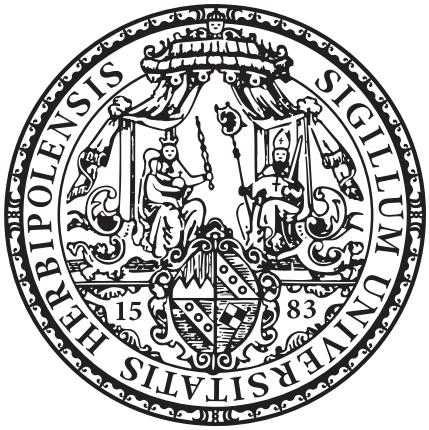
Julius-Maximilians-Universität Würzburg
Wirtschaftswissenschaftliche Fakultät



From Small to Large Data: Leveraging Synthetic Data for Inventory Management

Inauguraldissertation
zur Erlangung des akademischen Grades
doctor rerum politicarum (Dr. rer. pol.)

vorgelegt von
Simone Linda de Graaf geb. Buttler, M.Sc.
geboren in Gütersloh

Name und Anschrift:	Simone Linda de Graaf Kamper Str. 18 52064 Aachen
Erstgutachter:	Prof. Dr. Richard Pibernik
Zweitgutachter:	Prof. Dr. Christoph M. Flath
Datum der Einreichung:	24.05.2023

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my doctoral advisor, Prof. Dr. Richard Pibernik. Your relentless pursuit of excellence and your insightful questions and comments have significantly elevated the quality of my work. Your guidance has been invaluable, and I am deeply appreciative of the high standards you set, which have challenged me to push beyond my limits.

I would also like to thank my second advisor, Prof. Dr. Christoph Flath. Your support and constructive feedback have been crucial throughout this journey. Your expertise and encouragement have been greatly appreciated.

To my colleagues at the Chair of Logistics and Quantitative Methods in Business Administration and the Chair of Information Systems and Business Analytics, I highly appreciate the time we had together. Collaboration has not always been easy due to the COVID-19 pandemic, but the camaraderie and shared knowledge have been a source of inspiration and motivation. The collaborative environment we fostered together has been instrumental in the completion of this dissertation. I would like to express special thanks to Nikolai Stein and Andreas Philippi for the collaboration on our research papers and strong moral support. Christiane Kleespies for creating a loving and caring environment. Pascal Notz, Felix Schmidt, Toni Greif, Matthias Griebel, Kai Günder and Peter Wolf for all the fruitful discussions.

Finally, I would like to express my gratitude to my family and especially to my husband, whose endless patience and unwavering support have been my anchor through this process. Your encouragement and belief in me have kept me going even in the most challenging times. This dissertation would not have been possible without your love and support.

Thank you all for your contributions to this achievement.

Deutschsprachige Zusammenfassung (Summary in German Language)

In einer Welt des ständigen Wandels ist Unsicherheit zu einer alltäglichen Herausforderung für Unternehmen geworden. Die Covid-19-Pandemie hat deutlich gezeigt, wie schnell sich Marktumfelder verändern können und wie wichtig es ist, flexibel auf unvorhersehbare Ereignisse zu reagieren. In diesem komplexen Entscheidungsumfeld spielt das Operations Management (OM) eine entscheidende Rolle. Das Ziel des OM besteht darin, die Geschäftsprozesse von Unternehmen zu optimieren. Von der Standortplanung über die Produktionssteuerung bis hin zum Bestandsmanagement —OM befasst sich mit den strategischen und operativen Entscheidungen, die erforderlich sind, um den betrieblichen Erfolg sicherzustellen. Traditionell haben Unternehmen bei der Entscheidungsfindung theoretische Modelle aus Bereichen wie Mikroökonomie, Spieltheorie, Optimierung und Simulation genutzt (Mišić und Perakis, 2020). Doch angesichts der Fortschritte im Bereich des maschinellen Lernens und der mathematischen Optimierung der letzten Jahre eröffnete sich ein neues Forschungsgebiet: das datengetriebene OM.

Im datengetriebenen OM werden reale Daten, insbesondere Zeitreihendaten, herangezogen, um realistischere Modelle zu entwickeln, welche die Komplexität der Entscheidungsfindung besser erfassen können. Diese Daten können wertvolle Einblicke in vergangene Kundennachfrage und relevante Einflussfaktoren, wie Wetterbedingungen oder Börsentrends, liefern. Durch die Kombination von realen Daten mit Optimierungs- und maschinellen Lernverfahren können Unternehmen fundiertere und präzisere Entscheidungen treffen.

Jedoch besteht im Rahmen dieses neuen Forschungsgebiets eine Herausforderung: Studien haben gezeigt, dass einige Modelle optimale Lösungen liefern können, wenn diese eine umfangreiche Menge historischer Trainingsdaten zur Verfügung haben. Jedoch sieht die Realität häufig anders aus. Insbesondere in Teildisziplinen des OM, wie dem Kapazitäts- oder Bestandsmanagement, existiert häufig nur eine begrenzte Anzahl von historischen Beobachtungen, da Entscheidungen über den Einsatz von Ressourcen oder die Auffüllung des Bestands hier nicht in Echtzeit getroffen werden, sondern täglich, wöchentlich oder sogar nur monatlich stattfinden.

In anderen Anwendungsbereichen des maschinellen Lernens, in denen die Verfügbarkeit von Daten zum Trainieren von Modellen ebenfalls ein Problem darstellt, hat man damit begonnen, reale Daten durch synthetische Daten zu ergänzen oder sogar zu ersetzen. Synthetische Daten sind künstlich generierte Daten, die die Eigenschaften und Muster realer Daten nachahmen. Neuste Ansätze zur Generierung synthetischer Daten haben zum Ziel, den Entstehungsprozess echter Daten nachzuahmen. Das Verständnis des Entstehungsprozesses von Daten ist auch deshalb so wichtig, weil er kausale Zusammenhänge aufzeigen kann, die es ermöglichen, universellere Modelle zu entwickeln. Verstehen wir beispielsweise den Entstehungsprozess von Nachfragedaten für Bäckereiprodukte, kann dieses Wissen bei jeder Bäckerei der Welt angewandt werden, um beispielsweise die Menge der zu backenden Brötchen zu optimieren.

Diese Dissertation untersucht in drei inhaltlich abgeschlossenen Teilen, wie synthetische Daten genutzt werden können, um Trainingsdaten im Bereich des OM anzureichern und dadurch datengetriebene Modelle zur Entscheidungsunterstützung zu verbessern. Der Fokus liegt dabei auf dem Zeitungsjungenproblem, einem klassischen Problem der Bestandsplanung. Hierbei handelt es sich um ein einperiodiges Planungsproblem, bei dem es gilt, die optimale Bestellmenge zu ermitteln, sodass der Gewinn maximiert wird. Dabei muss berücksichtigt werden, dass unverkaufte Produkte am Ende des Tages einen Verlust bedeuten, aber auch ein zu schneller Ausverkauf potenzielle Einnahmen verpassen lässt.

Der erste Artikel, „A Meta Analysis of Data-Driven Newsvendor Approaches“, Kapitel 2, dient als Vorstudie zur Verwendung synthetischer Daten. Obwohl bisher in der Literatur mehrere datengetriebene, präskrip-

tive Ansätze vorgeschlagen wurden, ist es unklar, wie diese im Vergleich zueinander abschneiden. In dem Artikel wird ein reproduzierbares und einheitliches Bewertungsverfahren für datengetriebene, präskriptive Ansätze präsentiert. Das vorgestellte Bewertungsverfahren kann sowohl von Praktikern zur Modellauswahl als auch von Forschern zum Benchmarking neuer Ansätze verwendet werden. In diesem Artikel wird es in einer umfangreichen numerischen Studie verwendet, die mit einem großen und heterogenen Datensatz durchgeführt wird. Teil dieser Studie ist eine Robustheitsanalyse, um den Einfluss verschiedener Problemparameter zu bewerten, die die Leistung des Modells potenziell beeinflussen können, wie z.B. Eigenschaften des Datensatzes oder des zu lösenden Planungsproblems. Die Ergebnisse deuten darauf hin, dass die Leistung der evaluierten Modelle wenig robust ist und das zu verwendende Modell auf der Grundlage eines standardisierten Evaluierungsprozesses ausgewählt werden sollte, um bestmögliche Ergebnisse zu gewährleisten.

Im zweiten Artikel, „Application of Generative Adversarial Networks in Inventory Management“, Kapitel 3, wird die Verwendung synthetischer Daten, die durch *Generative Adversarial Networks* (GANs) erzeugt wurden, zur Lösung des Zeitungsjungenproblems untersucht. Der Einsatz datengetriebener, präskriptiver Verfahren hat zu einem wachsenden Bedarf an relevanten Trainingsdaten geführt, insbesondere wenn zusätzliche Informationen (Features) eingebunden werden. Daraus ergibt sich ein Bedarf an Techniken, die komplexe Beziehungen zwischen Nachfrage und Zusatzinformationen modellieren können und mit denen große Mengen synthetischer Daten erzeugt werden können. In diesem Artikel wird gezeigt, wie solche synthetischen Daten mit Hilfe von GANs - einem Ansatz des Deep Learning - erzeugt werden können. Da die Leistung von GANs häufig instabil ist, wird eine Selektionsstrategie als Vorstufe zur Anwendung der GAN-generierten Daten im Planungsproblem entwickelt. In numerischen Experimenten wird der vorgeschlagene Ansatz im Praxiskontext einer Bäckereikette angewandt und unter Variation verschiedener Experimentparameter untersucht. Er wird mit traditionelleren Ansätzen, wie dem *Distribution Fitting* und der *Sample Average Approximation* (SAA), verglichen. Die Ergebnisse legen nahe, dass die Anwendung von GANs eine vielversprechende Alternative zu diesen traditionellen Ansätzen darstellt.

Im dritten Artikel, „Combining Synthetic Data and Transfer Learning for Deep Reinforcement Learning in Inventory Management“, Kapitel 4, wird ein neuartiges, auf *Deep Reinforcement Learning* (DRL) basierendes Verfahren vorgeschlagen, das synthetische und reale Daten mittels *Transfer Learning* kombiniert, um OM-Entscheidungsprobleme zu lösen. Die Anwendung von DRL verspricht größere Flexibilität in der Problemdefinition als traditionellere, präskriptive Ansätze. Allerdings erfordert es auch große Mengen an Trainingsdaten. In diesem Artikel wird ein zweistufiges Verfahren vorgeschlagen, um mit weniger echten Trainingsdaten auszukommen. Zunächst wird ein generatives Modell trainiert, um die unbekannte gemeinsame Verteilung von Nachfrage und Features zu lernen. Dieses wird genutzt, um zusätzliche synthetische Trainingsdaten zu generieren. In einem zweiten Schritt wird ein DRL-Agent mit Hilfe des *Transfer Learnings* trainiert, wobei der DRL-Agent zunächst auf den synthetischen Daten vortrainiert wird und dann ein Feintuning auf der Grundlage eines kleineren realen Datensatzes erfolgt. Dieser Artikel evaluiert das vorgeschlagene Verfahren für ein Zeitungsjungenproblem in zwei verschiedenen numerischen Studiensettings. In dem kontrollierten Studiensetting ist die Verteilung der Daten bekannt, wodurch ein erster Schritt gemacht wird, zu verstehen, was die Leistung des vorgeschlagenen Verfahrens beeinflusst, z.B. die Qualität der generierten synthetischen Daten. Im Praxissetting, in dem die gemeinsame Verteilung der Daten unbekannt ist, wird das vorgeschlagene Verfahren auf Daten einer lokalen Bäckereikette angewandt. In beiden Fällen übertrifft das vorgeschlagene Verfahren die traditionelle präskriptive Methode. Es ist jedoch weitere Forschung erforderlich, um die Generalisierbarkeit dieser Ergebnisse zu beweisen.

Insgesamt zeigen die Ergebnisse dieser Dissertation, dass der Einsatz von synthetischen Daten Potential hat, Praxisanwendungen des maschinellen Lernens zu unterstützen. Die untersuchte Methode der Datengenerierung mit GANs ermöglicht die Modellierung komplexer Zusammenhänge in den Daten und unterstützt damit selbst die Anwendung von datenhungrigen Verfahren, wie DRL, zur Lösung von Planungsproblemen. Die Wahl eines guten GAN-Modells ist jedoch mit hohem Aufwand verbunden, sodass Kosten und Nutzen synthetischer Daten bei jeder Anwendung abgewogen werden sollten. Weitere Forschung ist notwendig, um die Generalisierbarkeit

der Ergebnisse zu gewährleisten.

Contents

1 Introduction

In a world of constant change and unpredictability, the ability to navigate uncertainty is a critical skill for organizations that want to succeed. Midst the whirlwind of shifting market demands, supply chain disruptions, and rapid technological advancements, businesses must remain agile and adaptive, which is where the concept of operations management (OM) comes into play.

OM is a multidisciplinary field that focuses on the planning, organization, and control of business processes to achieve optimal efficiency, productivity, and customer satisfaction. For instance, operations managers in the retail sector may need to determine when and how much inventory to restock to balance product availability and stock costs, all while customer demand remains uncertain and subject to fluctuation. Clearly, decision-making under uncertainty is a significant aspect of OM.

Mišić and Perakis, 2020 observe that research in OM has a long history of developing and applying theoretical models, typically leveraging microeconomic theory, game theory, optimization, or simulation techniques to identify optimal decisions for operational problems. However, given methodological advances in machine learning and optimization, made possible by greater computing resources, the need to complement theoretical models with real-world data has grown (Mišić and Perakis, 2020), giving rise to a new field of research: data-driven OM. Data-driven OM seeks to combine real-world data, optimization, and machine learning techniques to derive optimal decisions.

Real-world data can help researchers and decision-makers develop realistic models that can capture the complexity of decision-making under real-world conditions. Real-world data in OM usually takes the form of time series data, often presented in tabular format, and is collected over time with a fixed interval between each data point. Two types of data are usually

used: past observations of uncertain quantities of interest, which for many OM problems is customer demand, and past observations of "features" like weather or stock market developments that may influence these uncertain quantities.

In an effort to integrate both data and machine learning into decision-making, researchers follow one of two approaches.

The first, "predict-then-optimize", involves using a two-step approach to derive decisions from data, where a machine learning or statistical model leverages features to predict the future value of uncertain quantities of interest (e.g. demand for a certain product) before classical optimization techniques are used to derive a decision based on the prediction and uncertainty estimates. However, the separation between prediction and optimization may result in the loss of valuable information, as prediction approaches focus primarily on reducing prediction error, while optimization requires good estimates of prediction uncertainty. As Bertsimas and Kallus (2020) state, "it is not clear how to go from a good prediction to a good decision" (p. 2).

In response to this shortcoming, the second approach, prescriptive analytics, emerged to derive truly data-driven OM decisions. Prescriptive analytics approaches combine prediction and optimization into a single step, prescribing decisions directly from available data, often using methods inspired by machine learning. Some proposed approaches are asymptotically optimal when large amounts of data are available, but such is frequently not the case in practical settings, as the field of OM suffers from a paradox: While sensors, tracking devices, and business systems generate a plethora of data every day, recordings of uncertain quantities of interest, such as customer demand, are typically scarce, especially in disciplines like capacity and inventory management, where decisions to deploy resources or replenish inventory must be taken daily, weekly, or even monthly. Consequently, even though the number of features used to explain customer demand can increase, the length of the time series cannot be extended easily, as observations from the distant past are often irrelevant to today's decisions.

The need for data may be even more critical if OM wants to benefit from the next wave of improvement that is expected to emerge from leveraging deep learning models (Kraus et al., 2020). Traditional machine learning

methods (e.g. linear regression models, support vector machines, random forests) that are widely used in prescriptive analytics require significant human involvement (e.g. in selecting and constructing the features for the model to learn from or designing a policy that has to be learned). In contrast, the promise of deep learning is to have the machine work more independently on raw data, alleviating the need for human feature engineering (Goodfellow, Bengio, et al., 2016) and human formulations of the optimization problem (in the case of deep reinforcement learning (Boute et al., 2021)). However, these approaches are even more data-hungry, currently limiting their application in real-world settings (Kraus et al., 2020).

Other domains in which the availability of data to train machine learning models is an issue have started to complement or even replace real-world data with synthetic data that is artificially generated, rather than collected from real-world observations. One well-known domain that has benefited from the work on synthetic data is computer vision, where deep learning models are at the core of many advances (Shorten and Khoshgoftaar, 2019). Synthetic data can be generated using a variety of methods (Jordon et al., 2022), the first of which exploited the structure/geometry of images. New training samples were created by, for example, rotating, cropping, mixing, or blurring existing training images (Shorten and Khoshgoftaar, 2019). More sophisticated synthetic data generation methodologies have emerged since, that do more than modify existing data points, as they learn the joint distribution of all data variables to simulate how data is generated in the real world (Kingma, 2019). This method allows new data points to be created (e.g. in the case of computer vision, new image material). Understanding the generative process of data is also powerful because it can reveal causal relationships as they occur anywhere in the world (Kingma, 2019). For example, once we understand the generative process of demand for bakery products, we can apply this knowledge to bakeries anywhere.

One method that is at the forefront of new image generation is generative adversarial networks (GANs). In contrast to geometric approaches, GANs promise not to alter existing images but to generate new images that stem from the same distribution as the original images and cannot be distinguished from them. The success of GANs has been translated to

other domains, such as natural language processing (Shorten, Khoshgoftaar, and Furht, 2021), time series (Wen et al., 2021), and tabular data (Jordon et al., 2022). Advances in domains with structured data (e.g., time series and tabular data), such as OM, are especially important in increasing the application of artificial intelligence (Jordon et al., 2022). This dissertation makes a first step toward investigating the application of synthetic data in inventory management, a subfield of OM.

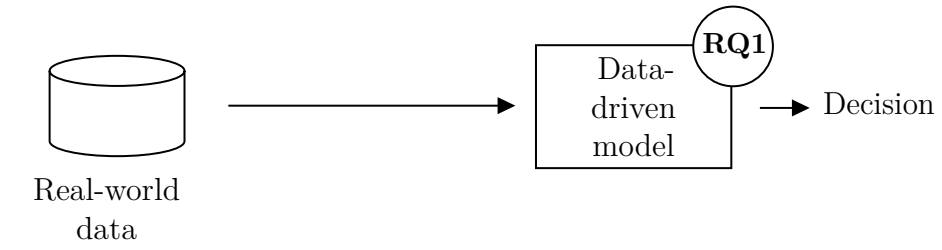
1.1 Synthetic Data for Data-driven Inventory Management

Inventory management is an important part of OM, as having effective inventory control policies in place is central to a business's remaining competitive in an unpredictable and constantly evolving business environment. However, a lack of relevant data makes it challenging to utilize sophisticated machine learning models for decision-making. For instance, the COVID-19 pandemic highlighted how rapidly customer behavior can change, rendering past observations unrepresentative of the future. Additionally, obtaining significant data to plan demand for newly introduced products can be challenging. Even if a company has a wealth of data at the organizational level, for example, due to many different products in its inventory, planning decisions must be made at the product level drastically limiting the amount of available data. In addition, replenishment decisions are typically made daily, weekly, or monthly, depending on the product's shelf life, which further limits the amount of data available to train sophisticated machine learning models, particularly when feature information is incorporated.

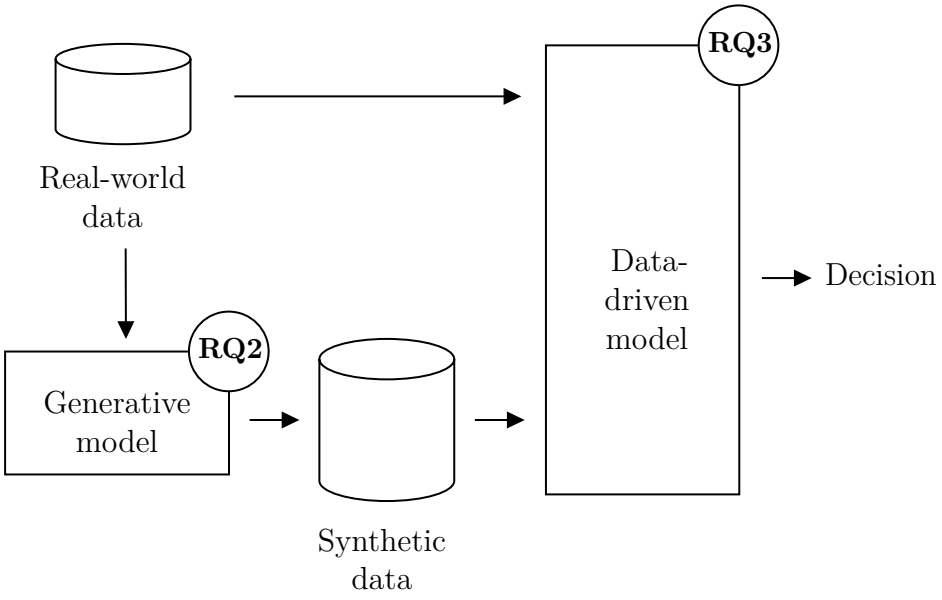
The generation of synthetic data promises to enrich inventory data sets and improve decision-making, so this dissertation investigates this guiding research question:

Guiding Research Question. *How can synthetic data be leveraged to improve decisions for data-driven inventory management?*

Figure 1.1 offers a conceptual representation of where this dissertation's



(a) Traditional data-driven approach



(b) Synthetically enriched data-driven approach

Figure 1.1: Overview of Research Questions

work anchors to include synthetic data in data-driven decision-making. The traditional approach to data-driven decision-making is shown in Figure 1.1a, where real-world data is utilized to train a data-driven model to prescribe a decision. Figure 1.1b, on the other hand, shows how synthetic data can enrich the traditional data-driven approach, potentially leading to improved decisions. In this case, a generative model is trained using real-world data to create a synthetic dataset, and both synthetic and real-world data are then used to train a data-driven model that prescribes a decision.

The work laid out in this dissertation focuses primarily on the single-period newsvendor problem, which has been studied extensively in the OM

research community and has several benchmarks. The newsvendor problem is a classic example of a stochastic optimization problem in inventory management. Here, the decision-maker's objective is to maximize profit by determining the optimal order quantity for a perishable product with uncertain demand. The expected profit is the difference between the revenue earned from selling the product and the cost of ordering or producing it. Any unsold inventory at the end of the selling period is a loss. Likewise, stockouts result in lost sales, reducing the expected profit. The newsvendor problem has served as a starting point for the development of data-driven methods in inventory management (Ban and Rudin, 2019; Bertsimas and Kallus, 2020; Oroojlooyjadid, Snyder, et al., 2020).

The first part of this dissertation is a pre-study of synthetic data for decision-making. Work proposed until now has predominantly applied data-driven approaches to small, often proprietary, real-world datasets and has lacked comprehensive benchmarking of their proposed methods and robust analyses of model performance across multiple real-world datasets. Consequently, which method should be selected as the state of the art for decision-making in inventory management remains uncertain. However, knowing the state of the art is central to assessing how well approaches that are enriched with synthetic data work.

Therefore, the first research question that this dissertation addresses is:

Research Question 1 (RQ1). *What data-driven methods achieve the best performance for a single-period inventory control problem and what factors can influence the model selection?*

Based on the findings from addressing Research Question 1, this dissertation examines how synthetic data can be employed to improve decision-making further. As in Figure 1.1b, an adequate generative model is required to generate synthetic data.

Prescriptive analytics methods use the joint distribution of demand and features to prescribe decisions, so generative models that can capture the complex relationship between demand and features are necessary. Generative models have been developed for various types of data, including time series and tabular data, but such models are not yet widely used in OM-

type problem settings. Therefore, the second research question that this dissertation addresses is::

Research Question 2 (RQ2). *How can generative models be employed to create synthetic data for OM-type problem settings, and how can such data be evaluated?*

Synthetic data are used to enrich the training data input of data-driven models (Figure 1.1b). Enlarging the amount of training data opens the possibility of applying data-hungry models that may work better than traditional data-driven models. Traditional data-driven methods are highly tailored to the problems they address, which requires an explicit formulation of the optimization problem. In complex problem settings like multi-period inventory management, formulating the optimization problem alone may not be feasible, especially if features are involved. Therefore, it is worth exploring deep reinforcement learning (DRL), a field of machine learning that has recently had many successes in solving complex decision-making problems, from learning to play games to performing robotic control tasks to autonomous driving. DRL may be especially useful in solving OM problems because it does not require an explicit formulation of the optimization problem. Instead, DRL algorithms can learn directly from experience by interacting with the environment and receiving feedback in the form of rewards, that indicate the desirability of an action taken by the DRL agent. DRL is even more data-hungry than classical supervised machine learning methods are, so to apply DRL in an inventory management context, this dissertation addresses a third research question:

Research Question 3 (RQ3). *How can deep reinforcement learning be used in combination with synthetic data to improve decision-making for a single-period inventory control problem?*

In summary, this dissertation investigates the application of synthetic data in inventory management, a subfield of OM. The three research questions are designed to explore how synthetic data can be leveraged to improve decision-making, including benchmarking of data-driven methods, using generative models to create synthetic data, and applying DRL in combi-

nation with synthetic data. By answering these research questions, this dissertation contributes to the growing field of data-driven decision-making in inventory management and provides insights into how synthetic data can be used to improve decision-making.

1.2 Structure of the Dissertation

This dissertation is composed of three independent articles that explore the guiding research question.

The first article, "A Meta Analysis of Data-Driven Newsvendor Approaches"[1] (Chapter 2), proposes a reproducible and unified evaluation procedure for data-driven approaches and conducts a large-scale numerical study in the context of a newsvendor problem setting. This work addresses research question 1. Although several data-driven approaches have been proposed, it is unclear how their performances compare, as authors use different benchmarks and various small and often proprietary data sets and do not make available the code they used to implement the approaches. In contrast, this article establishes a benchmarking setup that can be used by practitioners to choose a model and by researchers to benchmark new approaches. This setup is used in a comprehensive numerical experiment that is performed on a large and heterogeneous data set. A robustness analysis is conducted to evaluate the influence of various problem parameters that may affect model performance: properties of the data set, such as domain, size, or features included, and properties of the OM problem to be solved, such as parameters of the cost function. The study's results suggest that the models evaluated show little robustness and should be selected only based on a standardized evaluation process to guarantee state-of-the-art results.

The second article, "Application of Generative Adversarial Networks in Inventory Management"[2] (Chapter 3), investigates research question 2 by exploring the use of synthetic data generated by GANs to solve a classic inventory control problem, the newsvendor problem. The use of data-driven OM approaches has created a growing need for rich data sources in the

[1] This article is co-authored by Andreas Philippi, Nikolai Stein and Richard Pibernik.
[2] This article is co-authored by Nikolai Stein.

OM context. Although large amounts of auxiliary data, such as sensor information, tracking devices, customer relationship management (CRM), and enterprise resource planning (ERP) systems, are available, access to data on customer demand is often limited. This limitation creates the need for techniques that can model complex relationships between demand and auxiliary information and can be used to generate large amounts of synthetic data. Current approaches, such as distribution fitting and sample average approximation, are not suited to modelling such complex, high-dimensional relationships. This article shows how data can be generated synthetically using GANs, a type of deep learning approach. Since GANs' performance is known to be unstable, a selection strategy is used before applying the GAN-generated data to a data-driven OM context. In numerical experiments, the approach is applied in a real-world setting of a bakery chain to solve the newsvendor problem under various experiment parameters and compared to more traditional approaches, such as distribution fitting and sample average approximation. The results suggest that applying GANs is a promising alternative to these traditional approaches.

The third article, "Combining Synthetic Data and Transfer Learning for Deep Reinforcement Learning in Inventory Management"[3] (Chapter 4), proposes a novel DRL-based approach that combines synthetic and real data by means of transfer learning to solve OM decision-making problems. This work addresses research question 3. Although DRL promises a new way of learning decision-making policies, especially in settings in which an explicit policy cannot be defined, it requires large amounts of training data. This article proposes a two-step approach to alleviating this need. First, a generative model is trained to learn the unknown joint distribution of demand and features, which enables large amounts of synthetic training data to be created. Then a DRL agent is trained using transfer learning, where the DRL agent is pre-trained on the synthetic data and then fine-tuned based on a smaller real data set. The article evaluates the proposed approach for a newsvendor problem setting in two numerical studies: a controlled study, where the joint distribution is known, and a real-world study at a local bakery chain, where the joint distribution is unknown. The controlled setting makes a first step toward evaluating what drives the proposed

[3]This article is co-authored by Nikolai Stein and Richard Pibernik.

approach's performance, such as the quality of the synthetic data. In both settings, our proposed approach outperforms more traditional approaches, but more research is needed to establish the results' generalizability.

Table 1.1 presents an overview of the three articles' scientific contributions. Chapter 5 provides a summary of the findings and a conclusion, along with directions for future research.

Table 1.1: Overview of scientific contributions

Article	*Model focus*	*Methodological contributions*	*Conceptual findings*
A Meta Analysis of Data-Driven Newsvendor Approaches (Chapter 2)	Selected data-driven OM approaches	• Development of a reproducible, unified evaluation procedure for data-driven OM approaches • Structured comparison of data-driven OM approaches on a large and heterogeneous data set	• The evaluated approaches appear to have a low level of robustness against variations in problem parameters and across data sets • There is neither a dominating model nor a model that is always dominated
Application of Generative Adversarial Networks in Inventory Management (Chapter 3)	GAN	• Development of a GAN selection procedure tailored to OM problems • Structured evaluation of GAN-based wSAA approach	• The GAN-based synthetic data generation can model complex relationships between demand and auxiliary features, thus improving the decision-making made using traditional distribution fitting approaches • Computational costs should be considered
Combining Synthetic Data and Transfer Learning for Deep Reinforcement Learning in Inventory Management (Chapter 4)	GAN DRL	• Development of a transfer learning approach that leverages synthetic and real data • Evaluation of the proposed approach in settings with known and unknown data-generating process	• The proposed transfer learning approach appears to lead to better performance than using DRL on a small, real data sample alone

2 A Meta Analysis of Data-Driven Newsvendor Approaches

Recently, a number of publications in leading operations management and operations research journals proposed new models that combine machine learning and mathematical optimization techniques to predict inventory decisions directly from historical demand and additional feature information. This paper presents the results of a meta analysis of recent machine learning-based approaches for solving the most prominent problem in operations management, the newsvendor problem. We find that the reproducibility of existing studies is typically low because authors evaluate their new approaches based on small and proprietary data sets, do not share data and code, and use different benchmarks. We develop a reproducible, unified evaluation procedure and apply various models to a large and heterogeneous data set. Our results do not support the findings and claims of most of the recent papers, and, in several cases, we even obtain contradictory results. In general, the robustness of the newly proposed models appears to be low. To support both researchers and practitioners in the development and evaluation of new models, we provide extensive benchmark data and a Python library that contains open source implementations of most existing models.[4]

[4]This paper was published in *ICLR 2022 ML Evaluation Standards Workshop* Buttler et al. (2022). It is co-authored by Andreas Philippi, Nikolai Stein and Richard Pibernik.

2.1 Introduction

In today's fast-paced world, companies face considerable uncertainty when making important decisions in operations management, for example, when deciding upon capacity, inventory levels, transportation, and production schedules. However, with the rise of digitization, companies have gained unprecedented access to data related to their particular decision problem, offering the opportunity to reduce the degree of uncertainty. For example, in inventory management, the decision maker may have access to historical demand data as well as additional side information that may be predictive of the uncertain demand, such as weather data, calendar information, and data extracted from social media. Driven by the availability of such rich data sources, a stream of literature in operations management research has recently emerged called data-driven operations management (DDOM). Newly proposed DDOM approaches depart from the traditional predict-then-optimize (PO) paradigm that has been standard in operations management: Instead of first predicting the uncertain variable of interest and then solving a stochastic optimization problem, they integrate machine learning and optimization techniques to *directly* predict a cost-optimal decision from historical data.

As the classical single-period inventory management setting, the newsvendor problem naturally became a starting point for developing DDOM approaches. In the newsvendor problem, a decision-maker has to determine the optimal inventory quantity for a single period and incurs costs when demand is higher or lower than the inventory. The problem is different from "standard" regression problems studied in machine learning research in two ways: First, the loss function is non-symmetrical as over-predicting (i.e., having too much stock at hand) may incur different costs than under-predicting (i.e., not being able to fulfill all customers' demands). Second, identical absolute errors, in terms of units ordered, may not be of the same importance for all instances, because different products make different contributions to a company's bottom line. Therefore, models have to be evaluated based on the bottom-line impact and not on their predictiveness. Recently, a number of new DDOM approaches for solving the newsvendor problem have been published in leading operations management and operations research

journals. The common denominator of these papers is that they propose and analyze new DDOM approaches, and demonstrate that they outperform some benchmark model(s). However, the reproducibility of these studies is typically low because most authors only evaluate their approaches based on small and proprietary data sets and do not share data and code. In addition, they use a variety of different benchmarks. This renders a comprehensive comparison of the different DDOM models impossible and creates problems for researchers and practitioners: Researchers have no objective means to benchmark against state-of-the-art approaches, and practitioners can hardly identify the best approach for solving their real-life problems in a robust fashion.

