2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)
August 23-27, 2021. Lyon, France

Virtual metrology for semiconductor manufacturing: Focus on transfer learning

Rebecca Clain[1] and Valeria Borodin[1], Michel Juge[2], Agnès Roussy[1]

2021 IEEE 17th International Conference on Automation Science and Engineering (CASE) | 978-1-6654-1873-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/CASE49439.2021.9551567

***Abstract*— Nowadays, virtual metrology models for semiconductor manufacturing aim to be scalable. A Virtual Metrology (VM) system is intended to cover a wide spectrum of production contexts. However, due to the large numbers of possible combinations of recipes, tools and chambers, it becomes intractable to model each context separately. This work presents a VM modeling approach based on the paradigm of transfer learning in a fragmented production context. The approach exploits a 2-Dimensional Convolutional Neural Network (2D-CNN) architecture, namely Spatial Pyramid Pooling Network (SPP-net), to perform the transfer learning from source to target domains with input of different sizes. We implemented several transfer learning strategies on a benchmark dataset provided by the Prognostics and Health Management competition in 2016. The main key points of the proposed approach are discussed based on the findings gathered from the numerical analysis.**

I. Introduction

Metrology tools are in charge of ensuring the production quality in the semiconductor manufacturing industry. To overcome time and cost limitations involved by metrology tasks, VM systems are developed to predict metrology variables based on process and wafer state information.

Semiconductor manufacturing is highly fragmented, a high mix of products is routed through production lines including complex (multi-chamber) machines and following multiple recipes (machine settings) [1]. A VM system is intended to cover a wide spectrum of production contexts. However, due to the large number of possible combinations of recipes, tools and chambers, it becomes intractable to model each production context separately. Machine learning (ML) algorithms assume that training and test data are drawn from the same distribution. In reality, data collected from different production contexts are likely to come from different probability distributions [2]. A model trained for one production context can fail outside of it. Moreover, a ML algorithm requires sufficient amount of data to achieve a high predictive accuracy. Gathering sufficient amount of data for each production context can be very time consuming. As a result, most VM models are process-centric, single-task oriented and limited to a production context with sufficient instances.

To circumvent the aforementioned issues, we investigate the value of transfer learning (TL) approach in a multi-chamber production context to reach more accurate VM model without the need of lot of instances. TL leverages knowledge from a source task within a source domain to improve the predictive capability of a target task within a target domain. This work proposes a TL approach based on a SPP-net [3]. SPP-net is made of a CNN and a SPP layer on top of the last convolutional layer. CNNs have proven their competitiveness in different domains, from speech recognition [4] to text classification [5], being especially prevalent in image processing. To harness 2D-CNN capabilities, we propose to feed them with a 2D input representation of wafer records based on moving windows. CNN requires a fixed input size between training and test datasets. This constraint limits considerably TL within CNN applications. By definition, TL leverages whole or part of a previous trained model to train a new model. A fixed input constraint implies a same input size for baseline and new models. In practice, data collected from different chambers or tools do not have the same length and size. In particular, time series variables from sensors do not have the same length. One could cut the time series before applying the moving windows to obtain the same length, but it would lead to a loss of information and may hurt accuracy. To avoid the fixed-size constraint and loss of information from the time series, we use a SPP layer on top of the last convolutional layer, which pools the features and generates fixed-length outputs. It enables us to train a model on a source domain and to use it on a target domain with different input sizes. The TL approach studied in this paper go beyond the traditional approaches, limited to a narrow production context, by: **(i)** Leveraging the data fragmentation and transferring the knowledge from previously trained VM models for a given production context to a new VM model for other production contexts, **(ii)** Overcoming the lack of data for some chambers/tools. To this end, the proposed TL models exploit the predictive potential of the SPP-net.

In this paper, performances of different TL strategies with a SPP-net are evaluated with a focus on the key parameters of the successful strategies. Numerical experiments are conducted on a benchmark dataset provided by the Prognostics and Health Management (PHM) competition in 2016 [6].

II. Related work

Consistent with the scope of the current paper, this section reviews the literature related to regression problems and transfer learning in the field of VM. The literature on regression problems for VM is rich, while transfer learning for VM has not yet been much explored.

