Deep Feature Selection and Beyond

Milestone Report P6 – GS-IMTR

Yiwen Liao

May 10, 2022

GS-IMTR
Graduate School
Intelligent Methods for Test and Reliability

Advisor / main examiner: Prof. Dr.-Ing. Bin Yang

Co-examiner: Prof. Dr. rer. nat. Dirk Pflüger

Mentor: Jochen Rivoir and Raphaël Latty

Contents

1 Introduction

This technical report presents the accomplished, on-going and future research for the Project 6 (P6) of the Graduate School - Intelligent Methods for Test and Reliability (GS-IMTR). P6 started in January 2020 and is expected to finish in December 2022. The goal is to study and design Feature Selection (FS) approaches based on Deep Learning (DL) and one of the main use cases is variable selection for Post-Silicon Validation (PSV).

1.1 Brief Review

This section takes a brief review on the conducted research in a chronological order. A brief timeline for an intuitive overview is shown in Figure 1.

In the first half of 2020, we conducted broad literature review on the DL-based as well as representative conventional FS approaches. Additionally, in order to have a more intuitive understanding of the research target, we evaluated a few existing state-of-the-art FS methods [1, 2] respectively on the Maybach and Tuning datasets by Advantest. At the beginning of the second half of 2020, we had the first idea of the Feature Mask (FM) method [3], which became one of the core ideas for subsequent research. After a few months' iterative studies, the corresponding paper was submitted to IJCNN-2021 and finally accepted in April 2021. During the meantime, a two-stage weighting approach based on the FM-module was proposed to enhance the anomaly detection performance based on an autoencoder (AE). The corresponding paper [4] was afterwards published at CASE-2021.

In the first half of 2021, we mainly studied the extension capabilities of the FM method for PSV, including the feasibility of different feature representations and various downstream tasks (e.g. regression, classification and multi-task learning). In the mean time, an extended abstract on applying the FM-method to PSV was accepted by TuZ-2022 [5]. Moreover, instead of our two-stage method proposed in [4], we directly applied the FM-block to autoencoders in an end-to-end manner for One-Class Classification (OCC). The new approach significantly improved AE-based OCC performance, which has been accepted by IJCNN-2022 [6]. Inherited from the same idea, in 2022, we applied the FM-method to detect resistive open defects under process variations for combinational circuits in collaboration with AP1. Finally, as a first experiment, we applied the FM-method to multi-class classification for wafer map defect pattern recognition in collaboration with P10.

Currently, according to the special requirements from Advantest, we have been exploring the possibility of conditional feature selection based on the FM-method. An extended abstract has been accepted by the IMTR-Workshop at ETS-2022 [7].

1.2 Report Outline

This report first introduces a general framework of DL-based FS methods as basic knowledge in Section 2. Secondly, Section 3 presents the core idea for all of our subsequent studies, namely the FM method. Following that, we show the applications of the FM method to anomaly detection and OCC scenarios as well as two main collaborations in Section 4. Subsequently, Section 5 briefly discusses the on-going research on conditional FS. Finally, this report ends up with the plan for the next steps in Section 6.

