# Interpretation and Simplification of Deep Forest

Sangwon Kim, Mira Jeong, Byoung Chul Ko
Keimyung University, Daegu, South Korea
{eddiesangwonkim, mystroll24}@gmail.com, niceko@kmu.ac.kr

## Abstract

*This paper proposes a new method for interpreting and simplifying a black box model of a deep random forest (RF) using a proposed rule elimination. In deep RF, a large number of decision trees are connected to multiple layers, thereby making an analysis difficult. It has a high performance similar to that of a deep neural network (DNN), but achieves a better generalizability. Therefore, in this study, we consider quantifying the feature contributions and frequency of the fully trained deep RF in the form of a decision rule set. The feature contributions provide a basis for determining how features affect the decision process in a rule set. Model simplification is achieved by eliminating unnecessary rules by measuring the feature contributions. Consequently, the simplified model has fewer parameters and rules than before. Experiment results have shown that a feature contribution analysis allows a black box model to be decomposed for quantitatively interpreting a rule set. The proposed method was successfully applied to various deep RF models and benchmark datasets while maintaining a robust performance despite the elimination of a large number of rules.*

## 1. Introduction

Although the structures of recent deep neural networks (DNNs) continue to deepen and widen, resulting in improved recognition rates, several challenges remain: 1) When a DNN encounters a scenario that differs from the scenario used during the training phase, an instability occurs in that the structure cannot be modified based on the scenario. 2) A DNN is programmed on the basis of a small amount of knowledge and is superficial in that it does not have common sense regarding the world and human psychology [1]. 3) Recent DNN models continue to become wider and deeper to achieve a better performance, and may not be suitable for a variety of applications with limited memory or computational times. 4) A DNN system is greedy because it requires numerous training data. Finally, 5) because the output of a DNN is calculated through a black box, it cannot be accurately explained.

The first and second issues require more research to reduce the structural gap between a DNN and the actual human brain in terms of neuroscience, whereas the remaining issues can be solved by changing the current structure of the DNN. To reduce the size of the DNN network (issue 3), some studies have focused on compressing a DNN with a similar performance as the original models while reducing the size and width of the DNN network, e.g., using a knowledge distillation [2] [3], transferred/compact convolutional filters [4], low-rank factorization, and parameter tuning and sharing [5]. However, a compressed DNN model still requires a large number of parameters and a large amount of memory for processing the resources required for multiplication [6]. In addition, to create a deep model that can be trained using a small number of training data (issue 4) without a backpropagation, new approaches have been attempted for linking random forests (RFs) [7] [8] or random ferns [9] to layers instead of neurons in a deep model. These deep random classifier-based models link several ensemble algorithms to multiple layers with non-differentiable components and do not use backpropagation during training.

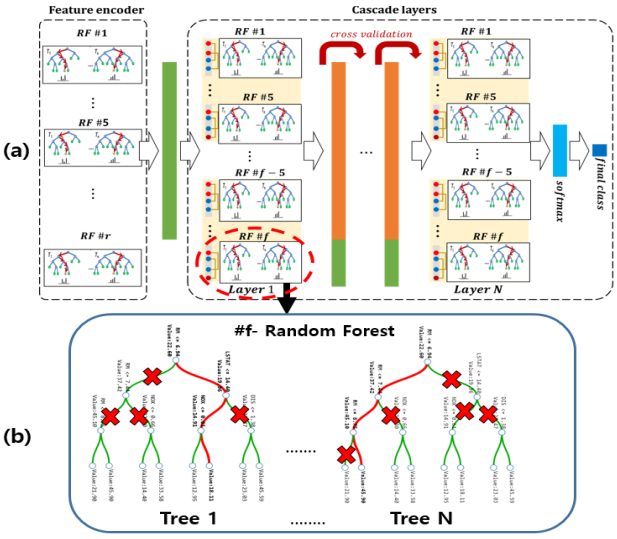


Figure 1. Overall architecture of interpretation using a deep random forest: (a) After a deep random forest is generated, each random forest consists of a large number of decision trees, and (b) a random forest can be explained and interpreted by decomposing rules based on the feature contribution.

Recently, studies on explainable or interpretable AI (XAI) have been actively conducted to improve the limitations of a black box model regarding the learning process (issue 5 above), which is an issue of deep learning.

XAI is a technology that allows humans to understand and correctly interpret the behavior and end result of an AI model to explain how the result is generated [10]. Therefore, unlike a black box model, users can check whether the decision made by an AI model is the best decision before making the final determination through a white box model.

Studies on the development and testing of an XAI learning model have been conducted to improve the explainability while maintaining a high-level learning ability by modifying the existing machine learning technologies or developing new ones. The technical approach for XAI can be divided into the following: (a) explaining a decision of the learning model (ELM) [11]–[15] and (b) interpreting the learning model (ILM)[16]–[18].

The ELM does not have full control over the model's structure, and focuses on explaining the model's decision by identifying the relevant input variable. Interpretation techniques are applied to the output of nonlinear machine learning models to produce a heat map (e.g., a highlighted image or text) of the interpretable input variables [11]. The ELM method gives users an extremely intuitive result by emphasizing the input variables through a prediction and redistribution of the learning model to determine which part of the input feature influenced the outcome. Bach et al. [12] proposed a way to visualize the contribution of a single pixel to kernel-based classifiers and multi-layer neural networks over the Bag of Words feature. These pixel contributions can be visualized through a heat map and provided to the users to intuitively verify the effectiveness of the classification decision. Montavon et al. [13] proposed a deep Taylor decomposition for interpreting generic multilayer neural networks by decomposing the network classification decision into contributions of its input elements. This method efficiently utilizes the structure of the network by backpropagating the explanations from the output to the input layer. Anders et al. [14] applied a deep Taylor/Layer-wise relevance propagation (LRP) technique to video data to understand the classification decisions of a deep network trained using this strategy. This method also identifies the tendency of the classifier to look mainly at frames that are close to the temporal boundaries of the input clip. Shi et al. [15] proposed the eXplainable and eXplicit neural modules that allow visual reasoning over scene graphs, as represented by different detection qualities. This method can insulate the "low-level" visual perception achieved by the modules, and can thus prevent a shortcut in reasoning of both the language and vision counterparts.

