RGB-X Classification for Electronics Sorting

FNU Abhimanyu, Tejas Zodage, Umesh Thillaivasan, Xinyue Lai, Rahul Chakwate, Javier Santillan, Emma Oti, Ming Zhao, Ralph Boirum, Howie Choset, Matthew Travers

Abstract—Effectively disassembling and recovering materials from waste electrical and electronic equipment (WEEE) is a critical step in moving global supply chains from carbon-intensive, mined materials to recycled and renewable ones. Conventional recycling processes rely on shredding and sorting waste streams, but for WEEE, which is comprised of numerous dissimilar materials, we explore targeted disassembly of numerous objects for improved material recovery. Many WEEE objects lack a well-annotated dataset for training an image classifier robustly, especially in the X-ray domain. This work introduces RGB-X, a multi-modal image classification approach, that utilizes key features from external RGB images with those generated from X-ray images to accurately classify electronic objects. More specifically, this work develops Iterative Class Activation Mapping (iCAM), a novel network architecture that explicitly focuses on the finer-details in the multi-modal feature maps that are needed for accurate electronic object classification. In order to train a classifier, electronic objects lack large and well annotated X-ray datasets due to expense and need of expert guidance. To overcome this issue, we present a novel way of creating a synthetic dataset using domain randomization applied to the X-ray domain. The combined RGB-X approach gives us an accuracy of 98.6% on 10 generations of modern smartphones, which is greater than their individual accuracies of 89.1% (RGB) and 97.9% (X-ray) independently. We provide experimental results to corroborate our results.

I. INTRODUCTION

Daisy [1], [2] and Dave [3], [4], Apple’s existing disassembly robots, disassemble electronic devices and components to enable the recovery of precious materials like rare earth elements, steel, and tungsten. Diverse object disassembly involves hundreds of decisions throughout the process, and object classification is the first major decision in the pipeline. In practice, precisely classifying disassembling objects is challenging as different objects of similar product line often have subtle visible differences leading to incorrect classifications that can cause sub-optimal material recovery if material compositions differ between the actual and predicted class. Conventional machine vision has to be highly accurate for recycling, however to classify all desired WEEE objects for disassembly using one line also requires that the system be capable of handling high variations seen in each class of multi-class WEEE such as cracks, bends, missing or replaced components, or other deformations that are difficult to predict and that inhibit conventional classification methods (e.g., RGB-based methods). These variations eliminate methods like weighing when numerous classes differ by only a few grams. Thus, this paper addresses challenges in classifying WEEE as they have subtle differences on the outside forcing a classifier to focus on extremely specific details on the outside or look inside for other distinguishing features.

In this work, we present a novel attention based iterative strategy, Iterative Class Activation Mapping (iCAM), to localize and then extract key features of relevant components from inside and outside of the devices for classification. This effectively guides the models to focus on the appropriate features for better decision making.

Also, to leverage both outside and inside features of electronic objects, we propose RGB and X-ray classification modalities individually and then uniquely combine them for their classification. Though there has been previous work on multi-domain classification, to the best of our knowledge, this work is the first to combine RGB and X-ray domains to perform classification for recycling purposes. This combination helps find common key features among the internal and external parts of the objects, making the classification robust to conditions like lighting, or wear and tear of the device.

The single-mode (RGB mode or the X-ray mode), as well as the joint training phase, needs an annotated dataset for training. Unlike other areas of image classification, used electronics lack a well-annotated dataset for training an image classifier robustly, especially in the X-ray domain.
The creation of annotated X-ray data is tedious and requires expert guidance. Thus, in this work, we propose a data randomization pipeline to generate synthetic X-ray data for training. We create synthetic X-ray images by projecting 3D-Computed Tomography (CT) scans into a 2D space and randomizing the relevant parameters like orientation, intensity settings, noise and background.