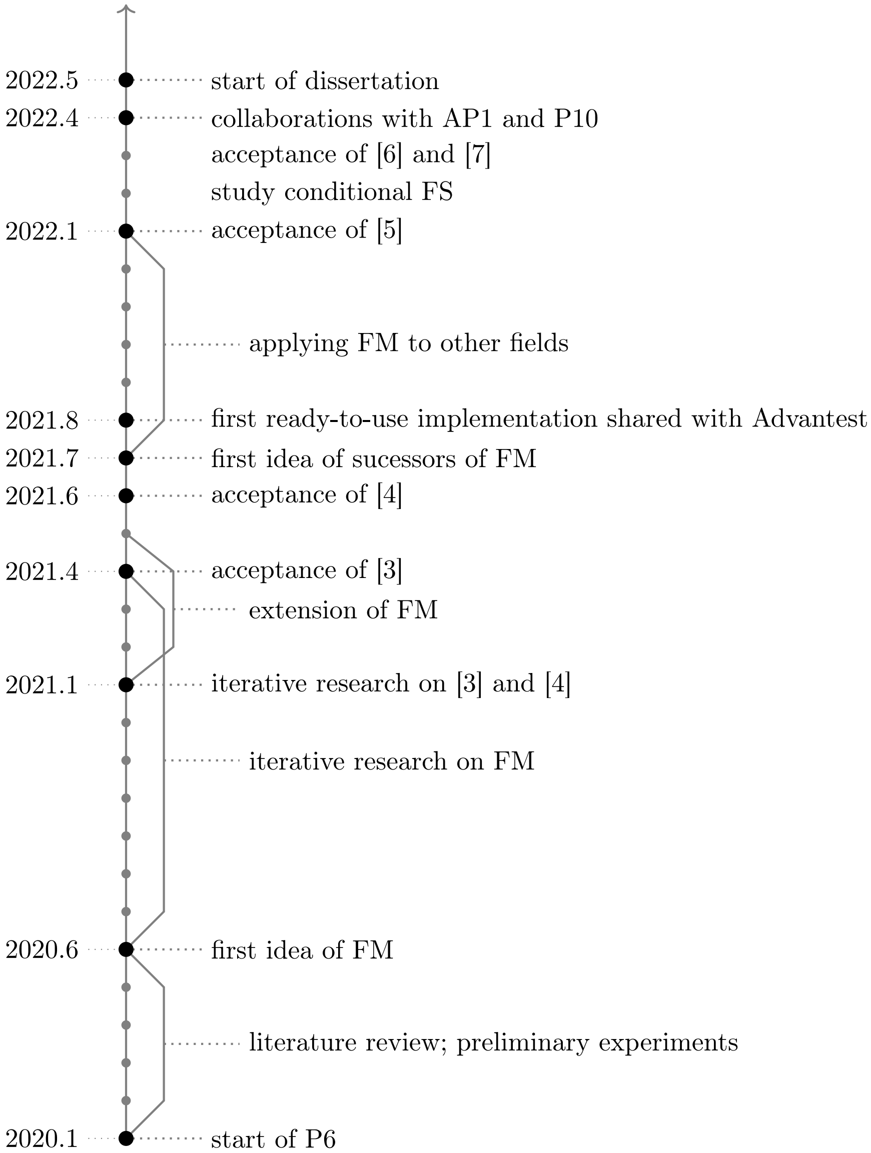

2022.5
start of dissertation
2022.4
collaborations with AP1 and P10
acceptance of [6] and [7]
study conditional FS
2022.1
acceptance of [5]
applying FM to other fields
2021.8
first ready-to-use implementation shared with Advantest
2021.7
first idea of sucessors of FM
2021.6
acceptance of [4]
2021.4
acceptance of [3]
extension of FM
2021.1
iterative research on [3] and [4]
iterative research on FM
2020.6
first idea of FM
literature review; preliminary experiments
2020.1
start of P6

Figure 1: A brief timeline for the conducted research of P6.

2 Background

This section presents a general structure of popular DL-based feature selection approaches in literature. Broadly speaking, most DL-based FS methods follow the structure shown in Figure 2. They simultaneously learn a feature mask $\boldsymbol{m}$ and a neural network $g_{\boldsymbol{\Theta}_n}(\cdot)$ that maximizes the prediction performance with carefully designed constraints on $\boldsymbol{m}$ and/or special generation process of $\boldsymbol{m}$.

Definition 1 (Feature Mask $\boldsymbol{m}$)**.** $\boldsymbol{m}$ is a vector, the dimension of which corresponds to the number of all candidate features. The value of each entry in $\boldsymbol{m}$ indicates the importance of the corresponding candidate feature. Typically, a larger value indicates more importance. $\boldsymbol{m}$ is also called *feature importance vector*, *feature weights* and *feature coefficients* in literature. It should noted that the value of $\boldsymbol{m}$ is typically non-negative and bounded. For example, common value ranges include $[0, 1]$ as in [8] and $(0, 1)$ as in [1].

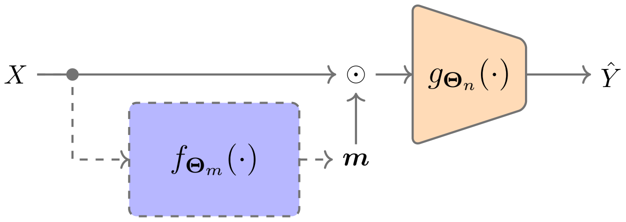

X
$\odot$
$g_{\boldsymbol{\Theta}_n}(\cdot)$
$\hat{Y}$
$f_{\boldsymbol{\Theta}_m}(\cdot)$
$\boldsymbol{m}$

Figure 2: A typical DL-based feature selection framework. It jointly learns the feature mask $\boldsymbol{m}$ and a learning network $g_{\boldsymbol{\Theta}_n}(\cdot)$. For some methods, $\boldsymbol{m}$ is dependent of the input data X and this path is illustrated in a dashed line.