[1]Mines Saint-Etienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS, CMP, Department of Manufacturing Sciences and Logistics, Gardanne, France {rebecca.clain, valeria.borodin, agnes.roussy}@emse.fr
[2]Department of Process Control, STMicroelectronics, 13106 Rousset, France michel.juge@st.com

A. Regression problems

Different classes of regression approaches for VM in semiconductor manufacturing can be found in the literature.

- **Linear:** are prevalent in the literature by virtue of their ease of implementation and interpretability. The authors of [7] compare several regression methods: simple linear regression, multiple linear regression, partial least square regression, and ridge linear regression algorithms. In [1], the authors combine regularization methods, ridge regression and LASSO, to tackle the high dimensionality in VM modeling.
- **Non-linear:** such as decision trees, KNN, SVR and neural network are well-known algorithms for VM, which have proved their efficiency in terms of computational time and accuracy (see, e.g. [8], [9], [10]). More recently, advanced ML models based on boosting and bagging methods have been explored in [11].
- **Deep learning:** Deep learning algorithms are gaining more and more attention, given their high prediction capabilities and their ability to handle raw data (see, e.g. [12], [13]).

B. Transfer learning

Given the large number of possible combinations and not always a sufficient available volume of data, developing a VM model for all the production contexts is arduous. To address this challenge, multitask learning and TL have been proposed in the literature related to VM for semiconductor manufacturing. These approaches are aiming to improve the generalization and adaptability of VM models, by leveraging exhibited knowledge from previous or concomitant tasks, either by parallel transfer of knowledge (multitask learning) or sequential TL.

- **Multitask learning:** The authors of [14] propose a hierarchical framework based on kernel methods solved by means of multitask learning. Two multi-step methodologies are presented in [1]: **(i)** *a process-based multi-step method*, where the collected information during all process steps are used as input of the VM model, and **(ii)** *a cascade multistep method*, where the input matrices are populated with the previous VM predictions for equipment that do not belong to the target step.
- **Transfer learning:** Only a few TL approaches have been explored in the framework of VM. In [15], the authors propose a common VM model for two identical-in-design chambers based on a deep learning architecture, called domain adversarial neural network, which incorporates a discriminator distinguishing the chamber under examination. In [10], the authors explore TL methods for identical-in-design equipment when the number of wafer records for target equipment is insufficient. None of these methods can be applied between domain not identical-in-design (e.g. different set of features and/or different time series lengths).

This paper presents a TL approach adapted to domains, which are not identical-in-design. Given the limited literature related to TL in VM, there is not many guidelines to find a good design to achieve TL. In this paper, findings gathered from the numerical analysis are used to discuss and provide guidelines.

III. Methodology

In this section, the 2D input representation is presented, followed by the SPP-net architecture and the TL approach used to address domain and task adaptation problems in a multi-chamber context.

A. 2D input representation

Usually, inputs of VM algorithms are cost-free data derived from Fault Detection and Classification (FDC) data (i.e. data collected by sensors) and production context data. The classical approach consists in summarizing temporal FDC data into a 1D feature space.

This study uses a 2D feature space representation to feed the SPP-net. A non-overlapping rolling windows approach is adopted to summarize temporal FDC data into a 2D representation. For every combination of wafer and stage, the rolling mean and median with a window step of ten records are extracted. For every combination of wafer and stage, several records are associated with a unique metrology variable. Fig. 1 illustrates the obtained 2D representation.

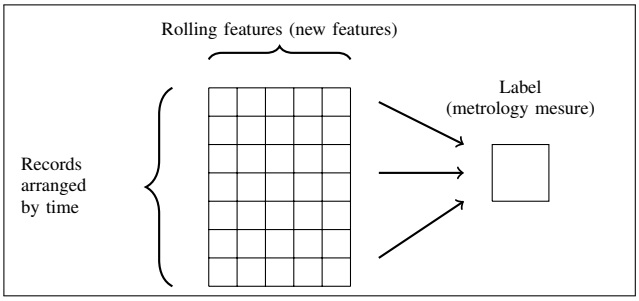

Fig. 1: 2D Input representation of FDC raw data

B. VM Prediction model: SPP-net

This study employs a SPP-net as a VM prediction model, which is composed of a CNN architecture and a SPP layer on top of the last convolutional layer. The SPP layer is a pooling layer, which *"performs some information aggregation at a deeper stage of the network hierarchy (between convolutional layers and fully-connected layers)"* [3]. For an exhaustive description of the SPP layer, the interested readers can refer to the original paper [3].