In this paper we perform a meta analysis of existing DDOM models for solving the newsvendor problem on large and very heterogeneous data using a standardized and reproducible procedure. In our study, we cannot reproduce the results of most of the previous papers and, in a number of cases, we even obtain contradictory results. We find that model robustness is low and that model performance depends on the specific product for which an inventory decision has to be taken, the parameters of the loss function, and the available feature information. More specifically, we observe that any one of the models under consideration can, under certain conditions, be optimal for an individual product—there is no dominating approach that can be established as state-of-the-art. Next to these important insights, we make several other contributions that can enhance future research on DDOM models: (1) We are the first in the field of DDOM to provide extensive benchmark data for the newsvendor problem. (2) We developed an open-source Python package that provides access to the most relevant DDOM models to allow for easy and efficient performance comparison and benchmarking. (3) We provide guidance on how to carry out an objective, structured, and reproducible evaluation of DDOM approaches that can also be applied to problems other than the newsvendor problem.

2.2 The Newsvendor and Data-Driven Solution Approaches

Among the many inventory control problems that have been addressed in the OM literature, the newsvendor problem is the most basic single-period inventory problem under demand uncertainty. As such, it became the natural starting point for developing DDOM approaches (Qi et al., 2020). In a newsvendor setting, the decision-maker decides upon the inventory of a single product for a single selling season. Any leftover demand at the end of the season leads to overage costs of c_o per unit. The decision-maker incurs underage costs c_u for each unit of demand that cannot be satisfied. Consequently, the decision-maker seeks to determine the order quantity q that minimizes the total expected costs. For a single product, the problem can be stated as follows:

$$\min_{q \geqslant 0} = \mathbb{E}_D[c_u(D-q)^+ + c_o(q-D)^+], \tag{2.1}$$

where D is the random demand and $(\cdot)^+ := \max\{0, \cdot\}$. If the demand distribution F is known, the optimal solution to (2.1), denoted by q^*, is given by the $c_u/(c_u + c_o)$ quantile:

$$q^* = F^{-1}\left(\frac{c_u}{c_u + c_o}\right), \tag{2.2}$$

where F^{-1} is the inverse cumulative density function (cdf) of D. In practice, the decision-maker cannot directly solve Equation 2.2, because he does not know the true distribution of D. However, historical demand data $d_1, ..., d_n$ is often available that can be used to solve the empirical counterpart of Equation 2.1:

$$q^* = \min_{q \geqslant 0} \frac{1}{n} \sum_{i=1}^{n} \left[c_u \left(d_i - q\right)^+ + c_o \left(q - d_i\right)^+\right] \tag{2.3}$$

The literature refers to this approach as sample average approximation (SAA) (Levi, Perakis, et al., 2015). In today's data-rich environments, companies not only have access to historical demand observations, but to potentially large data sets $S_n = \{(d_1, \boldsymbol{x}_1), \ldots, (d_n, \boldsymbol{x}_n)\}$ that contain historical demand

observations d_t and corresponding feature vectors $\boldsymbol{x}_t (t = 1, ..., n)$. The elements of the feature vectors can be any type of information that may be predictive of the uncertain demand. The new DDOM approaches addressed in this paper learn a decision function from S_n that predicts an inventory decision $q(\boldsymbol{x})$ for each new observation $\boldsymbol{x}$. The existing approaches can be classified into function approximation approaches that are based on the principle of empirical risk minimization (ERM), and approaches that integrate empirical conditional density estimation and optimization.

2.2.1 Empirical Risk Minimization-based Approaches

The approaches contained in this first class seek to learn a function $q(\cdot) : \mathcal{X} \rightarrow \mathcal{Q}$ that maps directly from the feature space $\mathcal{X}$ to a decision space $\mathcal{Q}$ by minimizing the empirical risk, which is defined as the average cost over the training data S_n. More formally, the problem to be solved is given by:

$$\min_{q(\cdot)\in\mathcal{F}} R_N(q(\cdot); S_n) := \frac{1}{n}\sum_{i=1}^{n}\big[c_u(d_i - q(\boldsymbol{x}_i))^+ + c_o(q(\boldsymbol{x}_i) - d_i)^+\big], \qquad (2.4)$$

where R_N is the empirical risk of $q(\cdot)$ and $\mathcal{F}$ is a function space. Given the function $q(\cdot)$, one can directly determine a decision $q(\boldsymbol{x})$ for each new observation $\boldsymbol{x}$. The solution to Equation 2.4 is equivalent to the solution of a high-dimensional quantile regression. To learn $q(\cdot) : \mathcal{X} \rightarrow \mathcal{Q}$, a number of different machine learning methods have been used in the literature. Beutel and Minner (2012) and Ban and Rudin (2019) restrict $\mathcal{F}$ to the space of linear functions. Oroojlooyjadid, Snyder, et al. (2020) and Huber et al. (2019) allow for a non-linear function space and determine $q(\cdot)$ by training deep neural networks that minimize the empirical risk in Equation 2.4. In the remainder of the paper, we use the acronyms LR to refer to the linear models proposed by Beutel and Minner (2012) and Ban and Rudin (2019), and DL to refer to the models proposed by Oroojlooyjadid, Snyder, et al. (2020) and Huber et al. (2019).

2.2.2 Conditional Density Estimation and Optimization

The approaches contained in this second class are based on the derivation of some data-driven sample weights from features and the optimization of SAA against a reweighting of the data, as expressed in Equation 2.5:

$$q^*(\boldsymbol{x}) = \arg\min_{q \in \mathcal{Q}} \sum_{i=1}^{n} w_i(\boldsymbol{x}) \big[c_u (d_i - q)^+ + c_o (q - d_i)^+ \big], \tag{2.5}$$

where $\boldsymbol{x}$ is the feature vector of a new instance and $w_i(\cdot)$ is a function that assigns a weight $w_i \in [0, 1]$ to each sample $(d_i, \boldsymbol{x}_i)$ based on the similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}$. We refer to this approach as conditional density estimation and optimization (CDEO). The optimal solution to (2.5) is

$$q^*(\boldsymbol{x}) = \inf \left\{ q : \frac{\sum_{i=1}^{n} w_i(\boldsymbol{x}) \mathbb{1}(d_i \leqslant q)}{\sum_{i=1}^{n} w_i(\boldsymbol{x})} \geqslant \frac{c_u}{c_u + c_o} \right\}. \tag{2.6}$$

In contrast to the ERM-based approaches, CDEO approaches define $q(\boldsymbol{x})$ point-wise. Thus, one has to first determine the sample weights w_i for each new instance $\boldsymbol{x}$, before solving the optimization problem in (2.5). Obviously, the performance of CDEO is driven by the way the sample weights are calculated. Multiple weight functions have been proposed in the literature, in particular by Bertsimas and Kallus (2020), who construct a number of different weight functions based on a variety of predictive machine learning methods, including k-nearest-neighbors, decision tree, and random forest regression. We refer to these approaches as KNNW, DTW, and RFW. Moreover, both Bertsimas and Kallus (2020) and Ban and Rudin (2019) propose to use kernel weight functions (KW).

2.2.3 Review of Competing Data-Driven Models

In Table 2.1 we give an overview of the relevant papers that proposed DDOM approaches applicable to the newsvendor problem, including the data sets and benchmarks that were used for evaluation purposes. While all models have shown superior performance compared to some benchmarks, under certain conditions, it still remains unclear how to compare their performances. The reason for this is twofold: On the one hand, most researchers use their own proprietary data set for evaluation. The data sets

Table 2.1: Overview of relevant DDOM papers (* denotes the models included in our evaluation)

Paper	Model	Benchmark	Data
Beutel and Minner	LR	OLS, MM	Proprietary retail chain data of 64 stores for 270 days; feature information includes price, weather data, and weekdays.
Ban and Rudin	LR*, KW*	PO, SAA	Proprietary data for a nurse staffing problem including demand for nurses in a hospital for 2644 time periods; feature information includes calendar data as well as lags.
Bertsimas and Kallus	KW, KNNW*, DTW*, RFW*	SAA, PP	Proprietary media vendor data including demand for various items and locations for 150 weeks; feature information includes 91 features with information on items, locations, dates, lags, as well as social media data.
Oroojlooyjadid, Snyder, et al.	DL*	KNNW, KW, RFW, LR, PO	Extract of Pentaho MySQL Foodmart Database including retail data from 24 different departments; feature information includes calendar data.
Huber et al.	DL	PO, LR	Proprietary bakery chain data of eleven products for five stores for 528 days; feature information includes calendar data, weather data, and locations of the stores.

vary in terms of the domain from which the data is drawn, the number and type of features, and the length of the corresponding time series. On the other hand, researchers use different models for benchmarking, making a direct comparison of results impossible. For instance, Beutel and Minner (2012) use retail chain data and benchmark their LR model against ordinary least squares (OLS) regression and the method of moments (MM). Ban and Rudin (2019) use empirical data from a nurse staffing problem to evaluate their models and show that both LR and KW outperform traditional PO approaches as well as SAA. Bertsimas and Kallus (2020) use data provided by a big international media vendor for testing. As benchmarks, they use random forest point predictions (PP) and SAA and show that their RFW

model performs best. Building on the work of Bertsimas and Kallus (2020) and Ban and Rudin (2019), Oroojlooyjadid, Snyder, et al. (2020) show that their DL model outperforms not only traditional PO approaches, but also existing data-driven models, including LR, KW, KNNW, and RFW, when applied to their extract of the Pentaho MySQL Foodmart Database. Huber et al. (2019) use proprietary data of a bakery chain to compare, among others, an LR and a DL approach to their PO counterparts. Their results suggest that for limited data availability the traditional PO approaches outperform their DDOM counterparts. The performance of DDOM models increases with data availability in terms of features and historical observations. They do not find any DDOM model that consistently outperforms conventional PO approaches.
In our study, we include the LR and the KW model of Ban and Rudin (2019), because their paper is published in the highest-ranking journal of the discipline and has been the first and most extensive paper on the "Big Data Newsvendor". We also include the KNNW, DTW, and RFW models of Bertsimas and Kallus (2020), as they provide the most extensive analysis and evaluation of CDEO methods. From the two similar papers that propose DL approaches, we select Oroojlooyjadid, Snyder, et al. (2020), because their implementation is similar to that of Huber et al. (2019), but in their evaluation, their model outperformed all other DDOM models, including those of Bertsimas and Kallus (2020) and Ban and Rudin (2019).

2.3 Experimental Setup

In the previous section, we briefly outlined that existing papers claim that their models outperform one or more benchmark approaches but that these claims can hardly be verified beyond individual data sets and particular benchmarks. The main goal of our study is an objective and fair performance comparison of the different models—we want to evaluate their robustness, validate the claims made in previous papers, and ascertain whether there is a model that can be recommended as state-of-the-art. To compare the models we propose a reproducible, unified procedure that is based on standards established in the machine learning community. We share code

and data via our GitHub repository [5] to make our experiments transparent and reproducible. We developed the open-source Python package *ddop* (Philippi et al., 2021), to provide easy and efficient access to the data-driven newsvendor models discussed in Section 2.2.

2.3.1 Data and Experiments

In our experiments, we use four heterogeneous data sets (Bakery, Restaurant, subset of M5, Store Item Demand (SID)). An overview of the data sets is provided in Table 2.2. More detailed information is included in section B.1. As one important step towards reproducibility, we make all of the data sets available [6]. The data sets cover various domains, are of different sizes in terms of the number of products and the length of the time series, and include different features.

Table 2.2: Overview of data set

Data set	Domain	Products × Shops	N	Features
Bakery	Bakery	3 × 5	1215	calendric, lag, weather, holidays
Restaurant	Restaurant	7 × 1	765	calendric, lag, weather, holidays, promotions
M5 (subset)	Retail chain	10 × 10	1942	calendric, lag, events
SID	Retail chain	50 × 10	1826	calendric, lag

To evaluate the robustness of the models in our numerical experiments, we follow a fractional factorial design and vary two specific dimensions: loss function and features.

Loss function: In most machine learning problems, a static loss function is used to evaluate the performance of different predictive models, e.g., the MSE or RMSE for regression problems. This is different in an

[5] https://github.com/opimwue/A-structured-evaluation-of-data-driven-newsvendor-approaches

[6] While M5 and SID were published on Kaggle as part of a competition, the other two data sets were provided by our industry partners and are accessible via our GitHub repository

operations management context. The loss function measures real-world costs and, therefore, depends on parameters that influence these costs. In our newsvendor setting, the loss function is determined by c_u and c_o as defined in Equation 2.1 and it may be asymmetric, depending on the particular values of these parameters. The relationship between c_u and c_o is captured by the so-called service level (sl), defined as $c_u/(c_u + c_o)$ (see Equation 2.2). Naturally, a good model should be robust across different service levels.

Features: Feature availability may vary across practical problems. Therefore, it is important to evaluate the robustness of different models with respect to the feature information. In our evaluation, we define three different feature categories: calendric, lag, and special features. Both calendric and lag features are used to capture the characteristics of the demand time series. Calendric features contain only the information that can be extracted directly from the date of the time series (e.g., weekdays, month, year). Lag features capture information from previous time periods, such as past demand observations, allowing the models to learn properties of the time series such as trend and seasonality[7]. In contrast to the first two feature categories, the special features depend on the respective data set and include domain knowledge, such as information about promotions, special events, or weather conditions.

We define three different experiments (see Table 2.3). We begin with a base case scenario that resembles a typical industry setting—90% service level (sl) with only calendric and lag features. Then we vary sl reflecting the ratio of the parameters c_u and c_o of the loss function. Finally, we vary the feature information to assess the models' robustness towards different levels of feature availability.

2.3.2 Evaluation Process

For each experiment, we first group each data set by product and store—we call a single product-store combination an instance. Subsequently, we apply

[7]To generate lag features, we use the Python library *tsfresh* (https://tsfresh.readthedocs.io/en/latest/index.html). More specifically, we compute basic descriptive statistics (e.g., minimum, maximum, mean, variance) for three different rolling windows of length 7, 14, and 28 days.

Table 2.3: Overview of experiments

	Loss Function	Features	Results
Base scenario	sl = 0.9	X = [calendar, lag]	Section 2.4.1
Loss function variation	sl ∈ {0.1, 0.25, 0.5, 0.75, 0.9}	X = [calendar, lag]	Section 2.4.2
Feature variation	sl = 0.9	X ∈ {[calendar], [calendar, lag], [calendar, lag, special]}	Section 2.4.2

each of the following steps for each instance, service level, and feature category.

Transformation: We apply one-hot encoding to transform all categorical features into their binary representation.

Train-test split: We split the data into a train set containing 75% of the data and a test set containing the remaining 25%. We do not apply shuffling to preserve the structure of the time series.

Scaling: We apply standardization (removing the mean and scaling to unit variance) to all continuous features using the *scikit-learn* standard scaler (Pedregosa et al., 2011). More specifically, we fit the scaler to the training data and then transform both the train and the test set.

Model training: For each model that provides hyperparameters, we apply a grid search on the train set with 10-fold cross validation to find the best parameters leading to the lowest average cross-validation cost. Subsequently, we fit the model to the entire train set. We provide the hyperparameter grids specific to each model in section B.3.

Model evaluation: To enable an objective comparison across data sets, we require the metrics used to be relative measures that operate on a universal scale. Therefore, we cannot use the empirical costs on the test set directly but normalize them by the SAA costs. SAA is a natural baseline as it does not include feature information. Given a model k, we compute the cost delta to SAA ΔC_k as follows:

$$\Delta C_k = 1 - \frac{R_{N_{test}}(q_k, S_{test})}{R_{N_{test}}(q_{SAA}, S_{test})} \tag{2.7}$$

where R_N is the empirical risk as defined in Equation 2.4.

Statistical significance test: In the course of our numerical evaluation, we also want to test our findings for statistical significance. In particular, differences in results obtained from model recommendation versus model selection. We use the one-sided Wilcoxon signed-rank test (Wilcoxon, 1945) that computes the paired differences between samples $d_i = a_i - b_i$ and tests whether $median(d) > 0$. In our case, whether the difference in relative performance improvement over SAA between model selection and model recommendation is greater than zero. It does not assume data to be normally distributed. We test for different levels of significance.

2.4 Results

This section presents the results of our numerical analysis, as described in Table 2.3. We first evaluate and discuss the performance and robustness of the models across the different data sets in the base case. Then, in Section 2.4.2, we evaluate how robust the models are to variations in the parameters of the loss function and feature availability. Finally, in Section 2.4.3, we evaluate how robust the models are relative to a model selection approach—that is, to an approach that selects a priori the best model for each instance, based on cross-validation.

2.4.1 Robustness Analysis across Data Sets

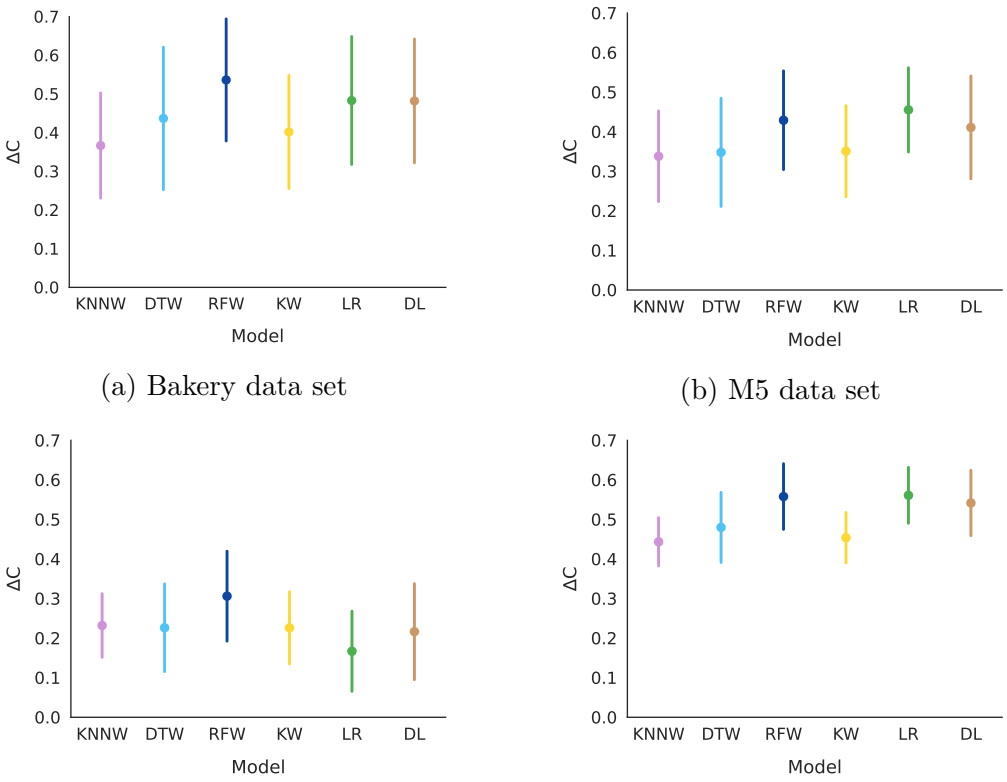
We first explore the performance and robustness of the different models in the base case setting. Figure 2.1 presents the relative performance improvement over SAA for all models by data set. The performances of the different models and their rank order, based on the mean performance, vary across data sets. RFW and LR lead to the highest mean performance improvements in three out of the four data sets. In the Restaurant data set, RFW leads to the highest mean performance, followed by DL. It is surprising that, in most cases, the relatively simple LR model leads to a better performance than models that can account for non-linear relationships between features and the decision variable. The results are not in line with those of Ban and Rudin (2019), where LR was inferior compared to KW,

and they do not support the findings of Oroojlooyjadid, Snyder, et al. (2020) where DL consistently outperformed all other models. Interestingly, in their study, LR was the model with the worst performance, with an average cost increase of 53% compared to DL. Our results are clearly contradicting these findings. Based only on the rank order per data set, either RFW or LR should be the models of choice. However, the overlapping error bars in Figure 2.1 suggest that this conjecture may not hold for individual instances. We carried out additional analysis at the instance level and found that in none of the data sets there is a single model that dominates all others (see the base case results in Figure 2.2); more importantly, we also observe that there is no dominated model—that is, each of the models considered in our study leads to the best performance for at least one instance. In Section 2.4.3 we address the performance impact of choosing the "optimal" model for each instance.

The discussion of the results of our base case analysis reveals two interesting insights: We are unable to reproduce the results obtained in previous studies and there is no model with robust performance across all data sets and instances.

2.4.2 Robustness Analysis depending on the Loss Function and Feature Availability

We now assess the models' robustness toward variations of the parameters of the loss function (reflected by the service level (sl)) and the availability of features X. To this end, we compare the breakdown of the optimal models in the base case to the breakdown under different service levels and feature categories. Selected results are displayed in Figure 2.2 and the detailed results are provided in section B.2. The results in Figure 2.2 suggest that the choice of the optimal model strongly depends on both the parameters of the loss function and the features considered. In the restaurant data set, for example, we see that the share of optimal models is strongly dependent on the availability of features (compare Figure 2.2 BC, Var3, Var4). From the results in the SID data set, we observe that the share of optimal models strongly depends on the parameters of the loss function (compare Figure 2.2 BC, Var1, Var2). These results support our initial conjecture in Section 2.4.1



(a) Bakery data set

(b) M5 data set

(c) Restaurant data set

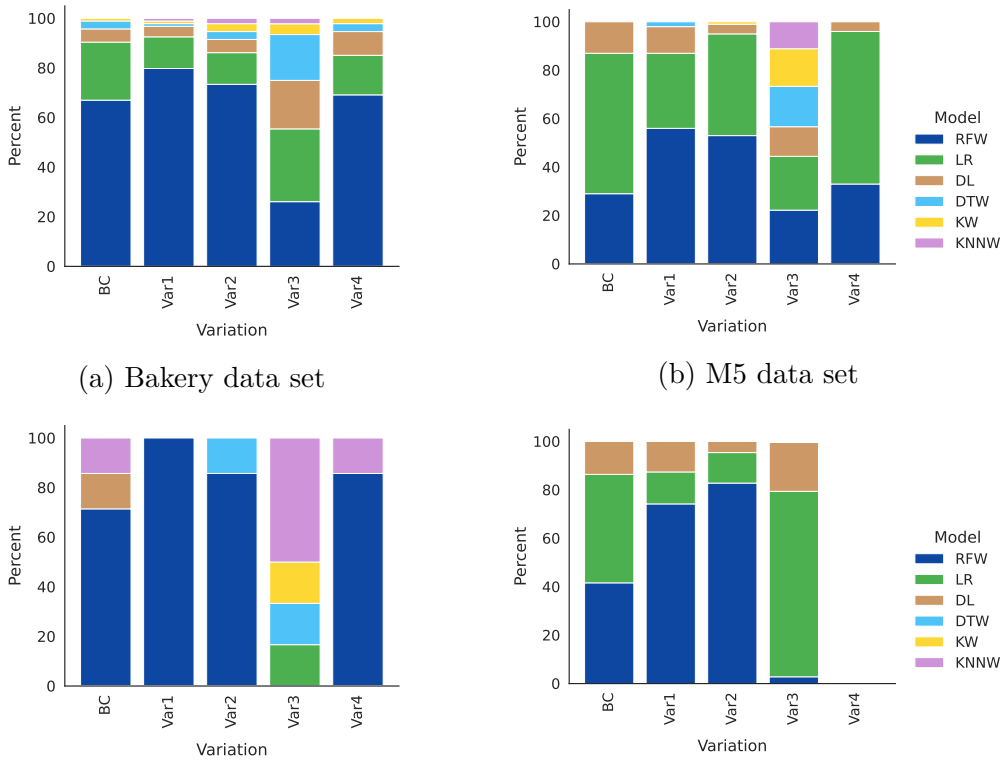
(d) SID data set

Figure 2.1: Best model per instance for base case per data set

that there is no single model that is robust across all data sets and instances under varying parameters of the loss function and feature availability. Of course, in this section, we only consider the optimality of different models. In the next section, we address the robustness of the different models in terms of their cost performance.

2.4.3 Robustness Analysis of Models' Cost Performance

To evaluate the robustness in terms of cost performance, we benchmark the individual models against a model selection approach. In the model selection approach, we identify the best model for each instance based on cross-validation. The difference between the performance of the model selection approach and the performance of an individual model serves as a measure of the model's performance robustness. In Figure 2.3, we report the pairwise cost difference between the model selection approach



(a) Bakery data set

(b) M5 data set

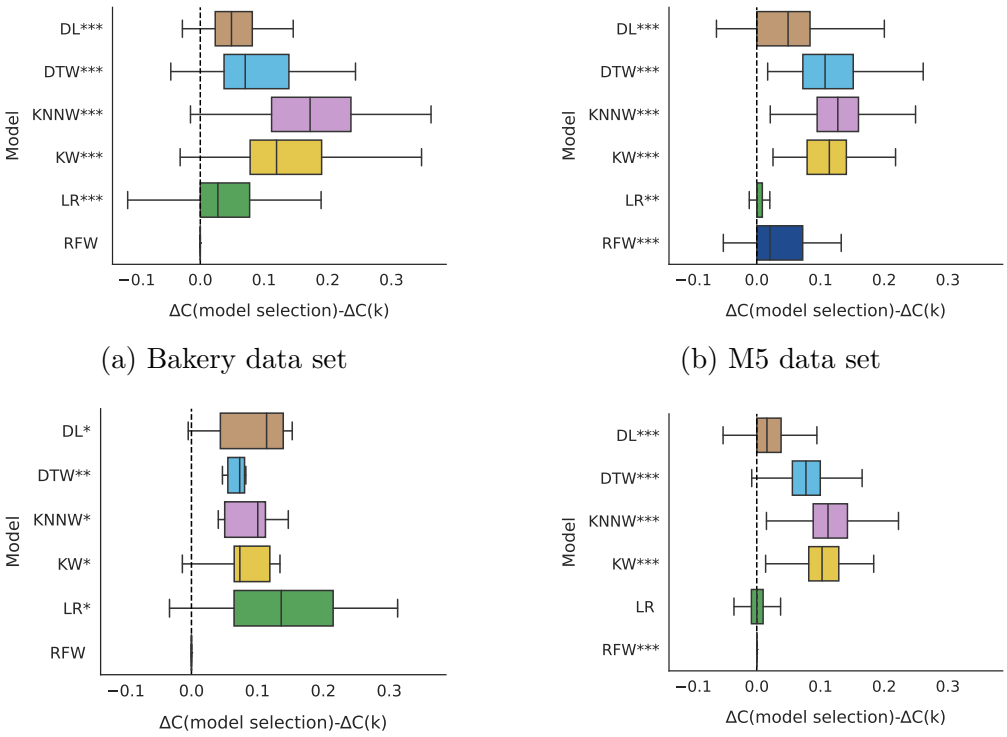
(c) Restaurant data set

(d) SID data set

Figure 2.2: Breakdown of optimal models
BC (base case): sl = 0.9, X = [calendar, lag]
Var1: sl = 0.75, X = [calendar, lag]
Var2: sl = 0.5, X = [calendar, lag]
Var3: sl = 0.9, X = [calendar]
Var4: sl = 0.9, X = [calendar, lag, special]

and the individual models for each instance. First of all, we see that in many cases the model selection approach leads to a statistically significant and substantial performance improvement compared to fixing one model a priori. However, the results are strongly dependent on the individual data sets. Although the results shown in Figure 2.2 suggest a high value of model selection for M5 and SID, we see that we do not experience a large performance impact from model selection. Choosing the LR approach a priori only induces small performance losses that are not statistically significant in the SID data set. On the contrary, in the other two data sets, fixing LR a priori has a very strong and significant detrimental impact on performance. Similarly, the RFW model performs well in the Bakery

and Restaurant data set, but significantly worse than model selection in M5 and SID. The results of this analysis displayed in Figure 2.3 support our initial hypothesis that there is no model that can be recommended as state-of-the-art and that the robustness of the individual models is low.



(a) Bakery data set

(b) M5 data set

(c) Restaurant data set

(d) SID data set

Figure 2.3: Difference in relative performance improvement over SAA between model selection and a given model in the base case
Wilcoxon's signed-rank test results: *: p<0.1, **: p<0.01, ***: p<0.001

2.5 Summary and Discussion

Recently, a number of papers have been published in leading operations management and operations research journals proposing new machine learning-based approaches for solving the newsvendor problem, the most prominent optimization problem in operations management. A key motivation of our study is that the reproducibility of these studies is typically low: They use proprietary data, different benchmarks, and do not share data and code. We are the first to conduct an extensive meta-analysis of these approaches.

Our evaluation is based on large and heterogeneous data and a unified evaluation procedure. Our findings do not support a number of claims made in previous papers with respect to the performance of the newly proposed methods. In some cases, we even obtain results that strongly contradict the conjectures made by the authors. Our results suggest that the performance robustness of the new methods is low. There is neither a dominant model nor a model that is always dominated. This indicates that an evaluation of established and new methods requires a standardized evaluation procedure, a large set of relevant benchmark data, and standards for sharing model implementations that enable reproducible comparisons. We take a first step in this direction by: (1) providing extensive benchmark data for the research community, (2) outlining an evaluation procedure that allows for a fair and objective comparison of alternative approaches, and (3) providing the open source Python library *ddop* (`https://github.com/opimwue/ddop`) that enables an easy and efficient comparison of different approaches.

3 Application of Generative Adversarial Networks in Inventory Management

With machine learning methods entering the space of operations management, a growing need for training data has emerged. Although big data exists in terms of tracing and logging data, historical demand data, in particular, are often highly irregular by their very nature. Furthermore, there may be confidentiality issues that limit access to training data. One way to alleviate this shortage is to augment training data by means of representative synthetic observations. However, current methods to create additional data samples such as distribution fitting suffer from the curse of dimensionality; esp. if we take into account features. We propose the use of generative adversarial networks to synthesize additional samples. In this research, we explore the suitability of this type of data generation in the context of the well-established and tractable newsvendor problem setting. We benchmark our GAN solution approach against distribution fitting, as well as sample average approximation on real data. The numerical evaluations provide promising first results in favor of our GAN-based approach.[8]

3.1 Introduction

In recent years, machine learning techniques have been successfully applied to solve important logistics problems, for example, in inventory management to prescribe replenishment decisions (Bertsimas and Kallus, 2020), or in supply chain management for demand forecasting (Carbonneau et al., 2008;

[8]This paper is co-authored by Nikolai Stein.

Punia et al., 2020). Although logistics systems typically generate large amounts of tracing and logging data (e.g., through sensors, tags, etc.) and benefit from the interconnection of different systems and data sources (ERP, CRM, etc.), the application of machine learning techniques is oftentimes hindered by limited access or availability of transaction data. This is particularly true for inventory and capacity management problems where decisions are made on a daily, weekly, or monthly basis and where customer demand is the random variable of interest that can potentially be explained by many different features. In most problems from practice, one does not have access to more than 1,000 historical demand observations and corresponding features. The reasons for this are multiple. Older data often becomes less relevant due to new products or changing customer behavior. Furthermore, there may be privacy concerns regarding linking data from different entities to establish larger data sets for machine learning. Another challenge is the introduction of new products where there exists no historical data at all, and one can only rely on historical data from similar products if such a product exists. Such "small data" likely rule out the use of data-intensive machine learning techniques. To overcome this problem, we propose to generate additional synthetic data samples to support the decision-making process. Therefore, synthetic data promises to be an entry point into the world of complex machine learning techniques. Data generation is done using a type of generative adversarial network (GAN). To validate our approach, we reduce the complexity of our decision-making task. We apply synthetic data generation to real-world data from a bakery chain. Our goal is to use synthetic data to decide how much product the bakery should produce the next day—the well-known newsvendor problem. This decision setting can easily be transferred to other decision problems as the data generation process is problem-agnostic. The contributions of our research can be summarized as follows:

1. We show how GANs can be used to generate synthetic data and how this data can be used to solve OM problems.

2. We apply the GAN-based weighted sample average approximation (wSAA) in a real-world setting.

3. We compare our approach with two traditional solution methods:

distribution fitting (DF) and sample average approximation (SAA).

In the following section, we provide an overview of related scientific work. After formalizing our problem in section 3.3, we describe our decision-making methodology in section 3.4. Hereafter, we present the setup of our numerical experiments (section 3.5) and their results (section 3.6). We conclude by summarizing our findings in section 3.7.

3.2 Related Work

Fundamentally, data should be the core of operations management (OM), as most problems are motivated by real-world challenges. However, OM research often avoids leveraging real data and instead resorts to data created by the mere assumption of a generic demand distribution (Balugani et al., 2019; Darwish et al., 2019; Yang et al., 2020; Çınar and Güllü, 2012) with a very stylized relationship to real data samples. Especially, approaches that require a lot of training data, such as deep reinforcement learning, resort to parametric assumptions for the demand distribution (Gijsbrechts et al., 2021). Others incorporate additional information in the form of features in their demand forecasts (Punia et al., 2020) but do not use this feature information in the decision-making process.

