CPSL
CONFERENCE ON PRODUCTION SYSTEMS AND LOGISTICS
CPSL 2020

1st Conference on Production Systems and Logistics

Part Based Mold Quotation With Methods Of Machine Learning

Sebastian Brede[1], Benjamin Küster[1], Malte Stonis[1], Mike Mücke[2], Ludger Overmeyer[1]
[1]*Institut für Integrierte Produktion Hannover gGmbH, Hannover, Deutschland*
[2]*Phoenix Contact GmbH & Co. KG, Blomberg, Deutschland*

Abstract

The dominating mass-manufacturing process of today is plastic injection molding. This production process uses economies of scale because parts are produced in seconds at marginal cost of plastics. However, upfront investment costs for the tooling of molds are the basis for deciding if a mold is tooled and hence if a part is viable for mass-production. If tooling costs are too high, a product may not viable for production. If tooling costs are estimated too low by the tool shop, contract implications may arise. Because injection molds differ in their complexity, price estimations for the tooling of molds are an ongoing quest. There are various methods for estimating the costs of injection molds such as rule based, analytical or data driven approaches. The advantage of data driven approaches is the ability of adjusting to historical production data as well as readjusting while training on new batches of recent data.

The focus of our research was to support the quotation process of tool shops. To this end, we studied a data driven machine learning approach.

The goal of this research is to develop a method with humanlike quotation accuracy, achieve standardization, factor in historic quotation data and shorten quotation process times. The machine learning approach developed is based on geometry data of parts and additional meta-information.

Within this research, a system was developed to interact with live production systems of an electronic part producing tool shop.

The method developed was trained and validated on production data in a case study. To enhance the quotation process, the method developed was embedded into a server-based application with a web user interface and interfaces to live production systems for the automation of processes.

Keywords

Machine Learning; Injection Molding; Cost Estimation

1. Introduction

1.1 Production by injection molding – quotation of molds

Injection molding of thermoplastics is a mass manufacturing process which enables an economic production of goods, i.e. a fast production process which needs little to no postprocessing of parts. To ensure an economic efficiency of production, costs must be considered as early as possible. With injection molding, there always needs to be a distinct mold for each part produced.

Production costs may be divided into the two categories of fixed costs and variable costs. While variable costs depend on production quantity such as material, energy and machine costs (as a product of machine rate and required process time), fixed costs are often created before the actual production of plastic parts begins. While the retrieval of variable costs is trivial, fixed costs may be unpredictable due to the manufacturing processes of molds and can create a great amount of debt that needs to be compensated during production.

This research is focussed on the estimation of fixed costs that result from the tooling of molds. The upfront costs of mold tooling not only consist of machining costs such as milling, sanding, drilling or electrical discharge machining but also of manual labour-intensive assembly and inspection. Because thermoplastic materials have a high shrinkage ranging from $s_l = 0.015 \frac{cm}{cm}$ to $s_h = 0.060 \frac{cm}{cm}$ [1], machining molds of tightly tolerated parts poses a challenge. To face the resulting difficulties, molds are often quoted by workers with a high level of experience in the field of tooling and quoting of molds. This experience-based cost-method is subject to a worker's individual abilities and therefore prone to errors. The aim of this research is to evaluate a method that supports the quotation process of injection molds and accounts for the characteristics of given historic data.

1.2 State of the art

During the last decades, various attempts at estimating mold costs have been made. The research presented in this section therefore focuses on methods that establish machine learning techniques for cost estimation.

In 2003, Wang et al. researched a case-based reasoning (CBR) system to aid workers' quotation of injection molds [2]. For retrieving similar parts, they used artificial neural networks (ANN) to calculate similarities between parts. The input parameters consisted of meta-parameters such as injection part type, tolerance requirements and parting plane complexity as well as direct properties such as part size and number of internal / external undercuts. The validation performance of the CBR-system Wang et al.' proposed was not disclosed.