The built CNN includes the following modules:

- 3 convolutional layers with 32 filters and a kernel size, respectively, of 7×7, 5×5 and 5×5.
- A batch normalization layer set after convolutional layers designed to automatically standardize the input of the layer.
- A rectified linear unit (Relu) function, located after the batch normalization layer, used as activation function to introduce non-linearity.

- A max pooling layer to obtain a feature map containing the most prominent features of the previous feature map.
- A SPP layer, which maps any-size input down to a fixed-size output.
- A flatten layer that converts the data into a 1D array in order to input it to the fully connected layers.
- 4 fully connected layers with output size respectively equal to 64, 32, 16 and 1.

C. Transfer learning

Due to the issues posed by the data fragmentation, ML approaches based on single-task models are prevalent in the VM-related literature. Single-task ML models can perform very well for a specific task on a specific domain, but presenting poor performances when applied to a related task or domain. This work addresses the issues induced by data fragmentation through the TL paradigm. TL leverages the knowledge gained in solving a source task in a source domain, and applies it to a new task and/or new domain. A complete description of the mathematical framework of TL can be found in [16]. Let us recall the definition:

Definition 1 (Transfer learning [16]): Given a source domain D_S and a learning task T_S, a target domain D_T and a learning task T_T, TL aims to improve the learning of the target predictive function $f_t(\cdot)$ in D_T by using the knowledge in D_S and T_S, where $D_S \neq D_T$ or $T_S \neq T_T$.

The common TL implementation includes two main steps: **(i)** To train a base network, and then **(ii)** To copy its layers to a target network. One can choose to transfer more or less number of layers and add, if necessary, fully connected layers on top of the transferred layers. The transferred layers can be left frozen or fine-tuned. In the first case, the pre-trained weights do not change, and in the second one, the pre-trained weights are used at initialization. The number of source layers to freeze depends on the domain similitude and remains a manual design choice. One can choose to keep or re-initialize the hyperparameter values from the source model.

IV. Computational Experiments

A. Data description

To evaluate the contribution of the proposed TL approach for VM modeling in the context of the data fragmentation, we conduct numerical experiments on a benchmark dataset from the PHM data competition in 2016. Data are derived from a Chemical-Mechanical Planarization (CMP) tool, that removes undesired material from the surface of wafers via a polishing process.

The studied dataset contains data from six chambers, which can be divided in two classes. The first class includes chambers 1-2-3, and the second class is composed of chambers 4-5-6. The chambers belonging to the same class are identical-in-design, but different from the chambers belonging to the other class. The dataset consists of data collected during various runs of the CMP tool for specified wafers over a period of time. FDC data are related to pressure, flow rates, component usage and context. The metrology variable to predict is the material removal rate (MRR). The provided dataset can be split into three subgroups. Each subgroup represents an individual dataset, divided into train and validation sets used to conduct experiments, as provided in Table I.

TABLE I: Data group PHM challenge 2016

	Chamber ID	Stage	Train size	Valid. size
`Dataset A`	1,2,3	A	204	39
`Dataset B`	4,5,6	A	680	143
`Dataset C`	4,5,6	B	756	173

According to the MRR distribution range in each subgroup, these three datasets can be separated into two different modes: the *low-speed mode* with MRR $\in [50; 110]$ nm/min, and the *high-speed mode* with MRR $\in [140; 170]$ nm/min. A tree representation of the PHM dataset is illustrated in Fig. 2.

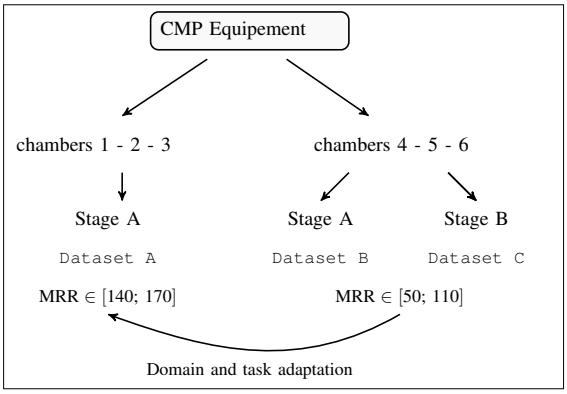

Fig. 2: Tree representation of PHM dataset

B. Design of experiments

To analyze the value of TL for VM modeling with domain and task adaptation, `Dataset B` and `Dataset C` are used as source domains and `Dataset A` as target domain. The small size of `Dataset A` heightens its interest for TL. TL with a small dataset is particularly relevant to overcome the possible lack of data.