The paper is organized as follows: Section II introduces the prior work reported for electronic objects’ image classification. Section III provides a detailed description on the dataset, iterative class activation mapping (iCAM) architecture and the training strategy. Section IV describes the experimental results on RGB-X classification. Section V discusses the conclusion and the future work.

II. PRIOR WORK

This section discusses the existing WEEE classification methods. It also discusses the prior work regarding the dataset generation process, fine-grained image classification and multi-modal classification.

A. Classification for WEEE sorting

WEEE is an existing problem where both classical techniques as well as deep learning architectures have been used recognize objects. UNU-KEYS [5] is a classification method that is being used to classify WEEE by using attributes such as average weight, material compositions and end-of-life characteristics. This works well for coarse level classification, but does not work well for similar looking devices which have subtle feature differences. To overcome that, state of the art deep learning methods have also being used to accurately classify WEEE items. Standard network architectures like Faster Squeeze-Net [6] and YOLOv3 [7] were used to sort electronics. The Swedish company, Refind, in collaboration with the Danish Institute of Technology has also demonstrated the feasibility of sorting different types of printed circuit boards, mixed electronic scrap, and batteries by capturing color images and applying a deep learning architecture [8], [9]. Use cases include applying deep learning to sort different types of coin cells [10], and additional imaging inputs such as thermal and X-ray imaging are being explored for WEEE recognition [11], [12]. While the prevalence of deep learning has allowed for advances in classification, these methods are still primarily used for component level classification and they use a single mode for classification, therefore making models sensitive to real world scenarios like lighting, noise, and deformations.

B. Domain Randomization

One of the obstacles of using a machine learning algorithm is the lack of readily available, large, annotated datasets. The machine learning community has developed the concept of domain randomization (DR) to synthetically generate large datasets instead of collecting and labeling data manually. DR aims to add large variability to the synthetic data generation process, attempting to capture the real-world variations.

Svetozar and Jianhong [13] show a useful analysis of important parameters to generate synthetic data for classification for RGB images. DR has also been used in non-classification tasks. Tobin et. al. [14] used DR to synthetically generate images of geometrical shapes for an autonomous robotic picking task. Additional applications include autonomous drone flight [15], object detection [16], viewpoint estimation [17], and human pose estimation [18]. These works demonstrate the application of DR in the RGB domain, but they do not explore their use in other domains like X-rays.

C. Fine-grained image classification

Fine-grained image classification captures discriminative features of fine-grained objects and uses those features in image classification. In this paradigm, a localization sub-network is designed for locating the key regions of interest followed by a classification network used for recognition. Previous works on this paradigm depended on adding dense part annotations for accurate part localization and then using them for image recognition [19], [20], [21]. Recent works require only image labels for accurate part localization for training the recognition sub-network [22], [23], [24]. Finally, additional techniques such as attention mechanisms [25] and multi-stage strategies [26] perform the joint training of the integrated localization-classification sub-networks.

D. Multi-domain classification

Multi-domain learning aims to improve the classification performance in general domains by making full use of the information in each domain. Prior work has focused on extracting hand-crafted features such as scale-invariant
Fig. 3: Classification network architecture with SP1 as the input image $I_0$. For iteration $i$, an image $I_i$ is passed as an input to the iCAM module. Internally, the iCAM module extracts feature maps ($FM_i$) of image $I_i$ and extracts the region of activation (i.e., the region most important for classification) using the CAM module. Based on the importance regions, an image $I_{i+1}$ is cropped from $I_i$, which is used for the iteration $i + 1$. For the final prediction, we aggregate the feature maps in the feature aggregator ($FA$) and pass it through a classifier block to generate a prediction. Subimages (a), (b) and (c) shows the heat map, the most important region of the image given the heat map, and the cropped image from the heat map.

feature transform (SIFT) from the multi-domain (RGB-D) image to get a combined image descriptor [27]. Blum et al. proposed an RGB-D descriptor that relies on a k-means-based feature learning approach [28]. However, the hand-crafted features are often dataset-specific and require a strong understanding of domain-specific knowledge [29]. To reduce the dependency on hand-crafted features, machine learning techniques are explored, and one of the most common techniques is using network parameter sharing [30].