In 2005, Xu et al. researched a Fuzzy ANN for estimating injection mold design time [3]. Their dataset consisted of 72 samples where 60 samples were used for training. Besides the direct parameters of part size, part precision and minimum thickness, they input the meta-parameters part structure and part shape. They report mean absolute percentage errors of 9 - 12 % for the models they researched.

Wang et al. researched the cost estimation of injection molded parts multiple times [4],[5],[6]. However, they did not focus their research on the mold's tooling but rather calculated the whole design to market costs, including the injection molding production process, of a given part. In 2007, they researched a back-propagation ANN with 1-2 hidden layers with 34- 36 hidden neurons [4]. The input parameters were part- and mold-specific; material, weight, surface area, parting line's projection surface area, bounding box dimensions, volume, wall-thickness and cavity quantity. They used an industry-related dataset with 1070 samples.

In 2010, Che from the same research institution as Wang et. al. evaluated a particle swarm optimization (PSO) based ANN on a bigger dataset containing 2100 samples using the same parameters as Wang et al. [5].

Wang et al. picked the PSO-ANN up again in their 2013 research [6]. The ANN studies mentioned above have a remarkably low relative error ("cost percentage error") of maximum 1.95 % [6], 0.72 % [5] and 1.1 % [6] in their 20 validation samples from a total dataset of 2100 samples although they incorporate only basic part features and a small ANN.

Florjanic et al. undertook expert's opinion to decide upon 22 input variables which they used for the estimation of total machining hours spent per part via ANN [7]. Besides part specific input-variables such as part material, surface area and complexity, they also considered incorporating mold variables such as

mold dimensions, parting line complexity, slider- and lifter-count. These mold variables created the need to include expert's opinion into the quotation process. Their final ANN consisted of one hidden layer with four hidden neurons. Their dataset consisted of 105 samples with 20 % used for evaluation. They encountered maximum relative errors of 38 % during their testing and advised a safety factor of 25 % to be applied to their model for performing predictions.

Eilert et al. researched an ANN method to predict manufacturing costs of sheet metal forming dies [8]. They automated the construction of dies from a given part and estimated tooling costs given on automatically constructed die features, such as ejector and stage count. They used a total of 30 samples as dataset for training and testing.

Kočov et al. developed an expert system for assistance with injection mold's cost estimation [9]. They designed their expert system with focus on guiding an expert worker through a quick, basic design of a possible injection mold for a given part. They then infer an estimated cost from the given mold design.

Börzel and Frochte investigated a method to estimate die-casting molds' tooling costs from small datasets [10]. They reduced geometry information from 3D computer aided design (CAD) data to scalar features. They considered common geometric properties as well as ratios of these and expert parameters (e.g. surface area influencing demolding). They performed a principal component analysis and found the first three principal components describing 96 % of the data variance. These principal components incorporate six parameters, namely: Part volume, surface area, projected bounding box surface area, bounding box volume and two specified bounding box side lengths. It is noteworthy that no expert parameter nor any of their defined ratios was part of the main principal components. With the mapped input parameters, they evaluated various machine learning techniques such as regression models trained by ANN, K-nearest neighbor regression and random forests regression (RFR). They grouped their 700 samples dataset into three groups of mold designs and trained distinct models for these. After this they performed evaluation on 20 % of their dataset of 700 samples. They identified outliers with more than 50 % error and achieved a mean relative uncertainty of 10 % to 14 %.

2. Conception of quotation method and model architecture

2.2 Conception of quotation method

With the methods researched by the authors mentioned above, parameters were used that describe only one single property of the part, e.g. part surface area and part volume. However, this poses a problem, because these parameters have a one-to-many relation: While, for example, a part's volume is dependent on a part's geometry and there exists exactly one volume for a given geometry, there may be infinite geometries possible for a given volume. This means, these parameters can only represent parts to a certain degree and will fail when parts with the same scalar parameters but different underlying geometries are considered.

