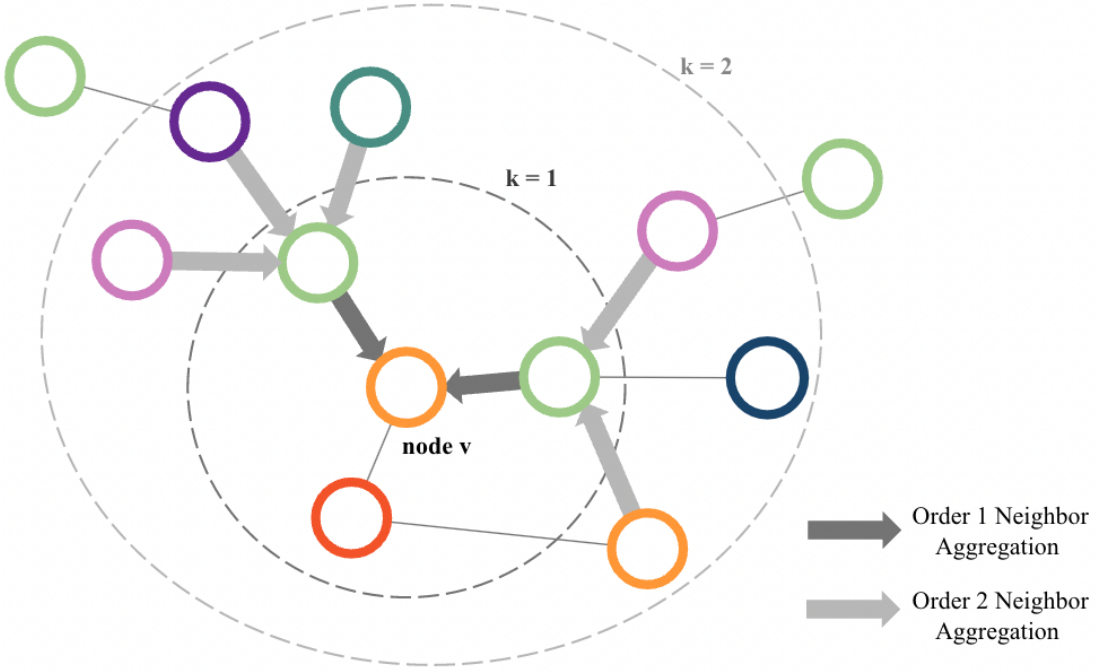

Figure 7: Schematic diagram of the sequence of aggregation process trained by mini-batch

Algorithm 1 GraphSAGE embedding generation (i.e. forward propagation) algorithm

Input: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$; input features $\{\mathbf{x}_v, \forall v \in \mathcal{V}\}$; depth K; weight matrices $\mathbf{W}^k, \forall k \in \{1, \ldots, K\}$; non-linearity σ; differentiable aggregator functions $\text{AGGREGATE}_k, \forall k \in \{1, \ldots, K\}$; neighbourhood function $\mathcal{N} : v \rightarrow 2^{\mathcal{V}}$.

Output: Vector representations $\mathbf{z}_v, \forall v \in \mathcal{V}$.

$\mathbf{h}_v^0 \leftarrow \mathbf{x}_v, \forall v \in \mathcal{V}$;
for $k = 1 : K$ **do**
  for $v \in \mathcal{V}$ **do**
    $\mathbf{h}_{\mathcal{N}(v)}^k \leftarrow \text{AGGREGATE}_k\left(\{\mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v)\}\right)$;
    $\mathbf{h}_v^k \leftarrow \sigma\left(\mathbf{W}^k \cdot \text{CONCAT}\left(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^k\right)\right)$
  end for
  $\mathbf{h}_v^k \leftarrow \mathbf{h}_v^k / \|\mathbf{h}_v^k\|_2, \forall v \in \mathcal{V}$
end for
$\mathbf{z}_v \leftarrow \mathbf{h}_v^K, \forall v \in \mathcal{V}$

Combined with this algorithm, it is easy to find that mini-batch training achieves the selection of node neighbours in batches, thus converting the training of the whole graph into the training of several groups of sub-graphs.

5.3 Max pooling

The aggregation strategy of the GraphSAGE layer was also modified in order to further improve the learning quality of the single layer in the GNN at the time of node embedding generation. In