In Figure 2, the dashed line illustrates the path from input $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^T \in \mathbb{R}^{n \times d}$ to the feature mask $\boldsymbol{m} \in \mathbb{R}^d$. This is because some prior works such as [9, 10] assume that $\boldsymbol{m}$ is independent of X during the forward propagation. In this case, $\boldsymbol{m}$ can be understood as a jointly trainable weights of the entire network. On the contrary, some methods such as [1, 3] leverage attention mechanism and $\boldsymbol{m}$ is thus directly dependent of the input data which is calculated as $\boldsymbol{m} = f_{\boldsymbol{\Theta}_m}(X)$. During training, for many methods such as [9, 10], additional regularization $\mathcal{R}(\cdot)$ must be applied to the feature mask $\boldsymbol{m}$ to guarantee certain properties like sparsity in $\boldsymbol{m}$. In total, the learning objective of DL-based FS methods can be formulated as

$$\operatorname*{argmin}_{\boldsymbol{m}, \boldsymbol{\Theta}} \mathcal{L}(g(\boldsymbol{x} \odot \boldsymbol{m}), y) + \lambda \cdot \mathcal{R}(\boldsymbol{m}), \tag{1}$$

where $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}_n, \boldsymbol{\Theta}_m\}$ and we omit the subscript for $g_{\boldsymbol{\Theta}_n}(\cdot)$ for simplicity. In the learning objective, the first term is a loss function towards the learning task and λ is a weighting factor balancing the two terms.

In should be additionally mentioned that the commonly used regularization (e.g. ℓ_2 weight decay) on the weights of neural networks can be added to the aforementioned learning objective to avoid overfitting, but such regularization is not directly related to the FS performance and we omit it for simplicity.

Overall, existing DL-based FS approaches typically aim to design special regularization terms or novel ways to better regularize or generate $\boldsymbol{m}$ and it is expected that important features have greater weights than unimportant features during the training. Finally, after training, we can select the top-k features by comparing the importance scores in $\boldsymbol{m}$.

3 Feature Mask (FM)

The core idea of this report as well as the whole P6 is the Feature Mask (FM) method based on a novel batch-wise attenuation and feature mask normalization[1]. The overall structure is shown in Fig. 3.

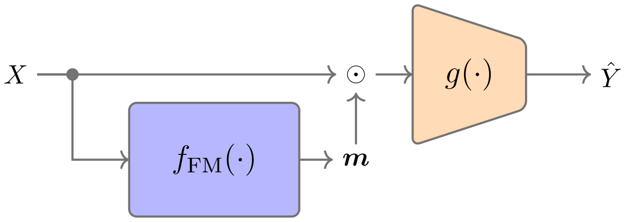


Figure 3: The feature selection framework based on the FM method.

3.1 Feature Mask Module

The FM-module, denoted as $f_{\text{FM}}(\cdot)$, generates *one* unique feature mask $\boldsymbol{m}$ for the entire dataset as: $\boldsymbol{m} = f_{\text{FM}}(X)$. As shown in Fig. 4, the FM-module consists of three submodules below.

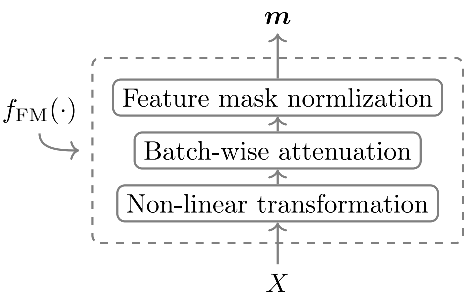


Figure 4: The structure of the FM-module.

Non-linear Transformation During training, the minibatch $X_B \in \mathbb{R}^{b\times d}$ is mapped to $Z_B = [\boldsymbol{z}_1, \boldsymbol{z}_2, \cdots, \boldsymbol{z}_b]^\top \in \mathbb{R}^{b\times d}$ under a non-linear transformation. In this way, the complex (non-linear) relations between different input features are expected to be captured during training. In this paper, as an example, the non-linear transformation is defined as

$$\boldsymbol{z_i} = W_2 \cdot \phi(W_1 \cdot \boldsymbol{x}_i + \boldsymbol{b}_1) + \boldsymbol{b}_2 \ , \tag{2}$$

where $W_1, W_2, \boldsymbol{b}_1, \boldsymbol{b}_2$ are of suitable dimensions and $\phi(\cdot)$ is a nonlinear function such as $\tanh(\cdot)$.