A recent branch of OM research—data-driven operations management—is concerned with incorporating feature information into decisions. The most promising work has been done by Bertsimas and Kallus (2020) by introducing methods that prescribe a decision based on historical observations of demand and related features. This creates the need for data sets that contain both—demand and feature data. To the best of our knowledge, research on synthetic data generation in an OM context has been restricted to forecasting time series of demand (Gonçalves et al., 2021; Guo et al., 2013; Seyedan and Mafakheri, 2020) or distribution fitting approaches (Gijsbrechts et al., 2021; Turrini and Meissner, 2019; Levi, Perakis, et al., 2015). Incorporating features into real-world approaches is not a trivial task and has two main issues. One, when adding to the number of features, we quickly suffer from the curse of dimensionality as we would have to model a time series/distribution for every feature combination in our historical data.

Two, there are only limited or even no samples in our training set for feature vectors that occur in the future. We propose an approach using generative adversarial networks that has the ability to learn an implicit representation of the joint distribution of demand and features and that allows us to sample from it. Originating in computer vision research, GANs have been shown to be a powerful method for artificial data generation Zhang et al. (2017). Goodfellow, Pouget-Abadie, et al. (2014) first proposed this concept of two opposing neural networks—a generator G and a discriminator network D. Since then, a wide range of models have been developed, focusing on different tasks such as high-resolution image generation (Brock et al., 2019; Karras et al., 2018), video (Vondrick et al., 2016) or even text-to-image translation (Zhang et al., 2017). For these purposes, a large number of network architectures and loss functions have been proposed that are suitable for the specific task at hand (Goodfellow, Pouget-Abadie, et al., 2014; Arjovsky et al., 2017; Gulrajani et al., 2017; Radford et al., 2016). Most recently, they have been adapted to augment tabular data—that is, data in the form of tables where each observation of the training data is resembled by a row in the table, while the columns are different features of the observation. This also makes them suitable for generating synthetic OM data that contains historical demand observations and categorical as well as continuous features such as the day of the week, temperature, lagged demands, etc. Tabular GANs have been used to augment medical patient data (Esteban et al., 2017), but to the best of our knowledge, they have not yet been used in OM. The two most recent models developed for tabular data are Tabular GAN (TGAN, Xu and Veeramachaneni (2018)) and its successor Conditional Tabular GAN (CTGAN, Xu, Skoularidou, et al. (2019)) which is able to mitigate common GAN issues such as mode collapse. A GAN model experiences mode collapse when, e.g., for categorical features, either one or the other category is generated but no mix between categories is achieved that reflects the natural appearance of categories within the data. We use CTGAN for our numerical experiments.

What makes the application of GANs in an OM context special in comparison to vision or language application is the existence of objective metrics. Theis et al. (2016) have compared common metrics in the field of image generation using GAN and concluded that they might yield dif-

ferent results. Still, manual visual inspection is a commonly used method to compare model results, and most articles include a report of samples generated using different GAN models (Arjovsky et al., 2017; Gulrajani et al., 2017; Karras et al., 2018; Brock et al., 2019). In OM, in most cases, we cannot inspect the data manually e.g., due to the high dimensionality of the features. However, the costs resulting from a decision derived based on GAN data allow for comparison of different GAN models.

3.3 Problem Description

We study the application of synthetic data generation to support OM decision-making. Many OM settings are concerned with the task of making a quantity decision (e.g., order placement, production plans, staffing) $\vec{q} \in \vec{\mathcal{Q}}$ to meet demand $\vec{y} \in \vec{Y}$. This decision is taken under uncertainty, since the demand is random following a joint distribution $p_{\vec{X}\times\vec{Y}}$ of demand and additional features $\vec{X}$. These feature variables $\vec{x} \in \vec{X}$ can be temporal (e.g., month, day of the week, public holiday), lagged target variables (e.g., past demand realizations), or anything else that potentially helps explain demand behavior (e.g., weather data, advertising spending, competition intensity). We denote our feature-dependent decision as a function $q(\cdot) : \mathcal{X} \to \mathcal{Q}$ that maps from the feature space to the decision space.

Our goal is to minimize the expected cost C of our decision, which can be formulated as

$$\min_{q(\cdot)\in\mathcal{Q}} \mathbb{E}_{\vec{Y}} \left[C(q(\vec{x}), \vec{D}) \mid \vec{X} = \vec{x} \right], \tag{3.1}$$

In practice, the joint distribution of demand $\vec{Y}$ and additional features $\vec{X}$ is unknown, and only a few historical samples $S_N = \{(\vec{x_1}, \vec{y_1}), \ldots, (\vec{x_N}, \vec{y_N})\}$ are available. Following the weighted SAA approach proposed by Bertsimas and Kallus (2020) we can determine q^* as:

$$q^* \in \arg\min \sum_{i=1}^{N} w_i \, C(q^*, \vec{y_i}) \tag{3.2}$$

where w_i are weights derived from historical data S_N. The weights can be seen as a measure of similarity between the given feature vector $\vec{x}_0$ and the

historical feature vectors $\vec{x}_i \in S_N$. For selected weight functions, weighted SAA has been shown to be asymptotically optimal for $N \to \infty$. Other methods, such as reinforcement learning, that could learn to prescribe an optimal decision also require vast amounts of training data. However, in most practical settings, we do not have access to unlimited amounts of historical data. Alternatively, there may be restrictions on using raw data that require privacy-preserving approaches. To address these issues, we study the use of synthetic data that mimics the behavior of real data to support OM decision-making.

3.4 Methodology

We propose the use of GANs for synthetic data generation. GANs learn an implicit representation of the joint distribution $p_{\vec{X} \times \vec{Y}}$ and sample from it. GANs consist of two competing neural networks—a generator G and a discriminator network D. The goal of the generator network is to create realistic samples that cannot be distinguished from the real training data by D. Both play a Minimax game with a joint loss function, as shown in Figure 3.1. The generator G transforms a noise vector z into a synthetic data sample. The discriminator D compares this generated sample with real-world samples and gives feedback on whether the sample shown is real or fake. The goal of D is to reject synthetic data samples as fake *and* to confirm that real-world data samples are valid.

As described in Section 3.2, GANs have been predominantly applied for high-resolution image generation. The tabular data that we want to generate have properties different from images. While image pixels lie in a fixed value range, tabular data is much more diverse; e.g., we have to distinguish between continuous and categorical columns. Therefore, we have to use a specialized GAN model. We selected the most recent model, conditional tabular GAN (CTGAN) proposed by Xu, Skoularidou, et al. (2019), for our evaluation. We describe CTGAN and its architecture in Section 3.4.1. Although GANs are a powerful tool, it is common knowledge that their training is a fairly challenging task. We describe the challenges and explain how we mitigate low GAN performance in our decision problem in Section 3.4.2.

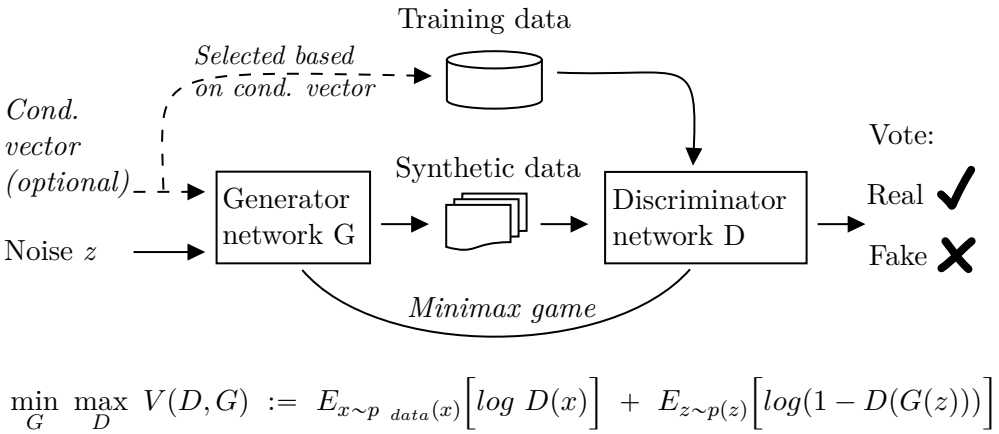


Figure 3.1: GAN concept

3.4.1 Conditional Tabular GAN

The selected model CTGAN is able to model continuous as well as categorical feature variables and mitigates the neglect of underrepresented categories in the training data. This is ensured by means of two methodological choices:

- The input data is preprocessed. Categorical columns are one-hot encoded. For each continuous feature, a variational Gaussian mixture model is fitted to encode the distribution modes and prevent mode collapse (Xu, Skoularidou, et al., 2019), which is a common GAN issue.
- Furthermore, during training, the generator is fed not only a noise vector as in Section 3.2 but also a condition vector. This vector is used to steer the attention of the generator as well as a filter for the real data sample that is shown as a comparison to the discriminator (also see Figure 3.1). This ensures that minority categories can also be learned.

The network architecture for both generator and discriminator was not changed from the original setup (Xu, Skoularidou, et al., 2019). A detailed description can be found in C.1.

In addition to the architecture, there are a number of other parameters—also called hyperparameters—that influence the GAN performance. The

hyperparameters that we vary are the batch size, the number of training epochs, the learning rates for the generator and the discriminator, and the random seed for all random processes. The batch size regulates how many training examples both networks see before a weight update is made. The number of training epochs determines the number of passes through the whole training set. Learning rates determine how quickly weights change in the direction of the loss gradient. The random seed steers random processes such as network weight initialization or weight dropout during training and is varied to avoid unfavorable settings. We provide more details on hyperparameter tuning in the following section.

3.4.2 Data Generation Strategy

In contrast to most deep learning techniques, the loss used to train the GAN does not give a direct indication of the quality of the generated data, and finding metrics to evaluate the performance of the GAN is a vivid field of research (Theis et al., 2016; Lucic et al., 2018). Especially in computer vision where GANs have been primarily used so far, synthetic data lacks an objective performance measure, and results are often only judged by human perception. This makes it difficult to objectively compare performance between different variants of GAN. In contrast, applying GANs in an OM setting has a clear advantage, as the quality of the data can be measured directly by the cost incurred by the problem solution solved using the synthetic data. However, solving the OM problem in each training step is too costly. However, we can use this property when selecting a trained GAN model.

The GAN training itself suffers from instability issues which make it difficult to determine a good set of hyperparameters that will work across several data sets. Instead, we have come up with a mitigation strategy that provides a high-quality GAN model while avoiding manual selection activities. We define a parameter grid that can be found in C.3. In the training phase, we train each model with a fixed hyperparameter configuration, resulting in m models in total (also see Figure 3.2—training phase). From these m models, we select the best model based on the performance in our selected decision task. As the goal of our GAN-based

synthetic data generation is to generate synthetic data that will enable better performance in the decision task at hand, we evaluated all m models on a validation set in the selection phase (see figure 3.2).

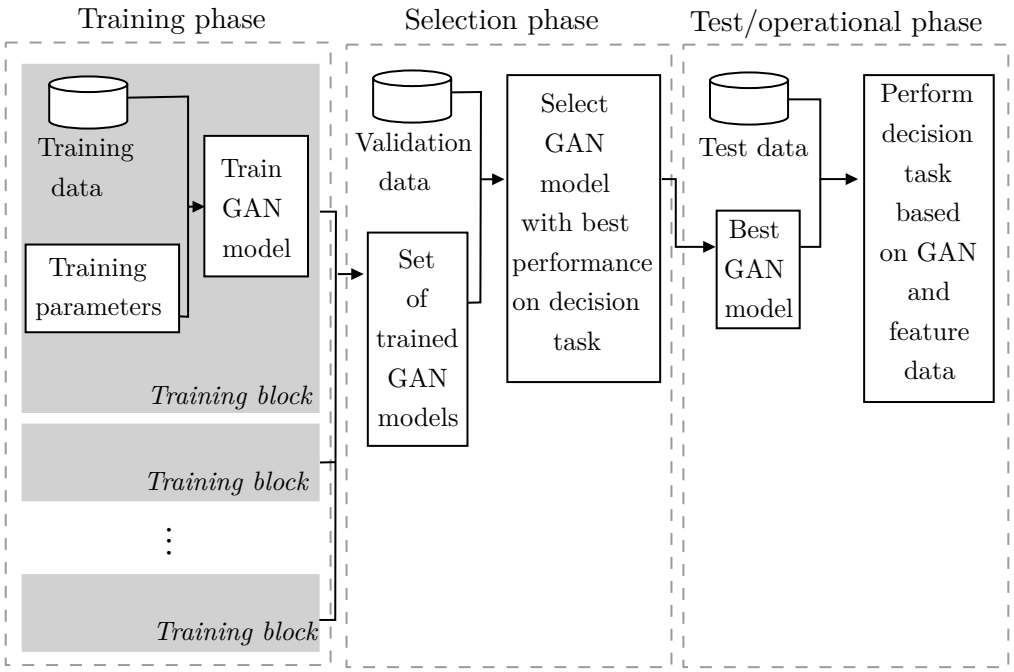
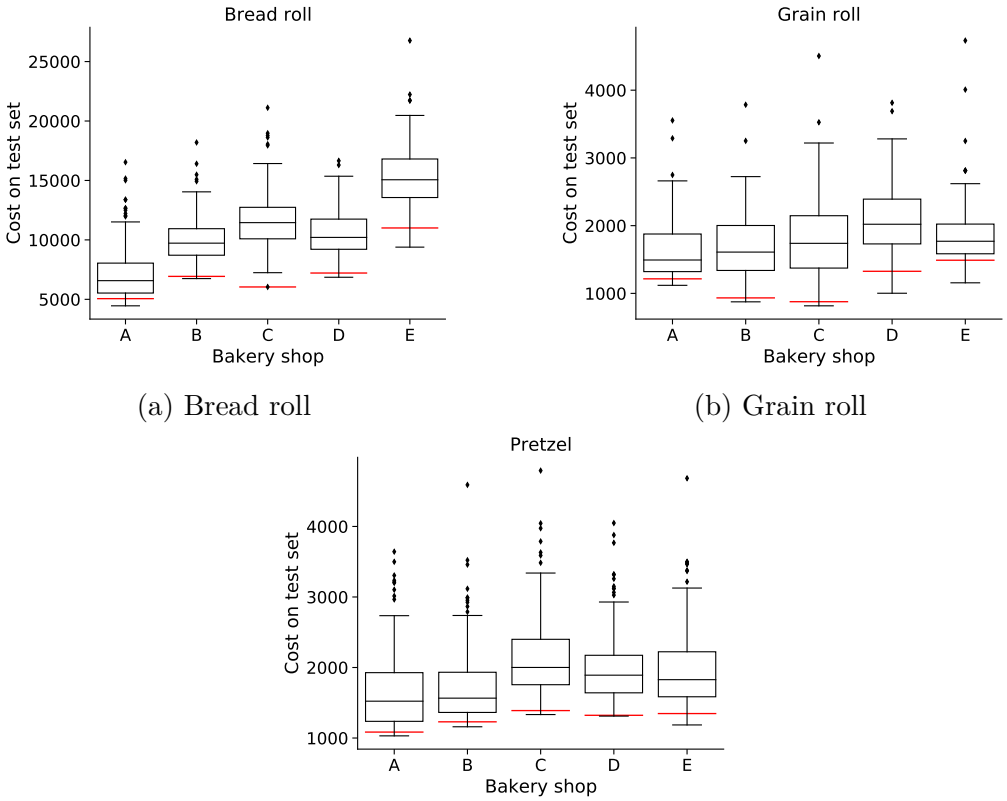


Figure 3.2: Schematic description of GAN selection procedure

This model selection is necessary to avoid unfavorable GAN performance and because we did not see convergence to a certain pattern of hyperparameters in our experiments. Figure 3.3 shows that our selection approach achieves a performance on the test set that is within the first quartile of all m models. In addition, the wide range of realizations in the box plots highlights the importance of effective model pre-selection.



(a) Bread roll

(b) Grain roll

(c) Pretzel

Figure 3.3: Boxplot of newsvendor cost for all 180 GAN models on test set Red mark indicates the performance of GAN model chosen during selection phase on the validation set.

In the operational phase, this best pre-selected GAN model is then used to generate data for the decision on a test set. Its performance in this phase can then be benchmarked against other models and methods which we describe in Section 3.5.2. This concept seems very simple at first, but can become very complex even in a small problem setting such as the newsvendor. The cost of the newsvendor's solution is driven by the desired service level. This entails that we have to select the optimal GAN model for each service level—this can be arbitrarily granular—that we want to evaluate. For other decision tasks where there are even more factors that can influence the cost of a decision, this is even more severe.

3.5 Experimental Setup

We apply GAN-based synthetic data generation in the real-world setting of a bakery chain. In Section 3.5.1 we formulate the OM decision problem and describe the data provided, including relevant features. Section 3.5.2 introduces two suitable benchmark methods for our decision problem, while Section 3.5.3 describes how the evaluation is done in our specific real-world setting.

3.5.1 Problem Setting

Our bakery chain's decision problem is concerned with determining the optimal order quantity $\vec{q}(\vec{x})$ of bakery goods for sale the next day. This problem is a variant of a well-known OM problem—the newsvendor problem. We use this problem as a proxy for more complex decision-making problems. We deliberately chose a problem that can be easily solved to focus on the data generation methodology itself. The newsvendor has to decide how many newspapers to order for the next day. Newspapers that are not sold the following day have to be disposed of and generate overage cost c_o, while unsatisfied demand generates underage cost c_u as sales are lost and customers might not come back. These assumptions are also true in the setting of our bakery, as bakery goods are perishable and cannot be sold the following day. Just as in the general case we described in Section 3.3, demand $\vec{y} \in \vec{Y}$ is not stationary, but follows a joint distribution $p_{\vec{X}\times\vec{Y}}$ of demand and additional features $\vec{X}$. For every feature vector $\vec{x}$, our bakery chain's goal is to minimize the total expected cost

$$\min_{\vec{q}(\vec{x})\in\mathcal{Q}} \mathbb{E}_{\vec{Y}}\left[C(\vec{q}(\vec{x}),\vec{Y})\right], \tag{3.3}$$

where

$$C(\vec{q}(\vec{x}),\vec{Y}) = c_u(\vec{Y}-\vec{q}(\vec{x}))^+ + c_o(\vec{q}(\vec{x})-\vec{Y})^+. \tag{3.4}$$

$(\cdot)^+$ is the $max(0,\cdot)$ and $\vec{q}(\vec{x})$ is a prescription of the order quantity $\vec{q}(\vec{x})$ as a function of features as well as the service level α which is given by $\frac{c_u}{c_o+c_u}$. Therefore, the cost function penalizes excess quantities with c_o and unmet demand with c_u. Given the feature vector $\vec{x}$ for the following day and the

joint distribution $p_{\vec{X}\times\vec{Y}}$, the optimal order quantity $\vec{q}^*$ can be analytically determined by solving

$$F_{\vec{Y}|\vec{X}}(\vec{q}^* \mid \vec{x}) = \alpha \tag{3.5}$$

where F is the conditional cumulative distribution function (cdf). However, in our practical setting this conditional cdf is unknown and we will resort to a weighted SAA approach as described in Section 3.3 to prescribe an optimal order quantity $\vec{q}(\vec{x})$. We determine the weights for the wSAA using a random forest model that is fit to the synthetic data generated by the GAN model. With the help of the random forest model, we can weigh our synthetic samples based on each feature vector $\vec{x}$ in the validation set. The weight w_i of each synthetic sample $s_i = (\vec{y_i}, \vec{x_i})$ is calculated as follows:

$$w_i(\vec{x}) = \frac{1}{T}\sum_{t=1}^{T}\frac{\mathbb{1}\left[\mathcal{R}^t(\vec{x}) = \mathcal{R}^t\left(\vec{x}^i\right)\right]}{\left|\left\{j : R^t\left(\vec{x}^j\right) = R^t(\vec{x})\right\}\right|} \tag{3.6}$$

The fraction nominator in the sum indicates whether the feature vector x of the validation set and a synthetic data sample x_i land in the same leaf node in the tree $\mathcal{R}^t$. The denominator counts how many other synthetic samples x_j also end up in this leaf node. In other words, the higher the weight assigned to a synthetic sample, the more frequently the random forest would make the same prediction for validation data samples and synthetic data samples. We used the *scikit-learn* random forest implementation with standard settings (Pedregosa et al., 2011).

We received sales data from 2016/03/02 to 2019/04/30 for different shops and products from the bakery chain. Examples of typical sales patterns are shown in Figure 3.4.

Although everyday products such as bread rolls are sold throughout the day, there are also products with more irregular sales patterns. Some products are only sold on certain days of the week, such as the raisin roll, while others seem to sell more in the first half of the day, such as the sandwich. When inferring demand from sales data, we have to be careful, as in most cases, the lost demand due to stockouts is not recorded. Sandwich sales are a good example of this problem. We cannot know whether there was really less demand for sandwiches in the afternoons or whether our products were sold out at that point. Therefore, it is likely that the demand

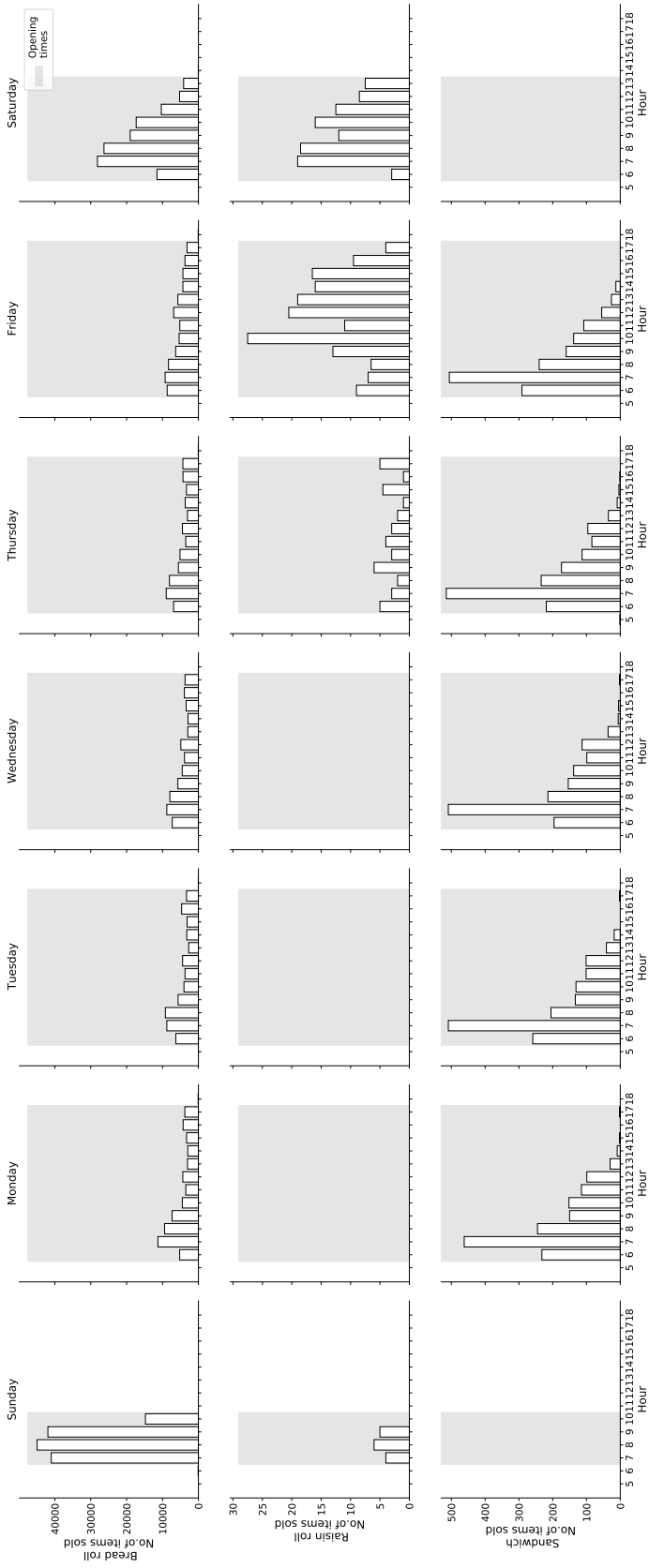


Figure 3.4: Sum of items sold by hour for a selected bakery shop

is censored in this case. If one has access to stock-out records, different methods can be used to infer uncensored demand (Sachs and Minner, 2014; Huber et al., 2019). However, since we do not have access to such data and in order not to distort our results, we selected three products that are unlikely to sell out and show similar behavior to bread rolls (3.4). Furthermore, we selected the five best shops with respect to total sales within the observed period. In terms of features, a detailed description can be found in C.2. We extracted several additional features from the date/time of sales such as month, day of week, and public holidays. Additionally, the bakery chain provided information on the sales campaigns that took place, for example, four for the price of three offers or general price reductions. We aggregated this information into a Boolean feature that indicates if a campaign had occurred in the current or previous week. As our chosen GAN model does not understand our data as a time series, we added features that function as a proxy for any trend in the data. Finally, we added information on the weather on the day of sale.

3.5.2 Benchmarks

We choose to benchmark our GAN results against a traditional parametric distribution fitting (DF) approach, as well as a data-driven sample average approximation (SAA) approach. Both seek to approximate the true demand distribution given a set of historical demand observations $\vec{Y}_N$. DF seeks a parametric distribution $p_\theta \in \mathcal{P}$[9] that minimizes the sum of squared errors between a selected distribution and a histogram fitted to $\vec{Y}_N$ such that

$$p_\theta \in \arg\min \sum_{i=1}^{N_{bin}} \big(p_\theta(i) - hist(i)\big)^2, \tag{3.7}$$

where N_{bin} is the number of histogram bins.

SAA replaces the parametric distribution p_θ fitted by the empirical distribution p_{emp} that naturally arises from the training data. In both cases, the optimal order quantity is given by:

[9] In our case $\mathcal{P}$ is the set of distributions implemented in Python's scipy.stats package (Virtanen et al., 2020)

$$\vec{q}^{*} = F^{-1}(\alpha) \tag{3.8}$$

where F^{-1} is the cdf of the respective distribution.

The advantage of both approaches is their computational efficiency in fitting the parametric or computing the empirical distribution, as well as sampling from it. However, when incorporating features, they quickly suffer from the curse of dimensionality as a distribution would have to be fit for every possible feature combination, leaving not enough training samples to learn from. Still, both are common approaches used within OM research—DF, for example, when performing simulation studies (Law, 2011) where most simulation software such as Tecnomatix Plant Simulation[10], FlexSim[11] or Risk Solver[12] provide it. SAA has also been widely studied from a theoretical perspective (Levi, Perakis, et al., 2015). For our experiments, we fit a distribution for each combination of product and bakery in the training set. Having an explicit representation of the demand distribution, we can determine the optimal order quantity using equation (3.8).

3.5.3 Evaluation Procedure

To be able to compare our results in different experimental settings, we fix the validation set used during the selection phase, as well as the test set used in the test/operational phase (Figure 3.2). The validation set comprises data from the period 2018/05/01 to 2018/10/31 and our test set from the period 2018/11/01 to 2019/04/30, respectively. The remaining data is used as training data. This data has three dimensions—time t, number of shops s, number of products p—which we individually reduce in our experiments to study the effect of small training data (see Figure 3.5). We evaluate GAN and distribution fitting, each using the same training data to fit a model.

For our evaluation in the bakery setting, we differentiate between decisions taken at a bakery shop level and implications/incurred cost at a bakery chain level. While the newsvendor problem has to be solved for every

[10] https://www.plm.automation.siemens.com/global/de/products/manufacturing-planning/plant-simulation-throughput-optimization.html
[11] https://www.flexsim.com/expertfit/
[12] https://www.solver.com/simulation-fit-distributions

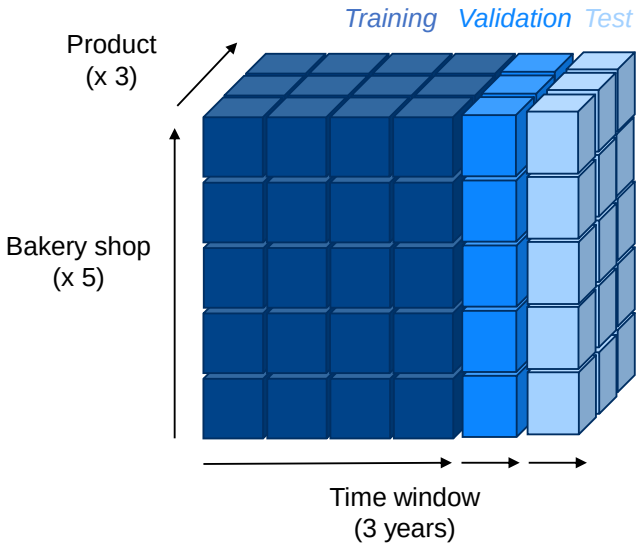


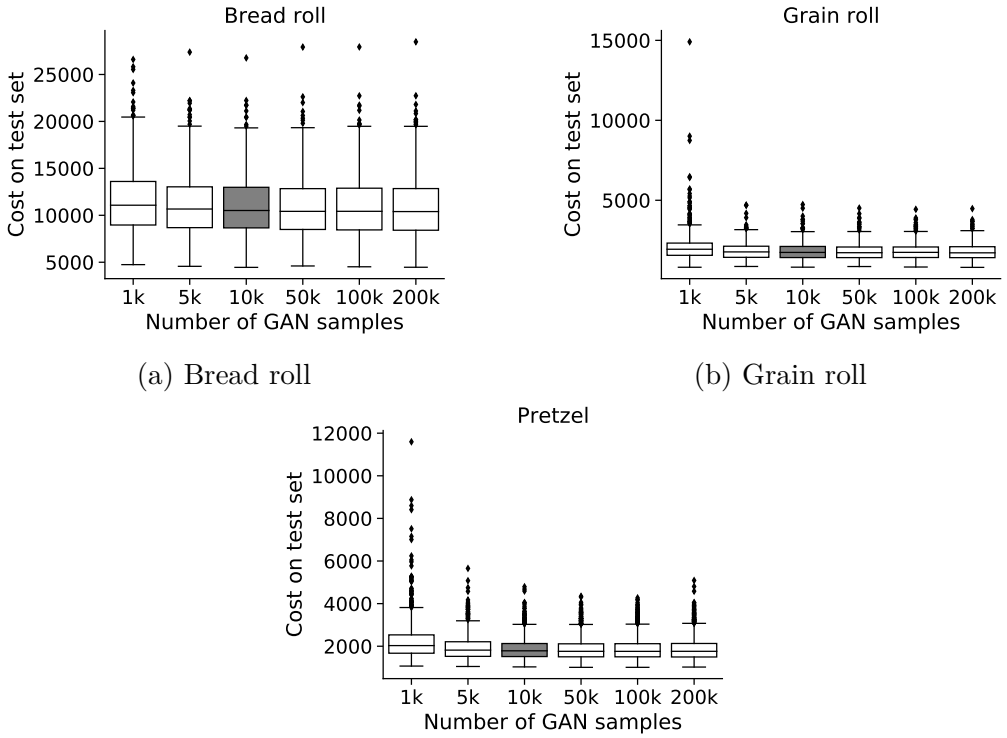
Figure 3.5: Data dimensions
Our training data can be reduced along 3 dimensions to evaluate the small data effect (Full period of data: 2016/03/02 - 2019/04/30)

product and bakery shop individually, we only consider the aggregated cost per product at the bakery chain level for our performance evaluation. This KPI is more relevant from a managerial point of view.

3.6 Experimental Results and Discussion

The objective of our experiments is to assess whether GAN-synthesized data is capable of guiding decision-making in an OM setting and how it behaves in even smaller data settings. We study this by varying different parameters, such as the service level α of our newsvendor problem or the size of our training data. For the latter, we study the reduction of training data along the three dimensions, time window t, bakery shop s, and product p (see Figure 3.5). Another challenge we faced when applying GANs in this applied setting was the question of the right number of samples necessary to achieve good performance in our decision task. We start our evaluation using the full-fledged training data set (two years, five bakery shops, three products each). The number of synthetic GAN samples not only improves the performance, but also increases the run time of our solution algorithm.

We found that in our setting the performance of the GAN models does not improve anymore once we sample more than 10,000 synthetic data samples (see Figure 3.6). We conclude that this represents a good empirical estimate of the implicit representation of our data distribution and will be used in further experiments.