Some authors added complexity parameters such as part complexity and parting line complexity [7]. But these parameters bear two problems: On one hand, these input parameters are a dimensionality reduction of the actual features which cannot be reconstructed by the input parameter's description. On the other hand, an expert's opinion is needed to decide on a part's level of complexity. Therefore, a method which accounts for the actual geometric features and their spatial relations is needed.

The seemingly natural way for inputting the part's geometric three-dimensional (3D) data into neural networks would be to input vertices' normalized position data into 3D convolution layers. However, this poses several challenges: On the one hand, 3D CAD data is used for part and mold construction. While 3D CAD data is unit driven, its surface-topology is secondary. It is defined / visualized during the tessellation stage by means of algorithms as marching cubes or marching tetrahedrons. That means, the resulting

topology which defines the vertices' quantities and positioning, is dependent on the algorithms used for exporting the part data.

On the other hand, 3D data can easily be corrupted, i.e. non-manifold surfaces or bad distribution of vertices. To overcome these difficulties, a visual representation of a part's 3D-data is proposed.

Because a 3D part has more than one possible side, a multi-view convolutional model, based on multi-view projections of the input part, is chosen for the processing of 3D data. These kinds of models were researched by Su et al. to classify various 3D models with a high precision compared to other methods such as 3D-voxel grids, 3D-shape descriptors or pointclouds [11].

To guarantee every part an equal image proportion, the part needs to be normalized to fit in set input layer boundaries. To account for meta-information not representable within a part's geometry - such as planned mold features – we propose an extra, parallel neural network.

Both model architectures are then merged to output the quotation estimate. The choice of meta-parameters to be included in the meta-information is made by examining expert's opinion, and by performing a statistical effect analysis on the data. The meta-parameters chosen are number of part-cavities in molds, mold-type, project type and runner-type.

2.3 Model architecture

The underlying model is separated into two parts, one convolutional part for image processing as well as one multi-perceptron part for meta-information processing. A schematic of the used model architecture is displayed in Figure 1. The convolutional part of the model (Figure 1, upper neural network part) is an adoption of Su et al.'s multi-view convolutional neural network, which in turn is an adoption of AlexNet, an early image processing model used for image recognition [12] with an additional view pooling layer [11].