Batch-Wise Attenuation Each row vector of the resulting Z_B depends on the corresponding input sample. However, feature selection generally requires that the given data should have the same important features. Hence, by explicitly averaging Z_B over the minibatch, a unique vector for all samples within a minibatch can be obtained during each training iteration. Specifically, for each minibatch, we calculate

$$\bar{\boldsymbol{z}} = \frac{1}{B}\sum_{i=1}^{B} \boldsymbol{z}_i \ . \tag{3}$$

Feature Mask Normalization The relative importance of different candidate features should be considered during training. In this work, we use the *softmax* function to normalize $\bar{\boldsymbol{z}}$:

$$\boldsymbol{m} = \mathit{softmax}(\bar{\boldsymbol{z}}), \text{ with } m_i = \frac{e^{\bar{z}_i}}{\sum_{j=1}^{D} e^{\bar{z}_j}}. \tag{4}$$

As can be seen above, the FM-module generates one *unique* feature mask $\boldsymbol{m}$ for the entire minibatch X_B during each training step. This idea can be interpreted as a novel Batch-Wise Attention mechanism.

[1] Some parts of this section are taken from our published work [3].

3.2 Main Results of FM

The proposed FM method achieved superior performance in comparison with other contemporary approaches in a supervised manner, as shown Figure 5. One of the main advantages of our method is that the FM-method does not require additional regularization $\mathcal{R}(\cdot)$ on $\boldsymbol{m}$, meaning we do not have as many hyperparameters as those of other reference methods. Furthermore, the FM-method can be used for both supervised and unsupervised feature selection tasks. We refer the interested readers to our full paper [3].

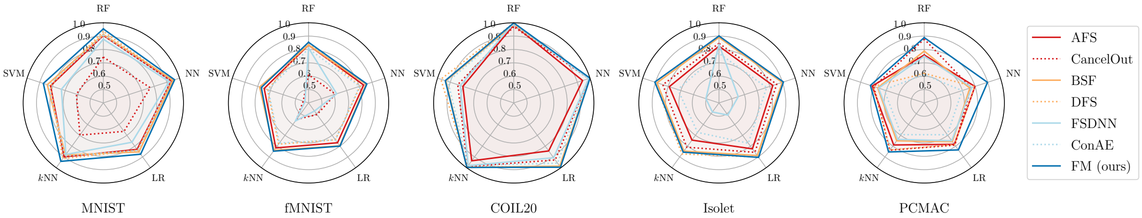


Figure 5: Feature selection performance of the FM method in comparison with the following recent reference methods: AFS [1], CancelOut [10], BSF [8], DFS [9], FSDNN [11] and ConAE [2].

4 FM-Derived Applications

As shown before, $\boldsymbol{m}$ actually denotes the importance for each individual feature. Therefore, a natural idea is to leverage the FM-module as a batch-wise attention block to different use cases[2]. This section briefly introduces the four conducted applications. More details can be found in [4, 6].