(a) Bread roll

(b) Grain roll

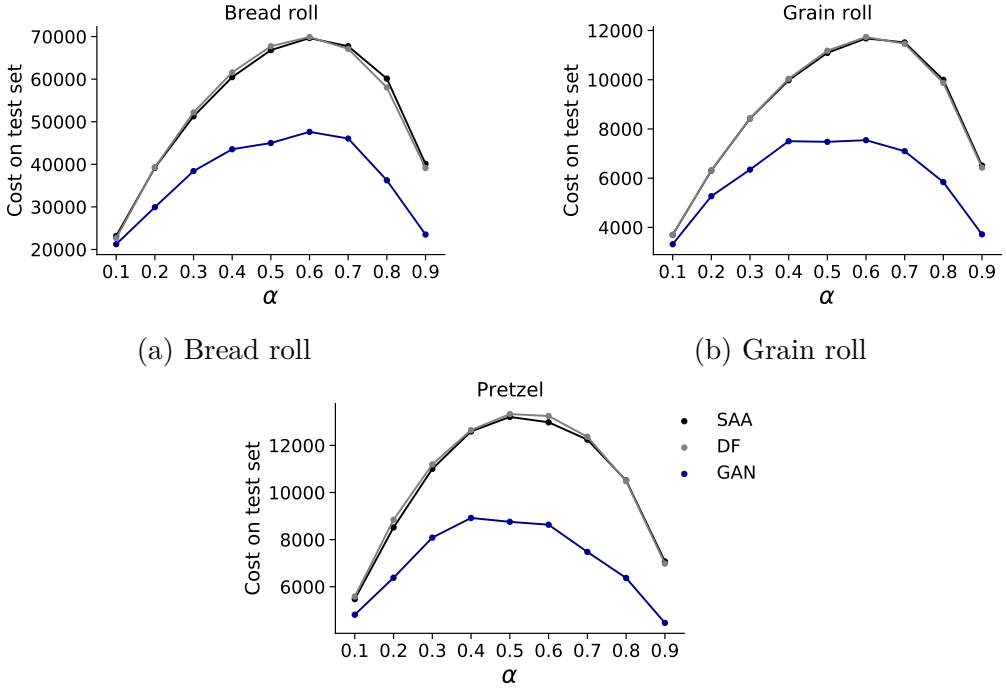
(c) Pretzel

Figure 3.6: Performance for different synthetic sample sizes

3.6.1 Implications of Service Level Variation

In the newsvendor problem, the service level has the greatest influence on cost performance as it determines how product underage and overage are priced. In a low service level setting, overage costs are high, which in turn implies that stock-outs are preferred over leftover products and vice versa in high service level settings. For a service level of 50%, the decision-maker is indifferent. Here, the quality of our demand forecast is of the greatest importance. Figure 3.7 shows that the GAN-based approach outperforms the distribution fitting and SAA benchmarks across

service levels. This indicates that GAN-based synthetic data generation can substantially improve decision-making. In a practical setting, a desired service level between 70-90% is most common. Therefore, we fix α to 80% for the following experiments.



(a) Bread roll

(b) Grain roll

(c) Pretzel

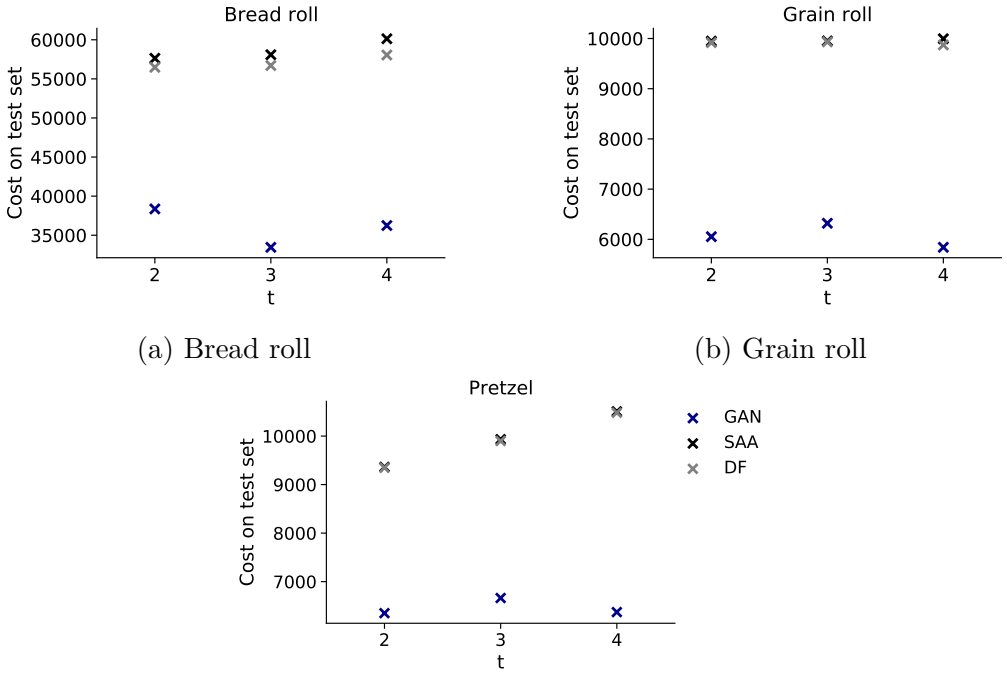
Figure 3.7: Performance per product on test set across service levels

3.6.2 Implications of Varying the Time Window

To assess the behavior of GAN-based synthetic data generation in a small data regime, we gradually reduce the number of training samples along different dimensions (see Figure 3.5).

We start by reducing the time axis t while keeping the number of shops $s = 5$ and products $p = 3$ fixed. In an OM setting, this dimension is often the main reason behind limited data availability, as most companies will have several products and even shops. We reduce the training data in two steps—from two years ($t = 4$) to one and a half years ($t = 3$) and finally to one year ($t = 2$). Further reduction is not possible because the GAN model can only learn the seasonal effects observed in the training data.

Therefore, the minimum data necessary to include the month feature is one year. Figure 3.8 shows that the relative order of the GAN-based approach and the benchmarks remains the same when changing the time axis. The GAN-based approach outperforms DF and SAA even in a small data setting with only a single year of training data. Furthermore, the GAN performance exhibits only minimal improvements as a result of the increase in data. This demonstrates that the method is useful even for short periods of training data.



(a) Bread roll

(b) Grain roll

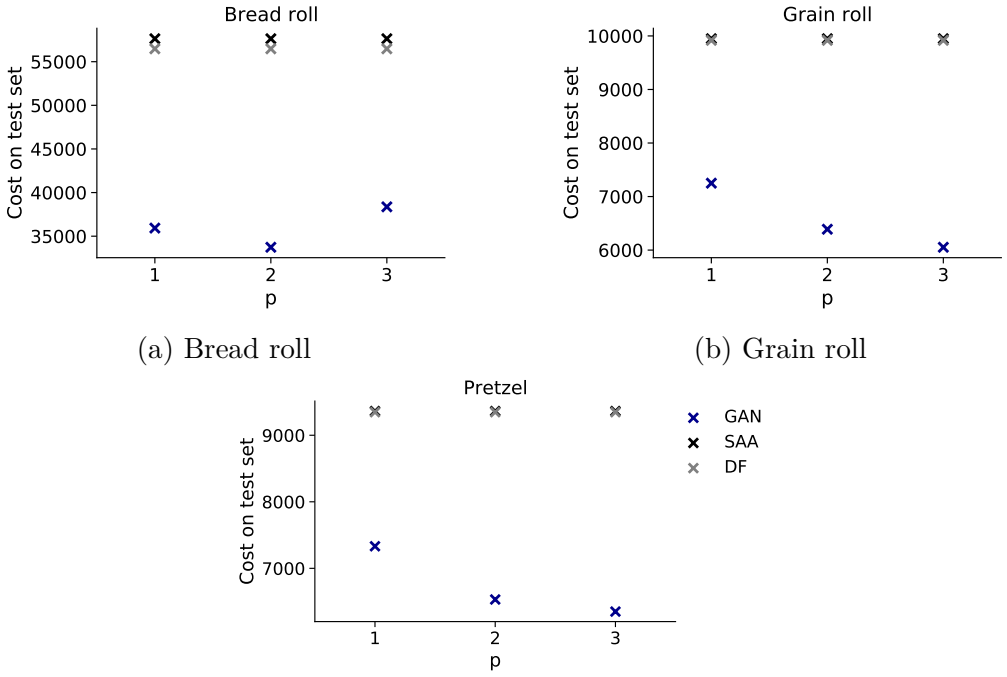
(c) Pretzel

Figure 3.8: Performance impact of varying t with p=3, s=5, α=0.8 fixed

3.6.3 Implications of Varying the Product Dimension

Most businesses do not sell a single product only but will be able to access data for multiple products, just like in our bakery setting where we consider three different products. In this section, we study the effect of varying the number of products p in our training data from three to two to one product while keeping $t = 2$ and $s = 5$ fixed. We find that again the relative order of the GAN-based approach and the two benchmark approaches

remains the same when changing the product axis (see Figure 3.9). When we take a closer look at the results of our GAN-based approach, we find that for two out of three products, the results of our GAN-based approach improve if we include more than one product in our training data. For the other product, the results only improve if we combine the data from two products in our training data. Performance deteriorates when adding a third product. We believe that this is due to the different demand scale of product 101. In general, the daily demand for product 101 is almost ten times the daily demand for product 109 or 110. In the case where our training data include all three products, our GAN might have learned a slight bias toward lower demand, leading to a cost increase. We conclude that the GAN-based approach is capable of using information from other products in decision-making, especially when their demand is on a similar scale.



(a) Bread roll

(b) Grain roll

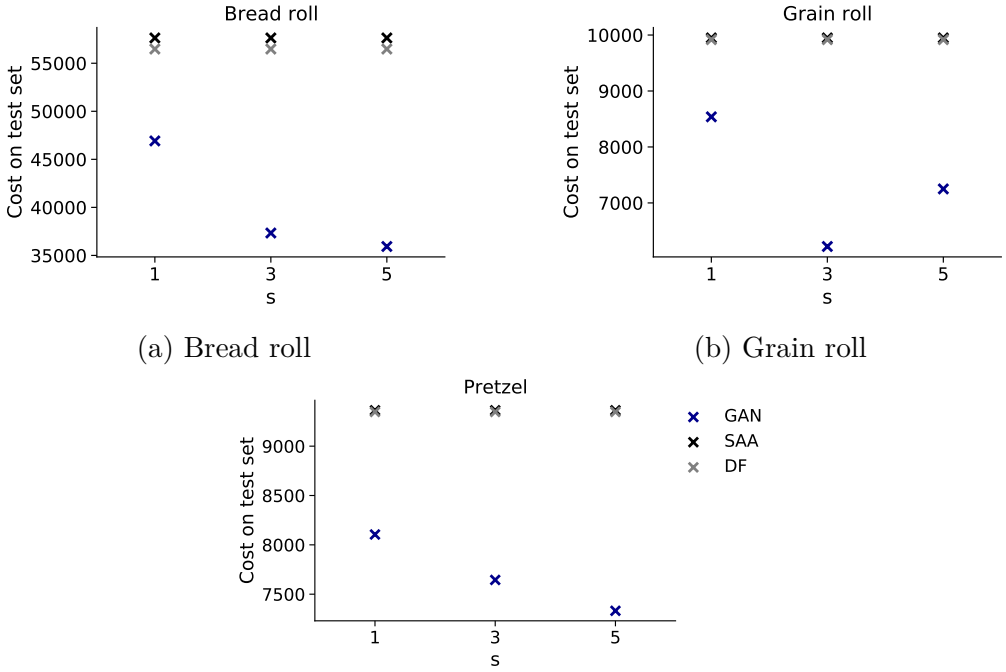
(c) Pretzel

Figure 3.9: Performance impact of varying p with t=2, s=5, α=0.8 fixed

3.6.4 Implications of Varying the Bakery Shop Dimension

Both from an academic and a practical standpoint, it is particularly interesting to evaluate whether decisions can be improved using data from other shops that sell the same products but might have a different demand pattern due to their location or customer base. To study this effect, we fix our training data to $t = 2$ and $p = 1$ and only vary the number of shops s that we include in our training data from five to three to one. We consider all possible permutations of shop combinations for the case of three shops.

Figure 3.10 shows that the relative order of the GAN-based approach and the two benchmarks remains the same when changing the shop axis. The DF approach stays constant as fitting happens at the product and shop level. Hence, the training data differs only along the time axis. For the GAN-based approach, learning across multiple shops turns out to be beneficial. Performance improves when more shops are used as training data.



(a) Bread roll

(b) Grain roll

(c) Pretzel

Figure 3.10: Performance impact of varying s with t=2, p=1, α=0.8 fixed

We conclude that our method is capable of extracting information patterns across shops that can support OM decision-making. This is good

news for multi-store chains, as it limits the cold start problem for AI-based decision-making in new stores.

In summary, all of our results confirm that the GAN-based approach for synthetic data generation is a promising alternative to traditional approaches such as DF and SAA.

3.7 Conclusion

Operations management has seen a recent influx of research on data-driven solutions for planning problems using machine learning. However, an often neglected aspect of this research is that most of the suggested approaches (e.g., reinforcement learning, deep learning) require large amounts of relevant planning data, which may not be readily available for decision-making problems due to a limited number of transactions or for reasons of confidentiality. To mitigate this problem, we propose to synthesize data samples using generative adversarial networks originating in the field of computer vision. Our approach leverages these networks to learn an implicit representation of the joint distribution of features and the variable of interest (e.g., demand). By sampling this joint distribution, we can use data-driven approaches to find robust solutions to complex planning problems. We illustrate and evaluate our model using a real-world data set from a bakery chain facing a classical inventory management problem. In particular, we solve the planning problem using synthetic data samples provided by a GAN trained on the real data. By means of extensive numerical evaluations, we show that our approach is capable of handling high-dimensional data and incorporating rich feature variables into the decision. This observation renders it a promising alternative to existing methods such as DF and SAA. However, replacing existing methods with GANs comes at the cost of computational complexity. While existing approaches can be performed efficiently, training and selecting a good performing GAN is not a straightforward process. We have outlined best practices to overcome the instability issues associated with GAN training and show how to select stable models that perform well.

Our research has proposed a novel usage of generative adversarial networks for operations management problems. Consequently, there are several opportunities for future research. We decided to focus on a relatively

simple planning problem (newsvendor problem) to understand how GANs should be trained and selected and how much data is needed to outperform traditional models. In the future, the proposed approach should be evaluated in more complex settings, such as multi-period inventory planning or pricing problems. Furthermore, the numerical evaluation should be extended to multiple data sets and feature settings. Furthermore, more research is needed on stabilizing GAN training to allow for a more straightforward selection procedure while simultaneously improving the robustness of the suggested approach.

4 Combining Synthetic Data and Transfer Learning for Deep Reinforcement Learning in Inventory Management

Deep reinforcement learning (DRL) is a machine learning technique that has been shown to be successful in resolving challenging decision-making problems. However, the application in practice, in particular in operations management (OM) contexts, is often hindered by a lack of large amounts of training data. In this work, we propose a transfer learning-based approach for solving OM planning problems that combines synthetically generated and real data to train DRL agents. In particular, our method pre-trains a DRL agent using synthetic data produced by a deep generative model, in this case a generative adversarial network (GAN). The agent is then fine-tuned using a small real data sample. In this way, in contrast to existing DRL-based approaches for OM, we are able to include feature information in the planning decision. We perform a numerical evaluation of our proposed approach using a classical single-period inventory management problem in two different settings. We obtain promising first results for our proposed procedure.[13]

4.1 Introduction

Over the last decades, machine learning and especially the subfield of deep reinforcement learning (DRL) have shown great successes in solving complex

[13]This paper is co-authored by Nikolai Stein and Richard Pibernik.

problems ranging from learning to play games (most prominent examples include chess and Go) to real-world applications (e.g., autonomous driving). Inspired by the successes in the field of machine learning, operations management (OM) researchers have started to use machine learning techniques to develop new data-driven approaches for decision-making under uncertainty. First approaches apply machine learning mainly to forecast the behavior of uncertain quantities of interest (e.g., demand for different products) based on additional information—so-called features. In a two-step "predict-then-optimize" approach, the forecast is then used to solve the planning problem with traditional optimization techniques. A different stream of research that has gained growing attention aims to resolve this disconnection between forecasting and optimization by developing prescriptive analytics approaches. These approaches take advantage of machine learning to prescribe decisions directly from available features. This incorporation of features is a key benefit of these approaches. A downside of prescriptive analytics approaches is that they often are highly problem-specific, e.g., they require an explicit formalization of decision policies or the optimization problem to be solved. For more complex problem settings, such as multi-period planning, this can become infeasible, especially if features are involved.

Especially, the strong interconnection of OM research with practice created the need to investigate technologies that offer more flexibility and can be customized more easily depending on the industry context. Recently, OM researchers have started to apply a method from machine learning that has been shown to be suitable for solving complex decision-making tasks: deep reinforcement learning (DRL). In contrast to existing approaches, the application of DRL does not require modeling policies for specific problem settings (with underlying assumptions), but it requires modeling environments. This is much closer to real-world settings, where environments (auxiliary information + outcome) are modeled in standard software (e.g., ERP systems). Several OM researchers have shown that DRL is versatile and can be applied to a number of problem settings (see Section 4.2). However, there are some limitations to their work. Current work assumes that the environment in which the DRL agent acts is known - this includes the distribution of the random variable of interest. This is problematic because "in many practical settings the distribution of the uncertainty is unknown and

must be derived from historical samples." (Gijsbrechts et al., 2021). Likely, this is especially problematic if we want to incorporate feature information (like prescriptive analytics approaches have done). We have already seen this problem for traditional prescriptive analytics approaches—if the number of samples n in D_{real} goes to infinity Bertsimas and Kallus (2020) and Notz and Pibernik (2022) have proposed prescriptive analytics approaches that are asymptotically optimal. However, there are no performance guarantees for $n << \infty$. Notz and Pibernik (2022) further show that the convergence rate decreases in the number of distinct features incorporated in the data.

Therefore, our work explores how features can be incorporated into DRL in settings with limited training data available. We investigate how the need for large amounts of real training data samples can be alleviated by enriching the small training data set with synthetic data samples. These synthetic samples are derived from a data-generating model whose goal is to learn the true underlying distribution of the real data. In this way, in any given environment we do not need assumptions of (parametric) distributions but learn the conditional distribution of uncertain quantities of interest (e.g., demand, lead time, prices, etc.) directly from data.

To formally describe our problem, let X and Y be random variables from which the observed features $x \in \mathcal{X}$ and the variables of interest $y \in \mathcal{Y}$ stem. We want to learn a synthetic data-generating model g_θ with parameters θ. To sample from the data-generating model we require a random variable $Z \in \mathcal{Z}$ with known prior distribution p_Z such that we can define g_θ as a function $g_\theta : \mathcal{Z} \to \mathcal{X} \times \mathcal{Y}$ that takes instances from Z as inputs and generates samples (x, y) from our data space $\mathcal{X} \times \mathcal{Y}$ as outputs. Our goal is to learn a synthetic data-generating model g_θ such that for every $(x, y) \in \mathcal{X} \times \mathcal{Y}$

$$p_\theta(x, y) \approx p_{X \times Y}(x, y) \tag{4.1}$$

where p_θ is the probability distribution of the samples generated from g_θ. We will discuss different approaches to modeling g_θ in Section 4.3.1. To find the optimal g_θ^* we solve the following learning problem:

$$g_\theta^* \in \arg\min_{g_\theta \in \mathcal{G}} \mathcal{D}\left(p_\theta,\, p_{X \times Y}\right) \tag{4.2}$$

with $\mathcal{D}$ a measure of distance of the distribution p_θ to the original distribution

$p_{X\times Y}$. As $p_{X\times Y}$ is unknown, different empirical approximations for (4.2) are used to find the optimal g_θ^*. We discuss this in more detail in Section 4.3.1. We explain how we train a DRL agent using synthetic data in Section 4.3.2. In Section 4.4 we present two numerical studies that we conducted following our proposed approach—one in which the data-generating process is known and one in which it is unknown. We conclude by summarizing our findings and pointing out avenues for future research in Section 4.5.

4.2 Related Work in Operations Management

As highlighted in the introduction, our work is predominantly related to two streams of research in OM: Prescriptive analytics and deep reinforcement learning. Related work on generative models is introduced in Section 4.3.1.

4.2.1 Prescriptive Analytics

In recent years, many prescriptive analytics approaches have been developed that, instead of making assumptions about demand distributions, work with real data from practice directly and, most importantly, include feature information to prescribe a decision. These prescriptive analytics approaches seek to prescribe decisions for uncertain quantities of interest that we model as a random variable $Y \in \mathcal{Y} \subset \mathbb{R}^{I_y}$ using a decision function that maps from the feature space $\mathcal{X} \subset \mathbb{R}^{I_x}$ to the decision space $\mathcal{Q} \subset \mathbb{R}^{I_q}$ and minimizes some cost function $C(q(X), Y)$. Often Y represents demand for different products and X contains additional information that can be temporal (weekday, month, year), lagged (historic demand observations), weather-related (temperature, hours of sunshine), or anything else that potentially adds to the description of Y's behavior. The learning problem to derive the optimal decision function can be formulated as follows:

$$\min_{q\in\mathcal{F}} \mathbb{E}_{X\times Y}\left[C(q(X), Y)\right]. \tag{4.3}$$

We focus this literature review on approaches for inventory management to reduce complexity, although the ideas in this paper can be applied to operations management problems in general. Bertsimas and Kallus coined the term "prescriptive analytics" with their seminal work "From

predictive to prescriptive analytics" (Bertsimas and Kallus, 2020). The classical single-period newsvendor problem provided a good starting point for the development and evaluation of such approaches. We provide selected examples in the following. Bertsimas and Kallus (2020) propose a reweighing of past observations to solve a stochastic optimization problem and showcase their approach for a replenishment problem related to a classical newsvendor setting where the inherent optimization problem can be solved efficiently. Here, the optimization step has to be repeated for each new decision to be taken. Other proposed approaches aim at finding a continuous function that prescribes a decision from a feature vector without the need for re-optimization and hence reduces inference time drastically. In the case of linear function spaces, Beutel and Minner (2012), as well as Ban and Rudin (2019), use linear programming models to learn a decision function. Oroojlooyjadid, Snyder, et al. (2020) use deep neural networks to define the decision function for non-linear function spaces. The approaches developed so far are highly tailored to the problem at hand and become inherently complex as the complexity of the problem to be solved increases. Qi et al. (2020) are the first to study prescriptive analytics in a more complex inventory management setting. They aim to solve a multi-period inventory problem with stochastic demand and vendor lead time. Similarly to Oroojlooyjadid, Snyder, et al. (2020), they use a deep neural network to predict optimal order quantities for each period in the planning horizon considered. This reduces complexity in the inference/ decision-making step. However, during training, their approach relies on the ability to compute ex post optimal order quantities efficiently for a given training set. For many different variants of multi-period inventory problems, a closed-form decision policy is unknown (Boute et al., 2021) and, therefore, they are traditionally solved by dynamic programming approaches, which become computationally intractable especially if we consider feature information (see Levi, Pál, et al. (2007) for a more detailed discussion). All of the approaches discussed have in common that they require a very precise problem and policy definition. This does not generalize well to other problem instances and even other data sets. Therefore, it is important to explore approaches that promise more flexibility in terms of addressing different classes of problems.

4.2.2 Deep Reinforcement Learning

Deep reinforcement learning (DRL) offers a different approach to learning complex decision policies. It has achieved great success, for example, from learning to play Atari games (Mnih et al., 2013) to beating human players in complex games such as chess and Go (Baxter et al., 2000; Silver et al., 2017). DRL is inspired by the way humans learn. In many situations, we are not told how to behave, but learn from trial and error. Similarly, in DRL an agent learns to take actions, or in our case, make decisions by observing its environment and acting accordingly. At each time step t, the agent observes a state s_t that can include not only our features X, but also additional information such as, for example, inventory levels or order pipelines. Based on s_t, the agent takes an action a_t that results in taking the agent to the next state s_{t+1}. The actions the agent takes are guided by a reward signal. The agent's goal is to maximize the reward. When choosing actions, the agent has to balance exploration and exploitation (Sutton and Barto, 2018). The agent tends to select actions that led to high rewards in the past, but it also has to explore alternative actions to learn more about the environment. The behavior of an agent is defined by a policy that tells the agent which action to select given a state. This is similar to the decision function $q(\cdot)$ that we want to learn for our OM problem. Therefore, we will use the same notation in the following. A huge benefit of DRL is that we need to specify neither the optimization problem nor a policy explicitly. The DRL agent learns through interaction with a problem environment that can be defined more liberally. Recently, deep reinforcement learning has received more and more attention from the operations management community. To our knowledge, Gijsbrechts et al. (2021) were the first to apply DRL in an operations management context. They apply DRL to three different operations problems: lost sales inventory, dual sourcing, and multi-echelon inventory. However, in contrast to the previously discussed prescriptive analytics approaches, they do not study real data for the lost sales and multi-echelon inventory problem. The practical case study for the dual-sourcing problem is only found in the appendix. In addition, Gijsbrechts et al. (2021) do not yet include feature information. Others have followed. Balaji et al. (2019) use deep reinforcement learning to solve a multi-period inventory

problem with deterministic lead time and stochastic demand drawn from a Poisson distribution. Gokhale et al. (2021) focus on a multi-warehouse multi-product replenishment problem with deterministic lead times and stochastic demand drawn from Poisson, normal, and gamma distributions. Moor et al. (2021) solve a periodic review multi-period inventory problem with deterministic lead time and stochastic demand drawn from a gamma distribution. They apply reward shaping to transfer knowledge from existing inventory policies to the deep reinforcement learning agent. Oroojlooyjadid, Nazari, et al. (2021) use deep Q-learning to play the beer game where demand is drawn from uniform as well as normal distributions. They also study real-world data sets. Here, they use the empirical distribution of the real data in order to be able to sample infinitely. They also include a small experiment on the effect of training sample size on the performance of their approach. They construct a "low-observation" setting by training on an empirical distribution constituted by only 100 samples from a uniform as well as a normal distribution in contrast to a "full-observation" setting where during training they sample directly from a uniform or normal distribution. They find that "the performance of two models [low-observation and full-observation] is quite similar". All of the DRL research discussed has in common that it focuses on training based on a known distribution, which is typically not transferable to practical settings. Real-world experiments are yet scarce, and if they are available, they do not include feature information, which has proven to be beneficial if we look at recently developed prescriptive analytics approaches. Boute et al. (2021) also call for their inclusion in the state space of DRL. However, incorporating features requires knowledge of the joint distribution of demand and features, or probably even more training real data. In practice, we do not know the joint distribution nor have access to large amounts of training data. Therefore, our research aims at closing this gap by proposing an approach that includes a synthetic data model for enriching the DRL agent's training process.

4.3 Approach

This section introduces our approach to train a flexible model for OM decision-making that is able to include feature information using synthetic

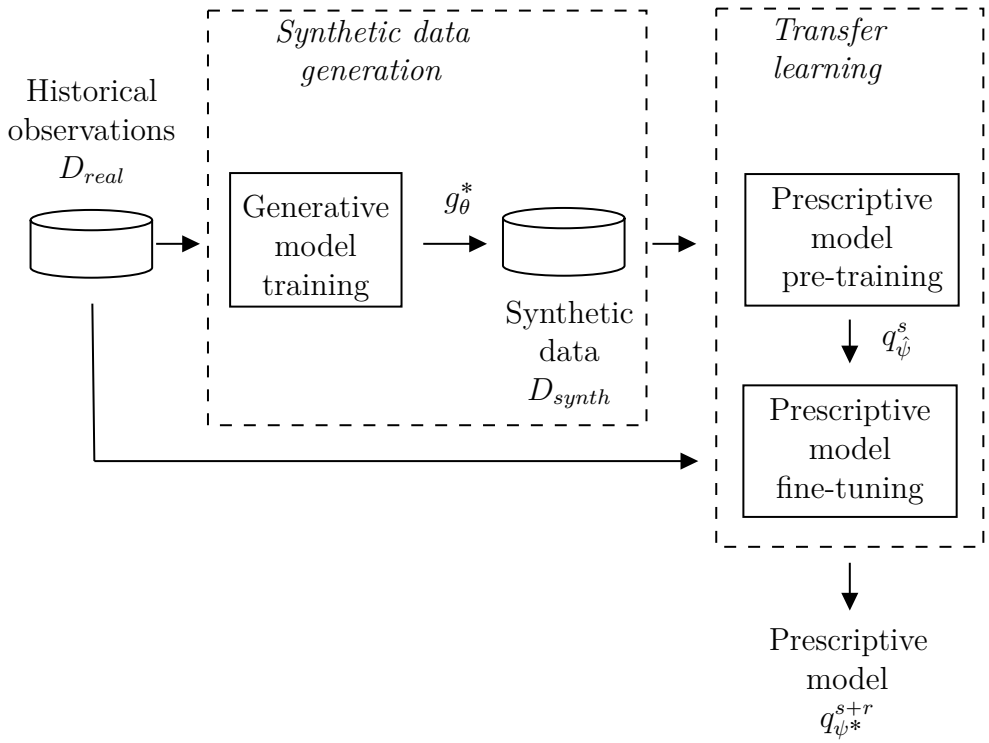


Figure 4.1: Leveraging synthetic data for decision-making

data. Our approach consists of two main elements (also see Figure 4.1): Synthetic data generation and a prescriptive model that utilizes deep reinforcement learning (DRL). In the synthetic data generation step, we learn a generative model g_θ with parameters θ that can be used to create a large set of synthetic data samples $D_{synth} = (X_{synth}, Y_{synth}) \in \mathcal{X}_{synth} \times \mathcal{Y}_{synth}$. In the next step, we utilize a transfer learning approach to arrive at a prescriptive model $q_{\psi^*}^{s+r}$ that leverages both synthetic and real data.

4.3.1 Synthetic Data Generation

For synthetic data generation, given a set of historical observations $D_{real} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ with $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ and $n < \infty$, our goal is to learn a generative model g_θ with parameters θ such that for every $(x, y) \in \mathcal{X} \times \mathcal{Y}$

$$p_\theta(x, y) \sim p_{X \times Y}(x, y) \tag{4.4}$$

where p_θ is the probability distribution of the samples generated from g_θ. Once we have the data-generating model, we can sample a synthetic data set D_{synth} of arbitrary size m from it. To find the optimal $g_{\theta*}$, one would like to solve the following learning problem:

$$g_{\theta*} \in \arg\min_{g_\theta \in \mathcal{G}} \mathcal{L}_{gen}\left(p_{X\times Y},\, p_\theta\right) \tag{4.5}$$

with $\mathcal{L}_{gen}$ a measure of the distance between the distribution p_θ and the original distribution $p_{X\times Y}$. As $p_{X\times Y}$ is unknown, different empirical approximations for (4.5) are used to optimize g_θ. A common choice to measure the distance is the Kullback-Leibler divergence $\mathcal{L}_{KL}$ with

$$\mathcal{L}_{KL}(p_{X\times Y} \parallel p_\theta) = \mathbb{E}_{(x,y)\sim p_{X\times Y}}[\log p_{X\times Y}\left((x,y)\right) - \log p_\theta\left((x,y)\right)]. \tag{4.6}$$

Optimizing the model parameters θ to minimize $\mathcal{L}_{KL}$ is equivalent to maximizing the log-likelihood that the original data D_{real} was generated by g_θ (Goodfellow, 2016), so that:

$$\theta^* \in \arg\max_{\theta} \mathbb{E}_{(x,y)\sim p_{X\times Y}}[\log p_\theta\left((x,y)\right)]. \tag{4.7}$$

There are multiple ways to solve the learning problem (4.5). Traditionally, OM did not consider features when solving planning problems and, absent of features, fitted some theoretical distribution to historical observations of the uncertain variable of interest Y (e.g., demand). However, the application of such methods is limited if features are involved. In most OM problems, we are concerned with real-world data, where we do not know the effect that X has on Y a priori. Features can interact in non-trivial ways not only with the uncertain variable of interest but also among one another. Furthermore, the features X often contain a mixture of continuous and discrete features. This makes it challenging to define a parametric multivariate distribution that models and learns this behavior. Therefore, we do not want to consider approaches that make assumptions about the underlying multivariate distribution $p_{X\times Y}$, but rather we rely on deep generative models for this task. Deep generative models use neural networks, which have been shown to be capable of learning complex

relationships. Most of the research, development, and evaluation of deep generative models focused on the generation of images (Alaa et al., 2022) that pertain to data that is different from OM-related data. Image data is made up of pixels that have continuous values. In addition, there is positional information that can be exploited; e.g., pixels that are located next to each other will assume similar values in order to visualize different geometric forms. Typical OM data come in the form of tabular data, that is, data that can be displayed as a table where the features are arranged as the columns of the table and each line contains a specific instance of feature information. In contrast to image data, it contains a mixture of continuous and discrete values. In addition, lines close to each other in the table can, but do not have to be related. In the same way, the order of columns in the table is arbitrary. Therefore, we require specific generative models that are tailored to these needs. Generative models can be broadly classified into two categories (Goodfellow, 2016): models that define an explicit density function p_θ vs. models that define p_θ implicitly. Many different models have been proposed in both categories. The two models that have attracted the most attention in recent years are variational autoencoders (VAE), which fall into the category of explicit models, and generative adversarial networks (GAN), which implicitly define the underlying distribution of g_θ. Both models have been appreciated for their flexible design (GM et al., 2020) and dedicated architectures have been proposed for OM-type data (time series and tabular data) (Esteban et al., 2017; Xu, Skoularidou, et al., 2019; Piatkowski et al., 2021). We will explain both models below.