Different TL strategies are compared to find out the appropriate design for TL when applied to the VM tasks in semiconductor manufacturing. We attempt to understand what make successful the transfer of the domain knowledge in the source dataset to the target dataset when TL is based on CNN. As mentioned in Section III-C, the transfer of knowledge passes through two channels: the transferred layers and the transferred hyperparameter values, especially the learning rate of the optimizer. We consider twelve different TL strategies, by varying the number of convolutional frozen layers and the value of Adam learning rate (Adam LR) (see Fig 3).

All the strategies are compared with a baseline model, where a SPP-net model is trained from scratch on the target dataset. The TL strategies are summarized in Fig. 3. To carry out the analysis, the rolling feature length within each dataset

were unified to ensure feature length matching. Forward feature selection on each dataset were implemented and model performances compared over a selected subset and the whole feature set. The baseline model on `Dataset A` performs better with a selected features set, whereas the models on `Dataset B` and `Dataset C` perform better with all the features. The dimension of `Dataset A` is different from the one of `Dataset B` and `Dataset C`. Firstly, because `Dataset A` is composed of a selected subset of features, whereas `Dataset B` and `Dataset C` are composed of the full set of features, and secondly due to the different time series lengths before applying the moving windows. A traditional CNN cannot be applied between `Dataset A` and the other datasets. A SPP-net, which overcome the fixed-size constraint, have to be used. Each feature were scaled with respect to its own dataset:

- `Dataset A`: 204 training 2D input matrices of size 18×27, defined by 18 selected features and 27 the most frequent rolling feature lengths.
- `Dataset B` and `Dataset C`: 680 and 756 training 2D input matrices of size 57×33.

The same CNN architecture (see Section III-B) and training design (see Algorithm 1) are used for all models, including the baseline model. We train incrementally each models for N iterations. The mean square error (MSE) is defined as the loss function. MSE is the average squared difference between the target value and the value predicted by the model and is defined as: $\frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n}$ The Adam optimizer is initialized with a learning rate equal to 0.001. For each iteration, the models are trained with 50 epochs. The number of epochs corresponds to the number of times that the learning algorithm will work through the entire training dataset. 20 epochs without improvement of the loss function is considered as the stopping criterion for one given training iteration. In our study, the dataset was divided into batches of 30 samples. An epoch is completed when the learning algorithm has passed through all the batches. During the training process, the train set was divided into 85% train set and 15% test set. The metrology accuracy of each model was evaluated in terms of MSE on test and validation sets. All the strategies are implemented based on Keras in Python.

Algorithm 1 Experiment training design

```
Divide dataset into train and validation sets
for iteration in 0 : N do
    Divide train dataset into train and test samples
    while stop criterion do
        Train neural network on train set
        Compute MSE on test sample
Compute MSE on validation set
```

C. Analysis of results

The performances of different designs of TL are evaluated and compared with respect to the baseline model. Fig. 4 presents the test results against training time in terms of MSE, with the source Adam LR for different numbers of frozen layers for the TL models from `Dataset B` (see Fig. 4 (a)) and `Dataset C` (see Fig. 4 (b)). A first conclusion from the graph, without focusing on the number of frozen layers, is that the TL models from B and C outperform the baseline model on the test set. They exhibit a rapid decline in MSE, by converging in less than 30 seconds. Their rate of convergence is faster, and they are able to reach smaller MSE. These good performances are reflected on the MSE on the validation set, where the TL models with the source Adam LR outperform the baseline model (see Tables II,III). It can be noted that MSE on the test set are around 20 or more, while MSE on the validation set are around 12. This shows that the validation set provided by the PHM competition is composed of *easier* instances than the test set. Sections IV-C.1-IV-C.2 deepen the understanding of the successful TL strategies. The analysis focuses on two known key determinants: the number of frozen layers on which depend the model weight update (Fig. 4 and Section IV-C.1) and the hyperparameter Adam LR value that controls how much the model change in response to the estimated error each time the model weights are updated (see Fig. 5 and Section IV-C.2).