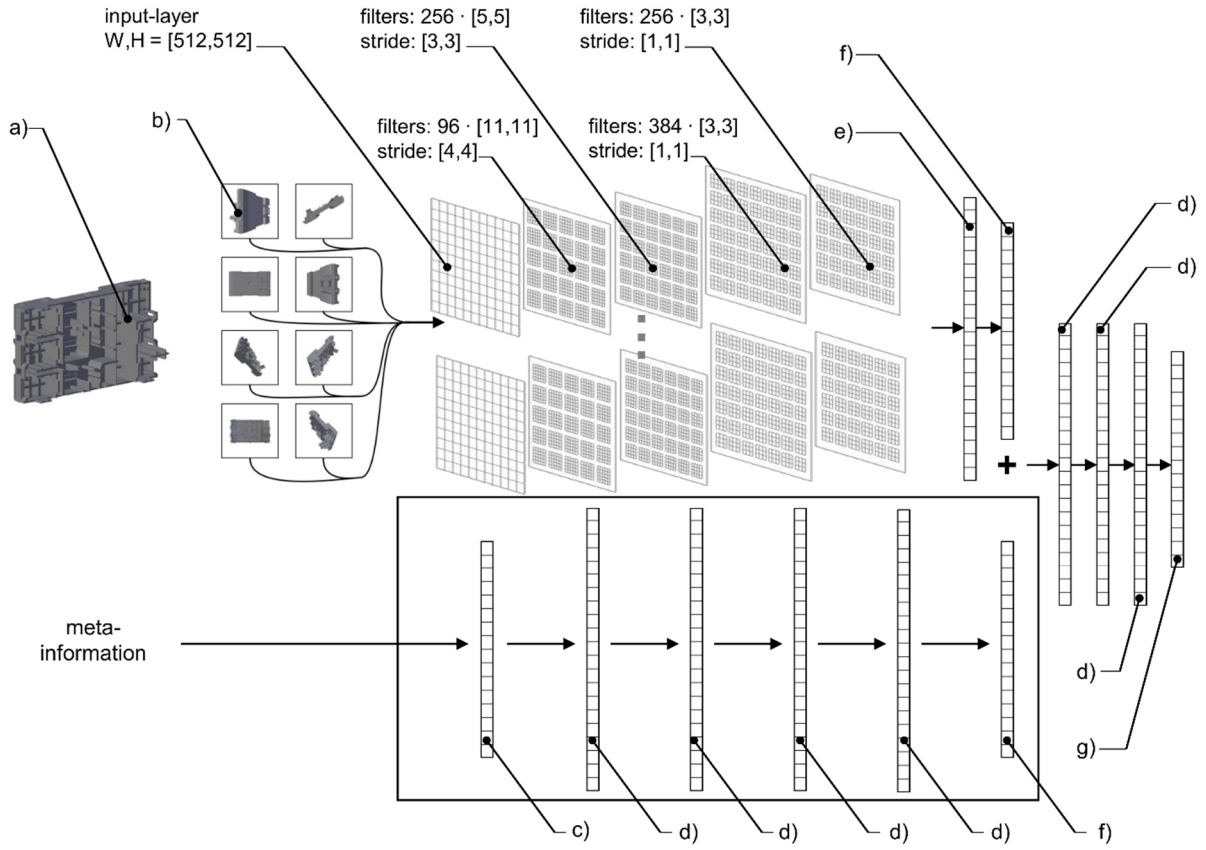

Figure 1: Underlying neural network model; a) part data; b) part renderings; c) extra parameters' input layer; d) flat layer, n = 2048 units; e) view-pooling layer, n = 20148 units; f) flat layer, n = 1024 units; g) output layer, n = 43 units

That means, each of the derived 20 images is fed into an individual convolutional neural network with a first layer of 512 px · 512 px as input layer. After input layer, a batch normalization layer to readjust the submitted images is inserted. After the batch normalization, various convolutional layers with differing filter sizes (as depicted in Figure 1) and differing strides, this means the distance step during filter convolution [13], follow. Between each filter layer, a maximum pooling layer with set pooling size of 3 and stride of 2 is inserted after each convolution layer except for layers 3 and 4. A spatial drop-out layer, to randomly drop filters and therefor counteract possible overfitting is placed between every convolution layer. To combine the activations of all 20 convolutional neural networks a flat, dense layer is used as a view pooling layer.

For additional meta-information, a flat, dense network architecture is chosen. This architecture is assigned once per submitted part. The flat layers start with a varying, dependent on chosen parameters, scaled input layer. After input layer, a normalization is made and several flat layers with architecture as depicted in Figure 1 follow. Between flat layers, only dropout layers are used.

To combine the view pooling layer as result of the convolutional neural network and the last layer of the flat, dense network for meta-information, both are concatenated. After this concatenation layer, a series of flat, dense layers follow. The last layer is the output layer, consisting of the number of classes created by mapping the costs to classes.

For activations, scaled exponential linear units, which are proposed as adding robustness during training by Klambauer et al. are implemented [14]. As cost function, cross entropy as well as absolute difference with equal weight is applied. As optimizing function, Adam optimizer, known for its ability to cope with sparse and noisy gradients, is utilized [15].

3. Case Study: Evaluation of proposed part cost quotation method

3.1 Software architecture overview

To accommodate for architectural difficulties arising from different operating systems and varying states of installed programs, the software to be developed is conceptualized as a web service. This enables to provide for good usability, rapid software rollouts, identical software used by every user and centralized data storage as well as user access-management. Because of the centralized data storage, it becomes possible to gather further data samples to retrain the quotation model on more recent data.