III. APPROACH

This section discusses the X-ray dataset generation process, the network architecture used for classification, and the joint training strategy used for combining the X-ray mode with the existing RGB mode for a combined RGB-X classification. In both these domains, we conduct experiments on 10 classes of modern smart phones (SP) classes.

A. Datasets

The 10 classes of modern smart phones will be referred to as SP1, SP2, SP3, SP4, SP5, SP6, SP7, SP8, SP9, SP10 in this work. For the RGB domain, the test dataset consists of 799 real RGB images. RGB training set includes a collection of synthetic and real images. The training dataset consists of a combination of 1000 synthetic and 250 real images per class. We collect RGB images using a Microsoft Azure Kinect camera mounted over a conveyor belt. We load the mesh of the phones in Blender [31] and use DR to create synthetic dataset by varying color, texture, and camera intrinsic and extrinsic parameters as suggested in [14].

For the entire X-ray domain, the test dataset consists of 749 real X-ray images and around $\approx$25000 synthetic X-ray images for training. Only synthetic X-ray images are used for training to demonstrate the efficacy of domain randomization and because of the lack of enough real X-ray present for each class. Test images are collected using an X-ray system equipped with 300kV micro focus and 180kV nano focus tubes, and a cesium iodide (CsI) detector. Both tubes are used to generate 2D and 3D images with resolutions ranging from 57.5 $\mu$m to 100 $\mu$m. The 2D and 3D images taken with the 300 kV tube use a peak energy of 220 kV with a current ranging from 200 to 300 $\mu$A, while images acquired with the 180 kV tube use 160 kV and 280 $\mu$A. For training, we utilize DR in order to generate a large scaled labelled dataset. To generate X-ray training data, we initialize generate 3D voxel models of 2-3 devices per class using off the shelf CT machines. CT scans consist of a scalar intensity assigned to each voxel based on the structure of the device. The voxel intensities are mapped to an opacity value and projected onto a plane using a simulated projector model to render X-ray images. The voxel intensity to voxel opacity mapping is known as a transfer function. A variety of X-ray machine settings can be simulated by the shape of the transfer function as shown in Fig 4. We use VTK library [32] to randomize the transfer function, and the intrinsic and extrinsic parameters of simulated projectors. Randomizing the transfer function arbitrarily can undesirably produce completely saturated images by assigning the same value to each pixel. To avoid this, we seed the randomization with 5 manually selected transfer function parameters. These manual seeds were selected using the visualization tool 3D Slicer [33]. We add randomness to these parameters and only select rendered images that have more than 100 SuperPoint [34] keypoints. Sample training
and testing images from the RGB and X-ray dataset are shown in Fig 2.

B. Network architecture and training strategy

This section describes the fine-grained classification network used for training the RGB and the X-ray mode, and the training strategy used in the single training and joint training phases. Let RGB image \((I_R)\) or X-ray image \((I_X)\) be of dimension \(H \times W \times C\) in the labeled dataset available for training the classification model, and \(y_i\) be the image label in one-hot encoding - i.e., \(y_i \in \mathbb{R}^m\) is a vector of dimensionality \(m\) (the number of classes) with \(y_i^{j=1} = 1\) and \(y_i^{j=0} = 0\) for the position \(j\) denoting the image label. We train the model using a three-stage approach: first training the 2 stream networks individually, followed by a joint fine-tuning stage.

1) Iterative class activation mapping (iCAM): As shown in Fig 3, our network has 4 modules: feature extractor module \((E)\), class activation mapping module \((CAM)\), feature aggregator module \((FA)\), and classifier module \((C)\). The feature extractor module \((E)\) and the \(CAM\) module are stacked together in an iterative manner followed by the feature aggregator module and classifier module.