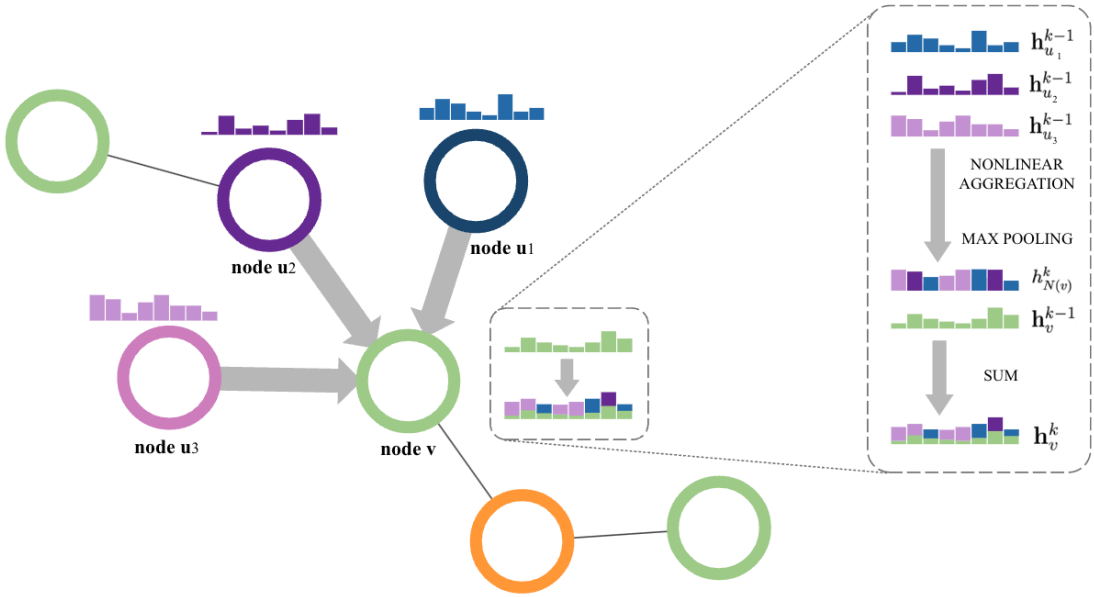

Figure 8: Schematic diagram of the aggregation principle of the GraphSAGE layer based on max pooling

phase 1, the embedding aggregation strategy used was inductive aggregation [17]. This means that each dimension of the target node and all its neighbours emebdding is averaged directly (replacing lines 4 and 5 in the pseudo-code) and then transformed non-linearly. As shown in equation 1

$$\mathbf{h}_v^k = \sigma\left(\mathbf{W}_{\text{self}}^k \mathbf{h}_v^{k-1} + \mathbf{W}_{\text{neigh}}^k \sum_{u \in \mathcal{N}(v)} \mathbf{h}_u^{k-1}\right) \tag{1}$$

where $\mathbf{h}_v^k$ is the representation of node v at layer k, $\mathcal{N}(v)$ is the neighbourhood of v which contains all neighbours u and $\mathbf{W}$ are the trainable wights of the GNN layers.

However, this embedding aggregation method may not be conducive to optimising node differentiation. Because during the embedding aggregation process, some salient node features may be buried by averaging with non-salient features from a large number of other nodes. Therefore, the aggregation strategy is changed to maximize pooling. The process is described in Fig. 8.

When we calculate the embedding of node v (green), we first aggregate the embeddings of its neighbouring nodes u1, u2 and u3 (dark blue, dark purple, light purple). Specifically, the embedding of each neighbour node at layer k-1 is first transformed non-linearly, and then max/mean pooling is applied by dimension, thereby capturing the outstanding performance on the set of neighbours in a particular aspect. The max/mean pooling representation of a neighbour node and the embedding of that node itself are then concatenated and updated to the embedding of the target node at layer k. The specific formula is as 2 and 3.

$$\mathbf{h}_v^{k-1} = max\left(\sigma\left(\mathbf{W}_{\text{neigh}}^{k-1}\sum_{u\in\mathcal{N}(v)}\mathbf{h}_u^{k-1} + b\right)\right) \tag{2}$$

$$\mathbf{h}_v^{k} = \sigma\left(\mathbf{W}^{k}\cdot \text{CONCAT}\left(\mathbf{h}_v^{k-1}, \mathbf{h}_{\mathcal{N}(v)}^{k}\right)\right) \tag{3}$$

After maximum pooling, the maximum values of each neighbouring node in each dimension is retained and the confounding effect of unimportant dimensions of other nodes on the generation of node embeddings is avoided. Comparing Eq. 1 with Eq. 2 shows that maximum pooling helps to keep the node embeddings distinctive and increases their discrimination in the dichotomous prediction task.

6 Benchmark Graph Neural Networks

6.1 Experimental results of improved GNN

To explore the benefits of the improved GNN in predicting customer return rates, several sets of experiments based on GNNs of different sizes were set up to generate results for benchmarking.

Similar to the experiments in phase 1, four models containing different number of layers were defined as Models A, B, C and D. All of the above models use GraphSAGE layers based on the Maxpooling aggregation strategy and all have a hidden dimension of 64 within the layers, the only difference being the number of layers. All four models are built on PyTorch Geometric, trained using mini-batches with a batch scale of 128, and using the Adam optimiser [21] . To prevent overfitting, the models were stopped early when the training epoch was 200. More information on the basic parameters of the experiment can be found in Table 12.

Based on the above configuration, the performance of the graph embedding model based on this graph is recorded for the GNN downstream prediction task without skip connections, as shown in Table 3.

Model	Test Scores			
	Precision	Recall	F1-score	CE Loss
A	**0.855**	0.737	**0.792**	**0.490**
B	0.819	**0.758**	0.787	0.491
C	0.819	0.750	0.783	0.493
D	0.818	0.751	0.783	0.494

Table 3: Summary of F1-scores and CE loss of improved GNN models evaluated on the full test data.