4.1 Two-Stage Semi-Supervised Anomaly Detection

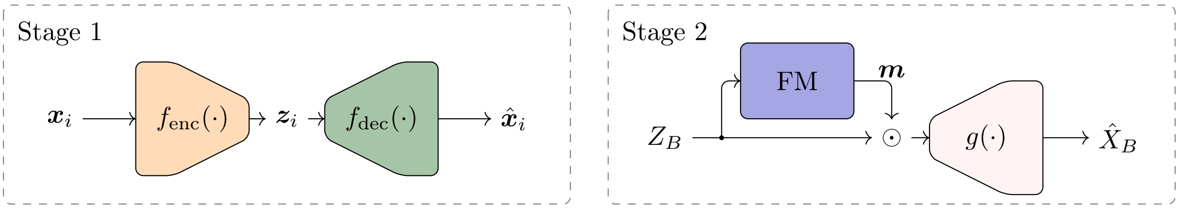


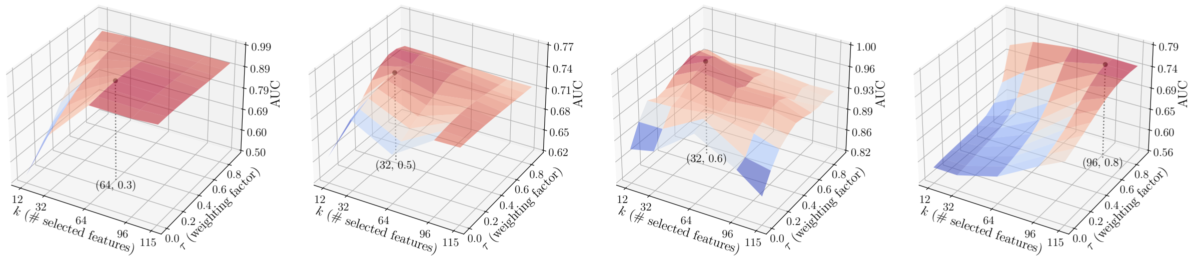
Figure 6: The two-stage semi-supervised anomaly detection framework based on the FM-module.

[4] is our first paper that has applied the FM-module to other fields beyond feature selection. Specifically, given a trained autoencoder (left of Figure 6), we train an FM-module jointly with a separate decoder to identify the critical learned latent dimensions. After training, we can specify the number of important latent features κ and a down-weight factor τ. Then, the less important latent features are down-weighted by multiplying τ. Subsequently, weighted latent representations are fed to the decoder to obtain reconstructions. Finally, the anomaly detection is performed by comparing the reconstruction errors as other AE-based anomaly detection algorithms.

This method reached 1.3% better AUC on MNIST and 6.9% better AUC on CIFAR-10 than an autoencoder in terms of anomaly detection. Figure 7 demonstrates the AUC versus κ and τ for four exemplary normal classes. The resulting surfaces were not flat. This indicates the necessity of the selection and weighting for a learned latent space.

This method is important because it for the first time shows the selection and weighting for a trained autoencoder still have contribution for better anomaly detection performance. Despite the improvements brought by the second stage training and weighting in latent space, this method still faces a few challenges. Firstly, it introduces two more hyperparameters which are data dependent and thus difficult to fine-tune.

[2] Some parts of this section are taken from [4, 6].



(a) MNIST (digit 8). **(b)** CIFAR10 (deer). **(c)** Solar cells. **(d)** MVTec (hazelnut).

Figure 7: AUC versus the two hyperparameters κ and τ.

Secondly, the training of $\boldsymbol{m}$ and the autoencoder is isolated. Therefore, it is effortful to ensure whether the autoencoder is correctly trained for the second stage.

The concerns and drawbacks mentioned above enable our end-to-end solution which is presented in the next section.

4.2 Weighted Autoencoder for One-Class Classification

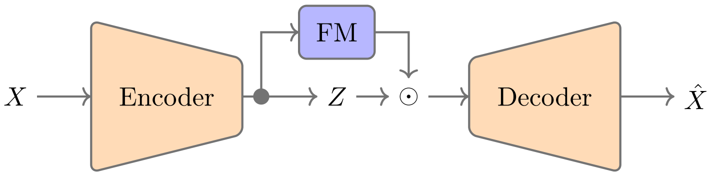


Figure 8: Autoencoder with weighted latent space.

In [6], we directly apply the FM-module to the latent space of an autoencoder to obtain an end-to-end model as shown in Figure 8. It should be noted that the FM-module is not used for selecting the latent features but weighting the latent features before feeding them to the decoder. In this way, it is expected the most critical and representative latent features can be more focused, while the misleading latent features are down-weighted.

This method is superior to the two-stage method [4] introduced above due to the absence of the two hyperparameters κ and τ. Nonetheless, the novel end-to-end method achieved more than 1.8% better AUC on MNIST and about 14.7% better AUC on the challenging dataset CIFAR-10 in comparison with a vanilla autoencoder with the same architecture. Furthermore, this work also discovered the performance degradation in autoencoders for one-class classification and the FM-module can significantly reduce the degradation. Figure 9 shows the comparison on the two benchmarking datasets between our method (BFW) and other reference methods.

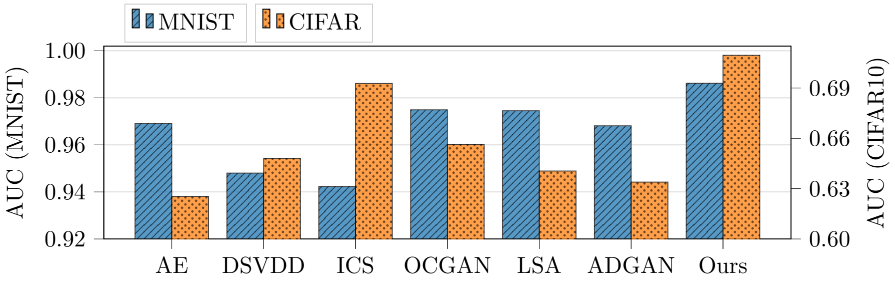


Figure 9: OCC performance comparison between our method (BFW) and the following reference state-of-the-art approaches: DSVDD [12], ICS [13], OCGAN [14], LSA [15] and ADGAN [16].

