Customer Order Behavior Classification via Convolutional Neural Networks in the Semiconductor Industry

Marco Ratusny, Maximilian Schiffer, and Hans Ehm

Abstract—**In the operational processes of demand planning and order management, it is crucial to understand customer order behavior to provide insights for supply chain management processes. Here, advances in the semiconductor industry have emerged through the extraction of important information from vast amounts of data. This new data and information availability paves the way for the development of improved methods to analyze and classify customer order behavior (COB). To this end, we develop a novel, sophisticated yet intuitive image-based representation for COBs using two-dimensional heat maps. This heat map representation contributes significantly to the development of a novel COB classification framework. In this framework, we utilize data enrichment via synthetical training samples to train a CNN model that performs the classification task. Integrating synthetically generated data into the training phase allows us to strengthen the inclusion of rare pattern variants that we identified during initial analysis. Moreover, we show how this framework is used in practice at Infineon. We finally use actual customer data to benchmark the performance of our framework and show that the baseline CNN approach outperforms all available state-of-the-art benchmark models. Additionally, our results highlight the benefit of synthetic data enrichment.**

Index Terms—**Image classification, pattern recognition, data processing, learning systems.**

I. Introduction

The SEMICONDUCTOR market is highly competitive and comprises customers from various industry sectors. To succeed in this market, semiconductor manufacturers (SMs) need to cope with high demand volatility in the whole market and short product lifecycles, see Table I. Compared to other industries, SMs face larger cycle times and at the same time daily order changes, which makes production planning particularly challenging; among others, because the supply chain is prone to the Bullwhip effect. Moreover, long manufacturing processes, spread across different facilities, lead to long lead times, which increase the necessity of a sufficiently long forecast period. Accordingly, a great deal of attention on designing and manufacturing innovative products is of highest importance for a SM to preserve a competitive business model.

TABLE I
Industry Comparison

	Semiconductor [1] [2]	Automotive [3] [4]	Steel [5]
Volatility	24 months	25 months	-
Cycle Time	10-15 weeks	4-6 weeks	10 days
Order Changes	daily	one time order	-

In this context, unpredictable demand remains a major obstacle that results mainly from the electronics market's volatility and rapid innovation cycles. For example, the global semiconductor market shrank by almost 40 percent in 2009, while it grew by over 40 percent in 2010 [6]. More explicitly, Ponsignon and Ehm [7] exemplify the volatility in the semiconductor industry more explicitly, comparing the fluctuation in GDP growth (representing market demand) of only 2-3% with the according change in semiconductor demand of -30% to 40%, an observation supported by the well-known bullwhip effect [8]. Due to these challenges, SMs need to adapt their operations to a rapidly changing environment, which requires their supply chains to be highly resilient and agile. To ensure resilience, SMs can use two different measures: SMs may secure operations by increasing their production flexibility or inventory. Here, both options remain resource intensive and are consequently not desirable. Alternatively, increasing a SM's capability to understand and accurately forecast customer order behavior (COB) remains an information-intensive but resource-inexpensive option to mitigate the Bullwhip effect and hence, constitutes a desirable option to ensure resilience.

In addition to the high volatility in the semiconductor industry, customers demand high-quality and on-time deliveries to avoid supply bottlenecks. In this context, customers often overestimate their demand in shortage situations, resulting in disrupted and inefficient supply allocation. Furthermore, lead times of more than a year urge SMs to predict demand accurately to enable optimal inventory management. Here, the SM needs to rely on each customer's forecast: underestimated demand leads to missing production capacities and raw materials, while overestimated demand leads to underutilized production capacities and increased capital binding cost.

Manuscript received 8 April 2022; revised 20 May 2022 and 10 June 2022; accepted 23 June 2022. Date of publication 29 June 2022; date of current version 4 August 2022. *(Corresponding author: Marco Ratusny.)*

Marco Ratusny and Hans Ehm are with the Supply Chain Innovation, Infineon Technologies AG, 85579 Neubiberg, Germany (e-mail: marco.ratusny@infineon.com; hans.ehm@infineon.com).

Maximilian Schiffer is with the School of Management and the Munich Data Science Institute, Technical University of Munich, 80333 Munich, Germany (e-mail: schiffer@tum.de).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSM.2022.3187100.

Digital Object Identifier 10.1109/TSM.2022.3187100

To overcome shortage situations or excessive stock levels, SMs have agreed with their customers to exchange forecasted demand information long before the actual delivery. However, customers may change their forecasts, respectively orders, on short notice. Accordingly, such changes may still result in capacity bottlenecks since SMs rely on the early forecast information for midterm inventory and capacity planning. Therefore, it is necessary to understand whether the early communicated customer demand tends to be stable or might be subject to change before the actual delivery, in order to anticipate and counteract potential bottlenecks as early as possible. Accordingly, an accurate understanding of COBs is crucial for improving product allocation and better accounting for future customer demand.

Against this background, we develop a general framework to identify and classify various COBs by transforming customer order data into a novel two-imensional (2-D) heat map representation, which allows to efficiently utilize a convolutional neural network (CNN) architecture to perform the respective classification task.

A. Related Work

Our work relates to COB and customer ranking, demand forecasting, and the application of image recognition in the semiconductor industry. In the following, we review related works in these fields concisely.