Table 3 shows the performance of the model improved by Mini-batching and Maxpooling under Precision, Recall, F1 score and CE Loss metrics. Clearly, by comparing Table 2 and Table 3, the improved GNN model has significantly improved the F1 score, Recall and CE Loss metrics compared to the original GNN model. Taking the improved GNN model A as an example,

its Precision reached 0.855, much higher than the 0.826 of the original GNN model A and the 0.805 of the best baseline model XGBoost, which proves that the improved means of GNN has effectively improved the reliability of GNN prediction values. Taken together, the improved model A achieves an F1 score of 0.792, while the CE Loss is reduced to 0.4902, which is a huge improvement over the weak improvement of the original GNN model over the baseline model. In summary, the improved GNN is able to capture the richer level of detail accessed by the original model and therefore shows higher predictive performance in

In addition, the improved GNN model shows a more significant degradation in performance as the number of layers grows. Specifically, Model A, which contains only 1 GraphSAGE layer, achieves the highest Precision of 0.855, much higher than the other three models with more layers, and ultimately achieves the highest F1 score of 0.792 and the lowest cross-entropy loss (0.490). This demonstrates that too deep a layer will make the set of neighbouring nodes to be aggregated by different nodes increasingly similar, reducing the differentiation of each set of node embeddings and thus leading to an over-smoothing problem. Therefore, we added a skip connection mechanism to the improved GNN model and documented the performance of the model in a downstream prediction task.

Model	Test Scores			
	Precision	Recall	F1-score	CE Loss
A	0.815	**0.752**	0.782	0.490
B	0.854	0.738	0.792	0.496
C	**0.878**	0.722	0.792	0.510
D	0.873	0.727	**0.793**	**0.489**

Table 4: Summary of F1-scores and CE loss of improved GNN models with skip connections evaluated on the full test data.

From the results in Table.4, the GNN with skip-connection shows the opposite performance trend in F1 score as the number of layers increases. When trained with only 1 layer, model A with skip-improved connections does not perform as well as model A without skip-improved connections in the table, with a score of 0.782. The only thing worth mentioning is that it achieves a Recall of 0.752, which is the best of all models. However, when increasing the number of layers of the model, the performance of the model with the skip connection mechanism evolves substantially. When the number of layers was 4, its performance reached the optimum, achieving the highest F1 score of 0.793 and the lowest cross-entropy loss of 0.4897.

The results shown in Table.3 and Table.4 demonstrate that skipping connections is an effective strategy for improving the degradation and oversmoothing problems that occur with increasing layer counts. From a graph structure perspective, skip connections ensure that more nodes are able to skip certain nodes for direct connections, ensuring the discrimination of their embeddings and the differentiation of their neighbour sets.

The Fig. 9 shows the trend of F1 score and Precision, Recall and other metrics of the improved GNN model during training. Compared with the original GNN model, the F1 score of the improved GNN converges very quickly, converging to an approximate final F1 score in the 12th epoch of training, while the original GNN model takes about 1500 epochs. In addition, the improved GNN remains stable after convergence, and there is no burr-like fluctuation in the F1

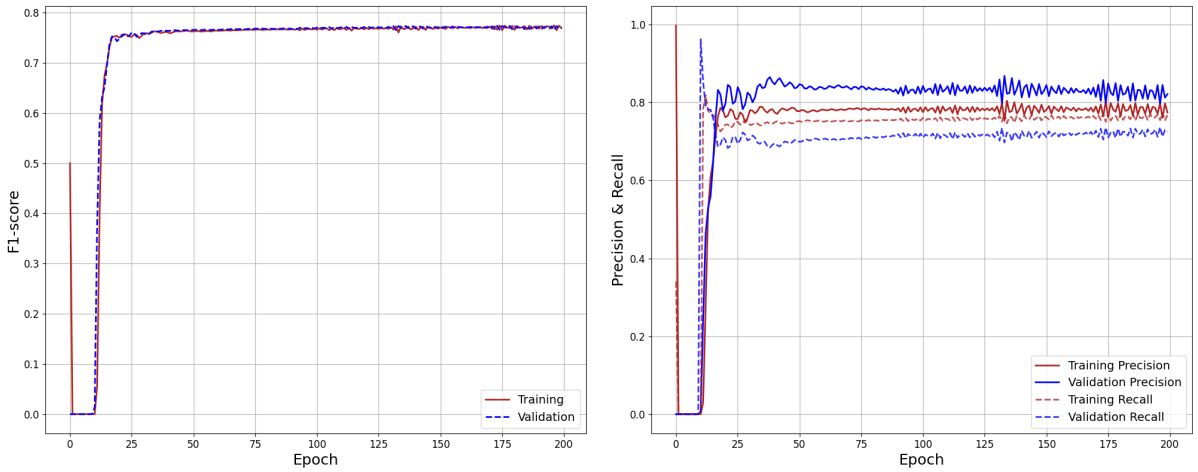

Figure 9: F1-score (left) and precision-recall (right) training and validation metrics across the training period for the improved GNN with skip connection.

score of the original GNN after convergence. This shows the impact of mini-batch training in improving the training efficiency and convergence speed of the model.

In addition, it is clear that the difference between the training Precision and Recall of the improved GNN model is smaller than that of the validation Precision and Recall in the right sub-graph, which represents a degree of overfitting of this improved model due to the increased complexity. However, this overfitting did not jeopardise its performance on the test set in the Table.4.

6.2 Experimental results based on individual countries

For the fashion company ASOS, the performance of the models across countries is of great commercial value, as the prediction of customer return rates from individual countries can be very informative for making business decisions. In order to investigate the predictive strength of GNN for individual countries, different models were trained using a dataset containing all countries, and the test results were divided into nine parts by country and metrics were calculated. The forecasting performance of the best performing GNN original model was recorded against the improved model and compared using the forecasting metrics of the baseline model in phase 1.

The results in Table.5 show the F1 scores of the different models for the nine individual countries. For all nine countries, the F1 scores of the GNN and boosted GNN models are significantly better than those of all other baseline models. This demonstrates the significant help of the graph-structured data for downstream individual country forecasting tasks and is a practical reference for analysing ASOS' marketing strategies in different countries.

