Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 180 (2021) 649–655

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry 4.0 and Smart Manufacturing

Prototyping Machine-Learning-Supported Lead Time Prediction Using AutoML

Janek Bender[a]*, Jivka Ovtcharova[b]

[a,b] *FZI Research Center for Information Technology, Haid-und-Neu-Strasse 10-14, 76135 Karlsruhe, Germany*

Abstract

Many Small and Medium Enterprises in the domain of Make-To-Order- and Small-Series-Production struggle with accurately predicting lead times of highly customisable orders. This paper investigates an approach using AutoML integrated into existing enterprise systems in order to enable Lead Time Prediction based on Machine Learning models. This prediction is based on both order data from an ERP system as well as real-time factory state informed by an IIoT platform. We used simulation data to feed the AutoML model generation and developed a lightweight web-based microservice around it to infer lead times of incoming orders during live production. Using industry standards, this microservice can be seamlessly integrated into existing system landscapes. The simplicity of AutoML systems allows for swift (re)training and benchmarking of models but potentially comes at the cost of overall lower model quality.

Keywords: Production; Lead Time Prediction; AutoML; Machine Learning

1. Introduction

As we observed in practice at partnering Small and Medium Enterprises (SMEs) in the domain of Make-To-Order- (MTO) and Small-Series- (SS) Production, one particularly problematic area is the job scheduling. Highly customisable orders with short delivery times lead to complex combinatory optimisation problems. Planning needs to be done swiftly and accurately while being flexible enough to react to high priority orders coming in at any time. Planning horizons, especially with automotive suppliers, often span weeks at most.

* Corresponding author. Tel.: +49-721-9654-501; fax: +49-721-9654-501.
E-mail address: bender@fzi.de

1877-0509
10.1016/j.procs.2021.01.287

In contrast, long-term orders are often rescheduled in favour of smaller, more urgent contracts. These SMEs struggle to balance the conflicting goals of maximising adherence to delivery dates while at the same time minimising storage load and setup times.

While there exists a number of different scheduling methods like First In First Out, Slack Time Rule, or Shortest Processing Time, a(n estimated) Lead Time (LT) is required as input. For the mentioned SMEs, the process of Lead Time Prediction (LTP) poses a problem which needs to be addressed before any optimisation in terms of scheduling may be performed. Especially in an MTO or SS scenario, this prediction is quite challenging because of the high product variance and the varying order parameters. Different approaches have been discussed in the literature as well as among practitioners.

This paper focuses on Machine Learning (ML) supported LTP and presents an approach tightly integrated into an existing system landscape using AutoML. Section 2 gives a brief overview of related works and highlights aspects that need to be researched more in depth. Section 3 presents the concept of LTP understood as a function of both order features and factory state. While section 4 provides a glimpse at a prototype, section 5 concludes this paper with an outlook into future work.

2. Related Work

We have studied several existing works with a mostly narrow view on ML-supported LTP with the goal to identify the best documented approaches to this problem.

2.1. Literature Reviews & Research Gaps

In a systematic literature review conducted by [1], the authors looked at Data Mining applications in the production domain. They noted a Regression Tree (RT) approach for LTP exists but further research is required. About nine years later, [2] conducted another literature review and found, while some progress had been made in LTP specifically, but also in the broader application of Data Mining and Artificial Intelligence (AI) in production, more research is needed as models tend to not generalise well. Furthermore, the authors stressed that more attention needs to be directed towards processing complex data types in real-time using highly integrated solutions. An emphasis on self-healing, -optimising, and -updating systems and models was made.

2.2. Previous Works on ML-supported LTP

In [3], the authors applied Regression Models (RM) and Artificial Neural Networks (ANN) to batch processing in the process industries. They generated randomised job data to validate their method. [4] proposed RTs for LTP in their hypothetical MTO job shop. They employed a sophisticated feature selection process for their model which was trained on data generated from a simulation. [5] applied Support Vector Machines (SVM) on LTP in a simulated MTO job shop and compared their approach to other Data Mining techniques. They have identified the overall state of the production system to be most influential in accurately predicting lead times. They too see model adaptability as an important alley for further research. [6] also used SVMs but on a real-world data set from the batch production of metallic components in the aerospace industry. They utilised 524 data sets and compared different models using between four and twelve features. [7] set their focus towards Waiting Time Prediction (WTP), an important part of LTP. The authors performed feature identification and cycle time estimation for the production of semi-conductors based on 123 simulated scenarios with a set of 50 features. They then employed Naïve Bayes Classifiers to select influential features and Conditional Mutual Information Maximisation to predict waiting times. [8] developed a knowledge-based system in the domain of Engineer-To-Order production. In said system, incoming orders were compared to a database of already completed orders using similarity measures. Similar completed orders were then utilised to predict process variables like LT. They validated their method on a real-world use case in the moulding industry. [9] simulated a linear semi-conductor production line and derived several RMs for LTP from it. [10] applied Bayesian Networks in combination with Decision Trees on real-world steel production data.