However, because ELM-based methods still depend on a complex black box model, it is difficult to explain what rules are used in the deep network to actually reach such a decision.

ILM aims to produce models that are inherently interpretable in a different way from a black box-based ELM approach. The representative model of an ILM is a rule-based algorithm, such as stochastic AND/OR graphs (AOGs), decision lists, and decision trees, because users can easily understand simple rules [16]. An AOG [17] generates an AND-OR relationship graph of the characteristics of the input data (e.g., sketch, color, texture, and position of an object in an image) and confirms the classification based on the node connected to the classification result. Liu et al. [18] proposed a rule-based regression algorithm that uses 1-norm regularized RFs. This approach simultaneously extracts a small number of rules from the generated RF and eliminates unimportant features. However, if the rules of the trees are excessively reduced to increase the analysis capability of the model, an issue occurs in that the performance is significantly reduced.

Bayesian rule lists (BRLs) [19] are based on a decision tree as a preliminary interpretable model providing a concise and convincing capability to gain the trust of domain experts. A BRL employs a prior structure to encourage sparsity and yield a posterior distribution over the possible decision lists. Lakkaraju et al. [20] proposed interpretable decision sets, which are sets of independent if-then rules, and a framework for building predictive models that are highly accurate and yet highly interpretable. Because each rule can be applied independently, decision sets are simple, concise, and easily interpretable. A scalable Bayesian rule list (SBRL) [21] was proposed as a faster variant of a BRL. An SBRL is used to build probabilistic rule lists that are two orders of magnitude faster than the previous BRL. Rule list algorithms are competitors to decision tree algorithms and are associative classifiers in that they are built from pre-mined association rules. However, such methods have an issue in that their performance is significantly degraded when excessively reducing the rules of the tree to improve the interpreting power of the model.

By contrast, other researches have tried to improve the interpretability by changing the structure of the NN. Yang et al. [22] proposed the use of an explainable NN subject to interpretability constraints in terms of the additivity, sparsity, orthogonality, and smoothness. A complex function is decomposed into sparse additive subnetworks and the projection indexes are forced to be mutually orthogonal such that the resulting subnetworks tend to be less confounded with each other. However, a NN-based method still depends on the backpropagation, which requires the use of a black box model during the learning process. In addition, in terms of transparency in a machine learning approach, the choice of hyper-parameters such as the learning rate and batch size has a more heuristic, non-transparent algorithmic nature [23].

In this study, we focus on the development of a new ILM-based interpreting method instead of an ELM-based approach to interpret and simplify a deep method, which

can maintain the important properties of the model structure and redefine the rules without sacrificing the performance. Unlike an ELM-based approach that focuses on a heat map of the input variables when using a DNN, we wish to understand why particular decisions were made and generate models explaining such decisions while maintaining the same predictive performance.

A new type of deep model, a deep RF, has been proposed to achieve an interpretable deep model and maintain a DNN-like performance. It links several RFs to multiple layers with non-differentiable components and does not use backpropagation during training [7] [8], as mentioned in the Introduction. Although deep RF generally achieves a high performance similar to that of a DNN, it generates a large number of rules because it is also composed of black box RFs. Therefore, a large number of rules are a significant obstacle to interpreting the results of the deep RF.

After a deep RF is trained and multi-layer networks are generated using several RFs (see, Fig 1 (a)), we first decompose the predictions of each decision tree in the RF into mathematically exact feature contributions. Individual predictions of the decision tree can be explained by breaking down the decision path into a single component per feature. This procedure is iteratively applied to find all rules of the entire RF layer by layer and saved to decision sets, which are sets of classification rules of an RF, (see the example in Fig. 1). Sequential covering then repeatedly maintains and eliminates rules of the decision set of an RF based on a combination of the rule contribution and feature pattern (frequency of feature). This regularization keeps only a small number of refined rules that are the most discriminative. After the sequential covering, we have the same number of decision sets per layer, but the numbers of rules and features are significantly reduced without decreasing the performance. Herein, we provide the qualitative and quantitative results demonstrating that our proposed interpreting method is highly reasonable and effective for improving the interpretability.

## 2. Related Works

As described in the Introduction, the purpose of this study is to propose a new interpreting algorithm based on ILM using a deep RF that shows a performance similar to that of a DNN but does not rely on a backpropagation. Therefore, this section introduces the related research focusing on a deep RF. Apart from the high recognition rate of a DNN, certain limitations such as an overly large number of hyper-parameters requiring parameter tuning, a black box model created through a gradient backpropagation, high processing costs, and the amount of training data are a significant burden to explain a DNN [24]. As an alternative approach, a deep ensemble classifier consisting of several RFs or ferns has been researched.

A multi-grained cascade forest called gcForest [7] was the initial trial to generate a deep forest ensemble with a cascade structure. To avoid a gradient backpropagation, the cascade levels are adaptively determined using an N-fold cross validation, which provides a performance similar to that of a DNN, although it was trained using only a small amount of data. A forward thinking deep random forest (FTDRF) [8] replaces the neurons of deep neural nets with decision trees instead of RFs. Input data are mapped forward through the layers to create a new learning problem. This process is repeated to convert the data of a single layer into multiple layers at a time. Multilayered gradient boosting decision trees (mGBDTs) [25] build blocks for each layer with an explicit emphasis on representation learning to learn hierarchical distributed representations through the stacking of several layers of a regression GBDT.

As the application of a deep RF, a Siamese deep forest [26] was proposed. This method defines the class distributions in a deep forest as the weighted sum of the tree class probabilities such that the weights are determined to reduce the distances between similar pairs of images and increase them between dissimilar points.

The lightweight multilayered random forest (LMRF) model [24] consists of a layer-to-layer RF. Each neuron of a DNN layer is replaced with an RF, and each layer consists of several types of RFs. Each layer consists of randomly generated heterogeneous RFs instead of uniform RFs to encourage diversity and maintain the generality, similar to the method used by gcForest [7]. In this study, a model was designed that uses only the output features of the previous layer as the new input features of the next layer without combining the transformed feature vector. As a replacement for deeper and wider networks, the LMRF model is applied to an embedded system in low-power and low-memory in-vehicle systems for the monitoring of driver emotions.