And for the comparison between the boosted GNN and the original GNN, it is easy to see that the boosted GNN shows better forecasting results in 8 out of 9 individual countries. For example, for customers from the UK, the changes to the GNN model topology and aggregation approach effectively helped improve their F1 score from 0.752 to 0.774, which is a considerable improvement.

Model	Australia	Austria	Denmark	France	Germany
Logistic Regression	0.635	0.776	0.658	0.593	0.812
Random Forest	0.655	0.785	0.672	0.606	0.817
MLP	0.680	0.792	0.691	0.626	0.819
XGBoost	0.731	0.806	0.717	0.664	0.827
GNN	0.725	0.803	0.724	0.673	0.826
Improved GNN	**0.757**	**0.821**	**0.744**	**0.732**	**0.842**

Model	Netherlands	Sweden	UK	US
Logistic Regression	0.729	0.747	0.673	0.671
Random Forest	0.745	0.759	0.717	0.683
MLP	0.754	0.765	0.727	0.696
XGBoost	0.772	0.778	0.751	0.728
GNN	0.771	**0.781**	0.752	0.727
Improved GNN	**0.801**	0.710	**0.774**	**0.744**

Table 5: F1 scores for individual countries from the GNN model and different baseline models

Model	Australia	Austria	Denmark	France	Germany
Logistic Regression	0.611	0.606	0.611	0.608	0.591
Random Forest	0.633	0.633	0.635	0.633	0.624
MLP	0.527	0.527	0.528	0.518	0.514
XGBoost	0.556	0.567	0.567	0.561	0.561
GNN	0.489	0.509	0.507	0.499	0.500
Improved GNN	**0.436**	**0.487**	**0.485**	**0.494**	**0.487**

Model	Netherlands	Sweden	UK	US
Logistic Regression	0.618	0.607	0.605	0.610
Random Forest	0.638	0.633	0.630	0.636
MLP	0.542	0.528	0.520	0.528
XGBoost	0.573	0.566	0.561	0.563
GNN	0.522	**0.507**	0.499	**0.504**
Improved GNN	**0.500**	0.585	**0.489**	0.505

Table 6: CE Loss for individual countries from the GNN model and different baseline models

Table.6 shows the tested cross-entropy losses for the different models based on individual countries. According to the results shown in the table, the cross-entropy losses of the original GNN model and the boosted GNN model are significantly lower in all countries, while the boosted GNN still performs better in all countries except Sweden and the USA. In addition, the forecasts made by the boosted GNN are more trustworthy and robust than the original GNN and the other baseline models for most countries.

6.3 Experimental results based on individual brand

The analysis on individual brands is of more practical relevance for ASOS executives to specify the corresponding marketing strategies. Similarly, comparative benchmarking experiments were designed for different models of separate brands, in which each event prediction was divided according to the brand to which the variant belonged and its F1 score and cross-entropy loss were tested separately as measures.

Model	ASOS Curve	ASOS DESIGN	ASOS Petite	Bershka	Collusion
Logistic Regression	0.734	0.771	0.771	0.760	0.690
Random Forest	0.751	0.781	0.781	0.768	0.718
MLP	0.757	0.758	0.791	0.775	0.746
XGBoost	0.779	**0.802**	0.802	0.787	0.770
GNN	0.786	0.776	**0.803**	0.789	0.773
Improved GNN	**0.805**	0.791	0.776	**0.826**	**0.779**

Model	New Look	Nike	Stradivarius	Topshop	Other
Logistic Regression	0.611	0.706	0.763	0.744	0.676
Random Forest	0.656	0.726	0.772	0.764	0.741
MLP	0.658	0.741	0.772	0.766	0.745
XGBoost	0.688	0.755	0.786	0.782	0.769
GNN	0.691	**0.756**	0.784	0.781	0.768
Improved GNN	**0.732**	0.711	**0.807**	**0.802**	**0.798**

Table 7: F1 scores for individual brands from the GNN model and different baseline models

Model	ASOS Curve	ASOS DESIGN	ASOS Petite	Bershka	Collusion
Logistic Regression	0.603	0.606	0.606	0.612	0.614
Random Forest	0.630	0.632	0.632	0.637	0.636
MLP	0.516	0.519	0.525	0.548	0.529
XGBoost	0.560	0.568	0.568	0.580	0.564
GNN	0.508	0.501	**0.510**	0.538	0.507
Improved GNN	**0.504**	**0.497**	0.513	**0.462**	**0.480**

Model	New Look	Nike	Stradivarius	Topshop	Other
Logistic Regression	0.609	0.611	0.614	0.608	0.597
Random Forest	0.634	0.635	0.638	0.633	0.625
MLP	0.534	0.542	0.552	0.532	0.503
XGBoost	0.571	0.574	0.586	0.572	0.549
GNN	**0.515**	0.518	**0.542**	**0.516**	0.476
Improved GNN	0.603	**0.482**	0.545	0.527	**0.461**

Table 8: CE Loss for individual brands from the GNN model and different baseline models

Table.7 shows the predicted F1 scores of the different models for the 10 individual brands. It is clear that the F1 scores of the original GNN model and the improved model show substantial

improvements relative to the baseline model for all nine brands except the ASOS DESIGN brand, while the improved GNN achieves better F1 scores for eight of the ten brands. For example, for the Bershka brand, the optimal baseline model XGBoost achieves an F1 score of 0.787, the simple GNN in phase 1 improves it slightly to 0.789, while the improved GNN model in this paper achieves an F1 score of 0.826, which is a significant improvement in prediction results.