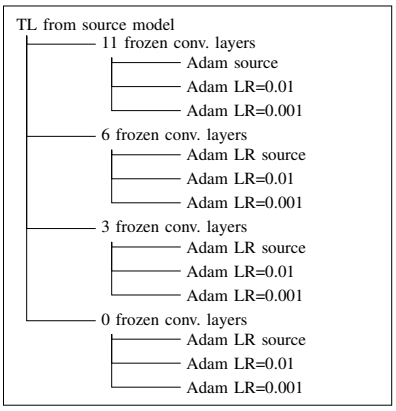

Fig. 3: TL strategies

1) Focus on weight update: Freezing layers, in the context of CNN is about controlling the way in which the weights are updated. By freezing different numbers of convolutional layers, the weight behavior of the convolutional layers for the target models is derived, all others things being equal. Freezing a different number of layers and keeping the source Adam LR do not affect results on the test (see Fig. 4) and validation sets (see Tables II-III). Fig. 4 shows the test curves with different numbers of frozen layers overlying.

The weights of the convolutional layers are not, or marginally, updated even when the possibility is given. It means the three groups of chambers share high-level features that do not need to be adjusted between them. However, the fine-tuning of the fully connected layers is necessary to adapt to the different datasets. The tests on the target dataset without training show systematically poor results. Finally, freezing layers should decrease the computational times, due to the absence of backward pass to those layers. However,

Fig. 4: Test curves for baseline model (no TL) and TL models with Adam LR transferred from sources models and different number of frozen layers for $N = 5$: (a) TL models from `Dataset B`, (b) TL models from `Dataset C`

TABLE II: Accuracy on validation set for TL strategies from `Dataset B` with different frozen layers and LR

	TL from `Dataset B`												baseline
	11			6			3			0			
LR / iteration	source	0.01	0.001	source	0.01	0.001	source	0.01	0.001	source	0.01	0.001	
1	11.53	27.41	46.92	11.53	15.64	16.94	11.53	**11.33**	18.3	11.53	27.64	16.80	14.01
2	11.62	20.81	28.82	11.62	15.32	16.44	11.62	**10.08**	17.92	11.62	14.64	16.49	15.40
3	12	19.58	22.96	12	14.69	20.76	12	**10.08**	17.92	12	14.63	16.48	14.42
4	12	19.59	22.65	12	14.69	20.76	12	**10.08**	17.92	12	14.63	16.48	12.32
5	12	19.59	22.66	12	14.69	20.76	12	**10.08**	17.91	12	14.63	16.48	12.74

TABLE III: Accuracy on validation set for TL strategies from `Dataset C` with different frozen layers and LR

	TL from `Dataset C`												baseline
	11			6			3			0			
LR / iteration	source	0.01	0.001	source	0.01	0.001	source	0.01	0.001	source	0.01	0.001	
1	**12.68**	24.02	64.59	**12.68**	28.62	20.65	**12.68**	21.56	16.09	**12.68**	16.21	14.34	14.01
2	**10.83**	37.64	34.04	**10.83**	12.61	21.75	**10.83**	28.15	17.59	**10.83**	25.09	16.59	15.40
3	10.73	25.27	24.68	10.73	18.61	20.94	10.73	23.42	17.00	10.73	**8.78**	17.29	14.42
4	**11.29**	32.26	22.64	**11.29**	12.68	22.24	**11.29**	23.08	16.39	**11.29**	13.28	15.93	12.32
5	13.10	32.26	22.32	13.10	14.05	28.02	13.10	22.90	15.56	13.10	11.36	19.31	**12.74**