Because the proposed quotation method is to be tested on real data in a real production environment, certain requirements, as a link to enterprise resource planning software, construction database and CAD software must be established as well as good usability. The schematic of the base softwares' architecture is displayed in Figure 2. The software is divided into a back-end server task and a front end. While the back end is designed as two separate, different server instances (creating separate server tasks), the front end is designed within web technologies HTML5 and JavaScript. The back end is realized with the Python programming language in its 3rd version and a micro framework flask for routing and managing user requests. Model serving and deriving of projections is coordinated via task queue module huey to decouple long lasting calculations from back end processes and to enable an enhanced user experience. Projections and measurements are either made with a call to proprietary CAD software Creo Elements Direct 3D Access executing a Common-Lisp script for files submitted in proprietary Creo file-format or with a call to open source software blender.

For gathering of training data, a link for retrieving historic records of part-data is made via web-interface of the construction data system. To ensure restricted access to model maintenance tasks and system/user database management a user access-rights system is implemented. The ANN is programmed with open source deep learning framework tensorflow in its version 1.13.

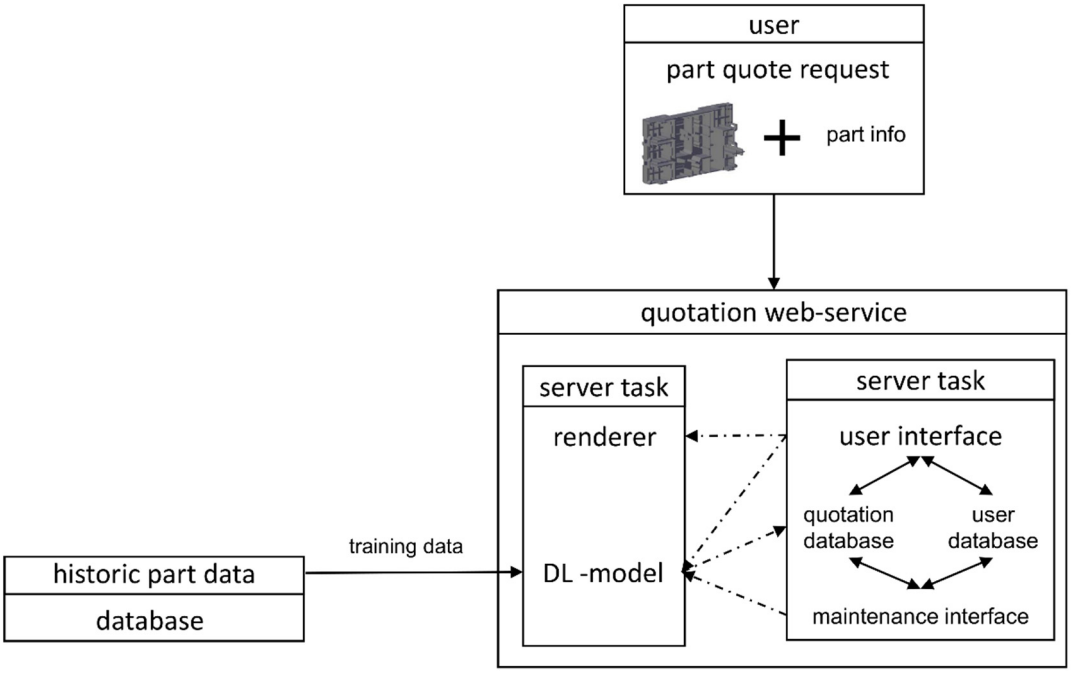

Figure 2: Basic schematics of developed system

3.2 Data pre-processing / model training

700 samples are created by gathering of 3D CAD data of injection mold parts and their corresponding meta-information. Each sample contains unique, historic part quotation data. For alignment of part's 3D CAD data, each part is normalized to a length of longest part side of $l_{part} := 1$ according to equations 1 and 2:

$$l_{max} = max(l_{part_x}, l_{part_y}, l_{part_z}) \tag{1}$$

$$l_{part_i} \Rightarrow \frac{l_{part_i}}{l_{max}} \tag{2}$$

There are 20 image projections derived via a rendering of the part's 3D CAD data as shown in Figure 3. The virtual camera alignment is adopted from Su et al.'s second camera setup; 20 cameras are placed on the vertices of a sphere around the part [11]. Hereby the images are rendered in 8-bit grayscale with black background and a resolution of 512 px · 512 px. Meta-information is prepared either by a categorization through grouping with a k-means algorithm for part properties with more than 3 levels or by normalizing to range of $\{i \, \epsilon \, \mathbb{R} \, | 0 \leq i \leq 1\}$.