Similarly, the cross-entropy losses shown in Table.8 for the 10 individual brands also show this trend. The GNN model has lower cross-entropy losses for all 10 brands, demonstrating that graph structure-based representation learning yields more robust prediction decisions than other baseline models that do not make use of graph structure.

7 Ablation & Future Work

7.1 Error Analysis

In Sect.6.2 considering the predictive performance of individual countries, it is clear to see that the improved GNN performs better for most of the nine individual countries relative to the original GNN and all baseline models. However, for Sweden, the F1 score of the improved GNN model is only 0.710, a very significant degradation relative to the original GNN model of 0.781 and even lower than the logistic regression of 0.747. The higher cross-entropy loss likewise has a very negative impact on ASOS in developing a robust marketing strategy against Sweden. This section therefore analyses the prediction errors for Sweden and selects the average values for Germany, the Netherlands and all countries as a comparison.

Fig.10 shows a comparison plot of the confusion matrices for all countries total, Sweden, Germany and the Netherlands. It is easy to see from the four subplots that the GNN model's predictions are characterised by very high precision and slightly high recall.

Comparing the higher F1 scores for Germany (bottom left) and the Netherlands (bottom right), the reason they maintain their high F1 scores is that their TP values are significantly higher than the average (top left). This means that for an event that is in fact a return, they have a higher chance of predicting it as a return (i.e. a higher-than-average recall). The underlying reason for GNN's poor predictive performance in the Swedish countries (top right), on the other hand, is that it has a much lower recall, with only about 63.34% probability of judging an event as a return when confronted with a fact return, which is why its F1 score is lower than that of the logistic regression model.

Fig.11 shows a schematic comparison of the ROC curves for all countries, Sweden, Germany and the Netherlands. It is easy to see from the curves that Germany (dark purple) has a smoother ROC curve and performs robustly at all classification thresholds due to the large number of customers. The Netherlands (light purple) and Sweden (cyan), on the other hand, have jagged curves overall due to their smaller sample sizes. However, GNN's predictions for Sweden underperform the average for all classification thresholds, while for the higher classification thresholds the Netherlands even outperforms the average for all countries. This shows the excellent predictive performance of GNN for some countries.

Table.9 shows the fundamental differences between the above countries on separate datasets, in terms of metrics such as number, total number of connections, average return rate and F1 score.

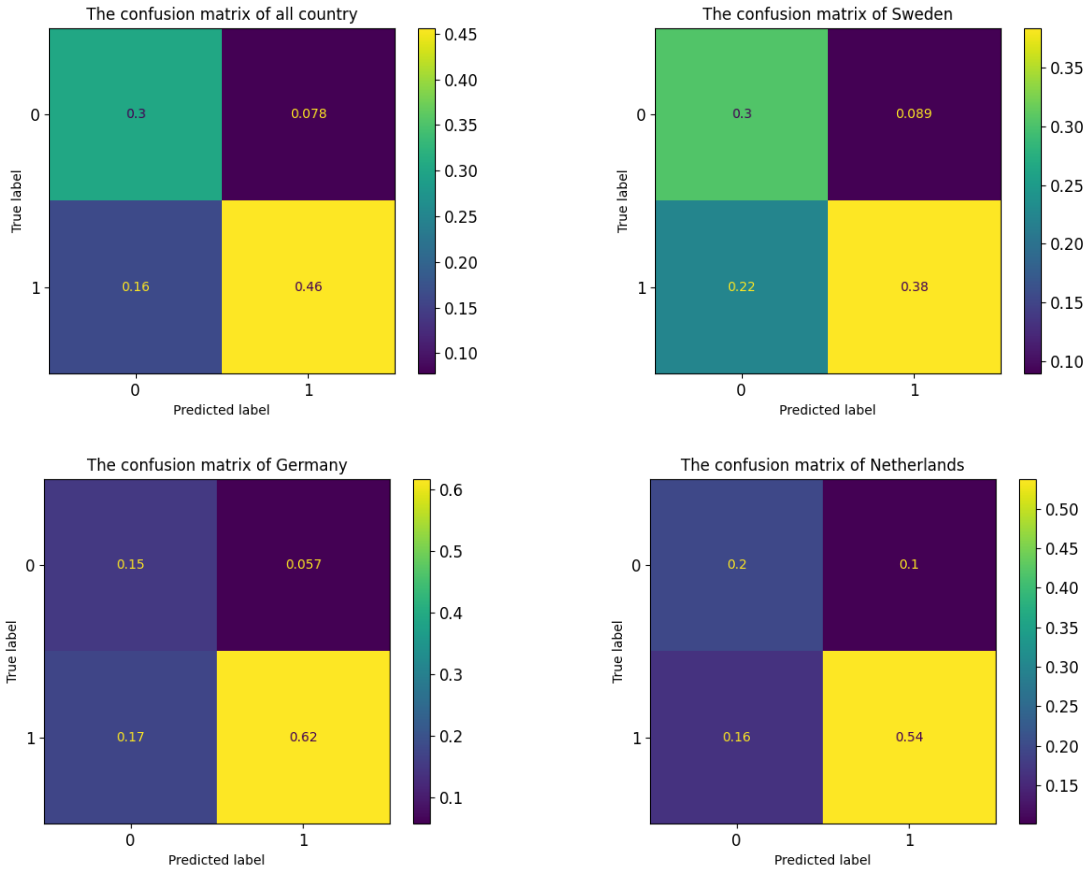

Figure 10: Schematic comparison of confusion matrices for all countries, Sweden, Germany and the Netherlands

Country	Number of customers	Total number of connections	Return Rate	F1-Scores	
				Test	Train
Sweden	112	112	0.607	0.711	0.804
Germany	1995	2012	0.798	0.842	0.859
Netherlands	158	158	0.703	0.802	0.817

Table 9: Summary of Error Analysis based on Sweden, Germany and Netherlands.