1) Customer Classification: Customer classification and ranking are widely used in customer relationship management, e.g., credit scoring. Wang *et al.* [9] developed a method to differentiate good creditors from bad creditors. Their approach expects to have a better generalization ability while preserving insensitivity to outliers. Wang *et al.* [10] analyzed the order behavior in a mobile shopping environment and showed that customers increase their order rate by adopting to mobile shopping. Chicco *et al.* [11] developed a clustering approach for electricity customer classification in order to provide specific tariff offers. They found an indicative number of customer classes between 15 to 20 fit the supplier's needs. There are many more works on customer classification and we refer the interested reader to [12], [13], and [14] for a profound overview.

In this work, we arrange customers into specific groups, similar to the approaches of Wang *et al.* [10] and Chicco *et al.* [11]. However, our work remains the first that lifts this approach to the analysis of a multi-week forecasting period instead of focusing on a single point in time.

2) Demand Forecasting: In recent years, machine learning and deep learning have been successfully applied in many areas, e.g., pattern recognition and time series forecasting. Abbasimehr *et al.* [15] proposed a forecasting algorithm based on multi-layer Long Short-Term Memory (LSTM) networks that performs well for predicting highly fluctuating demand data. Geng *et al.* [16] presented a spatiotemporal multi-graph convolutional network for ride-hailing demand forecasting, applied to regional-level demand forecasting. Kilimci *et al.* [17] introduced a demand forecasting system by combining time series analysis techniques, support vector regression algorithm, and deep learning models in order to predict the demand for a retail company. For a detailed overview, we refer to the overview of [18] and [19].

While our approach also bases on demand time series, it differs from the works mentioned above as it focuses on time series classification instead of prediction.

3) Image Recognition in the Semiconductor Industry: In the semiconductor industry, the detection and classification of wafer map defects, focusing on the application of CNNs, has been a popular research topic due to its high potential to prevent yield loss. Nakazawa and Kulkarni [20] used artificially created wafer maps to train a CNN for a pattern classification task. Tello *et al.* [21] and Santos *et al.* [22] used CNNs to improve classification on wafers with multiple defect patterns.

CNNs are widely used in the area of wafer defect detection and classification in the semiconductor industry. However, the approaches of Nakazawa and Kulkarni [20], Tello *et al.* [21], and Santos *et al.* [22] have a technical focus and have not been applied to analyze COBs early in the supply chain. We also use CNNs to take advantage of the general insights from the research of Nakazawa and Kulkarni [20], Tello *et al.* [21], and Santos *et al.* [22].

Concluding, the approaches of [9], [10], [11], [15], [16], [17], [20], and [22] are suitable for the studied use cases. However, these works focused so far on classification for the end customer market, demand forecasting, or fault and defect pattern recognition. Identifying COB during the forecasting period of a customer remains an open research question. With our study, we answer this question by combining and enhancing approaches from the works mentioned above to visualize and classify COBs.

B. Contributions

With this work, we introduce a general methodological framework to analyze COB in the semiconductor supply chain. More specifically, our contribution is fourfold. First, we develop a novel general COB classification framework. This framework leverages a novel alternative data representation in the form of heat maps and utilizes data enrichment via synthetical training samples to train a CNN model that performs the final classification task. Second, besides providing a state-of-the-art classification framework, the heat map representation contributes to a novel visualization format, which allows to intuitively comprehend a customer's forecasting behavior in practice. Third, we show how this framework is embedded at Infineon Technologies AG's supply chain ecosystem. Fourth, we provide a numerical study, showing that our framework outperforms all available state-of-the-art benchmark models. Moreover, our numerical study shows the benefit of integrating Synthetic Data (SD) and Actual Customer Data (ACD) in the training process of the utilized CNN architecture.

C. Organization

The remainder of this paper is as follows. Section II contextualizes our classification framework with respect to its embedding in Infineon's supply chain management before

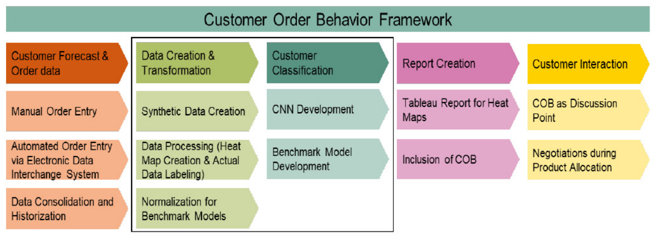
Customer Order Behavior Framework
Customer Forecast & Order data
Manual Order Entry
Automated Order Entry via Electronic Data Interchange System
Data Consolidation and Historization
Data Creation & Transformation
Synthetic Data Creation
Data Processing (Heat Map Creation & Actual Data Labeling)
Normalization for Benchmark Models
Customer Classification
CNN Development
Benchmark Model Development
Report Creation
Tableau Report for Heat Maps
Inclusion of COB
Customer Interaction
COB as Discussion Point
Negotiations during Product Allocation

Fig. 1. Framework Design.

Section III details our classification framework. Section IV details our computational experiments and Section V discusses our results. Section VI concludes this paper by summarizing its main findings and giving an outlook on future research.