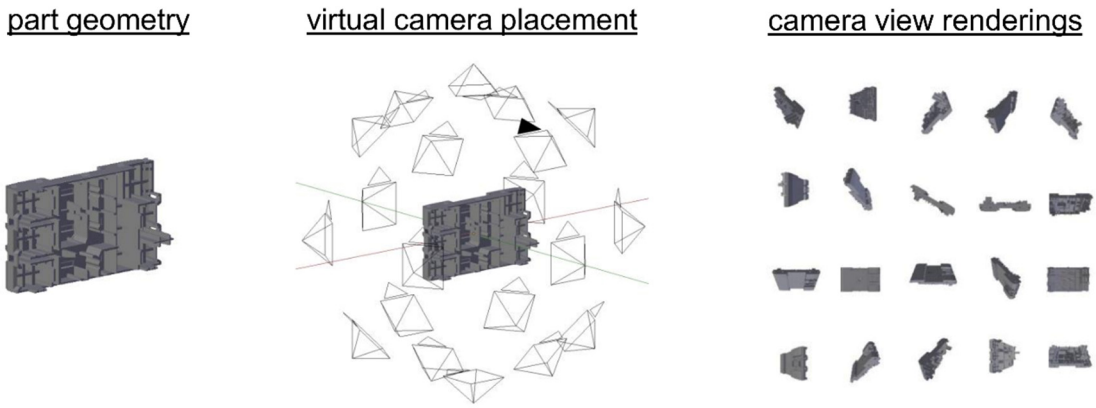

Figure 3: Rendering setup to derive 2d-projections from 3d part data

Each sample therefore consists of the two-dimensional projections and the normalized meta-information. Manufacturing cost data is used as label data. The cost data is rounded and mapped to the closest cost class with an equidistant cost distribution with 5.000 EUR steps. This means, the cost regression task is mapped onto a classification task. Before the conversion to a binary format for enhancing the data feeding pipeline, sample data is shuffled randomly and split into a training and testing dataset by a split of 80:20. The testing dataset is kept in a separate binary file reserved for model evaluation.

The model training is performed on gpu-hardware and is stopped after one week and after the convergence of the train-loss curve. By this time, $172 \cdot 10^3$ steps have been trained.

4. Evaluation of quotation method

4.1 Results of model evaluation

The trained model is evaluated on $n_{evalp} = 20$ % of total samples. For reference, a model trained on the same dataset but with only the convolutional part of the proposed model is compared to the model trained on parts' projections and meta-information. The results are displayed in Figure 4. The share of classes as a function of relative errors are shown. The relative error is calculated by equation 3 and grouped into classes for ease of display.