It is easy to see that the overall return rate of customers from Sweden is low at 0.607, much lower than Germany's 0.798 and the Netherlands' 0.703. The low return rate results in a much lower test F1 score than their training F1 score, i.e. the overfitting is much worse during training than in Germany and the Netherlands.

In conclusion, the underlying reason for the GNN model's poor performance in predicting Swedish customers may be due to the lack of examples of return instances for customers from Sweden. Therefore, oversampling for Swedish customers may be an effective way to address the poor performance of improved GNN model in this country.

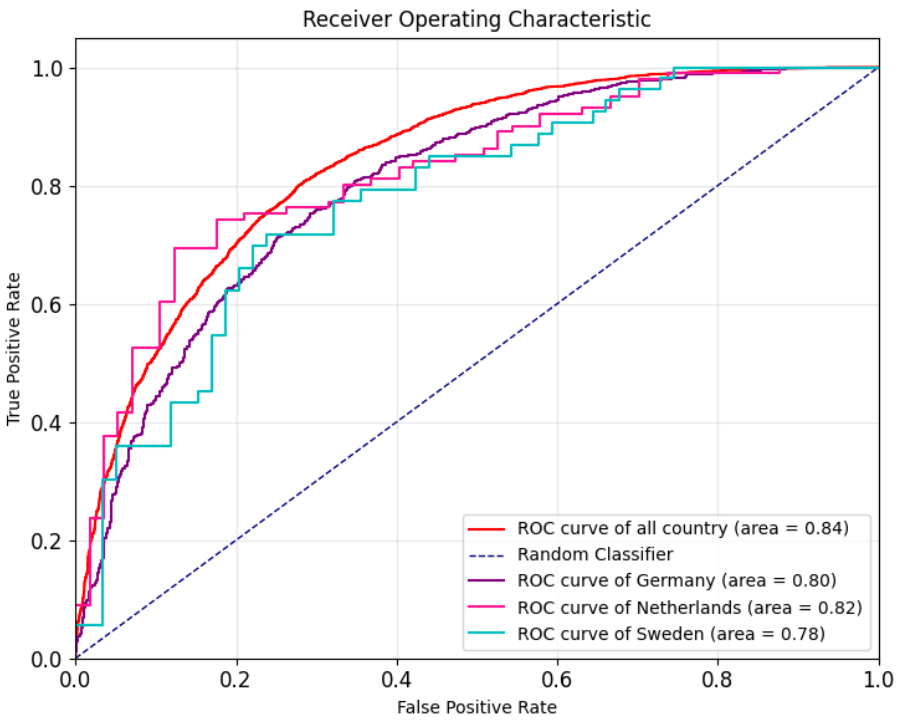

Figure 11: Schematic comparison of ROC curves for all countries, Sweden, Germany and the Netherlands.

7.2 Future prospects

In this phase, improvements to the structure-based, embedding aggregation and training approach of the simple GNN model in phase 1 have been implemented by introducing skip-connection, max-pooling aggregation and mini-batch training. Based on the results in Table.4, it has been demonstrated that the optimised model achieves good performance in prediction tasks in the fashion domain. However, this improved GNN model also has a number of areas for improvement.

First, regarding the embedding aggregation method, this model uses a GraphSAGE layer based on maximum pooling for the computation of node embeddings, in order to preserve the differentiation of node embeddings as much as possible. However, the graph structure shown in Fig. 2 used in this paper has an uneven number of connections of different kinds of nodes; for example, a country node may have more than 100,000 customer node connections, while a cold variant node may have only two or three connections. This uneven distribution of connections in density makes the importance of different neighbouring nodes for the embedding calculation not uniform. Therefore, introducing a graph attention mechanism to adjust the importance of different types of nodes by calculating the weight coefficients of different neighbouring nodes during embedding generation will help nodes to aggregate more important neighbouring embeddings into their own node embeddings.

Secondly, regularisation is important to solve the overfitting problem during model training. As

the dataset was cleaned and enhanced, most of products without too low or too high average return rates and non-regularised data were removed, and this operation resulted in a significant reduction in the training cross-entropy loss, while the validation and test F1 scores were not significantly improved, i.e. overfitting occurred. This would be mitigated by introducing regularisation to unify the scale of each shared weight in the forward calculation of the gradient.

Furthermore, although the improved GNN model achieved better results in the individual forecasting tasks for most countries and brands, for some countries and brands, such as Sweden, the improved GNN model instead performed worse. Therefore, in the future, the graph structure performance can be further improved by performing projection operations (i.e. linear transformation before aggregation), or by adding virtual nodes such as gender and age, or by trying other activation functions than ReLU, thus improving the embedding discrimination of this group of nodes. Additionally, by adding add-on edge features, it is possible to provide more specific context about customer returns and purchases. The above operations are very meaningful to awaken the graph neural network to the sensitivity of population nodes with low average return rate.