They were able to predict both production loads and production times. [11] used MES log data and simulations to feed a Multivariate Regression for LTP in flow-shop systems. [12] compared analytical methods, Linear Regression (LR), RTs, and SVMs for LTP using five features trained and tested on MES log data of a flow-shop system. They concluded that, while RTs performed best, LR was simpler and thus faster to apply. The authors also made the point that real-time data gathering and retraining of the models is a crucial factor in these methods providing the best results. Furthermore, they suggested tight integration with the Digital Twin. In another work by [13], the authors expanded on their approach to allow for online LTP and model retraining at runtime. [14] focused on comparing different regression algorithms for LTP, namely Linear, Ridge, and Lasso Regression, Multivariate Adaptive Regression, SVMs, the k Nearest Neighbour algorithm, RTs, as well as Random Forests. They applied these algorithms on a semi-conductor production with three process steps providing data feeding eight features. A novel approach took [15] in that they did not understand LTP as a regression but as a classification problem. They used an SVM to classify order data into eight different ranges, i.e. classes of expected lead time. In other related works by [16,17], instead of LTP, the prediction of transition times between different production steps was explored. They made the point that Waiting Times often make up large portions of overall LT but are hard to estimate due to their high variance.

2.3. Literature Conclusion & Open Issues

Summarising the related works, we note a few points. ML-supported LTP is still an ongoing research topic with several unresolved problems. Many authors were dealing with simulations rather than real-world data. Even when real-world data was used, it was often based on logs from MES. This likely indicates an insufficient integration of these models into production-grade MES. Efficient consolidation and pre-processing of real-time data from different data sources both on the shop floor and in upstream systems like ERP needs to be considered more strongly.

Furthermore, many works revolved around flow-shop systems, particularly in the semi-conductor production. Strict MTO or even small SS scenarios were often not at the core. Understandably so as generalisation in a true MTO scenario might not always be feasible.

Looking at the ML methods applied, most authors relied on supervised learning techniques. The models they used understand LTP as a regression problem and often underestimated factors outside of the order parameters, like production load or workforce availability. The human factor was rarely, if at all, considered. Many of the authors cited also noted that these models did not generalise well to new problems as LTP is highly reliant on company-specific circumstances. For that reason, authors stress more research needs to be done in the area of self-adapting models which are able to retrain at runtime or even use online training algorithms. In addition, these models should be tightly integrated with other state-of-the-art technologies and concepts like MES, IoT, and Digital Twins. Ideally, the ML pipeline is directly integrated with the entire automation pyramid.

Lastly, while many authors compared different algorithms, we did not see much work on automating the process of algorithm selection. Especially AutoML approaches are a promising avenue to look at. These show potentials in easing the process of retraining as well as enabling non-data scientists in the SMEs to monitor and train models in their own domains. With this paper and subsequent works, we aim to contribute to some of these points.

3. Problem Modelling

In this paper, we consider ML-supported LTP as a regression problem where the goal is to predict the LT of individual production steps. Where the sum of the LTs of all production steps forms the LT of the order. The two main inputs for our LTP are the orders and the factory state at the time of prediction.

3.1. Data Model

Orders in the MTO and SS domain provide several interesting features such as the product that is to be produced, its configurations and parameters, product quantities, and due dates. Note that we define product configurations as a list of selectable features and as such leading to categorical data while parameters refer to continuous number values or strings. Product variance can thus grow exponentially, leading to a different LT for any given order.

Although in reality, many technically possible product combinations are not offered to the customer in order to manage complexity or because of low demand.

Factory state on the other hand contains anything from resource availability (machines, labour, raw materials) to factory load, with additional parameters affecting LTs depending on the concrete problem (e.g. temperature on the shop floor or setup times). Fig. 1 illustrates the idea. We see individual orders as a set of parameters and selected configurations with defined priorities and due dates. Each order is varying in complexity, dictating different production steps along the factory layout. The factory is characterised by a limited set of both human and machine resources. The predictive model for LTP considering orders as well as factory state is directly integrated into the MES, where both data and control are consolidated.