II. Application in the Semiconductor Industry

We developed the presented framework in collaboration with Infineon, as it is of interest for Infineon to deeply understand the behavior of their customers during a 26-week lasting forecasting period. This forecast period is characteristic for SMs due to longer cycle times. It allows to align production and customer demand without requiring excessively high inventory levels. However, this longer forecast period may lead to more COB variants compared to other industries. Accordingly, a proper classification of the different COBs is necessary to support COB forecasting and operational planning. Infineon has agreed with its customers to start forecasting their demand at least 26 weeks before delivery. Currently, Infineon treats all customers' forecasts equally, not analyzing whether the customer's initially requested demand matches the ultimately ordered quantities. Due to that, it is necessary to determine how customers' forecasts evolve over the time horizon of 26 weeks before actual delivery. Our motivation is to increase data transparency by analyzing the COB of individual customers in more depth. This highly supports Infineon's customer logistics manager (CLM) department and empowers them to react appropriately to each customer based on the gained understanding of the customer through the COB classification.

Figure 1 details the embedding of our COB classification framework within the order management landscape of Infineon. We extract the customer forecast and order data from Infineon's SAP system, comprising customer, product, quantity, and time-related information. The time-related information is provided weekly and considers the last forecasts at the end of each week. Infineon uses an Electronic Data Interchange (EDI) system and manual orders via e-mail to obtain customers' forecasts or actual orders. This EDI system enables the customers to make changes to their orders on a daily level. However, we monitor differences that result from these changes on a weekly basis to reduce complexity and decrease the amount of data. Our framework's workflow includes two steps: *Data Creation & Transformation* and *Customer Classification*. Within the first step, we focus on the creation of SD, data processing, and data normalization. The data processing and normalization finally allow for our Customer Classification in the second step, based on a CNN architecture.

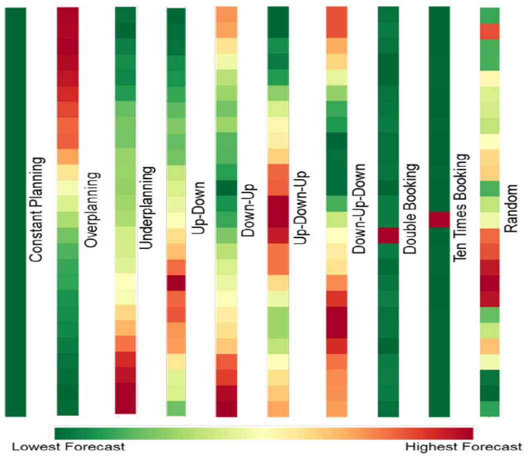

Fig. 2. COB heat map representations for one delivery week.

III. Methodology

In the following, we detail our COB classification framework that covers the planning tasks outlined in Section II and comprises four stages. The first stage focuses on data processing, particularly on transforming raw data into a heat map representation. The second stage utilizes the transformed data to characterize specific COBs, needed for the ACD labeling. The third stage focuses on data enrichment, explicitly leveraging the information gained from the second stage to complement the available data with SD to obtain a training data set that is more favorable for a machine learning application. Lastly, the fourth step leverages a CNN architecture to perform the COB classification task for unseen data.

A. Data Processing

In a first step, we transform raw customer data into heat maps (cf. [23]). Here, we use two characteristics to characterize the heat maps. The *forecasting horizon* describes the difference between the delivery target date and the date when the forecast was created. The *delivery week* is the target week at which the customer wants to receive the products. With these definitions, we characterize a forecast as the amount each customer forecasts for a specific delivery week in a particular forecasting horizon.

Based on this information, we create a heat map that represents the respective forecast transformed into a color gradient for each delivery week (x-axis) and forecasting horizon (y-axis). Figure 2 shows examples of such heat maps.

Converting raw data into a heat map requires several arithmetic steps. Algorithm 1 formalizes the heat map transformation: first, we normalize the data in the range of 0 to 1 (l.1 to 5). Specifically, we define the maximum, minimum, and median forecast value per Delivery Week (DW). Then we calculate *vrange* per delivery week, i.e., the maximum of the difference between the maximum and the median forecast value per DW and the difference between the median and

Algorithm 1 Heat Map Formalization

```
 1: $\text{Fmax}_{DW} = \max(\text{Forecast}_{DW})$
 2: $\text{Fmin}_{DW} = \min(\text{Forecast}_{DW})$
 3: $\text{center}_{DW} = \text{median}(\text{Forecast}_{DW})$
 4: $vrange_{DW} = max(Fmax_{DW} - center_{DW}, center_{DW} - Fmin_{DW})$
 5: $n\_forecast = \frac{(forecast_{DW,Diff} - (center_{DW} - vrange_{DW}))}{|(center_{DW} - vrange_{DW}) - (center_{DW} + vrange_{DW})|}$
 6: $cmax = \frac{(Fmax_{DW} - (center_{DW} - vrange_{DW}))}{|(center_{DW} - vrange_{DW}) - (center_{DW} + vrange_{DW})|}$
 7: $cmin = \frac{(Fmin_{DW} - (center_{DW} - vrange_{DW}))}{|(center_{DW} - vrange_{DW}) - (center_{DW} + vrange_{DW})|}$
 8: $q = \frac{(cmax_{DW} - cmin_{DW})}{256}$
 9: $Bin_{DW} = [q * 1, q * 2], [q * 2, q * 3], ..., [q * 255, q * 256]$
10: $n\_forecast \in Bin_{DW}$
```

the minimum forecast value per DW (l.4). Using these values, we normalize each forecast value per DW. Second, we create 256 bins to map each normalized forecast value to the corresponding bin by defining the range of each bin by *cmin* and *cmax* (l.6 to 10). Note that we use binning as a state-of-the-art appraoch to reduce the model architecture's complexity at the price of accepting a (limited) information loss. In fact, the information loss remains negligible in our case, because small demand changes rarely lead to a change of the COB classification. Reference [24] Finally, we map each bin to the corresponding color gradient - consisting of 256 colors - to receive the final heat map representation (l. 10).