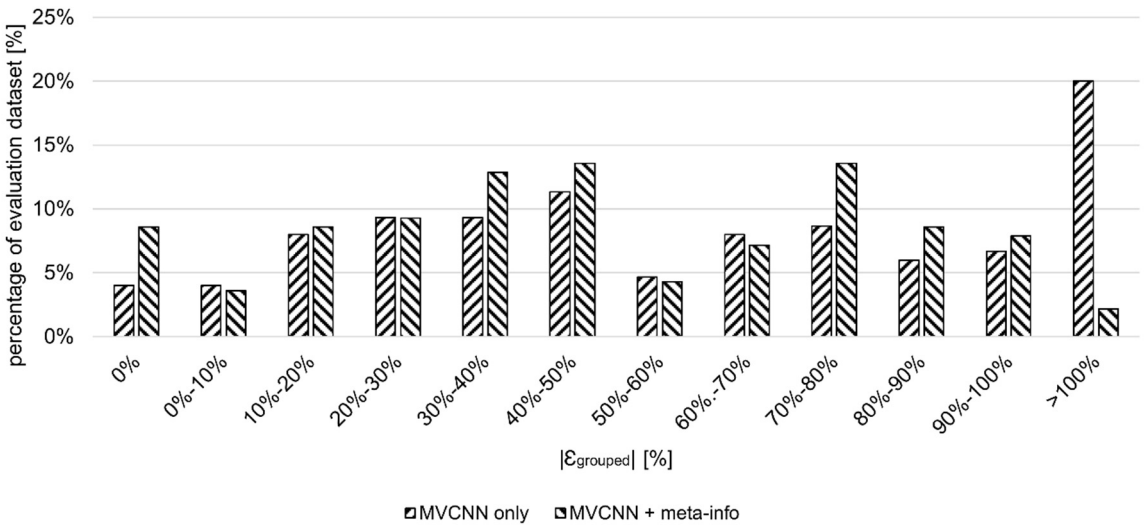

Figure 4: Display of model evaluation. Shown are grouped shares of relative errors of total evaluation dataset

The grouping is done in steps of 10 % with the final step being a group for all relative errors with $|\varepsilon_{rel}| > 100\ \%$.

$$|\varepsilon_{rel}| = \frac{|c_{pred} - c_{real}|}{c_{real}} \tag{3}$$

It can be noticed, that overall, the extended model scored twice as good as the model solely relying on image data. The model trained with additional meta-information scores twice as many parts without prediction error than the model solely relying on image data. Predictions with errors in class $\varepsilon_{rel} > 100\ \%$ by the model with additional meta-information are at just the tenth of the other models. The mean relative error is calculated as displayed in equation 4.

$$\overline{\varepsilon_{rel}} = \frac{\sum |\varepsilon_{rel}|}{n_{samples}} \tag{4}$$

For the model trained solely on images, the resulting mean relative error is calculated as $\overline{\varepsilon_{rel}} = 112\ \%$. The extended model with additional meta-information scores a mean relative error of $\overline{\varepsilon_{rel}} = 51\ \%$.

4.2 Discussion of results

As can be seen in the results, the mean relative error could be lowered to less than half of the model's error without the extra meta-information NN. Comparing the results to the models of Wang et al., their mean relative errors being $\varepsilon_{rel} < 2\ \%$, their achieved errors are comparably low [4], [6]. It is important to underline, Wang et al. not only estimated the fixed upfront mold costs, but also the production costs to produce a part via injection molding. Xu et al.'s evaluation's mean relative error is as low as $\varepsilon_{rel} = 12\ \%$ [3]. Florjanic et al's. work is closest to the results developed during this research with $\varepsilon_{rel} = 38\ \%$ [7]. For comparison, the total size of the datasets used and the validation sample size must be considered. For example, for Wang et al., the validation sample size is at 1.9 % of their total dataset [6]. On the other hand, NN models are data-driven, and comparing these models without access to underlying datasets is questionable.

In contrast to the cited authors' methods, the system we developed is designed not to need any mold specific parameters such as mold dimensions or parting line complexity, and its inputs do not rely on experts' opinion as in determining part complexity. To further enhance the model's performance, the server software system developed is designed to be extensible by adding further samples to the dataset.

5. Conclusion

During this research, a method to support the cost quotation process for injection molds was developed. The underlying ANN-model was trained on real world injection molding data. To evaluate the researched model, it was implemented as a stand-alone server-side application. This server-side application was interfaced to various production systems to have the possibility of gathering real data. The developed server-side application allows for continuous and automatic extension of the underlying dataset. Possibilities to further enhance the model's performance could be achieved by adding further data to the dataset or extending parameters in the dataset as well as further aligning historic data records.

Acknowledgements

The authors would like to thank the Phoenix Contact Foundation for supporting this research.