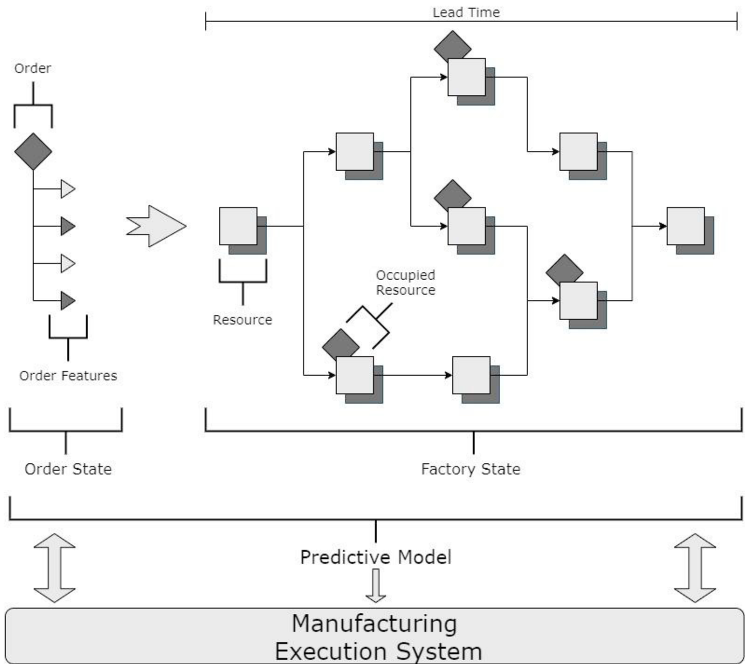

Fig. 1. Model for LTP.

4. Prototype

We have developed a prototype to explore the above concept combined with an AutoML approach. The prototype was validated on simulated data.

4.1. Architecture

Architecturally, our system is inspired by [13] insofar as both theirs and our design share some key components. We both rely on an ERP system in the background to provide order data and an MES to control the shop floor. And of course, there is an ML-component which handles training and inference. Fig. 2 provides an overview of the basic architecture. In our case, the simulation is enhanced by self-developed modular web-based microservices written in Java and Kotlin which are capable of consolidating the order data with the factory state in order to perform LTP and feed back the predicted LTs to up- and downstream systems. Commercial ERP systems and MES can be integrated using industry standards such as REST or OPC-UA.

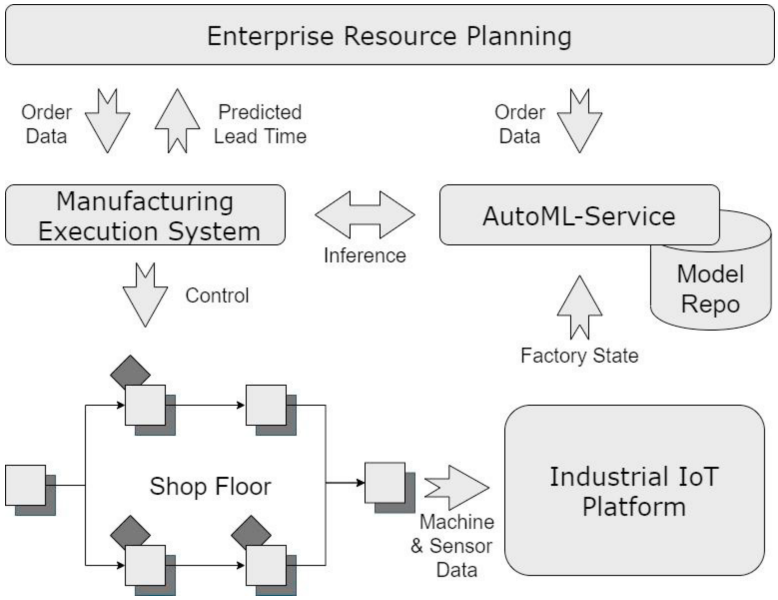

Fig. 2. Prototype Architecture.

The shop floor is monitored by an Industrial Internet of Things (IIoT) platform. At its core, we rely on *Apache StreamPipes (incubating)* [18] as a data stream processing tool. Machine sensors and controllers like PLCs can be connected by *StreamPipes Connect* [19]. The tool in return filters and processes this time series data from different sources and provides us with real-time factory state as well as various KPIs.

For the ML-component, we opted for the free open source version of *H20.ai* (https://www.h2o.ai/). Its firm base in the Java ecosystem allows us to seamlessly integrate the model inference into our microservice architecture. More complex interactions, such as model training, are conducted language-agnostically through its REST interface.