Finally, due to the limitations of the training system hardware resources and project time, only GNNs containing 4 layers of GraphSAGE were used for training. The possibility of varying the model size and complexity for improving the prediction performance is to be further explored in the future.

8 Conclusion

In this phase, an optimised GNN model was built to obtain better predictions of whether ASOS customers would return the product by introducing optimisation strategies such as skip connections, maximum pooling and mini-batch training. Comparing the optimal baseline model XGBoost with the simple GNN model of phase 1, the optimal overall prediction F1 score of the optimised GNN has reached 0.793, which has been improved substantially. The optimal cross-entropy loss has also been reduced from 0.4998 to 0.4897, demonstrating that the improved GNN has a better confidence level and can help make more robust decisions.

In addition, the performance of the improved GNN was improved for individual countries and markets. Specifically, the improved GNN model performs much better than the baseline model and the original GNN model in predicting some countries (8/9) and brands (7/10), which will improve the confidence level of ASOS in predicting whether its customers return the product or not. It can be expected that in A/B testing with real customers, the GNN model will provide more trustworthy predictions than the baseline model and further provide a more commercially valuable reference for ASOS to develop targeted marketing strategies.

Due to the limitation of computing hardware resources and project time, there is room for further optimisation of this GNN in terms of aggregation method, structural complexity and graph structure expressiveness. In the future, by introducing graph attention mechanisms, regularisation, increasing the depth of the model or adding more kinds of virtual nodes, there is room for further exploration to optimise the distribution of node weights and to awaken the sensitivity of the model to the embedding of less differentiated nodes.

Appendix

A The metrics

The following equations define the metrics used to evaluate the models,

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{4}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{5}$$

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{6}$$

where TP is the true positive counts, FP is the false positive counts and FN is the false negative counts.

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{i=1}^{N} y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \tag{7}$$

where N is the total number of predictions, y_i is the true class label (i.e. 0 or 1 for binary classification) of instance i and p_i is the predicted probability for the observation of instance i.

B Model Details

C Additional Figures

Model	Model Description
Logistic Regression	$C = 5.0$, tol. $= 10^{-4}$
MLP	2-layers hidden dim. $= 64$
Random Forest	# of estimators $= 100$, max. depth $= 6$, min. samples split $= 2$, min. samples leaf $= 1$, max. leaf nodes $= 10$
XGBoost	booster = gbtree, max. depth $= 4$, $\eta = 0.1$, $\gamma = 1$, min. child weight $= 1$, $\lambda = 2$, objective = Binary Logistic, early stopping rounds $= 5$

Table 10: Model descriptions for each of the baseline models considered in Sect. 4. Any parameters not listed for each of the models are set to there default settings in their corresponding packages.

Model A	1 Output SAGEConv layer (output dim. = 16)
Model B	1 SAGEConv layers (hidden dim. = 16) 1 Output SAGEConv layer (output dim. = 16)
Model C	2 SAGEConv layers (hidden dim. = 16) 1 Output SAGEConv layer (output dim. = 16)

Table 11: Experimental configuration parameters for the original GNN model

Model A	1 Output SAGEConv layer (output dim. = 64)
Model B	1 SAGEConv layers (hidden dim. = 64) 1 Output SAGEConv layer (output dim. = 64)
Model C	2 SAGEConv layers (hidden dim. = 64) 1 Output SAGEConv layer (output dim. = 64)
Model D	3 SAGEConv layers (hidden dim. = 64) 1 Output SAGEConv layer (output dim. = 64)

Table 12: Experimental configuration parameters for the improved GNN model

Hyper parameter	Value
Normalize layers	True
Remove Outliers	True
Low and high return of customers	0.3-0.7
Low and high return of products	0.3-0.7
Optimizer	Adam
Learning rate	0.01
Weight decay	0.0
Aggregate function	Max
Project	False
Activation function	ReLU
Dropout	0.2
Batch size	128
Training Epoch (with early stopping)	200

Table 13: Model description for the GNN models considered. Any parameters not listed are set to there default settings in their corresponding packages.

Product	Customer
Product ID	Customer ID
Variant ID	Year of Birth
Brand	Gender
Price	Country
Discount Value	Premier
Sales per Product	Sales per Customer
Returns per Product	Returns per Customer
Product Return Rate	Customer Return Rate
Return Reason Ratio: Looks different to image on site	Return Reason Ratio: Looks different to image on site
Return Reason Ratio: More than one size ordered	Return Reason Ratio: More than one size ordered
Return Reason Ratio: Late Delivery	Return Reason Ratio: Late Delivery
Return Reason Ratio: Poor Quality	Return Reason Ratio: Poor Quality
Return Reason Ratio: Doesn't fit	Return Reason Ratio: Doesn't fit
Return Reason Ratio: Doesn't suit	Return Reason Ratio: Doesn't suit
Return Reason Ratio: Incorrect item	Return Reason Ratio: Incorrect item
Return Reason Ratio: Parcel damaged	Return Reason Ratio: Parcel damaged
Return Reason Ratio: Faulty/Broken	Return Reason Ratio: Faulty/Broken
Return Reason Ratio: Too big/long	Return Reason Ratio: Too big/long
Return Reason Ratio: Too small/short	Return Reason Ratio: Too small/short
Return Reason Ratio: Changed my mind	Return Reason Ratio: Changed my mind
Return Reason Ratio: Missing item from multipack	Return Reason: Missing item from multipack
Product Type	

Table 14: A complete list of features associated with both customer and product instances. Note that 'Premier' corresponds to ASOS's subscription service which offers additional benefits to participating customers.