4.3 Collaborations

Unsupervised Resistive Open Defect Identification. Leveraging the weighted autoencoder presented in the last section, our recent work aims to detect resistive open defects under process variations in collaboration with AP1 of GS-IMTR. Experiments confirms that our method outperform other unsupervised anomaly detection algorithms as shown in Table 1. Moreover, as an unsupervised algorithm, our method even had comparable performance to the supervised reference methods. We refer the interested readers to our paper [17] for more details. This collaboration indicates that our FM-method can not only work for academic benchmarking datasets but also valuable for volume testing.

Table 1: Comparison with unsupervised methods.

	OCSVM [18]	Isolation Forest [19]	LOF [20]	AE	Ours
AUC	0.895 (± 0.000)	0.880 (± 0.008)	0.889 (± 0.000)	0.890 (± 0.006)	**0.942** (± 0.005)

Wafer Map Defect Pattern Classification. A recent collaboration with P10 combines the weighted autoencoder with an ordinary multi-class neural network for wafer map defect pattern recognition. In this way, the proposed framework simultaneously predict the defect pattern class as well as the state of each individual die in a wafer map. The novel architecture achieved comparable or better accuracy than other state-of-the-art DL-based wafer map classifiers. For example, Table 2 shows the overall recall comparison between our method and the other two popular methods. The collaboration with P10 implies the potential of the FM-method in multi-class classification problems.

Table 2: Comparison of the recalls for each defect type.

	None	Center	Donut	Edge-Loc	Edge-Ring	Loc	Random	Scratch	Near-Full
RF [21]	99.2%	87.0%	63.5%	58.3%	92.9%	27.1%	55.7%	9.8%	84.2%
DMC [22]	**99.4**%	90.7%	79.5%	**76.0**%	96.7%	**66.5**%	82.2%	21.6%	88.6%
Ours	**99.4**%	**93.1**%	**80.8**%	72.9%	**97.3**%	59.6%	**91.1**%	**28.6**%	**89.5**%

Comments. In total, the two recently accomplished collaborative projects have shown the effectiveness and robustness of the proposed novel FM-method in different domains and tasks. In addition, FM-method has been therefore evaluated in both supervised and unsupervised, one-class and multi-class classification scenarios. The conducted experiments present great prospects of our research.

5 On-Going Research

5.1 Conditional Feature Selection

The current research focuses on conditional feature selection, where candidate features should be selected given some preselected features. To enable this, we propose to fuse the information from the preselected and candidate features in the shallow layers of a neural network targeting a given learning task. As a result, the neural network acts as a guide to the feature selection module in a way that candidate features, which can minimize the training losses given the preselected features, should be assigned with greater importance scores.

According to user specifications, X is composed of two parts as $X = [X_p, X_n]$. That is, the preselected D_p features $X_p \in \mathbb{R}^{N \times D_p}$ and the D_c candidate features $X_c \in \mathbb{R}^{N \times D_c}$ with $D = D_c + D_p$. The overall framework is illustrated in Fig. 10. The FM-module generates $\boldsymbol{m} \in \mathbb{R}^{D_c}$. Afterwards, a transformation $f_c(\cdot)$ is applied to $X_c \odot \boldsymbol{m}$. In parallel, another independent transformation $f_p(\cdot)$ is applied to the preselected features X_p. Next, denoted by "concate" in Fig. 10, we fuse the outputs from both transformations by concatenating them together. The fused representations are fed to a neural network $g(\cdot)$ to obtain predictions $\hat{Y}$. Specifically, the preliminary research implements both encoding mappings f_p and f_c by simple dense layer with non-linear activation. Naturally, the design of the two encoding mappings can play a key role for the overall selection performance and we have been working on this issue.