4.2. Simulated Data

In order to test this first design, we relied on a simulation model based on one of our SME partner's use cases. In the simplified version of this use case, three somewhat similar but highly customisable base products, abstracted into *Simple, Advanced*, & *Complex Product*, are produced with different selectable configurations as well as settable parameters. In addition, orders generated for these products contain quantities, priorities, and due dates. For each run, we generated between 100 and 1.000 orders randomly with quantities ranging from single digits up to 100. These orders are then passed to a simple discrete event simulation integrated into our microservice architecture resulting in datasets of about 15.000 to 250.000 data points per simulation run.

4.3. Training & Inference using AutoML

The data generated by the simulation is then exported and used to train and compare a multitude of models. Note that we did not fully automate the training process as of yet. This data is a consolidation of both order data and factory state with currently 32 features. We use a simple label encoder to prepare the data. As the simulation data is complete and does not contain any corruptions, missing value imputation or further cleaning is not required. H20.ai's AutoML capabilities simplify the process of training and evaluating different models. To keep it basic, we do not perform any meaningful parameter tuning ourselves. Instead, we have H20.ai train hundreds of models at a time and select one for each of the three base products based on the lowest Root Mean Squared Error (RMSE). Note that we deliberately exclude Deep Learning to shorten the runtime. In our preliminary tests, the Gradient Boosting Machines (GBM) turned out to deliver the best overall results with an RMSE on the test set of about the statistical variance of our simulation model. We did not notice any relevant alterations of this figure with a growing dataset size.

However, as a disclaimer, the simulation model used to generate this data is very limited. We expect that imperfect real-world data provides a more challenging problem.

5. Conclusion & Outlook

In this paper, we presented a simple approach towards integrating ML-supported LTP into a greater enterprise architecture in a manufacturing SME using AutoML. A microservice architecture was developed that mimicked the order scheduling function of an MES which used H20.ai to infer LTs on newly arriving orders. The corresponding models were trained on simulation data by using H20.ai's AutoML capabilities. We preliminarily conclude that the concept of ML-supported LTP is technically feasible and beneficial for order scheduling in SMEs. The speed and simplicity of operating an AutoML system compared to regular ML frameworks is a significant advantage, even if it comes at the cost of lower quality models. However, a better integration of automated model monitoring and retraining needs to be investigated. Concept drift and poor generalisation pose significant challenges with these systems, especially in the domain of MTO- and SS-Production.

In the near future, we plan to validate the approach on real-world data. Obtaining said data is still a major struggle within the research community. While there are public data sets available for other areas in the production domain, such as Predictive Maintenance, there is none that fits our problem. Therefore, we are currently aggregating data from partnering companies manually by tapping ERP-, PDM-, and ME-Systems for order, product, and shop floor data accordingly. In this regard, fully connecting the shop floor in order to get meaningful data during production is still a challenge for many SMEs. We expect digitising shop floors in SMEs in a way that a holistic factory state can be inferred to remain a problem for the coming years.

Another important task is to better integrate the AutoML automated retraining aspect into the microservice landscape in order to address the concerns of concept drift. In this step, we will also explore the AutoML market more thoroughly and look at other frameworks such as TPOT (http://epistasislab.github.io/tpot/) or auto-sklearn (https://github.com/automl/auto-sklearn). Lastly, while our basic simulation is sufficient for this preliminary work, we seek to explore more sophisticated real-time simulation as applied in a previous work of ours [20].

References

[1] Choudhary AK, Harding JA, Tiwari MK. Data mining in manufacturing: a review based on the kind of knowledge. J Intell Manuf 2009;20(5):501–21. https://doi.org/10.1007/s10845-008-0145-x.

[2] Cheng Y, Chen K, Sun H, Zhang Y, Tao F. Data and knowledge mining with big data towards smart production. Journal of Industrial Information Integration 2018;9:1–13. https://doi.org/10.1016/j.jii.2017.08.001.

[3] Raaymakers W, Weijters A. Makespan estimation in batch process industries: A comparison between regression analysis and neural networks. European Journal of Operational Research 2003;145:14–30.

[4] Öztürk A, Kayalıgil S, Özdemirel NE. Manufacturing lead time estimation using data mining. European Journal of Operational Research 2006;173(2):683–700. https://doi.org/10.1016/j.ejor.2005.03.015.