B. Characterization of Customer Order Behaviors

We analyze the input data by classifying the available data into different patterns. The purpose of this analysis is twofold: first, associating each heat map with a specific class is necessary to enable the subsequent supervised learning task. Second, understanding the distribution of different COB patterns in the actual data eases the creation of meaningful SD.

Identifying appropriate patterns often requires involving an expert with domain knowledge. In our case, we included representatives from Infineon's CLMs department. We analyzed patterns for each delivery week, i.e., we analyzed each column of a heat map separately. Classifying data on this level of granularity allows us to observe changes in a customer's order behavior weekly, which equals the time horizon at which operational adjustments are made in practice.

Based on interviews with the CLMs, we identified ten different COB patterns: Overplanning (OP), Underplanning (UP), Up-Down (UD), Down-Up (DU), Up-Down-Up (UDU), Down-Up-Down (DUD), Double Booking (DB), Ten Times Booking (TTB), Constant (CO), and Random (R).

CO denotes a scenario in which a customer does not change its forecast values at all. DB implies that the customer accidentally forecasts the same quantities twice for the combination of a delivery week and a forecasting horizon. TTB usually implies that the customer accidentally forecasts ten times the same quantities for the combination of one delivery week and one forecasting horizon. TTB can also be seen as a constant behavior with one outlier since it results in the same heat map pattern. Together, DB and TTB represent the group of error-prone behaviors. OP indicates that the customer decreases its forecasts over time, whereas UP oppositely denotes an over time increasing forecasts. The behaviors DU, DUD, UD, and UDU, are combinations of OP and UP. In DU, an OP pattern is superposed by an UP pattern, while DUD comprises another subsequent OP pattern. UD contains the DU superposition in reversed order: an UP pattern is superposed by an OP pattern; analogously to DUD, UDU is a DU pattern, followed by an additional UP pattern. Class R implies that no pattern is directly observable: the customer orders in a way that does not fit any of the identified patterns. In practice, these COB classifications may trigger varying strategies to deal with specific customer types. CO eases the planning process and therefore, customers can be incentivized by receive a higher importance. This means that customers will prioritized with products in time of shortage situations. Error-prone behaviors (DB; TTB) allow CLMs to interact with the respective customers, to understand the reason of the (one-time) error to derive a strategy that mitigates future failures. OP and UP indicate that customers purposely follow a certain strategy, e.g., always ordering too much at the beginning of the forecasting horizon. In such cases, CLMs need to align with the customers to plan better and already provide reliable forecasts at the beginning of the forecasting period. The volatile behaviors (DU; DUD; UD; UDU) indicate that the customer provides a tactical demand, in order to receive the wished quantities in the end. The R pattern does not allow for a specific strategy in practice but ensures an exhaustive assignment of COB patterns to categories from a technical perspective.

C. Data Enrichment

1) Synthetic Data Creation: SD is artificially created data based upon actual events, which is often created purposely to replicate real-world scenarios. The creation of SD is popularly used in Artificial Intelligence (AI)/Machine Learning (ML) model training when real-world data is difficult to access or expensive to clean and label. Since CNNs are known to perform better when being trained on an extensive training data set, we generate SD, which we use to enrich our training data set. In this context, SD offers the possibility to imitate the patterns identified in Section III-B with greater flexibility. Moreover, considering the heterogeneous occurrence of these specific patterns in the actual data, we overcome another challenge by using synthetically generated data. Nakazawa and Kulkarni [20] states that synthetically generated data can be favorable to cover scarcely occurring patterns while training the model to obtain a better classification accuracy, particularly on instances that would otherwise remain invisible to the model.

Particularly, using only ACD often leads to heterogeneous and scarce data, since not all patterns occur with the same frequency in the ACD. Moreover, SD allow to distribute the number of samples per COB more uniformly. Furthermore, if only ACD is used to train the CNN, overfitting could occur since the algorithm does not appropriately learn all possible scenarios for the given COB pattern. Finally, unseen variants of SD can be created based on ACD. For this reason, the use of SD eases a faster upscaling of the sample size per COB.

TABLE II
CNN ARCHITECTURE

Layer	Kernel	Channels	Activation	Input	Output	Parameters
Input		3	ReLU	754x27	754x27	
conv1	7x7	32		754x27	377x13	4,736
pool1	2x2		ReLU	377x13	373x9	
conv1	5x5	64		373x9	186x4	51,264
pool1	2x2		ReLU	186x4	184x2	
conv1	3x3	128		184x2	92x1	73,856
pool1	2x2			92x1	92x1	
dropout				92x1	92x1	
flatten1				92x1	11776	
dense1			ReLU	11776	128	1,507,456
dropout				128	128	
dense2			Softmax	128	10	1,290
Total						1,638,602

To create SD, we proceed as follows: First, we generate raw data representing the exact theoretical behavior of the COB. It is only important that the created forecast values represent the behavior after the heat maps have been created. Note that we can generate this data without accounting for varying demand amplitudes due to the preceding normalization of the heat maps. Second, after creating clean heat maps, we include noise in the data, which is necessary to obtain training data that better represents the real COB, leading to a better CNN performance on actual data. We include noise by randomly perturbing forecast and order value with a factor between 0.9 and 1.1.