The deep random ferns (d-RFern) model [9] connects extremely randomized ferns to multiple layers to allow a high classification performance and a lightweight and fast structure. The input vector is first encoded as a transformed feature vector in the feature encoder layer and is then input to the cascade layers. The feature encoding process is similar to the DNN convolution and helps improve the performance of the final output layer. The cascade layer adjusts the number of ferns and layers required for the d-RFern adaptively, using only a small amount of data.

Additional approaches exist in which convolutional neural networks (CNNs) and decision trees [27]–[29] are combined to integrate the DNN architecture with a supervised forest feature detector. However, these differ from ensemble-based approaches that use ensemble trees as a layer-by-layer connection without the use of backpropagation during learning.

Although a deep ensemble classifier based deep model achieves a good performance similar to that of a DNN, one

RF must consist of a few hundred trees, and several RFs must form a single layer. FTDRF, however, consists of only two layers consisting of 2,000 decision trees per layer without the use of several RFs. However, this method also has a disadvantage in that the operation speed is slow owing to an excessive number of trees. In addition, they must be connected to multiple layers and have a similar length and parameter numbers similar to those of a DNN.

Among the numerous deep ensemble models available, in this study, the proposed rule elimination algorithm is applied to the LMRF, which is applicable to a real-time system because the numbers of RF neurons and layers are smaller than those of the other methods. This interpreted and simplified LMRF (iLMRF) is applied to various databases to prove that the performance is maintained even when the number of rules is drastically reduced.

## 3. Interpreting Deep Random Forest

Simplification of iLMRF is achieved through an elimination of weak rules based on an analysis of the feature contributions. The primary contribution of this study is to make the iLMRF interpretable/simple by creating a new contribution metric for interpreting the classifiers based on the feature contribution and frequency. This process is conducted from the second cascading layer except for the first feature encoding layer in the network, as shown in Fig. 1 (a). We demonstrate herein how the decision making processes of iLMRF consisting of a black box structure can be made explicable through two processes, namely, an estimation of the feature contribution and an elimination of unimportant rules.

### 3.1. **Growth phase**: Training of deep RF

As the first step, a non-NN style deep model, LMRF, based on an ensemble of RFs is trained. The LMRF consists of multiple layers $L^l(l \in \{1, \dots, N\})$ of RF $F_v^l(v \in \{1, \dots, V\})$, as depicted in Fig. 1 (a), where each $F_v^l$ consists of numerous decision trees $t$, and a $t$-th decision tree in a $F_v^l$ at layer $l$ is denoted as $dt_{v,t}^l$. In the first layer, the input vector is encoded as a transformed feature vector $\Phi^l$ by combining the output of an individual RF, $\Phi^1 = [\, P(\Phi_1^1|F_1^1), P(\Phi_2^1|F_2^1), \dots, P(\Phi_V^1|F_V^1)\,]$. From the second layer, each layer $L^l\ (l > 1)$ is trained using the encoded feature vector of layer $l-1$, and is also used to generate a new feature vector $\Phi^l$ for the next layer or to predict the final class at the final layer. With the LMRF, each neuron of a DNN layer is replaced with the RF, and each layer consists of several types of RFs. Each layer consists of randomly generated heterogeneous RFs instead of uniform RFs to encourage diversity and maintain the generality [24]. To determine whether to expand a layer, LMRF uses a K-fold validation to automatically determine the numbers of layers and parameters while reducing the risk of an overfitting. When the LMRF is converged through a K-fold cross-validation, the final class probability is determined by averaging the class probabilities predicted from each RF $[\, P(\Phi_1^N|F_1^N), \dots, P(\Phi_V^N|F_V^N)\,]$, and predicting the final class label with the highest probability.

### 3.2. **Sequential covering based on rule contribution**

Sequential covering is a common rule induction procedure that iteratively learns a single rule individually to create a decision set that includes the entire dataset [30]. After a densely coupled black box LMRF model is constructed, the rules of an individual RF should be iteratively saved in a decision set based on a sequential covering procedure.

The basic unit of an LMRF, i.e., a decision tree, is regarded as a rule-based model because the decision procedures that determine the final value depend on **if-then** conditions represented by the trained node. Each path from the root of the tree to a leaf is a rule that classifies a set of examples. When an instance $\mathrm{X}_n$ and its label $\mathrm{Y}_n$ (part of dataset $\mathcal{D} = \{(\mathrm{X}_1, \mathrm{Y}_1), \dots, (\mathrm{X}_\mathrm{N}, \mathrm{Y}_\mathrm{N})\}$) falls into a root node, $\mathrm{X}_n$ will be passed to a right or left child node that satisfies the split function with a threshold for a specific feature determined during the training step. These steps are repeated until the given data reach a leaf node that creates an optimal feature space. The node consists of pairs of specific feature indexes, a split function with a threshold, and a class distribution, and the chain of overall nodes (decision path or rule) per decision tree is stored in the decision set.

In this study, we modified the sequential covering algorithm to select the optimal rules from each tree and RF. In the classification problem, the feature contribution (importance) represents changes in the feature-specific distribution when instances are split up for a particular feature. To calculate the feature contribution, a decision tree traverses downward until it reaches a leaf. At every specific split, the feature contribution of the feature variable that determines the split is defined as the difference in class probability between a parent and child node. To obtain the final rule contribution, we follow the path from the root node to the leaf node of the data instance and sumall feature contributions of each node. This algorithm extracts the paths (rules) sequentially by looking for the best rule that has a high contribution score.

The rule contribution for the *i*-th rule $\mathrm{r}_i^t$ consisting of P depth on a *t*-th tree is then calculated as follows:

$$\mathrm{r}_i^t = \frac{\sum_{j \in P} feat.contrib(i,j)}{\sum \#class\ of\ t\text{-}th\ tree}\ , \qquad (1)$$

where the feature contribution of the *j*-th node $feat.contrib(i,j)$ of the *i*-th rule is calculated using the difference in class probability $\mathbf{Pr} = \{pr_1 \dots pr_{\#class}\}$ between a parent (*j-1*) and child (*j*) node.

$$feat.contrib(i,j) = \left(\mathbf{Pr}_{j-1} - \mathbf{Pr}_j\right) \qquad (2)$$

These rule contributions can then be normalized to a

value of between zero and one by dividing by the sum of all rule contributions of the *t*-th tree. A large positive or negative value of $r_i$ means that a rule consisting of several features contributes strongly to the decision class. By contrast, a small positive or negative value of $r_i$ means that a rule contributes weakly to the decision class. Values of zero in a contribution means that the feature does not contribute to the decision-making process.