Variational Autoencoders

Variational autoencoders (VAEs) have first been proposed by Kingma and Welling (2013) and Rezende et al. (2014). They use an encoding-decoding scheme to learn a generative model. The architecture of a VAE, in general, consists of two blocks—an encoder and a decoder—which both can be modeled as neural networks. Given D_{real} with underlying distribution $p_{X\times Y}$ and a random latent variable Z with a prior distribution, the encoder's goal is to learn the variational distribution $p_\phi(z|(x,y))$ by optimizing the parameters ϕ. The goal of the decoder D is to learn the distribution

$p_\theta((x,y)|z)$ by optimizing the parameters θ. This makes the decoder suitable as a generative model $g_{\theta*}$. As the prior distribution of Z is known, we can sample from it and pass those samples through the decoder to generate new data samples from $p_{X\times Y}$. It is common to choose $p(z) = \mathcal{N}(z, \mu, \sigma^2)$ as a Gaussian distribution (Kingma and Welling, 2013) so that the encoder only has to learn to predict mean and variance. By introducing an auxiliary variable $\epsilon \in \mathcal{N}(0, \mathbb{I})$, the reparameterization trick can be used to sample $z = \mu_\phi + \sigma_\phi \epsilon$. This enables optimizing parameters of the encoder and decoder jointly during training of the VAE via gradient descent (Piatkowski et al., 2021). VAEs are trained by maximizing the evidence lower bound (ELBO):

$$
\begin{aligned}
log\ p_\theta((x,y)) &\geqslant ELBO(\phi, \theta, (x,y)) \\
&= \mathbb{E}_{z\sim p_\phi(z|(x,y))} \left[\underbrace{log\ p_\theta((x,y)|z)}_{Reconstruction\ error} - \underbrace{\mathcal{L}_{KL}\big(p_\phi(z|(x,y) \parallel p_\theta(z)\big)}_{Regularization} \right]
\end{aligned}
$$

where the first term is the expectation of the reconstruction error that ensures that the original and decoded (x, y) are close and the Kullback-Leibler divergence $\mathcal{L}_{KL}$ that measures the difference between the variational distribution $p_\psi(z|(x,y))$ and the prior distribution $p(z)$. The latter serves as a regularizer. We refer to Kingma (2019) for further technical details.

Generative Adversarial Networks

Generative adversarial networks (GANs) belong to the division of models that implicitly define the density function. They consist of two opposing networks that play a minimax game with the objective of optimizing a joint loss function. The generator $G : \mathcal{Z} \longrightarrow \mathcal{X} \times \mathcal{Y}$ acts as a data-generating model that takes as input some noise vector z with distribution $p(z)$ from $\mathcal{Z}$ and maps it to a new data sample x from the data space $\mathcal{X} \times \mathcal{Y}$. Its goal is to produce samples that are as realistic as possible. In fact, they need to be so realistic that they can fool the second network—the discriminator $D : \mathcal{X} \times \mathcal{Y} \longrightarrow \{0, 1\}$—cannot distinguish them from samples from $p_{X\times Y}$. The discriminator's goal is to label samples that come from the real data set

as real and samples that are produced by the generator as fake. It produces the probability of a sample that originates from $p_{X\times Y}$. It is common to model G and D as neural networks with network parameters θ for the generator and parameters ϕ for the discriminator that are optimized during training by playing a minimax game with joint loss function $V(G_\theta, D_\phi)$:

$$\begin{aligned}\min_\theta \max_\phi V(G_\theta, D_\phi) = \mathbb{E}_{(x,y)\sim p_{X\times Y}((x,y))}[log\ D_\phi((x,y))] \\ + \mathbb{E}_{z\sim p(z)}[log(1 - D_\phi(G_\theta(z)))].\end{aligned} \tag{4.8}$$

From the discriminator's perspective, samples coming from $p_{X\times Y}$ should be assigned a probability of 1 and samples coming from p_θ a probability of 0. This will maximize $V(G_\theta, D_\phi)$ to a value of 0. The generator, on the other hand, wants to minimize $V(G_\theta, D_\phi)$. Therefore, from the generator's perspective, samples coming from p_θ should be assigned a probability of 1 by D_ϕ driving the second part of the equation towards $-\infty$.

Given an optimal discriminator $D_{\phi*}$, Goodfellow, Pouget-Abadie, et al. (2014) show that Equation 4.8 reduces to

$$\begin{aligned}\mathcal{L}_{gen} &:= \min_\theta V(G_\theta, D_{\phi*}) \\ &= -log(4) + \mathcal{L}_{KL}\left(p_{X\times Y} \,\middle\|\, \frac{p_{X\times Y} + p_\theta}{2}\right) + \mathcal{L}_{KL}\left(p_\theta \,\middle\|\, \frac{p_{X\times Y} + p_\theta}{2}\right),\end{aligned}$$

which reaches its global optimum if and only if $p_\theta = p_{X\times Y}$—that is, the generator perfectly learned the true joint distribution $p_{X\times Y}$ and thus serves as a generative model $g_{\theta*}$.

To the best of our knowledge, research so far has not provided substantial evidence on how best to model OM-type data. VAE use cases for tabular data focus on learning the underlying data distribution to impute missing data or detect anomalies in the data, while several GAN architectures have been developed specifically to model tabular data in order to generate synthetic data samples (Che et al., 2017; Xu, Skoularidou, et al., 2019; Yahi et al., 2017). From these especially, the conditional tabular GAN (CTGAN) architecture developed by Xu, Skoularidou, et al. (2019) stands out because it is highly flexible (being able to model continuous and categorical features)

and mitigates issues due to imbalance in training data by a conditional "training-by-sampling" approach. For our numerical analysis, we adopted the CTGAN architecture. However, we cannot rule out that there are more suitable generative models for our purpose.

Hyperparameter Tuning and Selection of Generative Models

The generative models proposed in the previous section require extensive hyperparameter tuning. Therefore, we need a metric that allows us to tune individual models and to compare the performance of alternative models. Unfortunately, we cannot use the log-likelihood (or the Kullback-Leibler divergence) in Equation 4.7, because it is either computationally intractable (in the case of VAEs, Jordon et al. (2022)) or not defined explicitly (in the case of GANs, Goodfellow (2016)). Instead, we need a metric to evaluate the quality of a sample $D_{synth} = \{g_1, \ldots, g_m\}$ generated by a generative model. Theis et al. (2016) as well as Piatkowski et al. (2021) evaluate different metrics, e.g., Parzen window estimates, nearest neighbors, maximum mean discrepancy, or Hausdorff discrepancy. Both come to the conclusion that the choice of metric has an influence on the kind of model that is selected. For example, in Piatkowski et al. (2021)'s numerical experiments, the quality of GAN-generated samples is higher than that of VAE-generated samples when evaluated by the maximum mean discrepancy, while it is the opposite way around when considering the Hausdorff discrepancy. Theis et al. (2016) stress that "[...] for generative models there is no one-fits-all loss function, but a proper assessment of model performance is only possible in the context of an application"(p.8). In our case, the goal of our synthetic data is to improve decision-making. A perfect generative model would lead to optimal prescriptions in Equation 4.3 because we have methods to solve Equation 4.3 that are asymptotically optimal (Bertsimas and Kallus, 2020; Notz and Pibernik, 2022). The greater the dissimilarity between distributions, the greater the cost increase. Therefore, the costs incurred in Equation 4.3 can serve as a quality measure for D_{synth}. However, depending on the problem setting, finding q^* can be computationally expensive, e.g., if it involves solving mixed-integer programs in different feature settings. This is especially problematic during hyperparameter tuning, where we have to

evaluate a large number of models. Therefore, we introduce a surrogate that is cheaper to compute—the pinball loss. For solving most OM problems, we optimize over some conditional density distribution. The pinball loss evaluates quantile forecasts and, therefore, transforms differences in the conditional distributions into costs. In general, the quantile is induced by the cost ratio between overestimating and underestimating demand in the OM problem. Therefore, to evaluate a synthetic sample D_{synth} we learn a surrogate function q^{sur} that, given a quantile u, maps from a feature space $\mathcal{X}$ to a quantile forecast $q^{sur}(x, u)$. We adapt the scaled pinball loss that was used as a central metric in the M5 "Uncertainty" competition (Makridakis et al., 2021) to evaluate the quantile forecast. We do not require scaling because we do not need to compare multiple time series against each other. Therefore, we only used the mean pinball loss and evaluated this on a holdout validation set $D^{val}_{real} = \{(x_1, y_1), \ldots, (x_v, y_v)\}$ of real data samples. Given a quantile u and a surrogate model $q^{sur}(x, u)$, we define the mean pinball loss as

$$meanPL(q^{sur}, u) = \frac{1}{v}\sum_{i=1}^{v} u(y_i - q^{sur}(x_i, u))^+ + (1-u)(q^{sur}(x_i, u) - y_i)^+ \quad (4.9)$$

where $(\cdot)^+ = \max(0, \cdot)$. The pinball loss is often also termed newsvendor loss.

In our numerical experiments, we investigate the question of how important the quality of our synthetic data D_{synth} is. It is conceivable that even poor data quality will suffice to learn the general problem structure, which can then enable us to use such a pre-trained model in combination with limited training data to achieve good results in real-world settings. We will further detail this in the next section.

4.3.2 Transfer Learning

The generative model provides us with the means to generate a large amount of synthetic data that can be used to train a DRL agent. However, we expect that $p_{\theta*}$ deviates from the true distribution $p_{X\times Y}$, and that this deviation has a negative effect on the prescriptive quality of our model. We term this

effect the "synthetic data gap". We propose using transfer learning to reduce this gap. The goal of transfer learning is to transfer knowledge gathered by performing a source task in a source domain to a target task in a related target domain. This is often done when the available data in the target domain is limited (Weiss et al., 2016). In our case, the target task represents the company-specific OM problem for which the decision-maker intends to obtain a solution, e.g., an inventory or management problem, and for which limited data, in the form of historical observations, is available in the target domain. Instead of solving the target task in the target domain—as done by traditional prescriptive analytics approaches, we propose to first solve the same task in a source domain, which is given by the synthetic data, and to then utilize the "knowledge", gathered in the source domain to better solve the (real) problem in the target domain. This form of transfer learning, where the source task and the target task are the same, but the source domain and the target domain differ, is referred to as transductive transfer learning (Csurka, 2017). More specifically, we first solve problem (4.3) in the source domain which is the synthetic data domain described by $p_{\theta*}$. We seek the parameters $\hat{\psi}$ such that

$$\hat{\psi} \in \arg\min_{\psi} \mathbb{E}_{(x_s,y_s)\sim p_{\theta*}}[C(q^s_{\psi}(x_s), y_s)]. \tag{4.10}$$

We learn $q^s_{\hat{\psi}}(\cdot)$ using deep reinforcement learning (DRL)—here $q^s_{\hat{\psi}}(\cdot)$ is referred to as a policy that an agent learns and ψ are the weights of the underlying neural network. The "knowledge gathered in the source domain" is reflected in the weights $\hat{\psi}$. Next, we want to transfer this knowledge to the target domain—in our case the real data domain, where we have a small sample D_{real} from $p_{X\times Y}$. Therefore, our goal is to solve problem (4.3) leveraging what we have learned from solving (4.10) to alleviate the synthetic data gap caused by the deviation between $p_{\theta*}$ and $p_{X\times Y}$. For this, we fine-tune $q^s_{\hat{\psi}}(\cdot)$ based on D_{real} to derive the final model $q^{s+r}_{\psi*}(\cdot)$. More formally, we search for the following:

$$\psi^* \in \arg\min_{\psi} \mathbb{E}_{(x,y)\in D_{real}}[C(q^{s+r}_{\psi}(x), y)] + \lambda(\psi, \hat{\psi}). \tag{4.11}$$

We initialize ψ with $\hat{\psi}$. To retain the knowledge gained in the source do-

main (by solving Equation 4.10), we regularize the training on the real data by $\lambda(\psi, \hat{\psi})$. This regularization term controls how strongly the knowledge gained in the source domain, i.e. $\hat{\psi}$, affects the parameters ψ^* of the final model. Depending on the selected model and training strategy, one can choose from a multitude of regularization approaches, for example, limiting the number of training iterations, limiting the learning rate, or freezing layers of the underlying deep neural network. In our approach, $\lambda(\psi, \hat{\psi})$ is a hyperparameter that must be tuned on the validation data during the training process. We detail this in Section 4.4.1.

4.3.3 Evaluation

From a practical point of view, the evaluation of our approach appears to be relatively straightforward. The decision-maker would like to know how our approach performs relative to a benchmark prescriptive model q^* applied to the real data D_{real}. Ideally, q^* is the optimal prescriptive model for a known $p_{X \times Y}$. However, in a practice setting, we often cannot compute the optimal model because $p_{X \times Y}$ is unknown. In this case q^* can represent some other benchmark model, e.g. the best-known prescriptive model, depending on the available data. Clearly, in many cases finding the best-known prescriptive model already poses a challenge, because, as stated before, the models' performances are data-dependent and we do not know a priori which prescriptive model performs best on a particular data set D_{real}. Therefore, identifying the best model to compute the expected costs of the sample $\overline{C}(q^*)$ may require a lot of effort for implementation, training, and testing. More formally, we denote the performance difference between $q^{s+r}_{\psi^*}$ and q^* by $\Delta_{overall}$ and define it as

$$\Delta_{overall} = \frac{\overline{C}(q^{s+r}_{\psi^*}) - \overline{C}(q^*)}{\overline{C}(q^*)}, \tag{4.12}$$

where $\overline{C}(q^{s+r}_{\psi^*})$ denotes the expected out-of-sample costs when applying the model learned in (4.11), and $\overline{C}(q^*)$ the expected out-of-sample costs associated with the benchmark model.

The value of $\Delta_{overall}$ is impacted by the performance of the generative model and the models used to solve (4.10) and (4.11)—both determine

$\overline{C}(q^{s+r}_{\psi*})$. We would like to divide $\Delta_{overall}$ into individual components that allow us to better understand the drivers that affect $\Delta_{overall}$. In addition, we would like to evaluate whether our $q^{s+r}_{\psi*}$ can mitigate what we term the "small data gap" that arises because we train a prescriptive model using only limited data.

We first turn to the impact of the generative model g_θ that is used to generate the synthetic data D_{synth}. We would like to measure what we termed the "synthetic data gap" in the previous section. A perfect generative model g^*_θ (with $p_\theta(x, y) = p_{X\times Y}(x, y)$) will lead to a zero synthetic data gap. With g^*_θ we can obtain $N \to \infty$ samples from $p_{X\times Y}$ and solve the underlying OM problem to optimality if we apply an asymptotically optimal prescriptive analytics model as proposed, for example, by Bertsimas and Kallus (2020). Imperfect generative models g_θ, for which p_θ deviates from $p_{X\times Y}$, will lead to a loss of optimality, because the optimal solution can be obtained for a sample from p_θ, but not for data generated from the true joint distribution $p_{X\times Y}$. We denote by $\overline{C}(q^s)$ the expected out-of-sample costs when applying an asymptotically optimal prescriptive analytics model to D_{synth} with $N \to \infty$, and by $\overline{C}(q^*)$ the expected out-of-sample costs when applying an asymptotically optimal prescriptive model to D^∞_{real} which contains $N \to \infty$ samples from $p_{X\times Y}$. The model q^* can now not only be the best known model, but also must be asymptotically optimal. The synthetic data gap can then be defined as follows:

$$\Delta_{synth} = \frac{\overline{C}(q^s) - \overline{C}(q^*)}{\overline{C}(q^*)}. \tag{4.13}$$

Obviously, Δ_{synth} is of limited practical relevance because, for real-world problems, the true joint distribution $p_{X\times Y}$ is unknown and one typically does not have access to sufficiently large data sets D^∞_{real}—else, synthetic data generation would not provide any value. However, as we show in the next section, Δ_{synth} is not only conceptually appealing but can provide important information to evaluate the performance of a generative model in controlled numerical experiments with known $p_{X\times Y}$.

Second, we want to capture the impact that the performance of the models that are used to solve (4.10) and (4.11) has on $\Delta_{overall}$. In our case, these are the DRL models with the weights $\hat{\psi}$ and ψ^* of the underlying

neural networks. Although DRL has achieved successes in a variety of application problems, there are no universal performance guarantees and it cannot be proven that the models converge to a full information optimum for $N \to \infty$. The performance of the DRL models is data- and problem-specific and depends on the choice of training algorithm and the training environment. The extensive hyperparameter tuning to find the best DRL model may be prohibitively expensive. Therefore, we cannot rule out the possibility that $\Delta_{overall}$ is negatively affected by a performance gap of the (DRL) model, which leads to inferior results in terms of $\overline{C}(q_{\psi*}^{s+r})$. To estimate such a potential "model gap", we suggest the following measure:

$$\Delta_{model} = \frac{\overline{C}(q_{\psi}^{r\infty}) - \overline{C}(q^*)}{\overline{C}(q^*)}, \tag{4.14}$$

where $\overline{C}(q_{\psi}^{r\infty})$ denotes the expected out-of-sample costs associated with the DRL model trained on D_{real}^{∞} that contains $N \to \infty$ samples from $p_{X\times Y}$, and $\overline{C}(q^*)$ is defined in conjunction with (4.13). Similarly to Δ_{synth}, we can only compute Δ_{model} in a setting where $p_{X\times Y}$ is known or we have a potentially infinite sample from $p_{X\times Y}$. In a practical setting, we could approximate Δ_{model} by training both models on D_{synth}. However, this would not allow for a strict separation of the synthetic data gap and the model gap, because we cannot assume that the DRL agent and the asymptotically optimal prescriptive model are affected equally by Δ_{synth}.

Finally, we want to assess how well $q_{\psi*}^{s+r}$ can mitigate performance differences that are due to limited training data, which is often a problem in practice. Even for asymptotically optimal models, the rate of convergence can be low, and we cannot make any inference about the model performance for $N << \infty$ (see Notz and Pibernik (2022) discussion). We capture this performance difference by defining

$$\Delta_{data} = \frac{\overline{C}(q^r) - \overline{C}(q^*)}{\overline{C}(q^*)}, \tag{4.15}$$

where $\overline{C}(q^r)$ is an asymptotically optimal model trained on D_{real} with $N << \infty$. Ideally, we can achieve lower out-of-sample costs with $q_{\psi*}^{s+r}$ compared to q^r. Thus, $\Delta_{overall} << \Delta_{data}$.

4.4 Numerical Experiments

In our numerical experiments, we study two different settings, inspired by theory and practice. In the first controlled experiment, we assume that we know the joint distribution $p_{X\times Y}$ and hence the data-generating process. Here, we want to understand how our approach"behaves", what performance gaps it produces, and how they can be explained along the lines of our measures in Section 4.3.3. In particular, we want to understand the quality of the generative model, the quality of a pure DRL model, and the quality of our transfer learning approach relative to a known optimum. Understanding this will help us interpret the results of a real-world analysis that we conduct in the second half of this section.

We keep the OM problem that we study in our numerical experiments simple to reduce the complexity induced by the problem setting itself. We consider the data-driven newsvendor that has been well studied in the domain of data-driven research, so that good benchmarks are available. We formulate the data-driven newsvendor as follows:

$$\begin{aligned}\min_{q(\cdot)\in\mathcal{F}} \mathbb{E}_{X\times Y}\left[C(\vec{q}(X),Y)\right] = \min_{q(\cdot)\in\mathcal{F}} \mathbb{E}_{X\times Y}\left[c_u(Y-q(X))^+\right. \\ \left. + c_o(q(X)-Y)^+\right]\end{aligned} \tag{4.16}$$

where $(\cdot)^+ = max(0,\cdot)$. The ratio between overage and underage costs is called the service level $sl = \frac{c_u}{c_u+c_o}$. We set $sl = 0.7$.

4.4.1 Experimental Setup for Known Data-Generating Process

In our first numerical experiment, we evaluated our approach for a known joint distribution $p_{X\times Y}$. Figure 4.2 shows an overview of our experimental setup. We detail our approach in the following. We start by generating four different data sets that all come from a linear data-generating process with known distribution $p_{X\times Y}$ which we describe in Section 4.4.1. We use the first two as training data sets, where D^{∞}_{real} with its 10,000 samples is our proxy for an unlimited data setting, and D_{real} is our proxy for a small data setting. In addition, we sample a validation set D^{val}_{real} that is used for

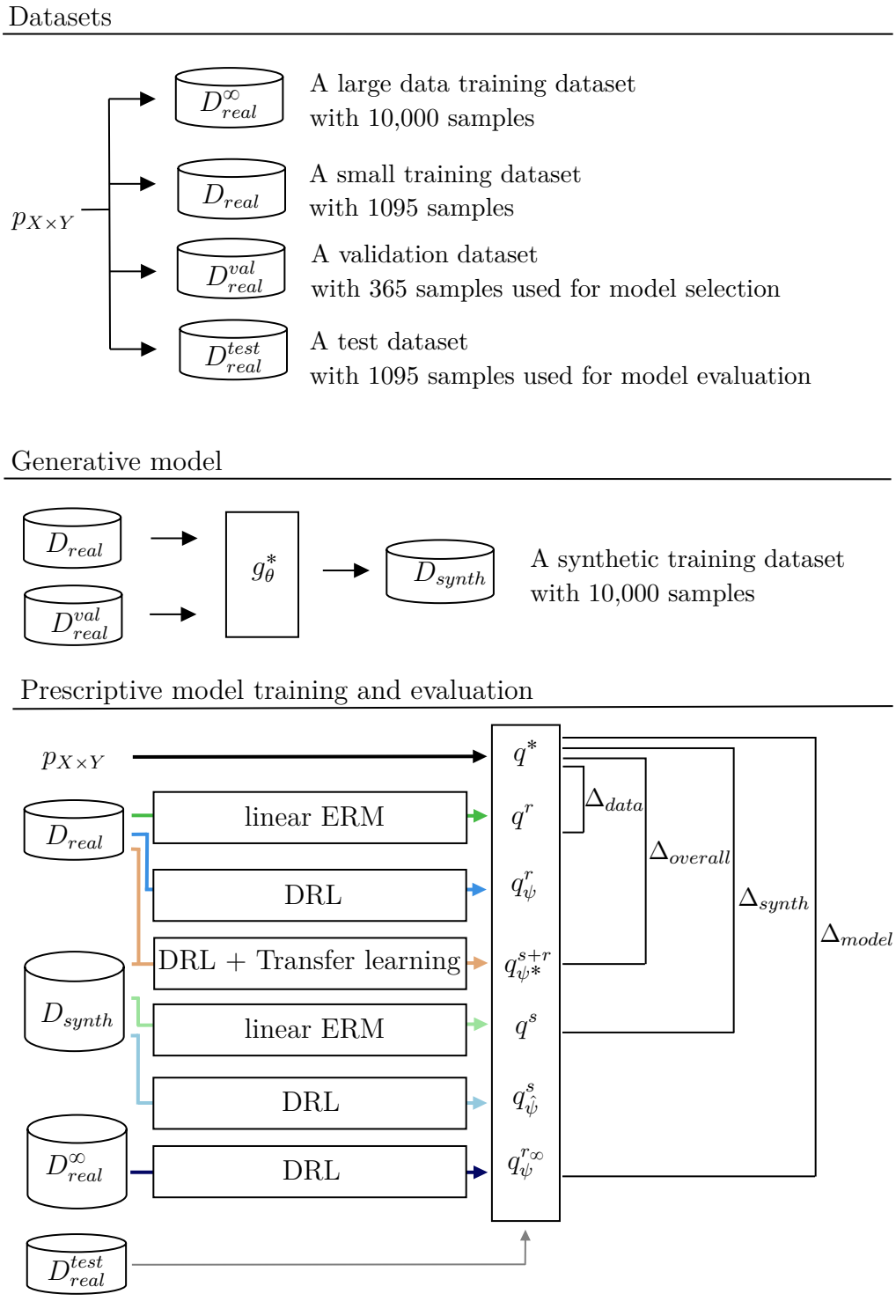


Figure 4.2: Experimental setup for known data-generating process

model selection and a hold-out test set D_{real}^{test} that is used to compute the performance gaps. We use D_{real} and D_{real}^{val} to train the generative model g_θ. We use g_θ to compute a large synthetic data set D_{synth}. We train different prescriptive models using the described data sets. Because we know the underlying distribution $p_{X\times Y}$ we can compute an optimal prescriptive model q^*. This is done by computing the feature-dependent demand and then adding 70% quantile of the noise distribution as a safety buffer. This model is the baseline to calculate the performance gaps described in Section 4.3.3 based on the hold-out test set D_{real}^{test}. To compute Δ_{data} we need to choose a prescriptive model that we train using D_{real}. We apply a model that assumes that the prescription function q comes from a linear function space, the empirical risk minimization-based approach proposed by Ban and Rudin (2019). We term this the linear ERM model. In this setting, the linear ERM model trained on D_{real} is q^r. Furthermore, we train a corresponding DRL-based model q_ψ^r. Details of the implementation of the DRL-based models can be found in Section 4.4.1. Next, we use D_{real} together with D_{synth} for our proposed transfer learning approach that results in $q_{\psi*}^{s+r}$. This model enables us to compute $\Delta_{overall}$. We evaluated Δ_{synth} using a linear ERM model q^s that is trained only on the generated data D_{synth}. Again, we train a corresponding DRL-based model $q_{\hat{\psi}}^s$ that we can use to assess how well the DRL agent can learn from the synthetic data. To evaluate Δ_{model}, we train a DRL agent based on D_{real}^{∞}.

Data Set

We use the make-regression function of the standard Python package *sklearn* (Pedregosa et al., 2011) to generate a data set in which the features X and demand y have a linear relationship of the form

$$y = \sum_i b_i x_i + b_0 + \epsilon \tag{4.17}$$

with b_0 the intercept of the linear model, b_i the scalar-valued coefficients[14] and ϵ random noise. In our experiments, we use 10 features, of which six are informative, that is, $b_i \neq 0$. We draw ϵ from a normal distribution $\mathcal{N}(0, 0.2 * \overline{y})$ with $\overline{y}$ the mean of y.

[14]The values of the coefficients b_i can be found in the Appendix D.1

We use this data-generating process to generate the data sets shown in Figure 4.2. To test the robustness of our approach, we performed 30 individual runs. In each run, we sample a new D_{real} from $p_{X \times Y}$ and evaluate the performance of our approach on a fixed test set D_{real}^{test}.

Generative Model

We use the conditional tabular GAN (CTGAN) architecture proposed by Xu, Skoularidou, et al. (2019) for the generation of the synthetic data set D_{synth}. As highlighted in Section 4.3.1, CTGAN appears particularly suitable in our context because it can generate tabular data with both continuous and categorical features and can deal with imbalanced categorical feature values, that is, certain values are over- or underrepresented. To tune and select a generative model, we perform a grid search of 180 unique hyperparameter combinations (see Appendix D.2) and select the model that achieves the lowest $meanPL$ on the evaluation data set D_{real}^{val}. For each trained GAN model, we generate 10,000 samples and use the linear ERM model as a surrogate model to compute $meanPL$, as described in Section 4.3.1. We use the linear ERM implementation and the default hyperparameter settings provided in the Python package *ddop* (Philippi et al., 2021). We use the model with the lowest $meanPL$ to generate D_{synth}.

Deep Reinforcement Learning and Transfer Learning

To train the DRL agents, we apply the proximal policy optimization (PPO) algorithm proposed by Schulman et al. (2017). PPO is a state-of-the-art DRL algorithm that has shown superior performance in many control tasks (Schulman et al., 2017) and promises stable training behavior (Schulman et al., 2017; Vanvuchelen et al., 2020). It has previously been applied in the context of OM (Balaji et al., 2019; Vanvuchelen et al., 2020). Our goal is to use a pragmatic approach for training that can be applied in a practical setting without requiring extensive computational effort, e.g., induced by hyperparameter tuning. Therefore, we use standard implementations and a rather straightforward training approach. We use the PPO implementation from *stable-baselines3* (Raffin et al., 2021) with standard hyperparameter settings. We use OpenAI's *gym* (Brockman et al., 2016) to implement the

DRL environment. For implementation details, see Appendix D.3. The training process for the models $q^s_{\hat{\psi}}, q^r_{\psi}$ and $q^{r_\infty}_{\psi}$ differs only in the training data set: $q^s_{\hat{\psi}}$ trains on D_{synth} and q^r_{ψ} trains on D_{real}. $q^{r_\infty}_{\psi}$ trains on D^{∞}_{real}. All models train 60,000 iterations without additional fine-tuning. In the transfer learning approach, we first train a DRL agent on D_{synth} for 60,000 iterations and derive the weights $\hat{\psi}$ of the underlying neural network. These weights are used to initialize a model q^{s+r}_{ψ} that is trained on D_{real} to obtain the final model $q^{s+r}_{\psi^*}$ with weights ψ^*. We regularize how strongly the knowledge gained in the source domain ($q^s_{\hat{\psi}}$ trained on D_{synth}) affects the parameters ψ^* of our final model $q^{s+r}_{\psi^*}$ by limiting the number of training iterations on D_{real}. In this regularization approach, the number of training iterations is a hyperparameter that has to be tuned on the basis of the available data. Figure 4.3 plots the average rewards on the validation set D^{val}_{real} for different lengths of the training process, reflected by the number of training iterations. We observe that updating the model on the data D_{real} of the target domain initially improves the overall model performance and that the maximum model performance is achieved at 30,000 training iterations. Beyond this value, overfitting on the target domain diminishes the performance of the model. Therefore, we set the number of training iterations at 30,000 in the rest of our experiments.

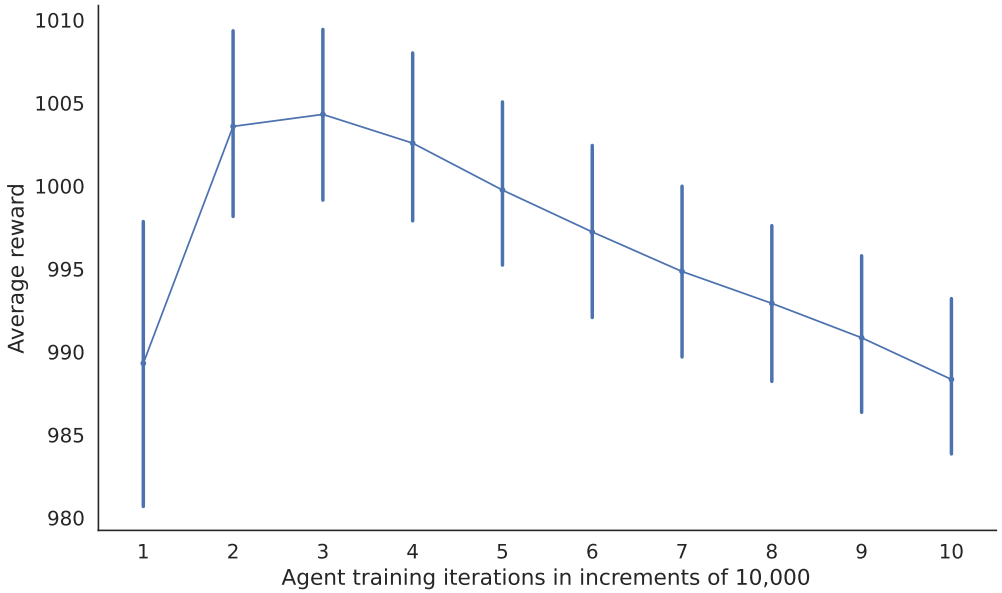


Figure 4.3: Average reward on validation set across experiment runs under variation of agent training length on D_{real}. Error bars show the standard deviation.

4.4.2 Results for Known Data-Generating Process

Figure 4.4 shows the percentage deviation from the average out-of-sample cost incurred by q^* for the different approaches across experiment runs along with the corresponding performance gaps. First, we discuss the performance gaps. We find that $\Delta_{overall}$ is smaller than Δ_{data} and Δ_{synth}. However, there is still a gap to Δ_{model} that can be exploited by improving different parts of the transfer learning approach.

We observe that the synthetic gap Δ_{synth} is particularly large. Together with the fact that the DRL-based model $q^s_{\hat{\psi}}$ also does not perform well, this indicates a low quality of the generated synthetic data. Δ_{synth} appears to be mitigated by our transfer learning approach. In fact, transfer learning with synthetic and real data $q^{s+r}_{\psi^*}$ outperforms the DRL model trained with only small data q^r_{ψ}, i.e., there is value in transfer learning, although the synthetic gap Δ_{synth} appears to be large.