[5] Alenezi A, Moses SA, Trafalis TB. Real-time prediction of order flowtimes using support vector regression. Computers & Operations Research 2008;35(11):3489–503. https://doi.org/10.1016/j.cor.2007.01.026.

[6] Cos Juez FJ de, García Nieto PJ, Martínez Torres J, Taboada Castro J. Analysis of lead times of metallic components in the aerospace industry through a supported vector machine model. Mathematical and Computer Modelling 2010;52(7-8):1177–84. https://doi.org/10.1016/j.mcm.2010.03.017.

[7] Meidan Y, Lerner B, Rabinowitz G, Hassoun M. Cycle-Time Key Factor Identification and Prediction in Semiconductor Manufacturing Using Machine Learning and Data Mining. IEEE Trans. Semicond. Manufact. 2011;24(2):237–48. https://doi.org/10.1109/TSM.2011.2118775.

[8] Mourtzis D, Doukas M, Fragou K, Efthymiou K, Matzorou V. Knowledge-based Estimation of Manufacturing Lead Time for Complex Engineered-to-order Products. Procedia CIRP 2014;17:499–504. https://doi.org/10.1016/j.procir.2014.01.087.

[9] Li M, Yang F, Wan H, Fowler JW. Simulation-based experimental design and statistical modeling for lead time quotation. Journal of Manufacturing Systems 2015;37:362–74. https://doi.org/10.1016/j.jmsy.2014.07.012.
[10] Mori J, Mahalec V. Planning and scheduling of steel plates production. Part I: Estimation of production times via hybrid Bayesian networks for large domain of discrete variables. Computers & Chemical Engineering 2015;79:113–34. https://doi.org/10.1016/j.compchemeng.2015.02.005.
[11] Pfeiffer A, Gyulai D, Kádár B, Monostori L. Manufacturing Lead Time Estimation with the Combination of Simulation and Statistical Learning Methods. Procedia CIRP 2016;41:75–80. https://doi.org/10.1016/j.procir.2015.12.018.
[12] Gyulai D, Pfeiffer A, Nick G, Gallina V, Sihn W, Monostori L. Lead time prediction in a flow-shop environment with analytical and machine learning approaches. IFAC-PapersOnLine 2018;51(11):1029–34. https://doi.org/10.1016/j.ifacol.2018.08.472.
[13] Gyulai D, Pfeiffer A, Bergmann J, Gallina V. Online lead time prediction supporting situation-aware production control. Procedia CIRP 2018;78:190–5. https://doi.org/10.1016/j.procir.2018.09.071.
[14] Lingitz L, Gallina V, Ansari F, Gyulai D, Pfeiffer A, Sihn W et al. Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer. Procedia CIRP 2018;72:1051–6. https://doi.org/10.1016/j.procir.2018.03.148.
[15] Lim ZH, Yusof UK, Shamsudin H. Manufacturing Lead Time Classification Using Support Vector Machine. In: Badioze Zaman H, Smeaton AF, Shih TK, Velastin S, Terutoshi T, Mohamad Ali N et al., editors. Advances in Visual Informatics. Cham: Springer International Publishing; 2019, p. 268–278.
[16] Schuh G, Prote J-P, Sauermann F, Franzkoch B. Databased prediction of order-specific transition times. CIRP Annals 2019;68(1):467–70. https://doi.org/10.1016/j.cirp.2019.03.008.
[17] Schuh G, Prote J-P, Hünnekes P, Sauermann F, Stratmann L. Impact of Modeling Production Knowledge for a Data Based Prediction of Transition Times. In: Ameri F, Stecke KE, Cieminski G von, Kiritsis D, editors. Advances in Production Management Systems. Production Management for the Factory of the Future. Cham: Springer International Publishing; 2019, p. 341–348.
[18] Riemer D, Kaulfersch F, Hutmacher R, Stojanovic L. StreamPipes. In: Eliassen F, Vitenberg R, editors. Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems - DEBS '15. New York, New York, USA: ACM Press; 2015, p. 330–331.
[19] Zehnder P, Wiener P, Straub T, Riemer D. StreamPipes Connect: Semantics-Based Edge Adapters for the IIoT. In: Harth A, Kirrane S, Ngonga Ngomo A-C, Paulheim H, Rula A, Gentile AL et al., editors. The Semantic Web. Cham: Springer International Publishing; 2020, p. 665–680.
[20] Bender J, Deckers J, Fritz S, Ovtcharova J. Closed-Loop-Engineering – Enabler for Swift Reconfiguration in Plant Engineering. In: Proceedings of the European Modeling and Simulation Symposium 2019, 138-144.