2) Usage of Actual Data: The usage of ACD in this paper is two-fold. First, we use parts of the labeled ACD set to include that into the training set and the synthetically created data. The training set of the labeled actual data is further used with the synthetically produced data to train the model. Furthermore, we use flipping as an augmentation technique. Flipping the data leads, in some cases, to different behaviors. e.g., OP becomes UP. We use another augmentation technique to slightly manipulate the forecast and order values, which changes the image's appearance while keeping the same pattern. Also, we flipped those manipulated data to receive even more data, which looks similar to actual customer data. Second, we use real customer data as the validation set to accurately interpret the developed models' results. No augmentation technique is used for the validation set, as we validate our approaches on ACD.

D. Convolutional Neural Network Architecture

For our COB classification, we apply a CNN architecture which we designed in previous work and refer to [23] for technical details. Our inputs, the heat maps, have a height of 754 pixels, a width of 27 pixels, and are RGB encoded. In a heat map, each square consists of 27 pixels in width and 29 pixels in height. Accordingly, as a heat map comprises 26 forecasts, the resulting image consists of 754x27 pixels. Table II details the baseline CNN's architecture, which comprises 12 layers. We use a 'ReLU' activation function for each convolutional layer, apply a 'ReLU' activation function before the last dropout, and utilize a 'Softmax' function in the last layer for the probability calculation of membership for each class.

IV. EXPERIMENTAL DESIGN

For training purposes, we use Infineon's compute farm, which is based on a CentOS 7.7.1908 operation system. Since we are focusing on an image classification task, GPUs are necessary to decrease the computational burden. Therefore, we use one Tesla P40 with 24GB of GDDR5 to train the neural network.

Our experiments base on a data set with 10,288 real customer heat maps, which we split into a training and a test set. We keep 1,050 heat maps for test purposes such that the training set contains 9,238 real heat maps. We then enrich the training set with 29,742 synthetically created heat maps (cf. Section III-C), such that the final training set contains 38,980 samples. With this data set, our experiments are twofold.

First, we benchmark the performance of our algorithm against 13 state-of-the-art approaches: a Fully Connected Network (FCN) and a Multilayer Perceptron (MLP) architecture [25], a ResNet-11 architecture with 11 layers and a time series input [26] and a ResNet-18 architecture with a heatmap input [27], an Inception model [28], an encoder network [29], a TLENET architecture [30], an MCDCNN network [31], an LSTM [32], as well as against a LSTM_FCN, a A_LSTM_FCN, and a M_LSTM_FCN architecture based on the work by [33]. In our experiments, we use the architectures and the training configurations as described in the respective papers. We transform the heat maps into a discrete multivariate time series to use our data set for these models. We resize the input size of 754x27x3 (height, width, channels) to 26x1x3, effectively sizing down the image to 78 values. This resizing transforms the image data into a multivariate time series of size (26, 3) (height, channels), which we can use as input for the various benchmark models. To train the baseline CNN, we use a 'Categorical Crossentropy' loss function and apply a 'rmsprop' optimization function with a learning rate of 0.001, and use a batch size of 64.

Second, we analyze the benefit of enriching the training set with SD. To do so, we analyze 44 different treatments. First, we compare the accuracy of the baseline CNN when being trained solely with ACD or SD or with a combination of both to isolate the impact of SD enrichment. Second, we conduct this comparison for various shares of ACD related to the overall amount of training data. The distribution of ACD to SD for the three groups is 5% to 95%, 10% to 90%, and 15% to 85%. Third, we repeat this study for training datasets of varying size, ranging from 2,500 to 60,000 samples, obtaining the upper limit of 60,000 samples based on the ACD shares analyzed and the maximum number of about 10,000 ACD training samples.

V. RESULTS

In the following, we discuss our results. First, we compare state-of-the-art approaches and provide detailed analysis for each behavior in Section V-A. Then, we analyze the benefits of enriching the training data by synthetic samples in Section V-B. Finally, we utilize SHapley Additive exPlanations (SHAP) in Section V-C to analyze the importance of specific heat map features for the classification task.

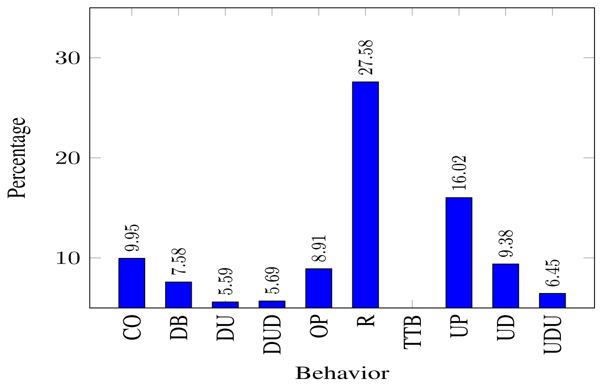

Fig. 3. Distribution of COBs.

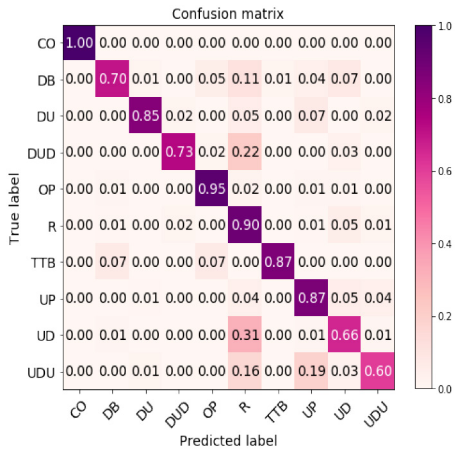

Fig. 4. Confusion Matrix.