The performance gap due to the small available real data Δ_{data} is greater than $\Delta_{overall}$. Hence, in this setting, we seem to mitigate performance losses due to scarce data using our transfer learning-based approach. However,

there is only a small difference of 0.5 percentage points between $\Delta_{overall}$ and Δ_{data}. However, the reason is not that DRL does not work well, as we can see from Δ_{model}. The linear ERM model q^r —which is the theoretically optimal model for our particular data—can "perfectly" learn the relationship between features and the dependent variable and only suffers from the limited amount of training data available. Δ_{data} can be purely attributed to the fact that we train linear ERM only on 1095 samples—the equivalent of 3 years of training data. DRL on its own (see results of q^r_ψ) is even less sample efficient.

In summary, our transfer learning approach improves on pure DRL, but suffers from the quality of the synthetic data expressed in Δ_{synth}. In this stylized setting, we can outperform q^r with $q^{s+r}_{\psi*}$. However, there is a caveat to this stylized setting—here we know that linear ERM is the best model because of the linear structure of our data-generating process. In a practical setting, we will not know the best model a priori. In addition, the problem setting might become too complex to apply "traditional" data-driven approaches. Further investment in improving transfer learning could help close the gap to $q^{r\infty}_\psi$.

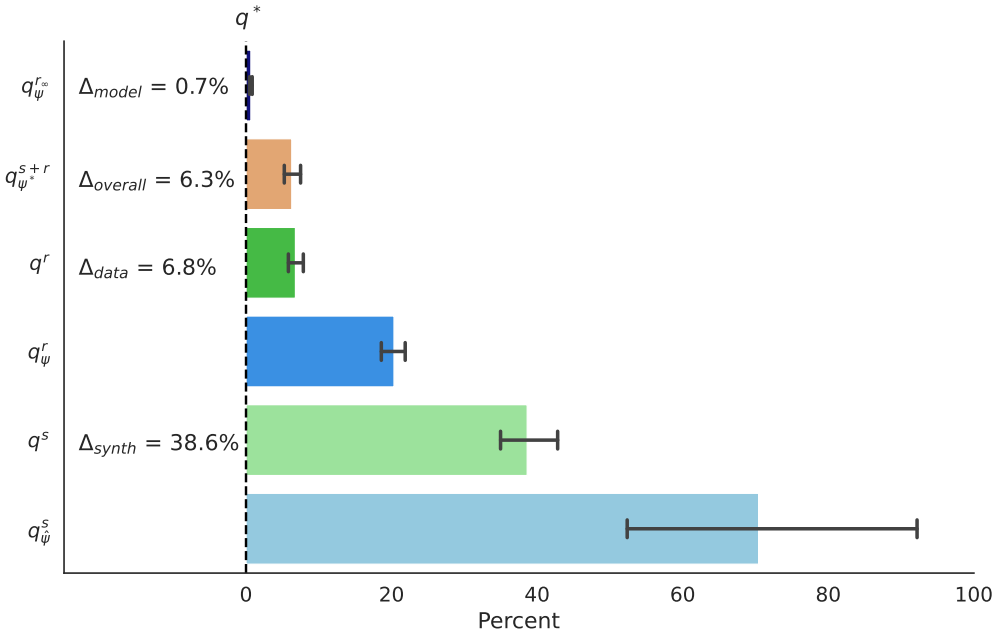


Figure 4.4: Percentage deviation from average out-of-sample costs of q^*. Error bars show standard deviation across 30 experiment runs. All Δ's are significant on a 99% confidence level under paired t-test

We start this investigation by evaluating the impact that the quality of GAN-generated data has on our transfer learning approach. We selected one experiment run and ranked the different GAN models trained in the synthetic data generation phase of our approach based on the pinball loss introduced in Section 4.3.1. For our evaluation, we chose GANs from different quantiles of this distribution and our original D_{synth} by the samples generated from these GANs. This means, for example, that $q^{s+r}_{\psi^*}$ now uses samples that were generated from the GAN in the 10% quantile in the pre-training phase. $q^s_{\hat{\psi}}$ and q^s use only these samples for training. Figure 4.5 shows the distribution of out-of-sample costs normalized by the average out-of-sample costs of q^* for models trained on GAN samples of different quality. To illustrate, the 0% quantile GAN is the GAN that performed the best in our model selection and the 100% quantile GAN is the GAN that performed the worst in our model selection. We observe that the performance of the DRL model $q^s_{\hat{\psi}}$ and the linear ERM model q^s depends on the quality of the GAN. Performance increases as the quality of the GAN increases. Especially for GAN data with lower quality (e.g., 90% and

100% quantiles), performance deteriorates. However, we do not observe this behavior for the transfer learning approach. Here, the performance appears quite stable and even independent of the GAN quality measured by the performance of q^s (also see Figure D.3 in Appendix D.5 for a focus on mean performance). For example, from the 10% quantile to the 25% quantile, the performance of both $q^s_{\hat{\psi}}$ and q^s deteriorates. This is also supported by comparing the demand distributions (see Figure D.4 in Appendix D.5). The distribution of the demand generated by the 0% and 10% quantile GAN strongly resembles the real data distribution, whereas the demand distribution of the 25% quantile GAN clearly looks different. Unexpectedly, the performance of $q^{s+r}_{\hat{\psi}^*}$ improves when pre-trained on the 25% quantile GAN compared to the 10% quantile GAN data.

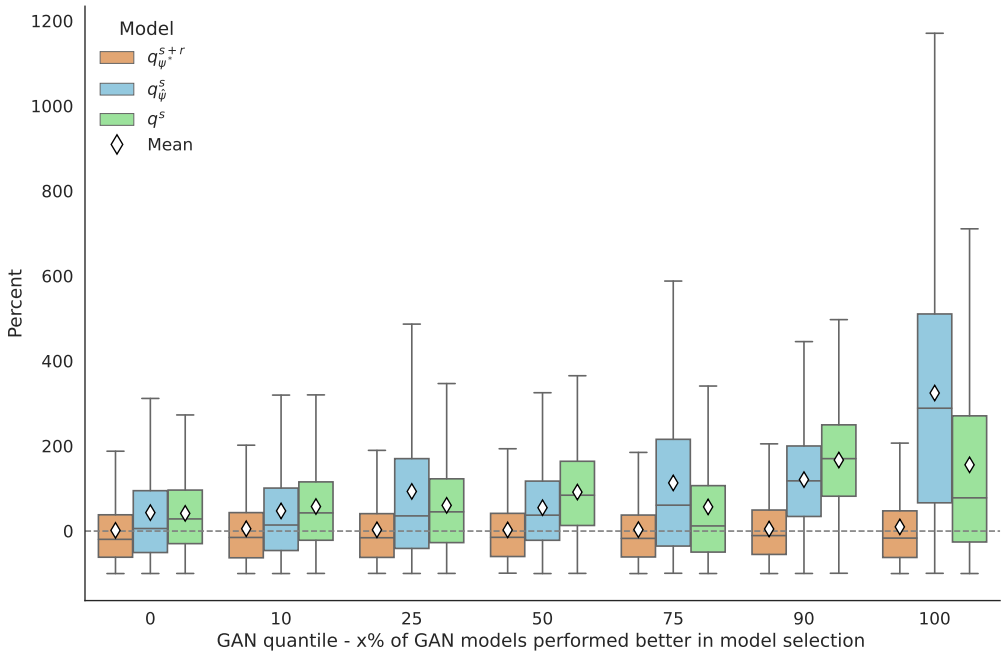


Figure 4.5: Percentage deviation from average out-of-sample costs of q^* per day in D^{test}_{real} under variation of the GAN quality for a selected experiment run. Outliers are omitted for better readability.

This motivates us to investigate how our transfer learning approach behaves in cases with more "extreme" synthetic data. We construct two different synthetic data sets independent of the GAN data. One with presumably lower quality than the GAN data and one with high quality. For low-quality synthetic data, we create a synthetic data sample that has

the same size as the GAN-generated D_{synth} but contains only constant demand values. The constant demand value is set to the mean demand of D_{real}. The features of this constructed synthetic sample all follow normal distributions that are fitted to the individual features of D_{real}. Hence, there is no connection between features and demand to be learned in this data.

In the second case, we use D_{real}^{∞} to pre-train $q_{\psi*}^{s+r}$. As D_{real}^{∞} is drawn from the true data-generating process, it resembles a perfect synthetic data sample.

Figure 4.6 shows the distribution of the out-of-sample costs normalized by the average out-of-sample costs of q^* for models trained on the best GAN samples, the synthetic data with constant demand, and the large real data sample. Similarly to the previous setting, $q_{\psi*}^{s+r}$ pre-trains on the given data and is fine-tuned on D_{real}, whereas $q_{\hat{\psi}}^{s}$ and q^s use only the given data for training. As an additional benchmark, we also plot the results for q^r—the linear ERM model trained on D_{real}. The results are rather disappointing. Even these "extreme" synthetic data cases appear to have little influence on the performance of $q_{\psi*}^{s+r}$. However, the performance of the other models is influenced by the quality of the data. We observe that for the two models $q_{\hat{\psi}}^{s}$ and q^s the performance on GAN as well as constant demand data is very similar. Especially if we consider mean performance. The interquartile range of the boxplots is slightly larger for the models trained on constant demand data. This indicates that in this quite simple make-regression setting, training a GAN model to generate synthetic data only generates small performance gains in comparison to generating synthetic data with a much simpler model. The mean percentage deviation from the out-of-sample costs of q^* is 1.6% for $q_{\psi*}^{s+r}$ pre-trained on the best GAN data vs. 2.7% for $q_{\psi*}^{s+r}$ pre-trained on the constant demand data. It is striking that even the model pre-trained on the constant demand data outperforms q^r. It is not clear which information the DRL agent can extract from the constant demand data during pre-training. When conducting this experiment, our initial hypothesis was that the performance of q^r should be a lower bound to the performance of the model pre-trained on the constant demand data. It is possible, however, that the DRL agent learns something about the general reward structure from the constant demand data.

In the large real data setting, we observe that although the boxplots

for the transfer learning approach $q^{s+r}_{\psi^*}$ and $q^{s}_{\hat{\psi}}$—the DRL model that only trains on the large real data—appear to be very similar, in terms of mean performance $q^{s+r}_{\psi^*}$ does not reach the performance of $q^{s}_{\hat{\psi}}$ (2.6% vs 0.4%). This provides an indication that additional effort might be necessary to tune the hyperparameters of the transfer learning approach. Still, all transfer learning approaches outperform q^r, which has a mean performance of 8.1%. Therefore, there appears to be a benefit of the pre-training that differentiates the transfer learning approach.

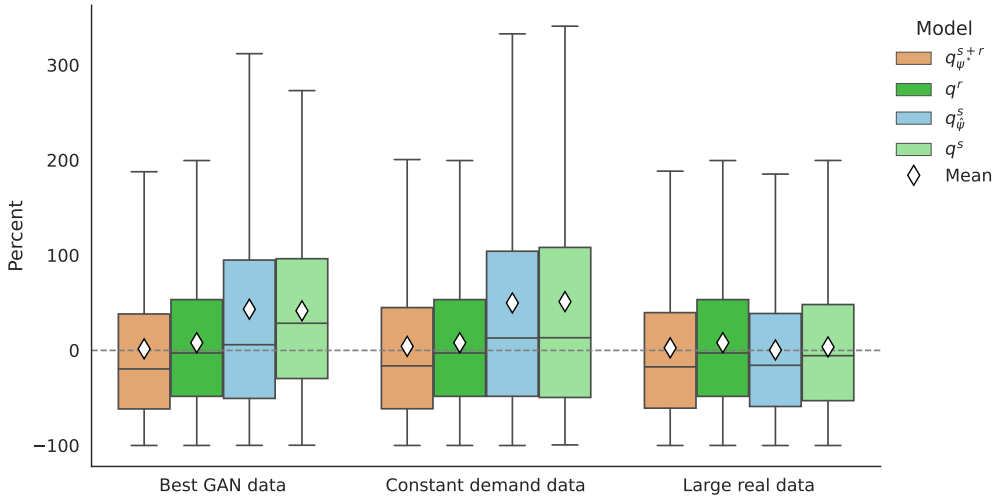


Figure 4.6: Percentage deviation from average out-of-sample costs of q^* per day in D^{test}_{real} under variation of the synthetic data used during training for a selected experiment run. Outliers are omitted for better readability.

4.4.3 Experimental Setup for Unknown Data-Generating Process

In a second analysis, we evaluate our approach based on real-world data from a bakery chain, where the data-generating process is unknown. In this setting, we do not know the optimal model q^* because we do not know the true underlying distribution $p_{X\times Y}$ or have access to an infinitely large sample from it. Hence, we cannot calculate performance gaps anymore. However, we can still receive an indication of the magnitude of the performance gaps by benchmarking our approach in a small sample setting with other DRL-based or other data-driven approaches.

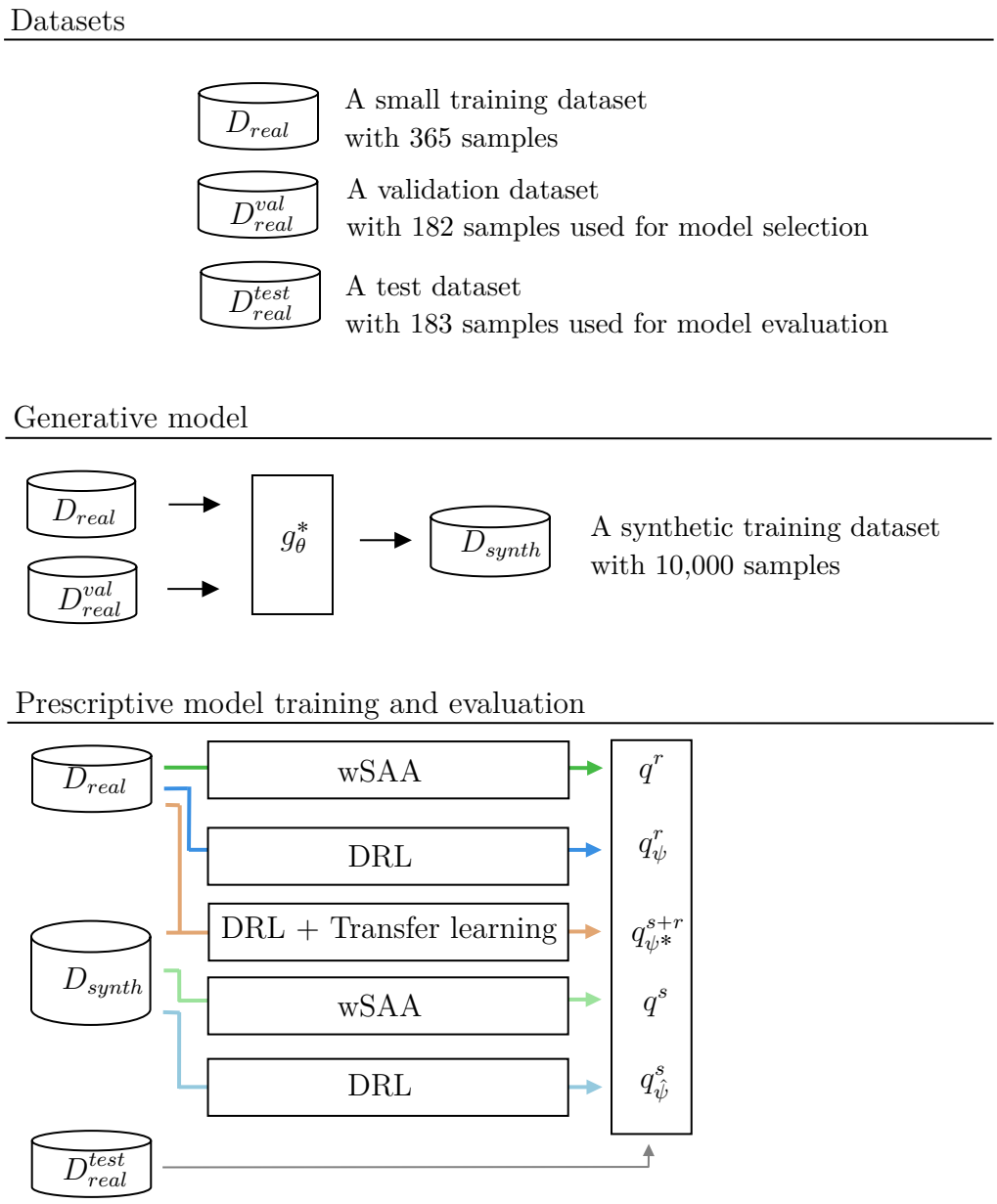


Figure 4.7: Experimental setup for unknown data-generating process

Our experimental setup is very similar to the setup in Section 4.4.1—an overview of the experimental setup is shown in Figure 4.7. In the following, we only point out the differences. Regarding the data sets, as we do not know $p_{X\times Y}$, we can only divide the available data into a small training data set D_{real}, a validation data set D_{real}^{val} and a test data set D_{real}^{test}. Regarding prescriptive models, we cannot train any model neither based on $p_{X\times Y}$ nor on a large sample from $p_{X\times Y}$. In addition, we exchange the asymptotically optimal "traditional" prescriptive model that we use to benchmark DRL-based approaches. In general, Buttler et al. (2022) found that the right prescriptive model strongly depends on the data set used and that model selection is important if we do not know the nature of our data. This can be a time-consuming process. However, since we use the same data set that Buttler et al. (2022) used in their analyses, we can rely on their results. Their study revealed that an approach proposed by Bertsimas and Kallus (2020) performed best. Their approach uses conditional density estimation and a random forest weight function that is used to re-weigh historical observations. We term this the weighted sample average approximation (wSAA) model. We use the random forest wSAA implementation and the default hyperparameter settings of *ddop* (Philippi et al., 2021).

Data Set

In our real-world setting, we use a data set from a local bakery chain. The data set contains sales data along with the corresponding features of five different bakery shops and three different products (bread rolls, grain rolls, and pretzels) from 05/01/2017 through 04/30/2019. Features capture information on sales time series, promotions, and weather. More detailed information can be found in Appendix D.6. The bakery needs to decide how many products to prepare for the next day. Unsold products are discarded at the end of the day. To apply our approach, we split the data into 1 year of training data D_{real}, half a year of validation data D_{real}^{val} and half a year of test data D_{real}^{test}. An individual prescriptive model is trained for each unique combination of product and store.

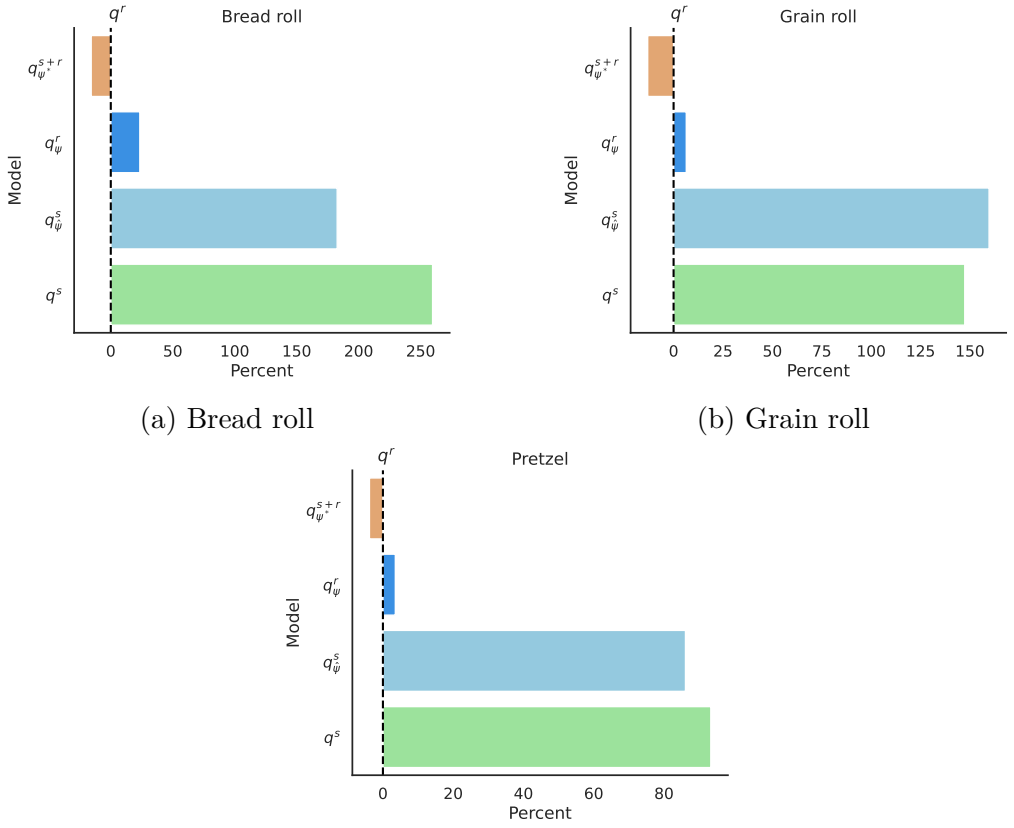
Generative Model

We choose the same specification for the generative model as in Section 4.4.1. However, we change the surrogate model that is used to select the best GAN model for the same reasons as described previously; we apply a wSAA model with a random forest weight function.

4.4.4 Results for Unknown Data-Generating Process

Figure 4.8 shows the percentage deviation from the out-of-sample costs of q^r per product (averaged across stores) for the different models. Although we cannot compute the performance gaps anymore due to the lack of an optimal model, we can still compare the performance of our pipeline to the "best known model", in this case q^r. We find that our transfer learning-based model $q^{s+r}_{\psi*}$ outperforms q^r and all other approaches for all products. On average, the costs incurred by $q^{s+r}_{\psi*}$ are between 4% and 16% (depending on the product) lower than the costs incurred by q^r. Furthermore, in line with the results in the previous section, we find that only leveraging D_{real} or D_{synth} for DRL is not enough to outperform the traditional data-driven model q^r. We find a cost increase of between 4% and 23% when using q^r_ψ instead of q^r. The performance gap between q^r and q^r_ψ is an indicator that DRL in this setting requires more samples for training—it is less sample efficient than wSAA—or, in other words, suffers more from small data. However, the performance difference between the two models is small (at least for grain rolls and pretzels), which we see as an indicator that the model gap is small. The performance gap between q^r and q^s is an indicator of the synthetic data gap and, therefore, of the insufficient quality of the synthetic data D_{synth}. It is interesting to see that on more complex data structures than in the one in Section 4.4.1 the DRL-based approach $q^s_{\hat{\psi}}$ seems to generalize better when trained on D_{synth} than q^s. The results are consistent on a shop level (see Appendix D.7).

To generate more granular insights, we vary the service level for a selected product in a single bakery shop. Figure 4.9 plots the scaled out-of-sample cost distribution of the models for different service levels. The performance of $q^{s+r}_{\psi*}$ and q^r appears to be quite stable across service levels and $q^{s+r}_{\psi*}$ consistently outperforms q^r. Interestingly, we observe that the



(a) Bread roll

(b) Grain roll

(c) Pretzel

Figure 4.8: Percentage deviation from average out-of-sample costs of q^r.

performance of the DRL-based model q^r_ψ that is only trained on D_{real} deteriorates with increasing service level. There are two hypotheses that can potentially explain this behavior. One, based on the small data set, the model q^r_ψ is only able to learn a good demand forecast but fails to calibrate the safety buffer correctly. The synthetic data in the transfer learning approach $q^{s+r}_{\psi*}$ is then used to learn the safety buffer. Two, q^r_ψ is not properly tuned and is overfitting on the small training data set. This would entail that the synthetic data in the transfer learning approach $q^{s+r}_{\psi*}$ is used for regularization.

In an effort to shed some light on these hypotheses, we conduct the same experiment as in the previous section and vary the synthetic data that the models use for (pre-)training. Again, we rank the GAN models trained during the model selection phase of the data generation and select GANs from different quantiles to generate synthetic data samples. In addition,

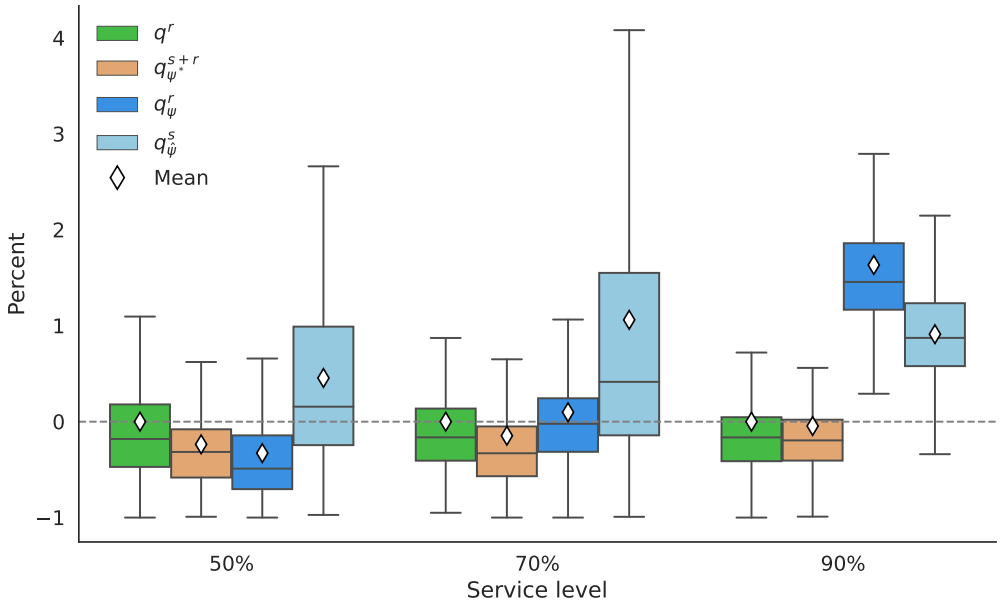


Figure 4.9: Percentage deviation from average out-of-sample costs of q^r per day in D^{test}_{real} under variation of service level for a selected product and bakery shop. Outliers are omitted for better readability.

we construct a synthetic data sample with constant demand. The constant demand value is set to the mean demand of D_{real}. To construct the features, we fit a normal distribution for each continuous feature of D_{real} and a uniform distribution for each categorical feature. Figure 4.10 shows the scaled out-of-sample cost distribution of $q^{s+r}_{\psi^*}$, $q^s_{\hat{\psi}}$ and q^s under variation of the synthetic data quality for a selected product in a single bakery shop.

Surprisingly, the DRL model $q^s_{\hat{\psi}}$ that trained only on the constant demand data outperforms all other $q^s_{\hat{\psi}}$ models that are trained on GAN data except for the data from the 10% quantile GAN. When using a traditional prescriptive model—in this case wSAA with a random forest weight function—the model trained on constant demand outperforms even all other traditional prescriptive models. This could indicate that, in this real-world setting, the GANs fail to learn a good estimate of the true underlying distribution $p_{X\times Y}$. The positive deviation from the average out-of-sample costs of q^r by $q^s_{\hat{\psi}}$ and q^s trained on the constant demand data indicates that there is information in the features to be exploited to make better decisions. Therefore, this discrepancy does not stem from the lack of predictive power of the features in the real data.

In addition, contrary to expectations, we observe that in this setting our proposed GAN model selection procedure does not appear to have worked well. If we compare mean performance, especially $q^s_{\hat{\psi}}$ trained on the 0% quantile GAN data, i.e. the best GAN data per our selection procedure, is outperformed by a number of models that train on worse data per our selection procedure, for example, $q^s_{\hat{\psi}}$ trained on the 10%, 25%, 50% and 75% quantile GAN data. The same is true if we compare the mean performance of the transfer learning models $q^{s+r}_{\psi^*}$. Here the transfer learning model pre-trained on the best GAN data is outperformed by all other transfer learning models that are pre-trained on GAN data. The best GAN data (0% quantile) transfer learning model has a mean performance of -14.6%, while the best of the other GAN data-based transfer learning models has a mean performance of -34.4%. However, the best GAN data transfer learning model still outperforms the transfer learning model pre-trained on the constant demand data which has a mean performance of -13.8%. In general, models based on transfer learning $q^{s+r}_{\psi^*}$ again show stable performance compared to models that train solely on synthetic data and consistently outperform q^r if we consider mean performance.

Finally, we cannot conclusively answer what drives the performance of our proposed transfer learning approach. The transfer learning model pre-trained on constant demand data outperforms q^r even though the model can only learn the mean demand in this setting, which it should also be able to learn from D_{real} alone. It is possible, therefore, that the pre-training acts as a way of regularizing the training of the DRL agent. However, the increase in performance of the transfer learning approaches when pre-trained on GAN data suggests that there is more information to be extracted from the GAN data than from the constant demand data that helps during pre-training.

However, only considering a selected product in a single bakery shop, caution must be applied to these conclusions, as the findings might not be representative of all products and bakery shops.

We conclude that our proposed approach also brings value in a practical setting. In all experiments, we see the same effect as before: Based solely on synthetic data, the reinforcement learning approach q^r_ψ does not perform well compared to q^r. We can remedy this through fine-tuning—in

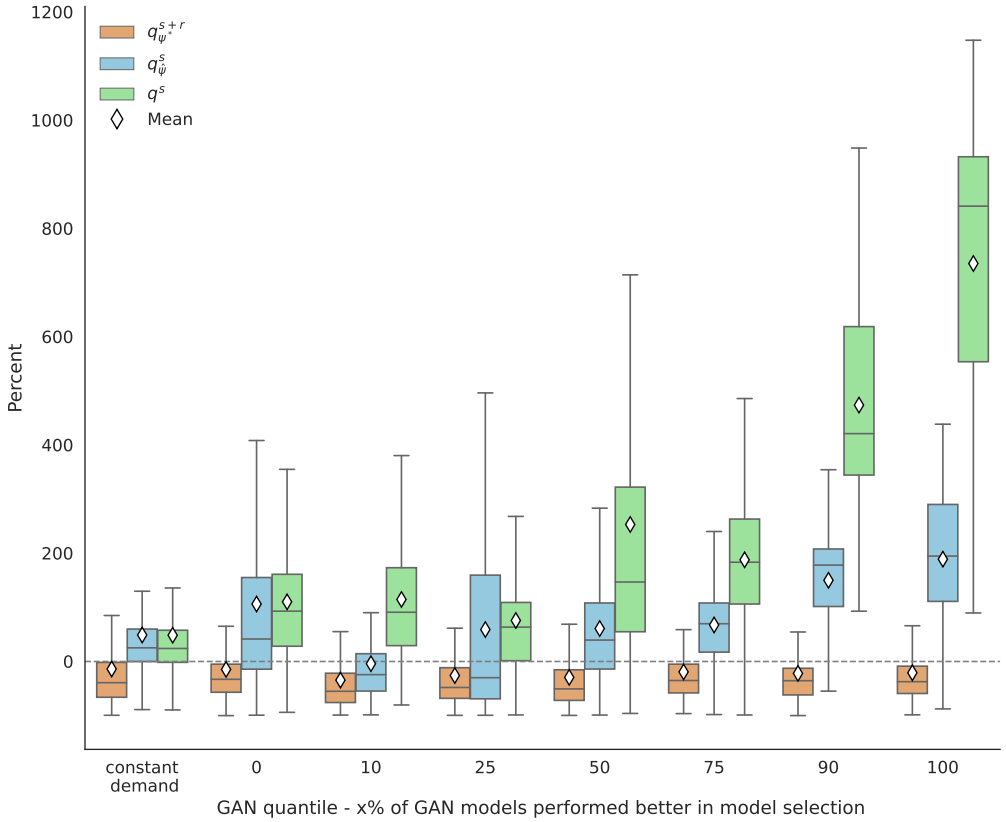


Figure 4.10: Percentage deviation from average out-of-sample costs of q^r per day in D_{real}^{test} under variation of the synthetic data quality for a selected product and bakery shop. Outliers are omitted for better readability.

fact, the combination of synthetic data and fine-tuning again renders the transfer learning approach superior. However, more research is necessary to investigate the driving forces behind this behavior.

4.5 Conclusion and Future Research Opportunities

This work proposes a transfer learning-based approach that combines synthetically generated and real data to train DRL agents in an OM context. In particular, our approach uses synthetic data that was generated by a deep generative model—in this case a GAN—to pre-train a DRL agent. Then, this agent is fine-tuned using a small sample of real data.

We illustrate and evaluate our proposed approach using a classical single-

period inventory management problem in two different settings. Using a controlled setting with known data-generating process, we make an effort to shed light on the performance drivers of our proposed approach. Using a real-world setting with data from a local bakery chain, we investigate the practical relevance of the proposed procedure.

Using extensive numerical evaluations, we demonstrate that our proposed approach is capable of improving the performance of DRL agents. However, we cannot conclusively derive what drives the performance of our proposed approach. Even though we observe that models trained on the GAN-generated data alone perform worse than models purely trained on small real data, a combination of both leveraging our proposed approach renders a positive performance impact independent of the data quality.

Our work has several limitations that should be addressed in future research.

First, it is difficult to obtain theoretical results for methods based on deep learning. Therefore, we resorted to a thorough numerical evaluation of our approach. However, this naturally limits our ability to draw general conclusions that go beyond the problem and data sets studied in this work.