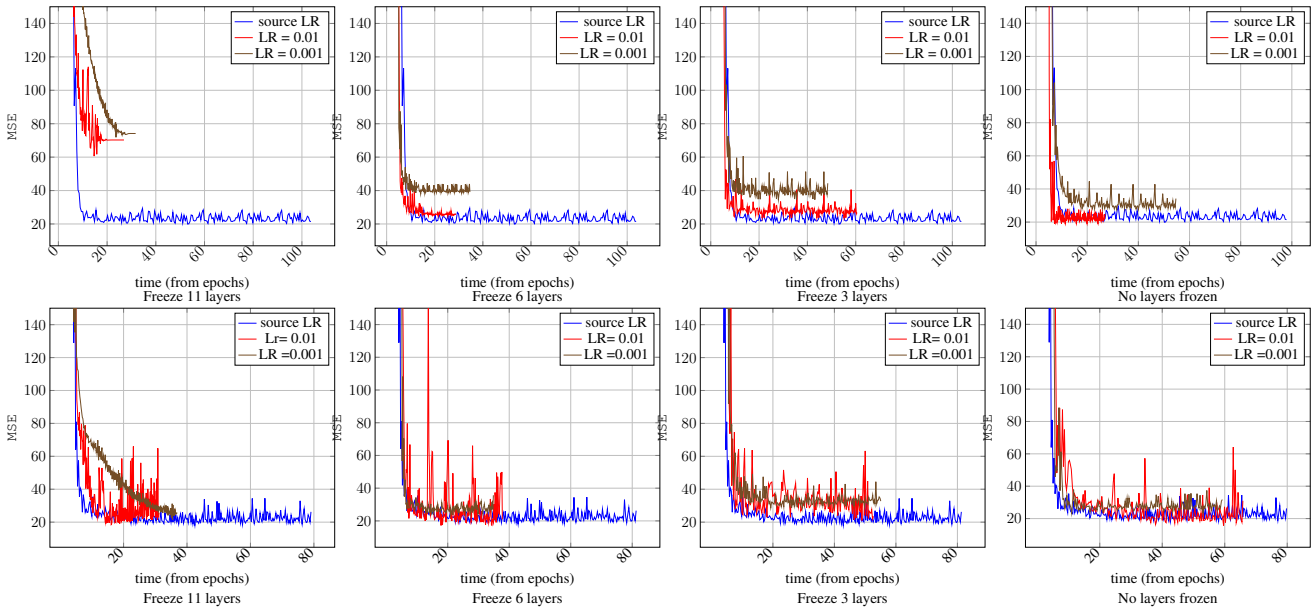

Fig. 5: TL strategies from B (top) and C (bottom) on test sets: Different frozen layers/Adam LR for $N = 5$ with *source LR* corresponding to Adam LR transferred from source model and LR corresponding to Adam optimizer initialized at 0.01 or 0.001

due to the small size of `Dataset A` the backward pass performs fast and the gain in computational time is marginal.

2) Focus on Adam hyperparameter: The Adam LR initialization has a central role in TL and is complementary with the weight initialization. Adam optimizer is an adaptive

learning rate method, which computes individual learning rates for each parameter of the model. Note that transferring the source learning rate previously initialized for the source model at 0.001 is not the same as initializing a new Adam LR at 0.001 for the target model. In the first case, all the adapted learning rates computed for the source model are transferred. In the second case, the adaptive learning rate from the source model are not transferred and an Adam optimizer is initialized and computes the adapted learning rate for the target dataset. To examine the impact of Adam LR on the TL models accuracy and computational time, Fig. 5 presents MSE against time on the test set with different numbers of frozen layers and different Adam LR values at initialization for the TL models from `Dataset B` (top) and `Dataset C` (bottom) for $N = 5$ iterations.

The models based on the source Adam LR outperform all the other strategies for the first iteration on the test and validation sets. Looking at the MSE test curves (Fig. 5), the models with source LR from Adam optimizer converge more quickly in less than 20 seconds. The results with the Adam LR initialized at 0.01 are ambiguous. These models fail with 11 frozen layers, but can possibly reach a good level of accuracy with fewer layers frozen at the expense of a training process instability. Big jumps in all of their MSE test curves are observed. As a result, the MSE on test and validation sets are unstable. Updating weights, which are already well initialized, with a big step leads to unstable training process. Finally, initializing the Adam LR with the same value as the source model (0.001) fails with all strategies. One can conclude that Adam optimizer adapts better in the source dataset than in the target dataset. For this reason, its pre-initialization is crucial in TL.

D. Concluding remarks

The small size of the target dataset is partly responsible for poor performances of the baseline model in test and validation sets. The TL models circumvents this lack of data and outperform traditional independent learning models. Sections IV-C.2-IV-C.1 provide two guidelines regarding the design of TL based on CNN, for multi-chamber equipment and small target dataset: **(i)** The fine-tune of conventional layers appears to not be necessary, because the chambers share high level features, but the fine-tune of the fully connected layer is mandatory to adapt to the marginal differences of each chamber, **(ii)** the Adam LR from source datasets has to be transferred.

V. Conclusions and perspectives

This work presents a VM modeling approach in a highly fragmented production context. The studied approach exploits a 2D-CNN architecture, namely SPP-net, capable to perform TL from a source to a target domain with input of different sizes. Several TL strategies are compared on a benchmark dataset provided by the PHM competition in 2016. The approach based on TL successfully increases accuracy of VM models in a domain and task adaptation context. The experiments show, conclusively, that their success relies on a suitable weight initialization and a corresponding optimizer LR.