TABLE III
RESULTS OF THE DEVELOPED MODELS

Algorithm	Accuracy	Training Duration
Baseline CNN	83.98%	300s
MCDCNN	83.69%	451s
LSTM	83.03%	3469s
Inception	82.70%	559s
M_LSTM_FCN	82.46%	1788s
LSTM_FCN	82.37%	1786s
Encoder	82.18%	2189s
ResNet-11	81.89%	2578s
FCN	81.80%	470s
A_LSTM_FCN	81.61%	1165s
VGG-11	80.38%	1747s
ResNet-18	79.05%	1502s
MLP	77.16%	780s
TLENET	73.65%	156s

A. *Classification Performance*

Figure 3 shows the distribution of the different COB patterns in the test set. As can be seen, the patterns in the test set are imbalanced as some COBs are underrepresented compared to others. Accordingly, the frequency of the most-occurring pattern (R) remains a naive benchmark for the classification accuracy that could be reached by naively classifying each heat map as R.

Table III displays the results for the baseline CNN and the benchmark models. As can be seen, all algorithms outperform the naive benchmark significantly, which gives proof to the quality of our algorithmic design choices. Moreover, the baseline CNN outperforms all other models and achieves an accuracy of 83.98% on the test set. Regarding the training duration, the baseline CNN is the second fastest model with a duration of 300 seconds for the training. Compared to the MCDCNN, the second-best performing algorithm with respect to accuracy, the CNN has a 120 seconds faster training time.

To analyze the classification accuracy of our framework for each predicted label, Figure 4 shows the corresponding confusion matrix, which compares the true label in each row with the labels classified by our algorithm in each column. As we can see, CO is the only class for which we reach a 100% classification accuracy. Moreover, OP and R show high classification accuracies of at least 90%, while the classification accuracy for DU, TTB and UP remains below 90% but above 85%. For DB and DUD, we reach a classification accuracy of at least 70%, while our algorithm performs worst for UD and UDU, yielding a classification accuracy below 70% but above 60%. All of these accuracies remain significantly above the naive benchmark.

To put these fragmented classification accuracies into context, some comments are in order. First, UP and R are the most occurring patterns over all customers such that having a high classification accuracy for those patterns is of high importance. Here, we achieve classification accuracies above 90% respectively 85%, which is above the average accuracy of our and all other state-of-the-art approaches for these COBs. Second, mixing up the classification of DU, DUD, UD and UDU does not impact Infineon's operations as these COBs are treated equally during negotiations with customers as volatile COBs. While the classification accuracy for DU is 85%, for DUD it is 73%, for UD it is 66% and for UDU it is 60%, the overall classification accuracy for this group of COB is 73%. Merging those patterns into one group mitigates some false positives and improves the overall classification accuracy by one percentage point to 84.9% compared to the classification of each individual COB. Third, obtaining a perfect classification accuracy for CO appears to be natural as CO patterns always look the same and are thus implicitly easy to detect. Fourth, DU, DUD, UD and UDU are sometimes classified as R although they exhibit a clearly distinctive pattern. Here, further algorithmic improvement to avoid unnecessary confusion remains a future research perspective.

B. *Comparison Between Synthetic and Actual Customer Data*

In the following, we analyze the impact of enriching the training data set with SD. To this end, we study the baseline CNN's accuracy for varying training set cardinalities when being trained solely with ACD, only with SD, or with a combination of both as outlined in Section IV.

Table IV shows (from left to right) the cardinality of the training set, the CNN's accuracy on the test set after being trained with either synthetic data only (SD), or Mixed Data (MD) consisting of SD and up to 5% (MD5), 10% (MD10), or 15% (MD15) of ACD, or being trained on

TABLE IV
INFLUENCE OF UP TO 15% ACD PER TRAINING SET

Num Training Samples	Acc. (SD)	Acc. (MD5)	Acc. (MD10)	Acc. (MD15)	Acc. ACD
2,500	64,00 %	65,40 %	65,53 %	68,06 %	59,14 %
5,000	67,40 %	69,10 %	70,33 %	70,33 %	63,31 %
7,500	68,53 %	72,80 %	73,93 %	72,90 %	63,51 %
10,000	69,40 %	73,74 %	74,31 %	74,88 %	65,30 %
15,000	71,60 %	75,92 %	75,23 %	75,60 %	
20,000	71,66 %	79,62 %	80,47 %	81,10 %	
30,000	73,80 %	81,40 %	80,86 %	81,62 %	
40,000	72,51 %	81,51 %	81,42 %	82,21 %	
50,000	71,66 %	81,04 %	81,33 %	81,36 %	
60,000	72,80 %	79,76 %	80,57 %	79,80 %	

ACD only. As can be seen, the CNNs trained on MD always perform at least 1.4 percentage points better than the CNNs trained on pure SD or ACD. In addition, the CNNs trained on pure SD achieve higher accuracies as the CNNs trained solely on ACD. We achieve the highest accuracy for a training set cardinality of 40,000 samples when using MD. For SD, we achieve the highest accuracy with a training set cardinality of 30,000. Moreover, we can observe that a higher amount of ACD leads to higher accuracy except in five cases. The difference in accuracy is marginal in these five cases, with less than one percentage point difference. Overall, the CNN trained on MD shows up to ten percentage points respectively nine percentage points better classification accuracy compared to its counterparts that have been trained solely on SD or ACD.