Additionally, we chose to evaluate our approach on a relatively simple planning problem. This choice was driven by the availability of good benchmarks that include features as well as the approachable structure of the cost function. Even though DRL can be adapted more flexibly to solve different planning problems than traditional prescriptive approaches, the environment in which the DRL agent learns still has to be adapted if we want to consider different problem classes. Therefore, we leave the evaluation of our approach in more complex problem settings such as multi-period inventory control to future research.

Furthermore, in our approach, we mostly relied on the standard hyperparameter settings of the DRL libraries used for our evaluation in line with Vanvuchelen et al. (2020) in order to limit the computational effort. Future work should consider in-depth approaches to tuning different hyperparameters using sophisticated hyperparameter tuning frameworks (e.g., *Optuna* or *Tune*).

On a similar note, finding the right transfer learning approach is an ongoing field of research and we do not claim that we have exhausted the

performance of q_{DRL}^{s+r} with the transfer learning approach that we applied, but leave this for future research.

Another limitation of our work is that we focused our evaluation on synthetic data that was generated using GANs. Although they have been successfully applied to generate tabular data (Xu, Skoularidou, et al., 2019; Esteban et al., 2017), there is no substantial evidence that they are the best deep generative model for OM data. Future work should explore the application of other sophisticated deep generative models to generate synthetic data.

Such deep generative models can have many benefits. They promise to learn complex relationships between demand and features. Some guarantee privacy, allowing us to train models without ever sharing real data. However, it is difficult to understand which interaction effects are actually present in the generated data. We made a first step towards isolating these effects with our case study of constant demand data. To further study the effect of the synthetic data used, other synthetic data sets with different characteristics should be created and evaluated, e.g. data with step changes in the demand due to categorical features or different correlations between demand and continuous features.

Our work is exploratory in nature, and we view it as a starting point for a thorough and in-depth evaluation of how to make DRL accessible in practical OM contexts. The potential to extend our work continues to be large.

The use of synthetic data is only one way to increase the size of the available training data. In many practical settings—even in our small bakery case study—there is data available for multiple products or multiple locations. Future work should consider whether pre-training on data from other products or locations is beneficial for guiding the learning process of DRL agents.

A second extension to our approach, but also to pure DRL-based approaches, is the consideration of feature uncertainty. Our current approach assumes that the features of the planning period are known. This is a reasonable assumption for part of the features, such as calendrical features (e.g., month or weekday) or features that describe product characteristics. However, in real-world settings, there are also many features for which only

forecasts are available. The most prominent example of this are weather-related features. A potential avenue for future research is how to include this additional level of uncertainty in decision-making approaches.

5 Summary and Conclusion

The development of data-driven OM approaches has increased the need for training data in OM. Although many rich data sources are available, recordings of variables of interest, which are demand observations in most OM cases, are scarce. Research on synthetic data generation offers the possibility to enrich or augment small amounts of real training data, thus promising to improve OM decision-making.

The goal of this dissertation is to address the guiding research question presented in Chapter 1:

Guiding Research Question. *How can synthetic data be leveraged to improve decisions for data-driven inventory management?*

Chapter 2 of this dissertation addresses research question 1, presented in Chapter 1, by performing a meta-analysis of selected data-driven OM approaches. The chapter starts by reviewing the papers in the field that are most relevant to the topic and comparing the data and benchmarks they use to evaluate newly proposed methods. The review reveals that the evaluations are highly heterogeneous and lack reproducibility, so the article presented in Chapter 2 establishes a reproducible, unified evaluation procedure. To shed light on the robustness of data-driven OM approaches under exogenous problem parameters like the service level or the available feature information, the proposed procedure is applied to a large and heterogeneous data set in a comprehensive numerical experiment in the context of a newsvendor problem setting[15]. The selected data-driven OM approaches are deployed in the form of the Python package *ddop* (Philippi et al., 2021)[16].

[15]The data set is made available to the research community for future evaluation studies.
[16]Also see Appendix A.

The results suggest that the evaluated approaches have a low level of robustness and that no model can be recommended as a state-of-the-art model, even in a low-complexity problem setting like the newsvendor, as no model consistently outperforms all others across problem parameters. Therefore, objective evaluation procedures and a consistent approach to identifying state-of-the-art models are needed. The work presented in Chapter 2 is a first step towards more reproducible research in OM, but more work is needed to understand the drivers behind model performance before the best model to use in which setting can be determined.

Chapter 3 of this dissertation addresses research question 2, presented in Chapter 1, by applying GANs to generate a large synthetic data set for model training purposes in the context of a single-period newsvendor problem setting. Using GANs enables complex relationships between demand and auxiliary features to be modelled, but GANs can be difficult to train and evaluate. The chapter's main methodological contribution is the development of a novel GAN selection procedure that is tailored to OM, as it provides an operational cost-based evaluation. The selected GAN model can then be used to generate data that can be used to train data-driven OM approaches. A wSAA approach is trained using data generated by a selected GAN and then applied to solve a newsvendor problem for a local bakery chain. This procedure is benchmarked against more traditional approaches, such as distribution fitting and SAA. The findings show that using GANs is a promising alternative to such conventional methods.

Chapter 4 of this dissertation addresses research question 2, presented in Chapter 1, by proposing a transfer learning approach for training DRL agents. The main motivation for this approach is the large training data requirement of DRL approaches. . This chapter's main methodological contribution is the proposed approach, which leverages a combination of synthetic and real training data. In addition, the work sheds light on the performance impact of various parts of the approach. In numerical experiments, the approach is applied to solve the newsvendor problem in two data settings: a controlled setting, where the underlying real data distribution is known, and a real-world setting from a bakery chain, where the real data distribution is unknown. The results show that, in both settings, the approach outperforms traditional prescriptive analytics approaches. This

research also finds that the DRL agent does not outperform traditional prescriptive analytics approaches when it is trained solely on synthetic data or when it is trained solely on a small amount of real data. Therefore, what the agent learns during the pre-training phase of the transfer learning approach cannot be determined conclusively. However, the proposed approach provides a foundation for connecting the worlds of synthetic data and real data.

In conclusion, the application of synthetic data for inventory management is a promising avenue for research. The results presented in this dissertation show that novel approaches to generating synthetic data are suitable for modelling complex interactions of features and demand in an inventory context. Their application supports the use of sophisticated machine learning and deep learning models like DRL in guiding and improving decision-making. However, the application also comes at the cost of high computational complexity that may not always be outweighed by its benefits. Clearly, several avenues for future research remain.

Among these is research that expands the application of the developed approaches to more complex problem settings. While this dissertation considers a range of data sets, many from practice, the problem class it considers is limited to the single-period newsvendor problem. Future research should expand the use of synthetic data in decision-making to more complex problem settings, such as multi-period inventory problems or even beyond the domain of inventory management. The developed DRL-based approach provides a natural starting point for expansion to other problem settings, as Gijsbrechts et al. (2021) showed in explaining that DRL can be applied to a variety of problem classes, albeit without considering feature information.

A second avenue for future research is to investigate the methods of synthetic data generation that can model joint distributions of demand and feature information. While GAN-based approaches like those applied in this dissertation have shown promising results, they also require considerable training effort and can be unstable during training. Therefore, GANs should be benchmarked against other sophisticated generative approaches, such as variational autoencoders, Bayesian networks, or normalizing flows.

A third avenue for future research is to incorporate the properties of

generative models into decision-making models to improve sample efficiency, that is, the model's ability to learn from limited data. The shift from predictive to prescriptive analytics eliminates the need for a forecasting model, which enables more information to be extracted from the real data in the decision-making model than is possible when the information is aggregated into a forecast before being given to the decision-making model. Similarly, more research should explore how to incorporate into the final decision-making model the knowledge the generative model has learned about the real data distribution without the detour of creating training samples. This step is similar to the step from predictive to prescriptive analytics, where the forecasting model already contains information that is lost if only the forecast is used for decision-making. Incorporating into the final decision-making model the knowledge the generative model has learned about the real data distribution promises to result in a single, flexible, and data-efficient prescriptive model.

A ddop: A Python Package for Data-Driven Operations Management

In today's fast-paced world, companies face considerable uncertainty when making important decisions in operations management, for example, when deciding upon capacity, inventory levels, transportation, and production schedules. However, with the rise of digitization, companies have gained unprecedented access to data related to their particular decision problem, offering the opportunity to reduce the degree of uncertainty. For example, in inventory management, the decision maker may have access to historical demand data as well as additional side information, such as social media data, customer behavior, weather forecasts or calendar data. Driven by the availability of such rich data sources there has recently emerged a stream of literature in operations management research called "data-driven operations management" (DDOM). The focus of DDOM is to combine machine learning and traditional optimization techniques to prescribe cost-optimal decisions directly from data. Various models have been developed and shown great performance on the dataset used. However, what is missing is efficient access to open-source code and datasets. With *ddop*, we provide a Python library that integrates well-established algorithms from the field of data-driven operations management, as well as standard benchmark datasets. Thus, *ddop* helps researchers in two ways:

- Researchers can efficiently apply and compare well-established DDOM models.
- Researchers can test newly developed models on benchmark datasets provided in the package.

The application programming interface (API) of *ddop* is designed to be consistent, easy-to-use, and accessible even for non-experts. With only a few lines of code, one can build and compare various models. In *ddop* all models are offered as objects implementing the estimator interface from *scikit-learn* (Buitinck et al., 2013). We thus not only provide a uniform API for our models, but also ensure that they safely interact with *scikit-learn* pipelines, model evaluation and selection tools.

The library is distributed under the 3-Clause BSD license, encouraging its use in both academic and commercial settings. The full source code is available at `https://github.com/opimwue/ddop`. The package can be installed via the Python Package Index using pip install *ddop*. A detailed documentation providing all information required to work with the API can be found at `https://opimwue.github.io/ddop/`.[17]

A.1 Statement of Need

With the growing number of publications in the field of data-driven operations management, comparability is becoming increasingly difficult. The reasons for this are twofold: One, most scientists work with proprietary company data which cannot be shared. Two, it is not yet standard that researchers share the code used to implement their models. Consequently, results are not directly reproducible and models have to be re-implemented every time a researcher wants to benchmark a new approach. This not only takes a lot of time but can also be a demanding process since such complex models are often challenging to implement. Against this background, there has recently been a call to take inspiration from the machine learning community, where great APIs like *scikit-learn* (Buitinck et al., 2013), *fastai* (Howard and Gugger, 2020), or *Hugging Face* (Wolf et al., 2019) have been developed that allow previous developed ML models to be effectively applied on different datasets. Following up on this, *ddop* is the first of its kind to integrate well-established data-driven models for operations management tasks. At the current state, this includes various approaches to solve the data-driven newsvendor problem, such as weighted sample average approxi-

[17] This paper was published in *The Journal of Open Source Software* Philippi et al. (2021). It is co-authored by Andreas Philippi and Nikolai Stein.

mation (**Bertsimas.2020)**), empirical risk minimization (Ban and Rudin, 2019), and a deep learning-based approach (Oroojlooyjadid, Snyder, et al., 2020). In addition, the library provides different real-world datasets that can be used to quickly illustrate the behavior of the available models or as a benchmark for testing new models. *ddop*'s aim is to make data-driven operations management accessible and reproducible.

A.2 Usage

Since all models in *ddop* implement the estimator interface from *scikit-learn* consisting of a *fit*, *predict*, and *score* method, usage follows the standard procedure of a *scikit-learn* regressor. First, a model is initialized by calling the class constructor from a given set of constant hyperparameter values, each describing the model or the optimization problem the estimator tries to solve. Note that for ease of use, all estimators use reasonable default values. It is therefore not necessary to pass any parameter to the constructor. However, it is recommended to tune them for the respective application, since this can often improve decision quality. After the model has been initialized, the *fit* method is used to learn a decision model from the training data (X_{train}, y_{train}). Once the training process is completed, the function returns the fitted model, which can then be used to make decisions for new data (X_{test}) by using the *predict* method. Finally, the *score* method can be used to access the decision quality of a model. The method takes as an input X_{test} as well as the corresponding true values y_{test} and computes the average costs between y_{test} and *predict*(X_{test}). Because all estimators follow the same interface, using a different model is as simple as replacing the constructor.

A.3 Future Work

There are several directions that the *ddop* project aims to focus on in future development. While at the current state there are only algorithms available to solve the newsvendor problem, the goal is to include models to solve other operations management tasks like multi-period inventory management

or capacity management. In addition, we aim to extend the library in terms of available datasets and tutorials.

B Appendix of Chapter 2

B.1 Data Sets

Bakery: This data set is provided by a local bakery chain. The data set contains sales data for three different products at 35 different stores over a period of 1215 days. Every evening, each store must order the products from a central factory that are delivered the next morning. Reordering during the day is not possible. Unsold goods have to be disposed of at the end of the day. Thus, the problem at hand can be considered a newsvendor problem. All products are everyday items with typically high stock levels making censored demand unlikely. We exclude 10 unique product-store combinations due to demand intermittency. Next to calendric and lag features, the data set contains information on weather, promotions, and holidays.

Restaurant: This data set contains daily sales data from a restaurant for 7 different main ingredients on 765 days. To prepare the meals, the restaurant must decide how much of the ingredients to defrost overnight. The defrosted ingredients must then be sold within the next day. Leftovers are disposed of. Thus, the problem at hand can be considered a newsvendor problem. During data recording, the store manager's strategy was to maintain a service level of almost 100%, which is why we consider censored demand not to be an issue. Next to calendric and lag features, this data set contains weather as well as special features.

M5: The M5 data set[18] contains daily sales from Walmart stores and was made available as part of the well known forecasting competition M5. The original data set contains sales records for 3,049 products across 3 product

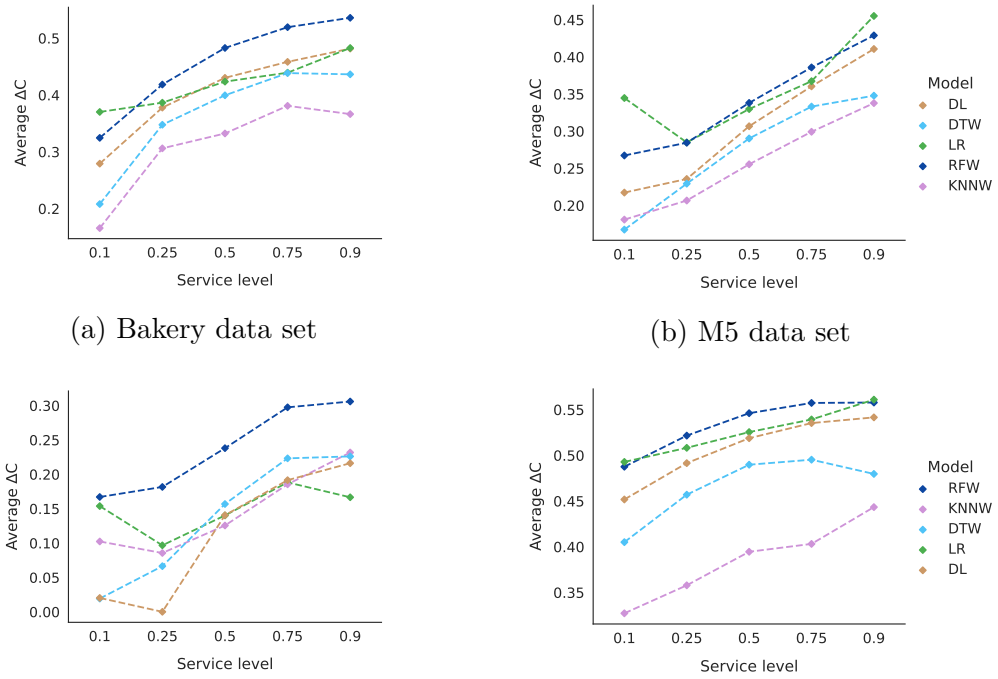
[18]https://www.kaggle.com/c/m5-forecasting-accuracy

categories, 7 departments, and 10 stores on 1942 days. For our analysis, we select only data that belongs to the product category "Foods" as this is most relevant in a newsvendor setting. To avoid intermittent demand, we select only the top 10 products which exhibit the least intermittency. This leads us to 100 unique product store combinations selected for the numerical evaluation. Next to calendric and lag features, this data set contains features that indicate special events such as sporting events or payout days.

SID: The store item demand data set was made available as part of the Kaggle Store Item Demand Forecasting Challenge[19]. It contains sales data from 50 different products in 10 different stores for 1826 days. It is our largest data set in terms of unique product store combinations but contains only calendric and lag features derived from the time series.

[19]https://www.kaggle.com/c/demand-forecasting-kernels-only

B.2 Relative Performance Improvement over SAA under Variation of Service Levels and Features



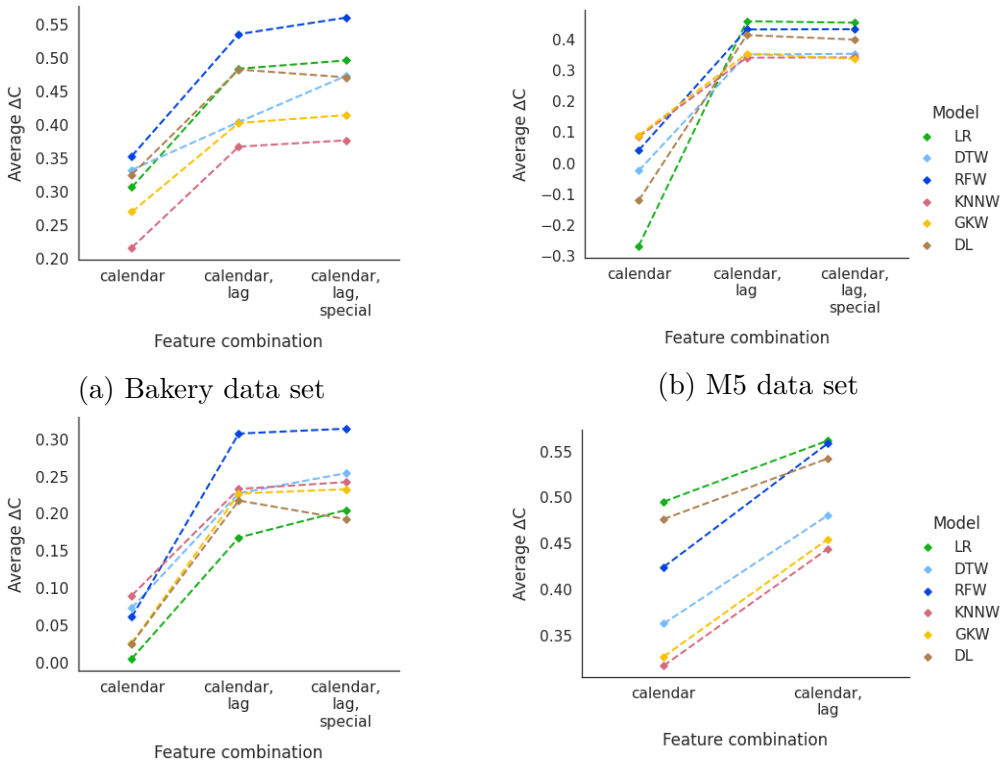
(a) Bakery data set

(b) M5 data set

(c) Restaurant data set

(d) SID data set

Figure B.1: Average relative performance improvement over SAA across service levels and data sets



(a) Bakery data set

(b) M5 data set

(c) Restaurant data set

(d) SID data set

Figure B.2: Average relative performance improvement dependent on feature availability

B.3 Hyperparameter Grids

The section states the list of hyperparameters tuned including their search spaces. If a model is not listed here, it has no hyperparameters to tune.

- DTW:
  - Maximum depth of the tree: [None, 2, 4, 6, 8, 10]
  - Minimum number of samples required to split an internal node: [2, 4, 6, 8, 16, 32, 64]
- RFW:
  - Maximum depth of a tree: [None, 2, 4, 6, 8, 10]
  - Minimum number of samples required to split an internal node: [2, 4, 6, 8, 16, 32, 64]
  - Number of trees in the forest: [10, 20, 50, 100]
- KNNW:
  - Number of neighbors to use: [1, 2, 4, 8, 16, 32, 64, 128]
- KW:
  - Kernel bandwidth: $[0.5, 0.75, 1, \ldots, \lceil\sqrt{n_{features}/2}\rceil + 0.25]$
- DL:
  - Optimizer: [“adam”]
  - Network architecture (number of neurons in each layer):
    - $(\lceil 0.5 \cdot n_{features}\rceil,\ \lceil 0.5 \cdot 0.5 \cdot n_{features}\rceil)$
    - $(\lceil 0.5 \cdot n_{features}\rceil,\ \lceil 0.5 \cdot 1 \cdot n_{features}\rceil)$
    - $(1 \cdot n_{features},\ \lceil 0.5 \cdot 1 \cdot n_{features}\rceil)$
    - $(1 \cdot n_{features},\ 1 \cdot 1 \cdot n_{features})$
    - $(2 \cdot n_{features},\ \lceil 2 \cdot 0.5 \cdot n_{features}\rceil)$
    - $(2 \cdot n_{features},\ 2 \cdot 1 \cdot n_{features})$
    - $(3 \cdot n_{features},\ \lceil 3 \cdot 0.5 \cdot n_{features}\rceil)$
    - $(3 \cdot n_{features},\ 3 \cdot 1 \cdot n_{features})$
  - Epochs: [10, 100, 200]

C Appendix of Chapter 3

C.1 CTGAN Architecture

CTGAN was implemented using the code provided along with Xu, Skoularidou, et al. (2019)'s paper and we refer to the original paper for details on the notation below.

Generator G network architecture:

$$\begin{cases} h_0 = z \otimes cond \\ h_1 = h_0 \otimes ReLu(BN(\mathrm{FC}_{|cond|+|z|\to 256}(h_0))) \\ h_2 = h_1 \otimes ReLu(BN(\mathrm{FC}_{|cond|+|z|+256\to 256}(h_1))) \\ \hat{\alpha}_i = tanh(\mathrm{FC}_{|cond|+|z|+512\to 1}(h_2)) & 1 \leqslant i \leqslant N_c \\ \hat{\beta}_i = gumbel_{0.2}(\mathrm{FC}_{|cond|+|z|+512\to m_i}(h_2)) & 1 \leqslant i \leqslant N_c \\ \hat{d}_i = gumbel_{0.2}(\mathrm{FC}_{|cond|+|z|+512\to |D_i|}(h_2)) & 1 \leqslant i \leqslant N_d \end{cases}$$

Discriminator D network architecture:

$$\begin{cases} h_0 = r_1 \otimes \cdots \otimes r_1 0 \otimes cond_1 \otimes \cdots \otimes cond_{10} \\ h_1 = drop(leaky_{0.2}(\mathrm{FC}_{10|r|+10|cond|\to 256}(h_0))) \\ h_2 = drop(leaky_{0.2}(\mathrm{FC}_{256\to 256}(h_1))) \\ \mathcal{C}(\cdot) = \mathrm{FC}_{256\to 1}(h_2) \end{cases}$$

Loss function: WGAN loss with gradient penalty (Gulrajani et al., 2017)

C.2 Detailed Description of Features

In this section, we give additional information about the features described in Section 3.5.1.

- demand: Daily sales for a bakery shop and product
- shop_no: Bakery shop identifier
- product_no: Product identifier
- shop_rank: Target encoding of shop_no
- product_rank: Target encoding of product_no
- isoweek: calendar week number
- weekday: day of week
- month: month
- is_schoolholiday: Boolean whether the day is a school holiday
- is_holiday: Boolean whether the day is a holiday
- is_holiday_next2days: Boolean whether the day is a holiday in the next two days
- promotion_currentweek: Boolean whether there is a sales campaign in the current week
- promotion_lastweek: Boolean whether there was a sales campaign in the previous week
- forecast_weekly: Forecast of Holt-Winter's exponential smoothing with 7 day trend
- forecast_monthly: Forecast of Holt-Winter's exponential smoothing with 30 day trend
- temp_avg_celsius: Average temperature on day of sales
- rain_mm: Total rainfall in mm on day of sales

C.3 Hyperparameter Grid

Hyperparameter	*Values*
Batch size	$\{n/100, n/50, n/25\}$
Number of training epochs	{100, 200, 300}
Learning rate G	$\{2e^{-04}, 4e^{-04}\}$
Learning rate D	$\{2e^{-04}, 4e^{-04}\}$
Random seed	{1-5}

Table C.1: Hyperparameter grid used for GAN training

D Appendix of Chapter 4

D.1 Coefficients of the regression model

Below we report the coefficients of the regression model used to create the controlled data setting in Section 4.4.3.

Coefficient	*Value*
b_0	482.6932146
b_1	21.6926114
b_2	46.47708831
b_3	49.31993817
b_4	9.4032502
b_5	0.
b_6	48.84516845
b_7	0.
b_8	53.89646799
b_9	0.
b_10	0.

Table D.1: Overview of regression coefficients

D.2 GAN hyperparameter grid

In the following, we report the search space of GAN hyperparameters for which we performed a grid search based on the pinball loss introduced in Section 4.3.1.

Hyperparameter	*Values*
Batch size	$\{n/100, n/50, n/25\}$
Number of training epochs	{100, 200, 300}
Learning rate G	$\{2e^{-04}, 4e^{-04}\}$
Learning rate D	$\{2e^{-04}, 4e^{-04}\}$
Random seed	{1-5}

Table D.2: Hyperparameter grid used for GAN training

D.3 DRL training environment

The training environment defines the world of the DRL agent. We use OpenAI's *gym* (Brockman et al., 2016) to implement the environment. The action space of our agent is continuous. However, we cap the action space such that the highest quantity that the agent can order is 20% above the maximal demand observed in the training set. Respectively, we set the minimum order quantity to 20% below the minimum demand observed. The state space is equal to our feature space such that every state s_t consists of a single feature vector x_t. After the agent has chosen an action/ order quantity, the agent receives a reward. We define the reward equivalent to Equation 4.16 but use a profit formulation:

$$reward(q_\psi(x_t), y_t) = min(q_\psi(x_t), y_t) * sales\_price - q_\psi(x_t) * purchase\_cost \tag{D.1}$$

where y_t is the demand that corresponds to feature vector x_t. We

set $sales\_price = \frac{1}{1-sl}$ with $sl = 0.7$ and $purchase\_cost = 1$. We scale rewards, actions, and states to $[-1, 1]$ to improve training.

D.4 Paired t-test results for known data-generating process

Below we report the paired T-test results for the different performance gaps computed.

Table D.3: Paired t-test results for known data-generating process

H_0		Mean	Std	CI95%	T	p-value	cohen-d
Δ_{model}	< 0	0.007	0.005	[0.01, inf]	7.573	1.193e-08	1.955
$\Delta_{overall}$	< 0	0.063	0.031	[0.05, inf]	10.789	5.730e-12	2.786
Δ_{data}	< 0	0.068	0.028	[0.06, inf]	12.942	7.070e-14	3.342
Δ_{synth}	< 0	0.386	0.113	[0.35, inf]	18.459	7.126e-18	4.766

Table D.4: Paired t-test results - $\Delta_{overall}$ versus Δ_{data}

H_0		Mean	Std	CI95%	T	p-value	cohen-d
Δ_{data}	$< \Delta_{overall}$	0.504	4.007	[-inf, 0.76]	-0.677	0.252	0.166

D.5 Evaluation of the influence of the synthetic data quality on the transfer learning approach for known data-generating process

In the following we report the results of supporting analysis that were performed to evaluate the influence of the synthetic data quality on our proposed approach

Figure D.1 shows the tuning results of the number of training iterations in the fine-tuning phase of our approach for different GAN qualities. We observe that the training iteration optimum ranges between 20,000 and 40,000. There is no strong correlation between the GAN quality and the duration of fine-tuning observable.

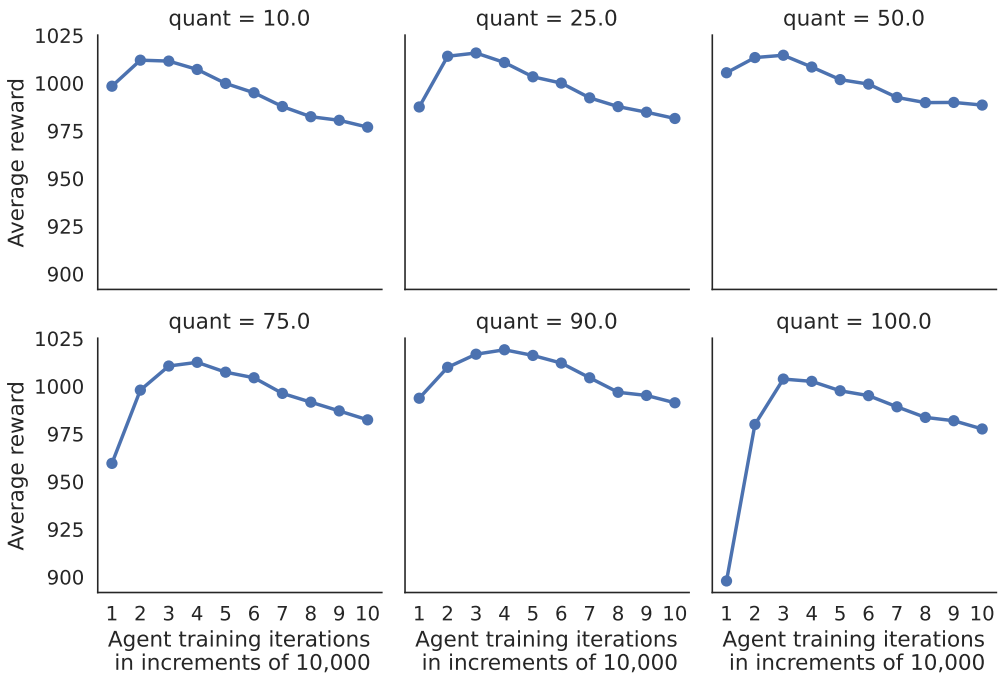


Figure D.1: Average reward on D_{real}^{val} under variation of agent training length on D_{real} for different GAN sample qualities

Figure D.2 is the same as Figure 4.5 but additionally shows outliers.

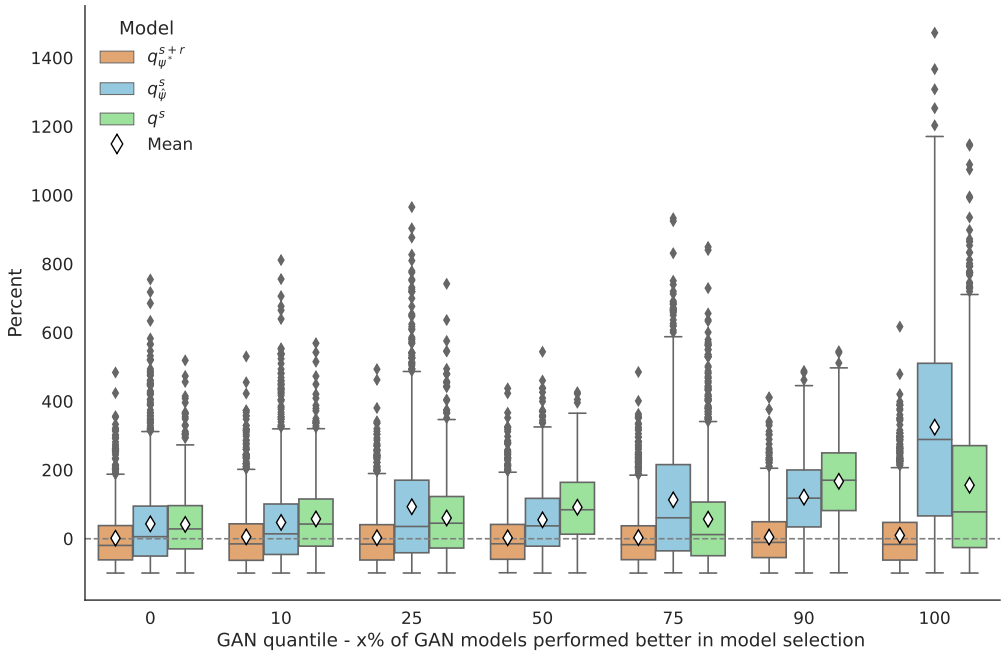


Figure D.2: Percentage deviation from average out-of-sample costs of q^* per day in D_{real}^{test} under variation of the GAN quality for a selected experiment run.

Figure D.3 focuses on the mean performance of $q^{s+r}_{\psi^*}$, $q^{s}_{\hat{\psi}}$ and q^{s} under variation of the GAN quality. In addition, it shows the standard deviation on D^{test}_{real}. We observe that the data quality has a strong influence on the models purely trained on synthetic data—performance deteriorates between 126 and 282 percentage points. However, the effect on $q^{s+r}_{\psi^*}$ is very limited—8 percentage points difference between best and worst mean performance.