A pre-mining of the feature pattern inspired by [19] is also used for the weight of the rule contribution. A feature pattern is the frequent occurrence of feature values (e.g., x = A). We extract frequently occurring feature patterns from all rules in a decision set $\mathrm{dSet}(v, l)$ of the *v*-th RF and *l*-th layer. The frequency of a feature pattern is measured based on its support in the decision set:

$$fre_v^l(x_j = A) = \frac{1}{|\mathrm{dSet}(v,l)|}\sum_{i\epsilon \mathrm{dSet}(v,l)} I(x_j^{(k)} = A) \quad (3)$$

where $|\mathrm{dSet}|$ is the cardinality of features in $\mathrm{dSet}(v, l)$, $fre(x_j = A)$ quantifies the frequency of feature patterns in the rules of $\mathrm{dSet}(v, l)$, and $I$ is an indicator function that returns a value of 1 if the feature $x_j$ of the instance $k$ is of level $A$; otherwise, a value of zero is returned. The feature pattern is a normalization of the number of overlapping features among all features included in $\mathrm{dSet}(v, l)$.

At the RF level, the final rule contribution $r^*{}_i$ in a decision set $\mathrm{dSet}(v, l)$ is estimated through a weighted combination of the feature contribution and feature pattern.

$$r^*{}_i = r_i \cdot \sum_{j\epsilon \mathbf{r_i}} fre_v^l(x_j) \quad (4)$$

In the equation, if the feature patterns included in each rule $\mathbf{r_i}$ have high frequencies, the final rule contribution $r^*{}_i$ increases in proportion to the feature contribution. The rules $\mathrm{indSet}(v, l)$ for each RF, $F_v^l$, are sorted and rearranged in ascending order according to the final rule contribution.

Table 1. One example of decision set reordering. The first three rules are extracted from the *v*-th RF and are rearranged based on the final contribution. Each rule has a pair of contributions and probabilities of a class.

| |
|---|
| **Initial Rules of dSet$(v, l)$** |
| *Rule* 1: $(x_2 > 2.59)$ *and* $(x_2 > 4.75)$ *and* $(x_0 <= 6.04)$ *and* $(x_3 > 1.84) \Rightarrow \{0\}$ [0, 0, 1] |
| *Rule* 2: $(x_2 > 2.59)$ *and* $(x_2 > 4.75)$ *and* $(x_0 <= 6.04)$ *and* $(x_3 <= 1.84) \Rightarrow \{-0.277\}$ [0, 0.5, 0.5] |
| *Rule* 3: $(x_3 <= 1.75)$ *and* $(x_3 > 0.7)$ *and* $(x_3 <= 1.55) \Rightarrow \{0.55\}$ [0, 1, 0] |
| ..... |
| **Reordered Rules of dSet$(v, l)$** |
| *Rule* 1: $(x_3 > 0.7)$ *and* $(x_3 > 1.55)$ *and* $(x_2 <= 4.95) \Rightarrow \{0.83\}$ [0, 0.6, 0.4] |
| *Rule* 2: $(x_3 <= 1.75)$ *and* $(x_3 > 0.7)$ *and* $(x_3 <= 1.55) \Rightarrow \{0.55\}$ [0, 1, 0] |
| *Rule* 3: $(x_2 > 2.59)$ *and* $(x_2 > 4.75)$ *and* $(x_0 <= 6.04)$ *and* $(x_3 <= 1.84) \Rightarrow \{-0.277\}$ [0, 0.5, 0.5] |

This procedure can be iterated until we extract all rules that cover the RFs of the *l*-th layer. Table 1 shows the reordering of learned rules in a decision set. The initial rules consist of the feature rule of the IF clause and the class probability pairs of the THEN clause. However, through the proposed sequential covering process, the rules are reordered according to the final contribution of each rule.

### 3.3. **Rule elimination phase:** Simplifying LMDF

We employed a feature contribution with a feature pattern for a rule contribution to represent the correlation between a trained feature and changes in the class probability. This approach helps with understanding which features, rules, and RFs affect the prediction results of an LMRF. However, an LMRF generates a large number of rules because it is also composed of several black box RFs. Therefore, the decision making processes of an LMRF can be made explicable through the elimination process of unimportant rules.

Weak rules in $\mathrm{dSet}(v, l)$ are eliminated according to the given final rule contribution $r^*{}_i$ and only the rules with a high contribution value remain. Rules included in $\mathrm{List}_{\mathrm{dSet}}[l]$ can be removed at the same rate for each RF according to the user input, or it can be adjusted for each RF depending on the required accuracy.

Algorithm 1 shows the overall rule elimination procedures based on the feature contribution and patterns for constructing an interpretable iLMRF. After completing the training of the iLMRF, test data are input into the first feature encoder layer. The outputs of the first layer are concatenated, and these transformed feature vectors, augmented with the class vector generated by the first layer, are input into the list dSet ($\mathrm{List}_{\mathrm{dSet}}$) of the *l*-th layer until the data are mapped to the final layer. The final layer averages the probability values of each class and determines the class with the highest probability value as the final class.