Apart from proving the superiority of using MD data to train our CNNs, our results allow for the following conclusions. First, we observe an accuracy increase up to 30,000 respectively 40,000 training samples and a slight decrease afterwards as the CNN starts to overfit. Accordingly, it is beneficial for the CNN to combine SD and ACD to obtain, first, a larger and more balanced data set and, second, to include more variants of ACD in the training set. Second, we observe that a CNN trained solely on SD performs better than a CNN trained solely on ACD. This suggests that a more balanced training set improves the training and consequently the CNN's classification accuracy on unseen data.

C. Classification Interpretation

To analyze which heat map features are important for the CNN classification, we use SHAP, which is a game theoretic approach to explain any machine learning model's output by connecting optimal credit allocation with local explanations. Reference [34] Figure 5 shows the SHAP explanations for three different heat maps (CO; UP; R). The first column displays unseen input images in the first column. Succeeding columns contain a grayscale image of the heat map, superposed by the visualization of the explanations. Here, pattern segments highlighted in red highlight a positive contribution to the SHAP values and accordingly to the activation when identifying a certain pattern. Consequently, a larger positive contribution in the column of the correct pattern points towards a more robust and reliable classification behavior. We observe the most positive activation for CO, indicating that the CNN

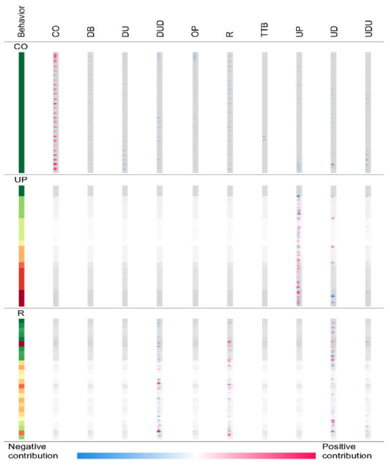

Fig. 5. Visualization of feature importance via shapley values.

classifies this pattern reliably. For the more complex behaviors - UP and R - predicting the correct class becomes harder, which is indicated by partly positive activation in classes to which the original heat map does not belong. Still, the CNN is able to classify these heat maps correctly, as we observe the highest positive activation related to the correct class.

VI. CONCLUSION & OUTLOOK

In this paper, we developed a general framework for COB classification. This framework combines an alternative data representation in the form of heat maps with data enrichment via synthetical training samples to generate a CNN architecture that performs the final COB classification. The used heat map transformation further represents an alternative possibility to intuitively visualize a customer's forecasting behavior in practice. Afterward, we demonstrated how we embedded our framework in the supply chain ecosystem of Infineon. We provided a numerical study, where we showed that the baseline CNN outperforms all available state-of-the-art benchmark models. Moreover, our numerical study showed the benefits of integrating SD and ACD in the training process of the CNN. In summary, our results illustrate the real-world relevance and emphasize the relevance of COB classification in the semiconductor industry. We note that although we have applied our approach to Infineon's ecosystem, it can easily be transferred to other companies.

In its operational process, Infineon can make use of the findings in three ways. First, they can use the heat maps of the COB as a means of visualization, resulting in increased information for the CLM since the heat maps provide an intuitive means of visualization. Second, a robust classification for the CO, DU, OP, R, TTB and UP is possible, which decreases the manual classification effort for the CLMs as those COB patterns occur most often. Lastly, an increased classification accuracy and interpretability of the results are reached through aggregating certain patterns as generally volatile behaviors, reducing the complexity for the CLMs. All that can be used to make the customers aware of their behavior, and possibly leads to a more stable behavior over time. The developed classification model supports the CLMs during discussions with customers to improve communication and helps to explain how a more stable behavior is more beneficial for demand planning and product allocation. Furthermore, the classification can be used to improve production planning, e.g., by reducing production volumes for products that are constantly subject to overplanning order behavior.

The successful development of this approach paves the way for future work. We plan to include contextual data into the model - effects, such as the product life cycle, can significantly impact the customer's forecasting behavior. COBs might change based on the status of the life cycle of a product such that integrating life cycle information remains another aspect for future work.

REFERENCES

[1] L. Mönch, R. Uzsoy, and J. W. Fowler, "A survey of semiconductor supply chain models part I: Semiconductor supply chains, strategic network design, and supply chain simulation," *Int. J. Prod. Res.*, vol. 56, no. 13, pp. 4524–4545, 2018.

[2] D. E. Sichel, S. D. Oliner, and A. M. Aizcorbe, "Shifting trends in semiconductor prices and the pace of technological progress," *SSRN Electron. J.*, pp. 1–44, Nov. 2006.

[3] H. Meyr, "Supply chain planning in the german automotive industry," *OR Spectr.*, vol. 26, no. 4, pp. 447–470, 2004.

[4] D. Sabadka, V. Molnár, and G. Fedorko, "Shortening of life cycle and complexity impact on the automotive industry," *TEM J.*, vol. 8, no. 4, pp. 1295–1301, 2019.

[5] S. M. Fahmi and T. M. Abdelwahab, "Case study: Improving production planning in steel industry in light of lean principles," in *Proc. Int. Conf. Ind. Eng. Oper. Manage.*, 2012, pp. 2489–2497.