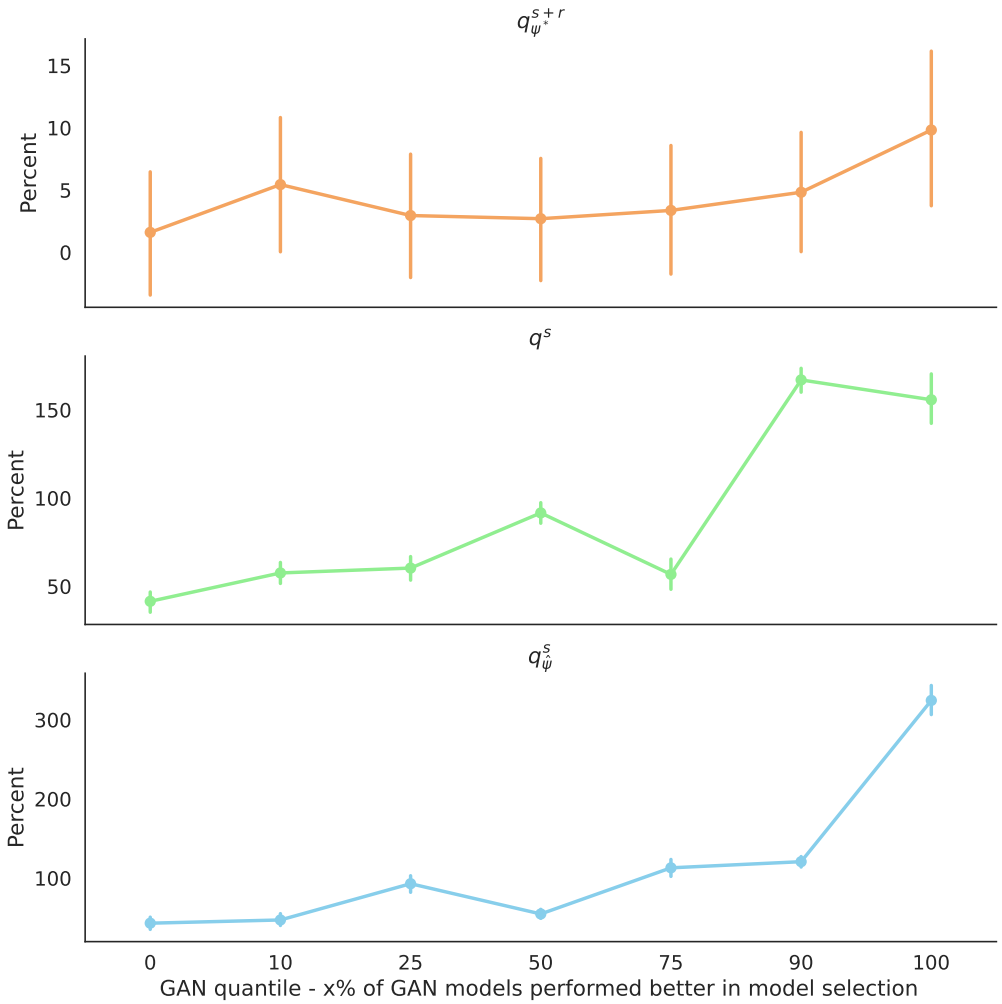


Figure D.3: Mean percentage deviation with standard deviation from average out-of-sample costs of q^* per day in D^{test}_{real} under variation of the GAN quality for a selected experiment run.

Figure D.4 shows the demand distribution of the real data and of samples generated by GANs of different quality. It does not include any feature dimensions. The figure validates that the demand distributions that the GANs have learned are different from one another.

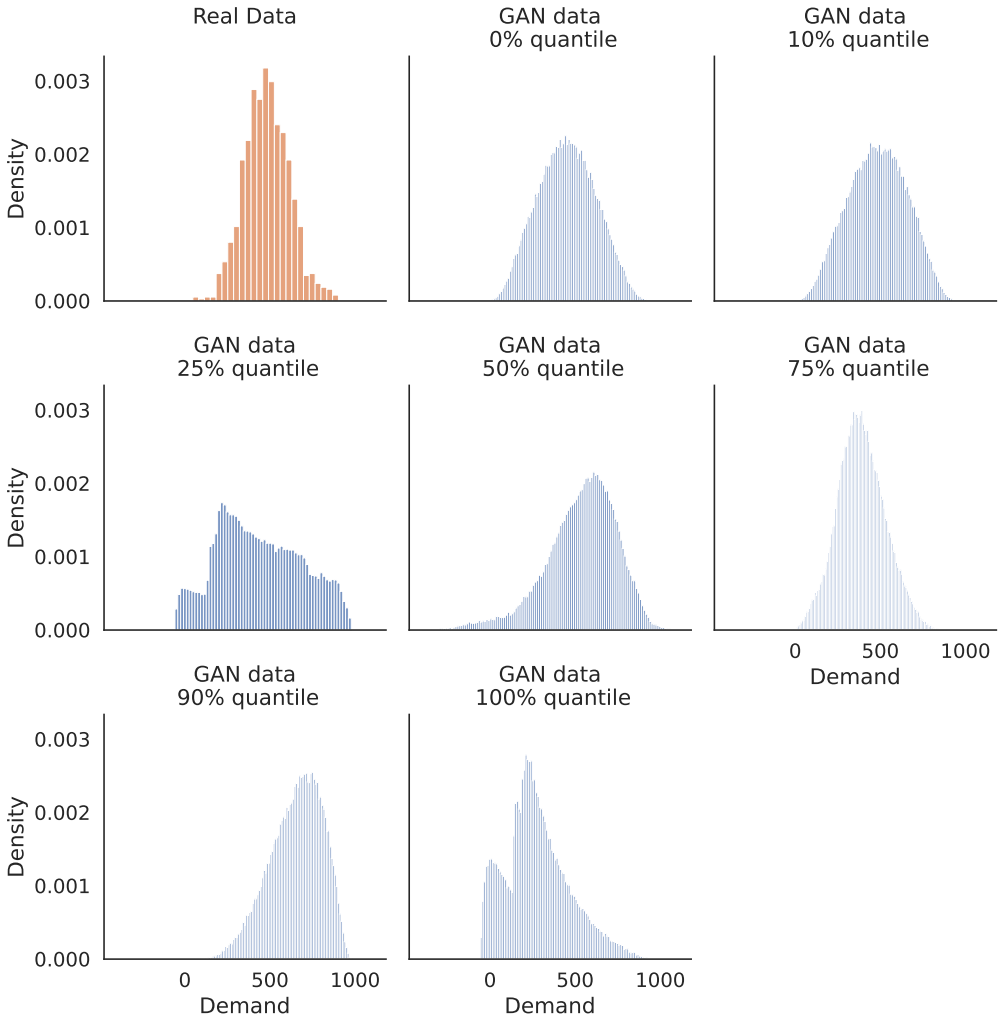


Figure D.4: Distribution of real demand vs. GAN generated demand

Figure D.5 is the same as Figure 4.6 but additionally shows outliers.

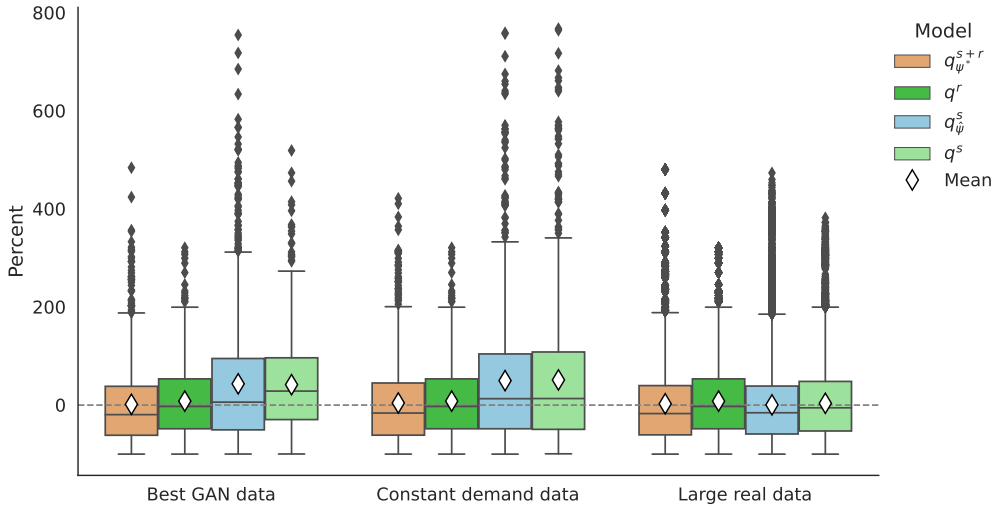


Figure D.5: Percentage deviation from average out-of-sample costs of q^* per day in D_{real}^{test} under variation of the synthetic data used during training for a selected experiment run.

D.6 Overview of bakery data feature information

In this section, we list the features contained in the bakery data set used in Section 4.4.3.

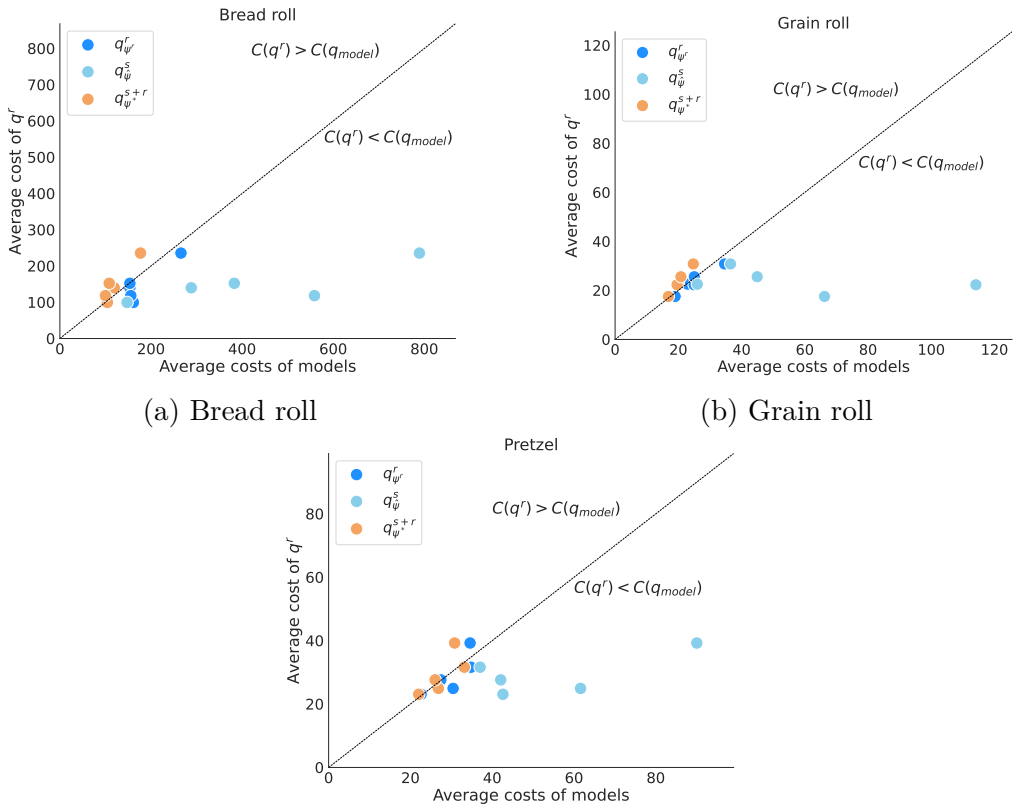
- shop_no: Bakery shop identifier
- product_no: Product identifier
- shop_rank: Target encoding of shop_no
- product_rank: Target encoding of product_no
- isoweek: Calendar week number
- weekday: Day of the week
- month: Month
- is_schoolholiday: Boolean whether the day is a school holiday
- is_holiday: Boolean whether the day is a holiday

- is_holiday_next2days: Boolean whether the day is a holiday in the next two days
- promotion_currentweek: Boolean whether there is a sales campaign in the current week
- promotion_lastweek: Boolean whether there was a sales campaign in the previous week
- forecast_weekly: Forecast of Holt-Winter's exponential smoothing with 7-day trend
- forecast_monthly: Forecast of Holt-Winter's exponential smoothing with 30-day trend
- temp_avg_celsius: Average temperature on day of sales
- rain_mm: Total rain in mm on day of sales

D.7 Evaluation of the transfer learning approach for unknown data-generating process

In Figure D.6 we plot the average costs of the DRL-based models $q^{s+r}_{\psi*}$, q^{r}_{ψ} and $q^{s}_{\hat{\psi}}$ versus the average costs of q^r on the bakery shop level. Dots left from the diagonal line indicate better performance than q^r and dots right from the diagonal the opposite. We observe that even though the performance of $q^{s}_{\hat{\psi}}$ fluctuates and is at the right half of the plot, applying our transfer learning approach to derive $q^{s+r}_{\psi*}$ brings us to the left side of the plot or at least close to the diagonal for all bakery shops.

In addition, Table D.5 provides the corresponding results of a paired t-test that tests whether the improvement of $q^{s+r}_{\psi*}$ over q^r is significant.



(a) Bread roll

(b) Grain roll

(c) Pretzel

Figure D.6: Average out-of-sample costs of DRL models per product on a shop level compared to q^r

Product	Shop	Mean	Std	CI95%	T	p-value	cohen-d
Bread roll	1	0.048	0.597	[-inf, 0.12]	1.078	0.859	0.068
Bread roll	2	-0.146	1.210	[-inf, 0.0]	-1.623	0.053	0.161
Bread roll	3	-0.288	1.219	[-inf, -0.14]	-3.173	0.001	0.295
Bread roll	4	-0.154	1.215	[-inf, -0.0]	-1.703	0.045	0.147
Bread roll	5	-0.249	1.007	[-inf, -0.13]	-3.321	0.001	0.279
Grain roll	1	-0.111	0.846	[-inf, -0.01]	-1.762	0.040	0.120
Grain roll	2	-0.122	1.412	[-inf, 0.05]	-1.156	0.125	0.111
Grain roll	3	-0.039	1.687	[-inf, 0.17]	-0.311	0.378	0.031
Grain roll	4	-0.186	0.894	[-inf, -0.08]	-2.795	0.003	0.201
Grain roll	5	-0.197	0.827	[-inf, -0.1]	-3.201	0.001	0.198
Pretzel	1	-0.046	0.718	[-inf, 0.04]	-0.853	0.197	0.051
Pretzel	2	0.050	0.763	[-inf, 0.14]	0.871	0.808	0.043
Pretzel	3	0.076	1.363	[-inf, 0.24]	0.753	0.774	0.066
Pretzel	4	-0.057	0.872	[-inf, 0.05]	-0.884	0.189	0.062
Pretzel	5	-0.215	0.899	[-inf, -0.1]	-3.207	0.001	0.244

Table D.5: Paired t-test results on bakery shop level with H_0: $C(q^{s+r}_{\psi^*}) < C(q^r)$.

Figure D.7 is the same as Figure 4.9 and Figure D.8 is the same as Figure 4.10 but both additionally show outliers.

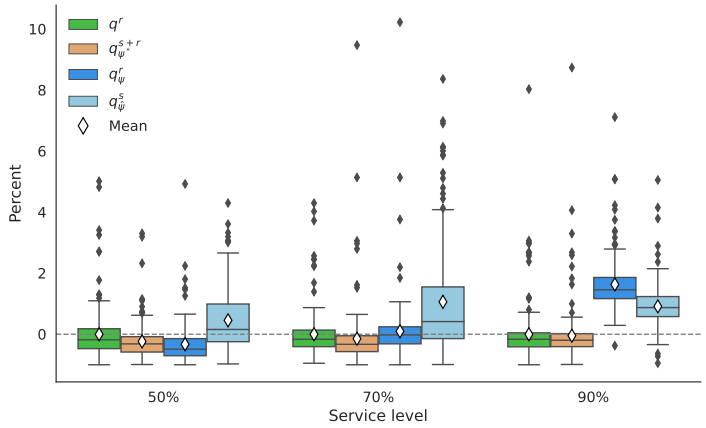


Figure D.7: Percentage deviation from average out-of-sample costs of q^r per day in D_{real}^{test} under variation of service level for a selected product and bakery shop.

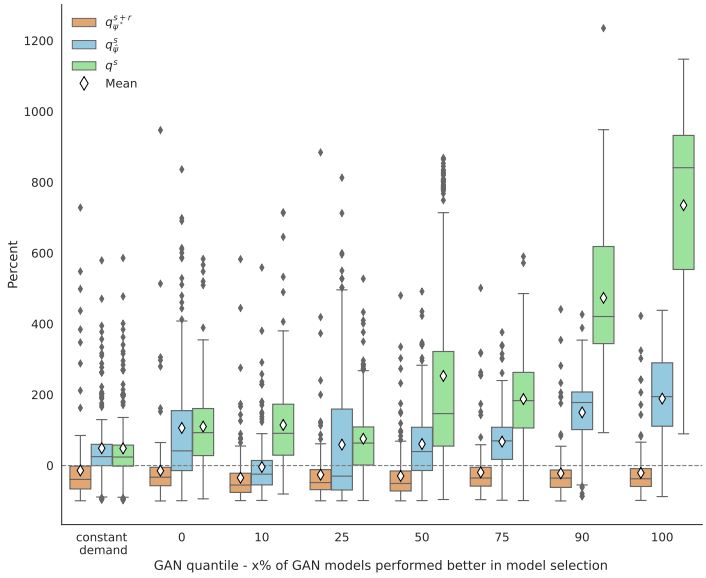


Figure D.8: Percentage deviation from average out-of-sample costs of q^r per day in D_{real}^{test} under variation of the GAN quality for a selected product and bakery shop.

List of Abbreviations

API	Application programming interface
cdf	Cumulative distribution function
CRM	Customer relationship management
CTGAN	Conditional tabular generative adversarial network
DDOM	Data-driven operations management
DF	Distribution fitting
DL	Prescriptive model proposed by Oroojlooyjadid, Snyder, et al. (2020)
DRL	Deep reinforcement learning
DTW	Decision tree-based weighted sample average approximation
ERM	Empirical risk minimization
ERP	Enterprise resource planning
GAN	Generative adversarial network
KNNW	K-nearest-neighbors-based weighted sample average approximation
KW	Kernel-based weighted sample average approximation
LR	Prescriptive model proposed by Beutel and Minner (2012) and Ban and Rudin (2019)
MM	Method of moments
MSE	Mean squared error
OLS	Ordinary least squares
OM	Operations management
PO	Predict-then-optimize
PP	Point-predictions
RFW	Random forest-based weighted sample average approximation
RMSE	Root-mean-square error
SAA	Sample average approximation
SID	Data set from Kaggle Store Item Demand Forecasting Challenge

sl	Service level
VAE	Variational autoencoder
WSAA	Weighted sample average approximation

List of Figures

List of Tables

Bibliography

Alaa, Ahmed, Boris van Breugel, Evgeny S. Saveliev, and Mihaela van der Schaar (2022). "How Faithful is your Synthetic Data? Sample-level Metrics for Evaluating and Auditing Generative Models". In: *International Conference on Machine Learning*, pp. 290–306.

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). "Wasserstein Generative Adversarial Networks". In: *International Conference on Machine Learning*, pp. 214–223.

Balaji, Bharathan, Jordan Bell-Masterson, Enes Bilgin, Andreas Damianou, and Chun Ye (2019). "ORL: Reinforcement Learning Benchmarks for Online Stochastic Optimization Problems". In: *ArXix preprint* arXiv:1911.10641.

Balugani, Elia, Francesco Lolli, Rita Gamberini, Bianca Rimini, and Mohamed Zied Babai (2019). "A Periodic Inventory System of Intermittent Demand Items with Fixed Lifetimes". In: *International Journal of Production Research* 57.22, pp. 6993–7005.

Ban, Gah-Yi and Cynthia Rudin (2019). "The Big Data Newsvendor: Practical Insights from Machine Learning". In: *Operations Research* 67.1, pp. 90–108.

Baxter, Jonathan, Andrew Tridgell, and Lex Weaver (2000). "Learning to Play Chess Using Temporal Differences". In: *Machine Learning* 40.3, pp. 243–263.

Bertsimas, Dimitris and Nathan Kallus (2020). "From Predictive to Prescriptive Analytics". In: *Management Science* 66.3, pp. 1025–1044.

Beutel, Anna-Lena and Stefan Minner (2012). "Safety Stock Planning under Causal Demand Forecasting". In: *International Journal of Production Economics* 140.2, pp. 637–645.

Boute, Robert N., Joren Gijsbrechts, Willem van Jaarsveld, and Nathalie Vanvuchelen (2021). "Deep Reinforcement Learning for Inventory Control: A Roadmap". In: *European Journal of Operational Research*.

Brock, Andrew, Jeff Donahue, and Karen Simonyan (2019). "Large Scale GAN Training for High Fidelity Natural Image Synthesis". In: *Proceedings of the 7th International Conference on Learning Representations*.

Brockman, Greg, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba (2016). "OpenAI Gym". In: *ArXix preprint* arXiv:1606.01540.

Buitinck, Lars, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake Vanderplas, Arnaud Joly, Brian Holt, and Gaël Varoquaux (2013). "API design for machine learning software: experiences from the scikit-learn project". In: *ArXix preprint* arXiv:1309.0238.

Buttler, Simone, Andreas Philippi, Nikolai Stein, and Richard Pibernik (2022). "A Meta Analysis of Data-Driven Newsvendor Approaches". In: *ICLR2022 ML Evaluation Standards Workshop*.

Carbonneau, Real, Kevin Laframboise, and Rustam Vahidov (2008). "Application of Machine Learning Techniques for Supply Chain Demand Forecasting". In: *European Journal of Operational Research* 184.3, pp. 1140–1154.

Che, Zhengping, Yu Cheng, Shuangfei Zhai, Zhaonan Sun, and Yan Liu (2017). "Boosting Deep Learning Risk Prediction with Generative Adversarial Networks for Electronic Health Records". In: *2017 IEEE International Conference on Data Mining*, pp. 787–792.

Çınar, Esra and Refik Güllü (2012). "An Inventory Model with Capacity Flexibility in the Existence of Advance Capacity Information". In: *Decision Support Systems* 53.2, pp. 320–330.

Csurka, Gabriela (2017). *Domain Adaptation in Computer Vision Applications*. Cham: Springer.

Darwish, M. A., M. Alkhedher, and Abdulrahman Alenezi (2019). "Reducing the Effects of Demand Uncertainty in Single-Newsvendor Multi-Retailer Supply Chains". In: *International Journal of Production Research* 57.4, pp. 1082–1102.

Esteban, Cristóbal, Stephanie L. Hyland, and Gunnar Rätsch (2017). "Real-Valued (Medical) Time Series Generation with Recurrent Conditional GANs". In: *ArXix preprint* arXiv:1706.02633.

Gijsbrechts, Joren, Robert N. Boute, Jan Albert van Mieghem, and Dennis Zhang (2021). "Can Deep Reinforcement Learning Improve Inventory Management? Performance and Implementation of Dual Sourcing-Mode Problems". In: *Manufacturing & Service Operations Management*.

GM, Harshvardhan, Mahendra Kumar Gourisaria, Manjusha Pandey, and Siddharth Swarup Rautaray (2020). "A Comprehensive Survey and Analysis of Generative Models in Machine Learning". In: *Computer Science Review* 38, p. 100285.

Gokhale, Apoorva, Chirag Trasikar, Ankit Shah, Arpita Hegde, and Sowmiya Raksha Naik (2021). "A Reinforcement Learning Approach to Inventory Management". In: *Advances in Artificial Intelligence and Data Engineering. Select Proceedings of AIDE 2019*. Ed. by Niranjan N. Chiplunkar and Takanori Fukao. 1st ed. 2021. Vol. 1133. Springer eBook Collection 1133. Singapore: Springer Singapore, pp. 281–297.

Gonçalves, João N.C., Paulo Cortez, M. Sameiro Carvalho, and Nuno M. Frazão (2021). "A Multivariate Approach for Multi-Step Demand Forecasting in Assembly Industries: Empirical Evidence from an Automotive Supply Chain". In: *Decision Support Systems* 142, p. 113452.

Goodfellow, Ian (2016). "NIPS 2016 Tutorial: Generative Adversarial Networks". In: *ArXix preprint* arXiv:1701.00160.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. MIT Press.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems* 27.

Gulrajani, Ishaan, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C. Courville (2017). "Improved Training of Wasserstein GANs". In: *Advances in Neural Information Processing Systems* 30.

Guo, Zhaoxia, W. K. Wong, and Min Li (2013). "A Multivariate Intelligent Decision-Making Model for Retail Sales Forecasting". In: *Decision Support Systems* 55.1, pp. 247–255.

Howard, Jeremy and Sylvain Gugger (2020). "Fastai: A Layered API for Deep Learning". In: *Information* 11.2, p. 108.

Huber, Jakob, Sebastian Müller, Moritz Fleischmann, and Heiner Stuckenschmidt (2019). "A Data-Driven Newsvendor Problem: From Data to Decision". In: *European Journal of Operational Research* 278.3, pp. 904–915.

Jordon, James, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N. Cohen, and Adrian Weller (2022). "Synthetic Data – What, Why and How?" In: *ArXix preprint* arXiv:2205.03257.

Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen (2018). "Progressive Growing of GANs for Improved Quality, Stability, and Variation". In: *Proceedings of the 6th International Conference on Learning Representations*.

Kingma, Diederik P. (2019). *An Introduction to Variational Autoencoders*. In collab. with Max Welling. Vol. 40. Foundations and Trends® in Machine Learning. Norwell, MA: Now Publishers.

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *ArXix preprint* arXiv:1312.6114.

Kraus, Mathias, Stefan Feuerriegel, and Asil Oztekin (2020). "Deep Learning in Business Analytics and Operations Research: Models, Applications and Managerial Implications". In: *European Journal of Operational Research* 281.3, pp. 628–641.

Law, Averill M. (2011). "How to Select Simulation Input Probability Distributions". In: *Proceedings of the 2011 Winter Simulation Conference (WSC)*, pp. 1389–1402.

Levi, Retsef, Martin Pál, Robin O. Roundy, and David B. Shmoys (2007). "Approximation Algorithms for Stochastic Inventory Control Models". In: *Mathematics of Operations Research* 32.2, pp. 284–302.

Levi, Retsef, Georgia Perakis, and Joline Uichanco (2015). "The Data-Driven Newsvendor Problem: New Bounds and Insights". In: *Operations Research* 63.6, pp. 1294–1306.

Lucic, Mario, Karol Kurach, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly (2018). "Are GANs Created Equal? A Large-Scale Study". In: *Proceedings of the 32nd International Conference on Neural Information*

Processing Systems. NIPS'18. Red Hook, NY, USA: Curran Associates Inc, pp. 698–707.

Makridakis, Spyros, Evangelos Spiliotis, Vassilios Assimakopoulos, Zhi Chen, Anil Gaba, Ilia Tsetlin, and Robert L. Winkler (2021). "The M5 Uncertainty Competition: Results, Findings and Conclusions". In: *International Journal of Forecasting*.

Mišić, Velibor V. and Georgia Perakis (2020). "Data Analytics in Operations Management: A Review". In: *Manufacturing & Service Operations Management* 22.1, pp. 158–169.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller (2013). "Playing Atari with Deep Reinforcement Learning". In: *ArXix preprint* arXiv:1312.5602.

Moor, Bram J. de, Joren Gijsbrechts, and Robert N. Boute (2021). "Reward Shaping to Improve the Performance of Deep Reinforcement Learning in Perishable Inventory Management". In: *European Journal of Operational Research*.

Notz, Pascal M. and Richard Pibernik (2022). "Prescriptive Analytics for Flexible Capacity Management". In: *Management Science* 68.3, pp. 1756–1775.

Oroojlooyjadid, Afshin, MohammadReza Nazari, Lawrence V. Snyder, and Martin Takáč (2021). "A Deep Q-Network for the Beer Game: Deep Reinforcement Learning for Inventory Optimization". In: *Manufacturing & Service Operations Management*.

Oroojlooyjadid, Afshin, Lawrence V. Snyder, and Martin Takáč (2020). "Applying Deep Learning to the Newsvendor Problem". In: *IISE Transactions* 52.4, pp. 444–463.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay (2011). "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12.85, pp. 2825–2830.

Philippi, Andreas, Simone Buttler, and Nikolai Stein (2021). "ddop: A Python Package for Data-Driven Operations Management". In: *Journal of Open Source Software* 6.66, p. 3429.

Piatkowski, Nico, Peter N. Posch, and Miguel Krause (2021). "How to Trust Generative Probabilistic Models for Time-Series Data?" In: *Learning and Intelligent Optimization. 15th International Conference, LION 15, Athens, Greece, June 20–25, 2021, Revised Selected Papers*. Ed. by Dimitris E. Simos, Panos M. Pardalos, and Ilias S. Kotsireas. Cham: Springer, pp. 283–298.

Punia, Sushil, Surya Prakash Singh, and Jitendra K. Madaan (2020). "From Predictive to Prescriptive Analytics: A Data-Driven Multi-Item Newsvendor Model". In: *Decision Support Systems* 136, p. 113340.

Qi, Meng, Yuanyuan Shi, Yongzhi Qi, Chenxin Ma, Rong Yuan, Di Wu, and Zuo-Jun Max Shen (2020). "A Practical End-to-End Inventory Management Model with Deep Learning". In: *SSRN Electronic Journal*.

Radford, Alec, Luke Metz, and Soumith Chintala (2016). "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *Proceedings of the 4th International Conference on Learning Representations*.

Raffin, Antonin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann (2021). "Stable-Baselines3: Reliable Reinforcement Learning Implementations". In: *Journal of Machine Learning Research* 22.268, pp. 1–8.

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Proceedings of Machine Learning Research 32. PMLR, pp. 1278–1286.

Sachs, Anna-Lena and Stefan Minner (2014). "The Data-Driven Newsvendor with Censored Demand Observations". In: *International Journal of Production Economics* 149, pp. 28–36.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov (2017). "Proximal Policy Optimization Algorithms". In: *ArXix preprint* arXiv:1707.06347.

Seyedan, Mahya and Fereshteh Mafakheri (2020). "Predictive Big Data Analytics for Supply Chain Demand Forecasting: Methods, Applications, and Research Opportunities". In: *Journal of big data* 7.1.

Shorten, Connor and Taghi M. Khoshgoftaar (2019). "A Survey on Image Data Augmentation for Deep Learning". In: *Journal of big data* 6.1, pp. 1–48.

Shorten, Connor, Taghi M. Khoshgoftaar, and Borko Furht (2021). "Text Data Augmentation for Deep Learning". In: *Journal of big data* 8.1, p. 101.

Silver, David, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis (2017). "Mastering the Game of Go without Human Knowledge". In: *Nature* 550.7676, pp. 354–359.

Sutton, Richard S. and Andrew Barto (2018). *Reinforcement Learning: An Introduction*. Second edition. Adaptive computation and machine learning. Cambridge, MA: The MIT Press.

Theis, Lucas, Aäron van den Oord, and Matthias Bethge (2016). "A Note on the Evaluation of Generative Models". In: *International Conference on Learning Representations*.

Turrini, Laura and Joern Meissner (2019). "Spare Parts Inventory Management: New Evidence from Distribution Fitting". In: *European Journal of Operational Research* 273.1, pp. 118–130.

Vanvuchelen, Nathalie, Joren Gijsbrechts, and Robert Boute (2020). "Use of Proximal Policy Optimization for the Joint Replenishment Problem". In: *Computers in Industry* 119, p. 103239.

Virtanen, Pauli, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, van der Walt, Stéfan J., Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vanderplas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul

van Mulbregt (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17.3, pp. 261–272.

Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba (2016). "Generating Videos with Scene Dynamics". In: *Advances in Neural Information Processing Systems* 29.

Weiss, Karl, Taghi M. Khoshgoftaar, and DingDing Wang (2016). "A Survey of Transfer Learning". In: *Journal of big data* 3.1, pp. 1–40.

Wen, Qingsong, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu (2021). "Time Series Data Augmentation for Deep Learning: A Survey". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*. Ed. by Zhi-Hua Zhou. International Joint Conferences on Artificial Intelligence, pp. 4653–4660.

Wilcoxon, Frank (1945). "Individual Comparisons by Ranking Methods". In: *Biometrics Bulletin* 1.6, p. 80.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush (2019). "HuggingFace's Transformers: State-of-the-art Natural Language Processing". In: *ArXix preprint* arXiv:1910.03771.

Xu, Lei, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni (2019). "Modeling Tabular Data using Conditional GAN". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Xu, Lei and Kalyan Veeramachaneni (2018). "Synthesizing Tabular Data using Generative Adversarial Networks". In: *ArXix preprint* arXiv:1811.11264.

Yahi, Alexandre, Rami Vanguri, Noémie Elhadad, and Nicholas P. Tatonetti (2017). "Generative Adversarial Networks for Electronic Health Records: A Framework for Exploring and Evaluating Methods for Predicting Drug-Induced Laboratory Test Trajectories". In: *ArXix preprint* arXiv:1712.00164.

Yang, Ya, Huihui Chi, Wei Zhou, Tijun Fan, and Selwyn Piramuthu (2020). "Deterioration Control Decision Support for Perishable Inventory Management". In: *Decision Support Systems* 134.

Zhang, Han, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas (2017). "StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks". In: *Proceedings of the IEEE International Conferenec on Computer Vision*, pp. 5908–5916.