References

[1] Brydson, 1999: Plastics materials. 7th ed. Oxford, Boston, Butterworth-Heinemann, 158-183, dx.doi.org/10.1016/B978-075064132-6/50049-8

[2] Wang, Wang, Wang 2013.: Cost estimation of plastic injection molding parts through integration of PSO and BP neural network, Expert Systems with Applications 40, 2, 418–428, dx.doi.org/10.1016/j.eswa.2012.01.166

[3] Xu, Yan 2006: An intelligent estimation method for product design time, The International Journal of Advanced Manufacturing Technology 30,7, 601–613, dx.doi.org/10.1007/s00170-005-0098-6

[4] Wang 2007: Application of BPN with feature-based models on cost estimation of plastic injection products, Computers & Industrial Engineering 53, 1, 79–94, dx.doi.org/10.1016/j.cie.2007.04.005

[5] Che 2010: PSO-based back-propagation artificial neural network for product and mold cost estimation of plastic injection molding, Computers & Industrial Engineering, 58, 625–637, dx.doi.org/ 10.1016/j.cie.2010.01.004

[6] Wang, Che, Wang 2007: Cost estimation of plastic injection products through back-propagation network, Proceedings of the Eighth Conference on WSEAS International Conference on Neural Networks, 8, 54-60

[7] Florjanic, Govekar, Kuzman 2013: Neural Network-Based Model for Supporting the Expert Driven Project Estimation Process in Mold Manufacturing, Strojniški vestnik – Journal of Mechanical Engineering 59, 1, 3–13, dx.doi.org/10.5545/sv-jme.2012.747

[8] Eilert, Ullmann, Overmeyer 2012: Umformwerkzeuge automatisiert konfigurieren und kalkulieren - Herstellkostenermittlung von Blechumformwerkzeugen mit Methoden der künstlichen Intelligenz auf Basis einer automatisierten CAD-Modellerstellung, ZWF - Zeitschrift für wirtschaftlichen Fabrikbetrieb, 107, 11, 801-807, dx.doi.org/10.3139/104.110851

[9] Kočov, Spiroski, Tutes 2015: EXPERT SYSTEM FOR MOLD QUOTATION, Journal for Technology of Plasticity Vol. 40, 1, 33-46

[10] Börzel, Frochte 2019: Case Study on Model-based Application of Machine Learning using Small CAD Databases for Cost Estimation, Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, SCITEPRESS - Science and Technology Publications, 258–265, dx.doi.org/10.5220/0007979802580265

[11] Su, Maji, Kalogerakis, Learned-Miller 2015: Multi-view convolutional neural networks for 3d shape recognition, Proceedings of the IEEE international conference on computer vision, 945–953, dx.doi.org/10.1109/iccv.2015.114

[12] Krizhevsky, Sutskever, Hinton 2017: ImageNet classification with deep convolutional neural networks, Communications of the ACM 60, 6, 84–90, dx.doi.org/10.1145/3065386

[13] Michelucci 2019: Fundamentals of Convolutional Neural Networks, Advanced Applied Deep Learning: Convolutional Neural Networks and Object Detection, Apress, 79-123, dx.doi.org/10.1007/978-1-4842-4976-5_3

[14] Klambauer, Unterthiner, Mayr, Hochreiter 2017: Self-Normalizing Neural Networks, Advances in Neural Information Processing Systems 31, 1-10

[15] Kingma, Ba 2015: Adam: A method for stochastic optimization, Proceedings of the 3rd International Conference on Learning Representations, 1-15

Biography

Sebastian Brede (*1988) is project engineer at the IPH - Institute of Integrated Production Hannover (non-profit limited company)

Benjamin Küster (*1988) leads the department of production automation at the IPH

Dr.-Ing. Malte Stonis (*1979) is managing director of the IPH

Dr. Mike Mücke (*1982) is technology development manager for software solutions at Phoenix Contact

Prof. Dr.-Ing. Ludger Overmeyer (*1964) heads the Institute of Transport and Automation Technology (ITA) at the Leibniz Universität Hannover and is member of the management board of the IPH