[6] A. Seitz, H. Ehm, R. Akkerman, and S. Osman, "A robust supply chain planning framework for revenue management in the semiconductor industry," *J. Revenue Pricing Manage.*, vol. 15, no. 6, pp. 523–533, 2016.

[7] H. Ehm and T. Ponsignon, "Future research directions for mastering end-to-end semiconductor supply chains," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, 2012, pp. 641–645.

[8] H. Lee, V. Padmanabhan, and S. Whang, "Information distortion in a supply chain: The bullwhip effect," *Manage. Sci.*, vol. 50, no. 12, pp. 1875–1886, Apr. 1997.

[9] Y. Wang, S. Wang, and K. K. Lai, "A new fuzzy support vector machine to evaluate credit risk," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 6, pp. 820–831, Dec. 2005.

[10] R. J.-H. Wang, E. C. Malthouse, and L. Krishnamurthi, "On the go: How mobile shopping affects customer purchase behavior," *J. Retailing*, vol. 91, no. 2, pp. 217–234, 2015.

[11] G. Chicco, R. Napoli, and F. Piglione, "Comparisons among clustering techniques for electricity customer classification," *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 933–940, May 2006.

[12] L. Zuo and J. Guo, "Customer classification of discrete data concerning customer assets based on data mining," in *Proc. Int. Conf. Intell. Transp. Big Data Smart City (ICITBS)*, 2019, pp. 352–355.

[13] X. Li and C. Li, "The research on customer classification of B2C platform based on k-means algorithm," in *Proc. IEEE 3rd Adv. Inf. Technol. Electron. Autom. Control Conf. (IAEAC)*, 2018, pp. 1871–1874.

[14] S. Sa'adah and M. S. Pratiwi, "Classification of customer actions on digital money transactions on PaySim mobile money simulator using probabilistic neural network (PNN) algorithm," in *Proc. 3rd Int. Seminar Res. Inf. Technol. Intell. Syst. (ISRITI)*, 2020, pp. 677–681.

[15] H. Abbasimehr, M. Shabani, and M. Yousefi, "An optimized model using LSTM network for demand forecasting," *Comput. Ind. Eng.*, vol. 143, May 2020, Art. no. 106435.

[16] X. Geng *et al.*, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3656–3663.

[17] Z. H. Kilimci *et al.*, "An improved demand forecasting model using deep learning approach and proposed decision integration strategy for supply chain," *Complexity*, vol. 2019, pp. 1–15, Mar. 2019.

[18] X. Zhu, G. Zhang, and B. Sun, "A comprehensive literature review of the demand forecasting methods of emergency resources from the perspective of artificial intelligence," *Nat. Hazards*, vol. 97, no. 1, pp. 65–82, 2019.

[19] K. Benidis *et al.*, "Neural forecasting: Introduction and literature overview," 2020, *arXiv:2004.10240*.

[20] T. Nakazawa and D. V. Kulkarni, "Wafer map defect pattern classification and image retrieval using convolutional neural network," *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 2, pp. 309–314, May 2018.

[21] G. Tello, O. Al-Jarrah, P. Yoo, Y. Al-Hammadi, S. Muhaidat, and U.-H. Lee, "Deep-structured machine learning model for the recognition of mixed-defect patterns in semiconductor fabrication processes," *IEEE Trans. Semicond. Manuf.*, vol. 31, no. 2, pp. 315–322, May 2018.

[22] T. Santos *et al.*, "Feature extraction from analog wafermaps: A comparison of classical image processing and a deep generative model," *IEEE Trans. Semicond. Manuf.*, vol. 32, no. 2, pp. 190–198, May 2019.

[23] M. Ratusny, A. Ay, and T. Ponsignon, "Characterizing customer ordering Behaviors in semiconductor supply chains with convolutional neural networks," in *Proc. Winter Simulat. Conf. (WSC)*, 2020, pp. 1931–1942.

[24] K. Coussement, S. Lessmann, and G. Verstraeten, "A comparative analysis of data preparation algorithms for customer churn prediction: A case study in the telecommunication industry," *Decis. Support Syst.*, vol. 95, pp. 27–36, Mar. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923616302020

[25] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," 2016, *arXiv:1611.06455*.

[26] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: A review," *Data Min. Knowl. Disc.*, vol. 33, no. 4, pp. 917–963, 2019.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*.

[28] H. I. Fawaz *et al.*, "InceptionTime: Finding AlexNet for time series classification," *Data Min. Knowl. Disc.*, vol. 34, no. 6, pp. 1936–1962, 2020.

[29] J. Serrà, S. Pascual, and A. Karatzoglou, "Towards a universal neural network encoder for time series," 2018, *arXiv:1805.03908*.

[30] A. L. Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *Proc. ECML/PKDD Workshop Adv. Anal. Learn. Temporal Data*, 2016, pp. 1–8.

[31] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Time series classification using multi-channels deep convolutional neural networks," in *Web-Age Information Management*, F. Li, G. Li, S.-W. Hwang, B. Yao, and Z. Zhang, Eds. Cham, Switzerland: Springer Int., 2014, pp. 298–310.

[32] D. Smirnov and E. M. Nguifo, "Time series classification with recurrent neural networks," in *Proc. 3rd ECML/PKDD Workshop Adv. Anal. Learn. Temporal Data*, Sep. 2018, pp. 1–80.

[33] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate LSTM-FCNs for time series classification," *Neural Netw.*, vol. 116, pp. 237–245, Aug. 2019. [Online]. Available: https://doi.org/10.1016

[34] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Assoc., 2017.