**Algorithm 1:** Rule elimination based on feature contribution and feature pattern

**Input:** The number of layers $N$, the number of RFs $V$, the number of trees $T$, random forest $RF$, list of dSets $\mathrm{List}_{\mathrm{dSet}}$

Start with an empty list of dSets, $\mathrm{List}_{\mathrm{dSet}} = \emptyset$
Learn LMRF
**For** each *l* layer**:**
  **For** each *v* RF:
    **For** each *t* tree:
      -Split a *i*-th rule from a decision tree
      -Calculate feature contribution of a *i*-th rule $feat.contrib\,(i,*)$
      -Calculate rule contribution for *i*-th rule $r_i^t$
      -Add rule and its $r_i^t$ to $\mathrm{dSet}(v, l)$
    **End**
    -Compute feature pattern $fre_v^l(x_j = A)$ by splitting rules in $\mathrm{dSet}(v, l)$
    -Re-compute a new rule contribution $r^*{}_i$
    -Sort rules in $\mathrm{dSet}(v, l)$ according to $r^*{}_i$
    -Add $\mathrm{dSet}(v, l)$ to $\mathrm{List}_{\mathrm{dSet}}$of *l*-th layer
      $\mathrm{List}_{\mathrm{dSet}}[l] = \mathrm{List}_{\mathrm{dSet}}[l] + \mathrm{dSet}(v)$
  **End**
**End**
**Output:** The $\mathrm{List}_{\mathrm{dSet}}[l]$ consists of *l* layers

Table 2. Comparison of accuracy, number of rules, number of parameters (#Param.), and number of operations (#Op.) between DF models according to the rule ratios using the CK+ dataset

| Rule | Accuracy (%) | | | Rules (M) | | | # Param. (M) | | | # Op. (M) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ratio | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF |
| 1.0 | 93.60 | 89.71 | 92.41 | 0.12 | 0.16 | 0.13 | 0.53 | 2.90 | 2.51 | 0.0060 | 0.0381 | 0.0233 |
| 0.9 | 92.86 | 90.00 | 92.15 | 0.11 | 0.15 | 0.12 | 0.51 | 2.78 | 2.39 | 0.0060 | 0.0381 | 0.0232 |
| 0.8 | 92.50 | 89.92 | 92.24 | 0.09 | 0.13 | 0.10 | 0.47 | 2.59 | 2.22 | 0.0059 | 0.0380 | 0.0231 |
| 0.7 | 91.87 | 89.92 | 92.18 | 0.08 | 0.12 | 0.09 | 0.44 | 2.38 | 2.03 | 0.0059 | 0.0379 | 0.0230 |
| 0.6 | 91.05 | 89.73 | 92.04 | 0.07 | 0.10 | 0.08 | 0.39 | 2.16 | 1.83 | 0.0058 | 0.0377 | 0.0228 |

Table 3. Comparison of accuracy, number of rules, number of parameters (#Param.), and number of operations (#Op.) between DF models according to the rule ratios using the MNIST dataset

| Rule | Accuracy (%) | | | Rules (M) | | | # Param. (M) | | | # Op. (M) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ratio | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF |
| 1.0 | 97.98 | 98.77 | 98.57 | 0.08 | 0.94 | 0.26 | 2.12 | 26.42 | 7.17 | 0.0089 | 0.0852 | 0.0296 |
| 0.9 | 97.77 | 98.73 | 98.57 | 0.08 | 0.85 | 0.23 | 2.00 | 24.79 | 6.76 | 0.0088 | 0.0851 | 0.0294 |
| 0.8 | 97.41 | 98.74 | 98.57 | 0.07 | 0.76 | 0.21 | 1.86 | 22.98 | 6.28 | 0.0087 | 0.0850 | 0.0293 |
| 0.7 | 96.86 | 98.76 | 98.47 | 0.06 | 0.66 | 0.18 | 1.71 | 20.98 | 5.75 | 0.0087 | 0.0849 | 0.0292 |
| 0.6 | 96.00 | 98.75 | 98.39 | 0.05 | 0.57 | 0.16 | 1.54 | 18.86 | 5.19 | 0.0086 | 0.0850 | 0.0291 |

## 4. Experiments

In this section, we check the interpretability and simplification of the iLMRF model and compare the performance when the same rule elimination is applied to other deep RF- and DNN-based approaches. From the experiment, we prove that the compressed iLMRF maintains a similar performance, not only the original LMRF, but also DNN-based algorithms, although the iLMRF removes a significant percentage of the rules. To prove the coherence of the performance and examine the interpretability of the compressed iLMRF, we conducted a test using the following five datasets.

### 4.1. Databases

**CK+ dataset** [31]: The expanded Cohn–Kanade (CK+) dataset is a public benchmark dataset for facial expression recognition (FER) and has 327 image sequences from 118 subjects and seven facial expression labels based on FACS. The feature vector consists of 84 dimensional distance ratios and 88 dimensional angles that are extracted from the facial landmarks [32].

**MNIST dataset** [33]: The Modified National Institute of Standards and Technology (MNIST) dataset contains images of handwritten digits and is also widely used for an evaluation in the field of machine learning. The images were normalized to a 28 pixel × 28 pixel resolution with grayscale values. The MNIST dataset includes 60,000 training samples and 10,000 testing samples.

**IRIS dataset** [34]: The IRIS dataset includes three iris species (setosa, versicolor, and virginica) and four dimensional feature vectors (sepal length, sepal width, petal length, and petal width) and consists of 150 samples.

**WDBC dataset** [35]: The Wisconsin Diagnostic Breast Cancer (WDBC) dataset provides the diagnosis results of the Wisconsin University Hospital. It is composed of two category labels, malignant and benign, with 212 and 357 images, respectively. The feature vectors of the WDBC dataset consist of 32 variables, including the patient id, diagnosis, radius, texture, perimeter, area, smoothness, compactness, concavity, and symmetry.

**ORL dataset** [36]: The Orivetti Research Lab (ORL) dataset contains a set of facial images taken at AT&T Laboratories Cambridge. It offers 400 grayscale images with a pixel resolution of 64 × 64 captured from 40 distinct subjects. Some images were taken at different times, with varying lighting and facial emotions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses).

During the experiments, CK+, IRIS, and WDBC dataset use different type of feature vectors without the feature encoder layer (first layer). In other two datasets, image is inputted into a feature encoder layer for feature extraction.

### 4.2. Evaluation of deep RF models

One way to prove the interpretability of a model is to show its simplicity. Therefore, during this experiment, we first compared the numbers of rules, parameters, and operations used in the model, and the accuracy, while reducing the model size for the deep RF based methods.