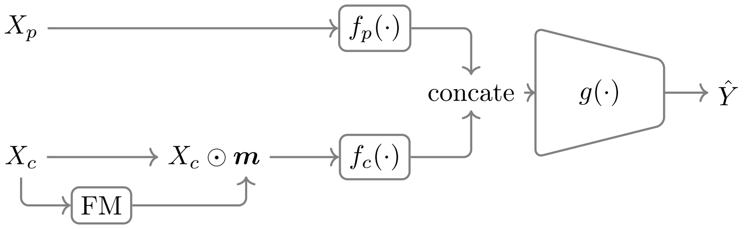


Figure 10: The proposed framework for conditional feature selection.

In this scenario, we primarily focus on the case where we select features based on some given preselected features (as conditions). Nevertheless, the whole idea in Figure 10 can be easily extended to more generic cases. For example, the "condition" is no longer or not limited to preselected features, but can be additional data, expert knowledge and new attributes to the given datasets. For all the cases mentioned before, we can simply use f_p to encode them into suitable representations as conditions to train the whole algorithm to obtain conditioned $\boldsymbol{m}$. This is an important on-going work by us.

5.2 Preliminary Results

Case I: Voltage Selection for Open Circuit Detection This experiment investigates delays under which features (varying voltages) are critical for defect identification given the delays under the voltage of 0.9V. In total, we had respectively 1000 defective and 1000 non-defective samples with $D_c = 11$ and $D_p = 1$, meaning that the conditional feature selection is performed towards a binary classification problem. Fig. 11 shows the resulting accuracy over different subset sizes K from 1 to 12. $K = 1$ means that we selected 0.9V as the only feature, while the other sizes denote that we select 0.9V and $K - 1$ different voltages. Overall, our method shows similar performance as an exhaustive search, while the exhaustive search is not feasible in practice due to its exponential computational complexity.

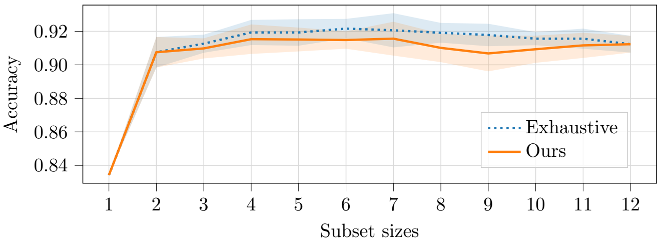


Figure 11: Accuracy over different subset sizes for our method and an exhaustive search.

Case II: Feature Selection for Post-Silicon Tuning In this experiment, the goal was to find out the four most important candidate features under the condition that t_2 was already preselected, i.e. $D_c = 10$ and $D_p = 1$. For the exhaustive search, we need to evaluate $\binom{10}{4} = 210$ different candidate feature combinations and trained a 3-layer multilayer perceptron by minimizing the MSE loss between the predictions and the Figure-of-Merit (FoM). In total, it took about 86 minutes for the exhaustive search and the optimal selection result was the combination of t_1 to t_5. On the contrary, our method was trained only once by feeding all 10 candidate features and preselected features to the model. The learned feature importance vector is shown in Figure. 12. We can see that t_1, t_3, t_4 and t_5 obtained the greatest importance scores and were considered as the best candidate features given the preselected feature t_2, which matched the aforementioned exhaustive search results. It should be emphasized that it took about only 25 seconds (i.e. 0.4 minutes) to perform the full training for our method, corresponding to about 0.48% time consumption of an exhaustive search with the same learning network architecture. The experimental results show that our method is promising to efficiently solve conditional feature selection tasks in test and reliability.

Note that the detailed information about both datasets above can be found in [5, 17].

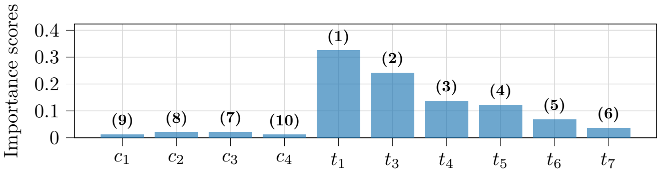


Figure 12: The learned importance scores of our method for a real-world dataset. Larger scores indicate higher importance of the corresponding candidate features.

6 Next Steps

Due to the time limitation, the main task in the rest time (about 8 months on the assumption of no extensions) of this project is working on the doctoral dissertation. Therefore, the next steps for this project are planned from the theoretical and implementation perspectives.