References

[1] *ASOS*. URL: https://www.asos.com/.

[2] *ASOS PLC ORD 3.5P (ASC.L) Income Statement - Yahoo Finance*. URL: https://finance.yahoo.com/quote/ASC.L/financials/.

[3] L Breiman. 'Random Forests. Machine Learning 45, 5–32'. In: (2001). DOI: 10.1023/A:1010933404324.

[4] Deli Chen et al. 'Measuring and relieving the over-smoothing problem for graph neural networks from the topological view'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 3438–3445.

[5] Jie Chen et al. 'Universal Deep GNNs: Rethinking Residual Connection in GNNs from a Path Decomposition Perspective for Preventing the Over-smoothing'. In: *arXiv preprint arXiv:2205.15127* (2022).

[6] Tianlong Chen et al. 'Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[7] Tianqi Chen and Carlos Guestrin. 'XGBoost: A Scalable Tree Boosting System'. In: *CoRR* abs/1603.02754 (2016). arXiv: 1603.02754. URL: http://arxiv.org/abs/1603.02754.

[8] Meha Desai and Manan Shah. 'An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (MLP) and Convolutional neural network (CNN)'. In: *Clinical eHealth* 4 (2021), pp. 1–11.

[9] Yafen Dong et al. 'Recognition of imbalanced underwater acoustic datasets with exponentially weighted cross-entropy loss'. In: *Applied Acoustics* 174 (2021), p. 107740.

[10] Driss El Alaoui et al. 'Deep GraphSAGE-based recommendation system: jumping knowledge connections with ordinal aggregation network'. In: *Neural Computing and Applications* (2022), pp. 1–12.

[11] Pantelis Elinas and Edwin V Bonilla. 'Addressing Over-Smoothing in Graph Neural Networks via Deep Supervision'. In: *arXiv preprint arXiv:2202.12508* (2022).

[12] *Fashion with integrity*. 2022. URL: https://www.asosplc.com/fashion-with-integrity/.

[13] Kao Ge, Jian-Qiang Zhao and Yan-Yong Zhao. 'GR-GNN: Gated Recursion-Based Graph Neural Network Algorithm'. In: *Mathematics* 10.7 (2022), p. 1171.

[14] Amur Ghose et al. 'Generalizable Cross-Graph Embedding for GNN-based Congestion Prediction'. In: *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. IEEE. 2021, pp. 1–9.

[15] Sonia Esther González-Moreno, Jesús Manuel Palma-Ruiz and Luis Ever Caro-Lazos. 'Marketing Strategies for Esports'. In: *Esports and the Media*. Routledge, 2022, pp. 52–68.

[16] Parisa Hajibabaee et al. 'An empirical study of the graphsage and word2vec algorithms for graph multiclass classification'. In: *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2021, pp. 0515–0522.

[17] Mustafa Hajij et al. 'High skip networks: A higher order generalization of skip connections'. In: *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*. 2022.

[18] William L. Hamilton. 'Graph Representation Learning'. In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14.3 (), pp. 1–159.

[19] William L. Hamilton, Rex Ying and Jure Leskovec. 'Inductive Representation Learning on Large Graphs'. In: (2017). DOI: 10.48550/ARXIV.1706.02216. URL: https://arxiv.org/abs/1706.02216.

[20] Hiroki Kanezashi et al. 'Ethereum Fraud Detection with Heterogeneous Graph Neural Networks'. In: *arXiv preprint arXiv:2203.12363* (2022).

[21] Diederik P Kingma and Jimmy Ba. 'Adam: A method for stochastic optimization'. In: *arXiv preprint arXiv:1412.6980* (2014).

[22] Blerina Lika, Kostas Kolomvatsos and Stathes Hadjiefthymiades. 'Facing the cold start problem in recommender systems'. In: *Expert systems with applications* 41.4 (2014), pp. 2065–2073.

[23] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. 2013. URL: http://arxiv.org/abs/1301.3781.

[24] Federico Monti et al. 'Fake news detection on social media using geometric deep learning'. In: *arXiv preprint arXiv:1902.06673* (2019).

[25] Bryan Perozzi, Rami Al-Rfou and Steven Skiena. 'DeepWalk: Online Learning of Social Representations'. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '14. New York, New York, USA: Association for Computing Machinery, 2014, pp. 701–710. ISBN: 9781450329569. DOI: 10.1145/2623330.2623732. URL: https://doi.org/10.1145/2623330.2623732.

[26] Zhejiang Ran et al. 'Accelerate graph neural network training by reusing batch data on GPUs'. In: *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. IEEE. 2021, pp. 1–8.

[27] Alvaro Sanchez-Gonzalez et al. 'Learning to simulate complex physics with graph networks'. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 8459–8468.

[28] Artur M Schweidtmann et al. 'Physical Pooling Functions in Graph Neural Networks for Molecular Property Prediction'. In: *arXiv preprint arXiv:2207.13779* (2022).

[29] Sebastian Tillmanns and Manfred Krafft. 'Logistic regression and discriminant analysis'. In: *Handbook of market research*. Springer, 2021, pp. 329–367.

[30] Yidi Wu et al. 'Seastar: vertex-centric programming for graph neural networks'. In: *Proceedings of the Sixteenth European Conference on Computer Systems*. 2021, pp. 359–375.

[31] Lingxiao Zhao and Leman Akoglu. 'Pairnorm: Tackling oversmoothing in gnns'. In: *arXiv preprint arXiv:1909.12223* (2019).