To verify the effectiveness of the rule elimination scheme, we compared its performance with that of two representative deep RF based methods by varying the ratios of the rules from 1.0 to 0.6: (1) iLMRF, (2) gcForest [7], and (3) FTDRF [8]. Tables 2 and 3 demonstrate the performance according to the rule ratios using the CK+ and MNIST datasets, respectively. As we can see from Table 2, when iLMRF is trained using fully connected rules with CK+ facial landmark features, the accuracy is somewhat higher than that of gcForest (3.89%) and FTDRF (1.19%) despite using a slightly smaller number of trees (rules) and RFs. The numbers of parameters and operations of iLMRF are also 5.5- and 6.4-times lower than those of gcForest, and 1.9-and 3.9-times lower than those of FTDRF, respectively. However, when we reduced the rule ratio by 40%, the

accuracy of iLMRF decreased by 2.1% compared with the fully connected rules, although the relative accuracy is still higher than that of gcForest and FTDRF. Because iLMRF originally used fewer rules, the more rules that are removed, the lower the performance compared to the other methods. However, the required numbers of parameters and operations for a compressed iLMRF are 5.4- and 6.4-times smaller than those of gcForest, and 4.6- and 3.9-times smaller than those of FTDRF.

As shown in Table 3, the three algorithms used a feature encoder layer to transform the MNIST images into a new input vector. The original iLMRF using fully connected rules has a slightly lower accuracy than that of gcForest and FTDRF because it uses a smaller number of trees (rules) and RFs. For example, gcForest and FTDRF use 3.25- and 11.8-times more rules than iLMRF, respectively. In addition, iLMRF also has 12.5- and 9.6-times fewer parameters and operations than gcForest, and 3.4- and 3.3-times fewer parameters and operations than FTDRF. When we reduce the rule ratio by 40%, the accuracy of iLMRF is only 1.9% lower than that of gcForest and FTDRF. However, the number of rules learned by iLMRF is approximately 11-times lower than that of gcForest and 3-times lower than that of FTDRF. In addition, the numbers of parameters and operations of iLMRF are 12.3- and 9.7-times lower than those of gcForest, and 3.4- and 3.4-times lower than those of FTDRF, respectively. From the results, we can see that the proposed rule elimination effectively reduces the number of parameters and operations, and a finer gap in the accuracy of iLMRF may be sufficiently acceptable for a real-time embedded system.

To evaluate the performance of the algorithm on more diverse datasets, we conducted the same experiment on the IRIS, WDBC, and ORL datasets. As shown in Tables 4, 5, and 6, although iLMRF uses much fewer parameters and operations, it demonstrates a similar accuracy as gcForest and FTDRF. Based on the experiment results, we confirmed that iLMRF achieves an efficient rule compression among deep RF models in terms of both memory and the number of computations. Exceptionally, our approach has slightly less maintainability in terms of accuracy than gcForest and FTDRF according to the changing rule elimination ratio for the ORL dataset. The reason for this is that the original iLMRF consists of small networks with only a few core rules, although gcForest and FTDRF models have a higher rule redundancy in the network. Therefore, although the rules of the two comparison methods are reduced, most of the duplicated rules are removed, and thus the performance is not significantly reduced.

Overall, although the three methods commonly remove numerous rules compared to their original model, gcForest and FTDRF still contain larger rules from a minimum of 1.6- (gcForest of ORL) to a maximum of 7.5-times (gcForest of IRIS) those of iLMRF, although the accuracies remain similar.

Table 4. Comparison of accuracy and numbers of rules among three deep RF models according to changes in rule elimination using IRIS dataset

| Rule ratio | Accuracy (%) | | | # Rules (M) | | |
|---|---|---|---|---|---|---|
| | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF |
| 1.0 | 100.00 | 98.00 | 100.00 | 0.0046 | 0.0362 | 0.0137 |
| 0.9 | 100.00 | 98.00 | 100.00 | 0.0046 | 0.0362 | 0.0137 |
| 0.8 | 100.00 | 98.00 | 100.00 | 0.0046 | 0.0329 | 0.0136 |
| 0.7 | 98.00 | 98.00 | 100.00 | 0.0039 | 0.0287 | 0.0120 |
| 0.6 | 100.00 | 98.00 | 100.00 | 0.0033 | 0.0248 | 0.0096 |

Table 5. Comparison of accuracy and numbers of rules among three deep RF models according to changes in rule elimination using WDBC dataset

| Rule ratio | Accuracy (%) | | | # Rules (M) | | |
|---|---|---|---|---|---|---|
| | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF |
| 1.0 | 96.49 | 95.21 | 97.34 | 0.0063 | 0.0450 | 0.0256 |
| 0.9 | 96.49 | 95.21 | 97.34 | 0.0063 | 0.0450 | 0.0247 |
| 0.8 | 96.49 | 95.21 | 97.34 | 0.0057 | 0.0385 | 0.0227 |
| 0.7 | 96.49 | 95.74 | 97.34 | 0.0050 | 0.0350 | 0.0199 |
| 0.6 | 96.49 | 95.74 | 96.81 | 0.0044 | 0.0298 | 0.0168 |

Table 6. Comparison of accuracy and numbers of rules among three deep RF models according to changes in rule elimination using ORL dataset

| Rule ratio | Accuracy (%) | | | # Rules (M) | | |
|---|---|---|---|---|---|---|
| | iLMRF | gcForest | FTDRF | iLMRF | gcForest | FTDRF |
| 1.0 | 97.50 | 97.50 | 90.00 | 0.0595 | 0.0957 | 0.1133 |
| 0.9 | 97.50 | 97.50 | 90.00 | 0.0543 | 0.0893 | 0.1045 |
| 0.8 | 97.50 | 97.50 | 90.00 | 0.0481 | 0.0786 | 0.0925 |
| 0.7 | 87.50 | 97.50 | 90.00 | 0.0422 | 0.0702 | 0.0809 |
| 0.6 | 87.50 | 97.50 | 90.00 | 0.0362 | 0.0594 | 0.0696 |

### 4.3. Comparison with state-of-the-art methods

An additional experiment was conducted on the CK+ dataset to test whether the proposed algorithm effectively recognizes the facial expressions, and the performance was compared with other state-of-the-art-methods, namely, an AlexNets-based FER approach [37]; a 3D CNN-based FER approach with deformable facial action parts constrained (3DCNN-DAP) [38]; a DNN-based approach that uses multiple inception layers [39]; a 3D Inception-ResNet (3DIR) with LSTM for the FER [40]; a fast FER based on a hierarchical weighted RF (H-WRF) [27]; three DRF-based methods, i.e., gcForest [7], FTDRF [8], and LMRF [24]; and the proposed iLMRF. The deep RF based methods, gcForest, FTDRF, and iLMRF, exploited a feature vector consisting of an 84-dimensional distance ratio and an 88-dimensional angle ratio [27] without using the entire image.