6.1 Theoretical Perspective

In order to maintain the consistency through different proposed approaches, we will mainly focus on the following topics:

- Extensive evaluation of the proposed methods (i.e., FM, CFM and conditional FM) as well as possible new approaches on more public benchmarking datasets provided by [23] to obtain an objective justification of the feature selection capability of our methods.
- Due to the paper length limitations, we omitted comprehensive study of hyperparameters in some published and under-review papers. Thereby, next steps must include thorough study of hyperparameters of proposed methods on representative public benchmarking datasets as well as confidential datasets from Advantest.
- Further study and comparison between the proposed FM-method and other tightly related techniques such as popular (sample-wise) attention techniques [24] as well as Batch Normalization [25].
- Although there are a few great survey papers in feature selection such as [23, 26], to the best extent of our knowledge, there is no survey papers targets DL-based feature selection. However, from the top machine learning conferences, we can clearly see the increasing interest in using DL to solve feature selection problems. In the rest time of this project, we try to keep reviewing as many DL-based FS papers as possible to make the background section of the dissertation more solid and comprehensive.

6.2 Implementation Perspective

Currently, we already have a (revised) ready-to-use implementation for the FM method. The existing implementation supports:

- End-to-end training towards different downstream learning tasks, e.g. classification, regression and multi-task learning;
- Customized latent dimensions for the FM-block;
- Customized learning networks $g(\cdot)$;
- Several other DL-based FS algorithms.

Accordingly, the final version must consider the following requirements and functions:

- The Complementary Feature Mask method should be included, in which different and customized complementary feature masks as well as losses for the complementary path should be supported;
- The final version must support conditional feature selection, in which both the preselected and candidate features can be encoded according to user's requirements;
- Automatic hyperparameter tuning based on Bayesian Optimization should be included;
- Easy interfaces for P2 (visualization).

References

[1] Ning Gui, Danni Ge, and Ziyin Hu. Afs: An attention-based mechanism for supervised feature selection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3705–3713, 2019.

[2] Muhammed Fatih Balın, Abubakar Abid, and James Zou. Concrete autoencoders: Differentiable feature selection and reconstruction. In *International conference on machine learning*, pages 444–453. PMLR, 2019.

[3] Yiwen Liao, Raphaël Latty, and Bin Yang. Feature selection using batch-wise attenuation and feature mask normalization. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9. IEEE, 2021.

[4] Yiwen Liao, Alexander Bartler, and Bin Yang. Anomaly detection based on selection and weighting in latent space. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 409–415. IEEE, 2021.

[5] Yiwen Liao, Jochen Rivoir, Raphaël Latty, and Bin Yang. A deep-learning-aided pipeline for efficient post-silicon tuning. In *2022 34th workshop on Test Methods and Reliability of Circuits and Systems (TuZ)*, 2022.

[6] Yiwen Liao and Bin Yang. To generalize or not to generalize: Towards autoencoders in one-class classification. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 409–415, 2022.

[7] Yiwen Liao, Tianjie Ge, Raphaël Latty, and Bin Yang. Conditional variable selection for intelligent test. In *2022 Workshop on Intelligent Methods for Test and Reliability at European Test Symposium*, 2022.

[8] Andrii Trelin and Aleš Procházka. Binary stochastic filtering: feature selection and beyond, 2020.

[9] Yifeng Li, Chih-Yu Chen, and Wyeth W Wasserman. Deep feature selection: theory and application to identify enhancers and promoters. *Journal of Computational Biology*, 23(5):322–336, 2016.

[10] Vadim Borisov, Johannes Haug, and Gjergji Kasneci. Cancelout: A layer for feature selection in deep neural networks. In *International Conference on Artificial Neural Networks*, pages 72–83. Springer, 2019.

[11] Debaditya Roy, K Sri Rama Murty, and C Krishna Mohan. Feature selection using deep neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2015.

[12] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402, 2018.

[13] Patrick Schlachter, Yiwen Liao, and Bin Yang. Deep one-class classification using intra-class splitting. In *2019 IEEE Data Science Workshop (DSW)*, pages 100–104, 2019.

[14] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2898–2906, 2019.

[15] Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent space autoregression for novelty detection. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 481–490, 2019.

[16] Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 3–17. Springer, 2018.

[17] Yiwen Liao, Zahra Paria Najafi-Haghi, Hans-Joachim Wunderlich, and Bin Yang. Efficient and robust resistive open defect detection based on unsupervised deep learning, 2022.

[18] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[19] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008.

[20] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[21] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[22] Tsung-Han Tsai and Yu-Chen Lee. A light-weight neural network for wafer map classification based on data augmentation. *IEEE Transactions on Semiconductor Manufacturing*, 33(4):663–672, 2020.

[23] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017.

[24] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[25] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[26] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.