Future research focuses on extending TL sensitivity analysis to other applications involving large-scale real-life datasets. TL between both, different datasets in size and very dissimilar tasks (e.g. two different processes), will be consider to investigate it full interest.

Acknoledgment

This paper is conducted in the framework of the project MADEin4, which has received funding from the ECSEL JU (Electronic Components and Systems for European Leadership Joint Undertaking) under grant agreement No 826589. The JU receives support from the European Union's Horizon 2020 research and innovation program and France, Germany, Austria, Italy, Sweden, Netherlands, Belgium, Hungary, Romania and Israel.

References

[1] G. A. Susto, S. Pampuri, A. Schirru, A. Beghi, and G. D. Nicolao, "Multi-step virtual metrology for semiconductor manufacturing: A multilevel and regularization methods-based approach," *Computers and Operations Research*, vol. 53, pp. 328–337, 2015.

[2] S. H. Bang, R. Ak, A. Narayanan, Y. T. Lee, and H. Cho, "A survey on knowledge transfer for manufacturing data analytics," pp. 116–130, jan 2019.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 1904–1916, 9 2015.

[4] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1d & 2d cnn lstm networks," *Biomedical Signal Processing and Control*, vol. 47, pp. 312–323, 1 2019.

[5] "Comparative effectiveness of convolutional neural network (cnn) and recurrent neural network (rnn) architectures for radiology text report classification," *Artificial Intelligence in Medicine*, vol. 97, pp. 79–88, 6 2019.

[6] P. D. C. P. Society, "Phm data challenge," https://www.phmsociety.org/events/conference/phm/16/data-challenge, 2016.

[7] H. Purwins, B. Barak, A. Nagi, R. Engel, U. Hockele, A. Kyek, S. Cherla, B. Lenz, G. Pfeifer, and K. Weinzierl, "Regression methods for virtual metrology of layer thickness in chemical vapor deposition," *IEEE/ASME Transactions on Mechatronics*, vol. 19, pp. 1–8, 2 2014.

[8] D. Teixidor, M. Grzenda, A. Bustillo, and J. Ciurana, "Modeling pulsed laser micromachining of micro geometries using machine-learning techniques," *Journal of Intelligent Manufacturing*, vol. 26, pp. 801–814, 8 2015.

[9] P. Kang, D. Kim, H. J. Lee, S. Doh, and S. Cho, "Virtual metrology for run-to-run control in semiconductor manufacturing," *Expert Systems with Applications*, vol. 38, pp. 2508–2522, 3 2011.

[10] P. Kang, D. Kim, and S. Cho, "Semi-supervised support vector regression based on self-training with label uncertainty: An application to virtual metrology in semiconductor manufacturing," *Expert Systems with Applications*, vol. 51, pp. 85–106, 6 2016.

[11] C. H. Chen, W. D. Zhao, T. Pang, and Y. Z. Lin, "Virtual metrology of semiconductor pvd process based on combination of tree-based ensemble model," *ISA Transactions*, vol. 103, pp. 192–202, 8 2020.

[12] K. B. Lee and C. O. Kim, "Recurrent feature-incorporated convolutional neural network for virtual metrology of the chemical mechanical planarization process," *Journal of Intelligent Manufacturing*, vol. 31, pp. 73–86, 1 2020.

[13] M. Maggipinto, M. Terzi, C. Masiero, A. Beghi, and G. A. Susto, "A computer vision-inspired deep learning architecture for virtual metrology modeling with 2-dimensional data," *IEEE Transactions on Semiconductor Manufacturing*, vol. 31, pp. 376–384, 8 2018.

[14] A. Schirru, S. Pampuri, C. D. Luca, and G. D. Nicolao, "Multilevel kernel methods for virtual metrology in semiconductor manufacturing," vol. 44. IFAC Secretariat, 2011, pp. 11 614–11 621.

[15] N. Gentner, A. Kyek, Y. Yang, M. Carletti, and G. A. Susto, *Enhancing scalability of virtual metrology: a deep learning-based approach for domain adaptation*.

[16] S. J. Pan and Q. Yang, "A survey on transfer learning," pp. 1345–1359, 2010.