As shown in Table 7, although 30% of the rules are removed through a rule elimination from the original iLMRF model, the resulting accuracy is only 1.3% less than that of the approaches described in [39] and [40]. However, the overall numbers of parameters and operations are significantly reduced compared to the two DNN-based algorithms. In the second comparison, we conducted a test on the MNIST dataset and compared the performance

Table 7. Comparison of accuracy (Acc.), numbers of parameters (# Param.), and numbers of operations (#Op.) with the state-of-the-art methods using the CK+ dataset

| Methods | Acc. (%) | # Param.(M) | # Op. (M) |
|---|---|---|---|
| AlexNets [37] | 92.2 | 62.3 | 720 |
| 3DCNN-DAP [38] | 92.4 | 70 | 174 |
| Multiple Inception [39] | 93.2 | 12.36 | 23.7 |
| 3DIR with LSTM [40] | 93.2 | 10.90 | 18.9 |
| H-WRF [27] | 92.6 | 0.25 | 0.0067 |
| gcForest [7] | 89.7 | 2.90 | 0.0381 |
| FTDRF[8] | 92.4 | 2.51 | 0.0233 |
| LMRF (1.0)[24] | 93.6 | 0.53 | 0.0060 |
| iLMRF (0.8) | 92.5 | 0.47 | 0.0059 |
| iLMRF (0.7) | 91.9 | 0.44 | 0.0059 |

Table 8. Comparison of accuracy (Acc.), numbers of parameters (# Param.), and numbers of operations (#Op.) with the state-of-the-art methods using the MNIST dataset

| Methods | Acc. (%) | # Param. (M) | # Op.(M) |
|---|---|---|---|
| ResNet-101[41] | 98.3 | 42 | 212 |
| ShuffleNetV2[42] | 97.0 | 1.3 | 2.7 |
| MobileNetV2[43] | 98.5 | 2.24 | 11.02 |
| MobileNetV3[44] | 98.6 | 1.66 | 16.59 |
| gcForest [7] | 98.8 | 26.42 | 0.085 |
| FTDRF[8] | 98.6 | 7.17 | 0.029 |
| iLMRF (1.0) | 98.0 | 2.12 | 0.0089 |
| iLMRF (0.8) | 97.4 | 1.86 | 0.0087 |
| iLMRF (0.6) | 96.0 | 1.54 | 0.0086 |

between the three state-of-the-art CNN-based methods and two deep RF methods, namely, ResNet-101 [41] and two CNN compression networks, shuffleNetV2 [42] and MovileNetV2 [43]; deep RF-based methods, i.e., gcForest [7], FTDRF [8], and LMRF [24]; and the proposed iLMRF.

Table 8 shows that the accuracy of the original iLMRF is similar to that of the state-of-the-art methods. When 40% of the rules of the iLMRF are removed, only a 2% decrease in the existing accuracy occurs. The reduced accuracy can be overcome when considering the effectiveness of the numbers of parameters and operations of the iLMRF when compared against other CNN-based compression algorithms [42] [43]. For example, in the case of MobileNetV3, the accuracy is 2.6% higher than that of iLMRF (0.6), although the number of operations of iLMRF is 1,929-times higher.

Through the two experiments, we know that the proposed method can derive outstanding compression performances in terms of the numbers of parameters and operations while maintaining the level of accuracy, and will be an opportunity to extend the range of iLMRF applications to low-end systems.

### 4.4. Feature interpretability

As another example indicating that the proposed iLMRF is interpretable, we graphically presented the contributions of the features used to classify the classes in the two RF nodes in the first layer. As shown in Fig. 2, when using the IRIS dataset for classification, *petal length* and *petal width* are indicated as important features in the first RF, whereas

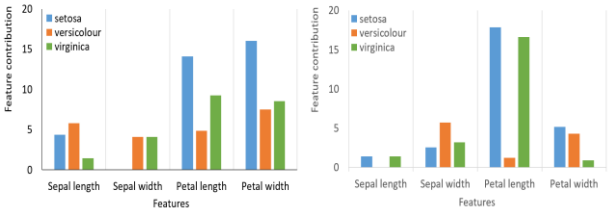


Figure 2. Feature contribution in classifying IRIS dataset using iLMRF: (a), (b) feature contributions of two RFs in the first layer.

only *petal length* is marked as an important feature in the second RF.

We also analyzed the interpretability of the feature contribution using the ORL dataset. The iLMRF redistributes the class prediction backwards using the local feature contribution until it assigns a relevance score to each input variable, similar to a heat map [14]. In Fig. 3, we can see which feature variables (pixels) are valuable for classifying objects from the input image. From the results, we can confirm that the feature contribution used for the rule elimination also provides a heat map for intuitively verifying the results along with the interpretation.

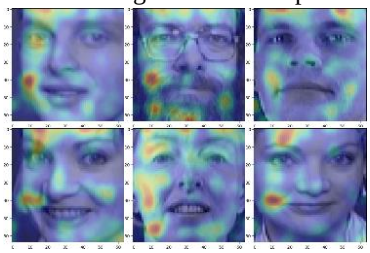

Figure 3. Visual attention heat map based on feature contributions in the ORL dataset. A heat map identifies pixels that are pivotal for the classification of an individual person.

## 5. Conclusion

In this paper, an interpretation and simplification method for a black box deep RF model using rule elimination based on the feature contribution and pattern was proposed. The model interpretation and simplification are achieved by analyzing the importance of the features on the sub-optimal space of each node from a fully trained iLMRF and eliminating the low contribution rules. Although DNN-based model compression methods should consider the trade-off regarding the number of parameters and the performance, the experimental results proved that the proposed method effectively reduces the number of rules, parameters, and operations without a decrease in performance. In addition, unlike a black box model, we can interpret which features contribute most to the decision making of the iLMRF before making the final decision through the rule elimination process. However, the proposed iLMRF interpretation method is not a fully white model because it still contains numerous rules and feature parameters. A future study will focus on the design of a fully interpretable model that is human understandable through a depth-wise analysis of the rules.

References


[1] G. Marcus. Deep Learning: A Critical Appraisal. *arXiv:1801.00631*, 2018.

[2] Y. Chen, N. Wang, and Z. Zhang. Darkrank: Accelerating deep metric learning via cross sample similarities transfer. In *AAAI*, 2018.

[3] G. E. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015.

[4] Y. Wang, C. Xu, C. Xu, C. Xu, and D. Tao. Learning Versatile Filters for Efficient Convolutional Neural Networks. In *NIPS*, 2018.

[5] F. Tung and G. Mori. CLIP-Q: Deep Network Compression Learning by In-Parallel Pruning-Quantization. In *CVPR*, 2018.

[6] S. J. Kim, S. Y. Kwak, and B. C. Ko. Fast Pedestrian Detection in Surveillance Video Based on Soft Target Training of Shallow Random Forest. *IEEE ACCESS*, 7:12415-12426, 2019.

[7] Z.-H. Zhou and J. Feng. Deep forest: towards an alternative to deep neural networks. In *IJCAI*, 2017.

[8] K. Miller, C. Hettinger, J. Humpherys, T. Jarvis, and D. Kartchner. Forward Thinking: Building Deep Random Forests. *arXiv:1705.07366*, 2017.

[9] S. Kim, M. Jeong, D. Lee, and B. C. Ko. Deep coupling of random ferns. In *CVPR Workshop*, 2019.

[10] G. David. Explainable Artificial Intelligence (xai). *DARPA*, 2017.

[11] G. Montavon, W. Samek, and K-R. Muller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1-15, 2018.

[12] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek. On Pixel-wise Explanations for Non-Linear Classifier Decisions by Layer-wise Relevance Propagation. *PLoS ONE*, 10(7):1-46, 2015.

[13] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K. R. Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[14] C. Anders, G. Montavon, W. Samek, and K. R. Müller. Understanding Patch-Based Learning of Video Data by Explaining Predictions. *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, Springer, pp. 297-309, 2019.

[15] S. Jiaxin, H. Zhang, and J. Li. Explainable and explicit visual reasoning over scene graphs. In *CVPR*, 2019.

[16] O. Biran and C. Cotton. Explanation and Justification in Machine Learning: A Survey. In *IJCAI Workshop*, 2017.

[17] Z. Si and S.-C. Zhu. Learning AND-OR templates for object recognition and detection. *IEEE TPAMI*, 35(9):2189-2205, 2013.

[18] S. Liu, S. Dissanayake, S. Patel, X. Dang, T. Mlsna, Y. Chem, and D. Wilkins. Learning accurate and interpretable models based on regularized random forests regression. *BMC Systems Biology*, 8:1-9, 2014.

[19] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan. Interpretable classifiers using rules and Bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350-1371, 2015.

[20] H. Lakkaraju, S. H. Bach, and J. Leskovec. Interpretable decision sets: a joint framework for description and prediction. In *ACM SIGKDD*, 2019

[21] H. Yang, C. Rudin, and M. Seltzer. Scalable Bayesian Rule Lists. In *ICML*, 2017.

[22] Z. Yang, A. Zhang, and A. Sudjianto. Enhancing Explainability of Neural Networks through Architecture Constraints. *arXiv:1901.03838*, 2019.

[23] R. Roscher, B. Bohn, M. F. Duarte, and J. Garcke. Explainable Machine Learning for Scientific Insights and Discoveries. *arXiv:1905.08883*, 2019.

[24] M. Jeong, M. Park, and B. C. Ko. Intelligent Driver Emotion Monitoring Based on Lightweight Multilayer Random Forests. In *INDIN*, 2019.

[25] F. Ji, Y. Yu, and Z.-H. Zhou. Multi-layered gradient boosting decision trees. In *NIPS*, 2018.

[26] L. V. Utkin and M. A. Ryabinin. A Siamese Deep Forest. *Knowledge-Based Systems*, 139:13-22, 2018.

[27] Y. Kong and T. Yu. A Deep Neural Network Model using Random Forest to Extract Feature Representation for Gene Expression Data Classification. *Scientific Reports*, 8(1):16477, 2018.

[28] Y. Ioannou, D. Robertson, D. Zikic, P. Kontschieder, J. Shotton, M. Brown and A. Criminisi. Decision Forests, convolutional networks and the models in-between. *arXiv:1603.01250*, 2016.

[29] N. Frosst and G. E. Hinton. Distilling a neural network into a soft decision tree. *arXiv:1711.09784*, 2017.

[30] C. Molnar. Interpretable Machine Learning. *Lean Publishing*, 2019.

[31] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *CVPR Workshop*, 2010.

[32] M. Jeong and B. C. Ko. Driver's Facial Expression Recognition in Real-Time for Safe Driving. *Sensors*, 18(2):1-18, 2018.

[33] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278-2324, 1998.

[34] D. Dua and C. Graff. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml.

[35] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1-18, 1998.

[36] F. Samaria and A. Harter. Parameterisation of a Stochastic Model for Human Face Identification. In *WACV*, 1994.

[37] A. Krizhenvsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

[38] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen. Deeply learning deformable facial action parts model for dynamic expression analysis. In *ACCV*, 2014.

[39] A. Mollahosseini, D. Chan, and M.H. Mahoor. Going deeper in facial expression recognition using deep neural networks. In *WACV*, 2016.

[40] B. Hasani and M.H. Mahoor. Facial Expression Recognition Using Enhanced Deep 3D Convolutional Neural Networks. In *CVPR Workshop*, 2017.

[41] K. He, X. Zhang, S. Ren and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[42] N. Ma, X. Zhang, H. T. Zheng, and J. Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *ECCV*, 2018.

[43] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen. Mobilenetv2: Invberted residuals and linear bottlenecks. In *CVPR*, 2018.
[44] A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, and Q. V. Le. Searching for mobilenetv3. In *ICCV*, 2019.