In our method, we use class activation mapping (CAM) [35] using the global average pooling (GAP) to indicate the important image regions identified by CNNs to identify the respective classes. For a given image \(I \in \mathbb{R}^{H \times W \times C}\), let \(FM(x, y) = \sum_{i=1}^{k} f_{m_i}(x, y)\) represent the activation of the last convolutional layer at spatial location \((x, y)\) with \(k\) channels. \(\theta_C^c\) denotes the weight corresponding to class \(c\) in the classifier module \(C\). We define \(m_c\) as the class activation map for class \(c\) as:

\[
m_c(x, y) = \sum_{j=1}^{k} \theta_C^c f_{m_j}(x, y).
\]

Thus, \(m_c(x, y)\) is an unnormalized probability distribution directly indicating the importance of the activation at spatial location \((x, y)\). More details on CAM can be found in [35].

In our experiments, InceptionNet-v3 [36] up to the Mixed7C layer is chosen as the feature extraction module with \(\theta_E\) as a learnable parameter. The feature map \((FM_i)\) and the class activation map \((m_{c,i})\), for every iteration \(i\), is calculated using Eq.1. The activation map is upsampled to the original image size \(H \times W \times C\), and then a square region around the most dominant pixel of the map is selected resulting in image \(I_{i+1}\). The cropped image is then upsampled to the original image dimension \(H \times W \times C\) using bilinear upsampling function in Pytorch. After running the feature extraction and the CAM module for a pre-defined number of iterations \(n\), \(F_{0:n}\) is passed to the feature aggregator module followed by the classifier module to get the final classification score \(S\):

\[
FM(x, y) = \sum_{i=1}^{n} \theta_{FA} FM_i(x, y),
\]

\[
S = \text{softmax}(f_{\theta_C}(FM)),
\]

where \(FM\) is the aggregated feature map, and \(\theta_{FA}\) and \(\theta_C\) are the learnable parameters of the feature aggregator and the classifier module. For our experiments, we choose \(n=3\).

2) Training strategy: We proceed by training the iCAM network to minimize the negative log likelihood, \(L\), of the training data. While training we solve for:

\[
\theta_{E}, \theta_{FA}, \theta_C = \argmin_{\theta_E, \theta_{FA}, \theta_C} \sum_{i=1}^{N} L(i_{ICAM}(I, y_i; \theta_E, \theta_{FA}, \theta_C), y_i), \tag{3}
\]

We individually train the \(iCAM_R\) and \(iCAM_X\) for \(I_R\) and \(I_X\) images on RGB and X-ray datasets. For both the streams, the \(\theta_E\) is initialized with InceptionNet-v3 weight pretrained on ImageNet and were trained for 40 epochs each. The implementation is highly parallelized and performs full-batch gradient descent using the Stochastic Gradient Descent [37] optimizer in the Pytorch Autograd library [38], with a batch size of 16 with a learning rate of 0.001. At the end of this individual training stage, \(iCAM_R\) and \(iCAM_X\) have different sets of weights which are later fused in the joint training stage.

Once both domains are trained individually, we use the parameter sharing method to combine the information of both the domains as shown in Fig 1. During the joint-training, the softmax activation is discarded and the output from the classification layers are concatenated. Their individual responses \(FM_R\) from \(iCAM_R\) and \(FM_X\) from \(iCAM_X\), are fused and fed through an additional stream \(FS([g_R, g_X])\) with parameters \(\theta_{FS}\). This fusion network again ends in a softmax classification layer. During the joint-training phase, \(\theta_{E,R}, \theta_{FA,R}, \theta_{E,X}\) and \(\theta_{FA,X}\) are frozen and only \(\theta_{C,R}\), \(\theta_{C,X}\) and \(\theta_F\) are open to training. In this stage of training, we solve:

\[
\theta_{C,R}, \theta_{C,X}, \theta_{FS} = \argmin_{\theta_{C,R}, \theta_{C,X}, \theta_{FS}} \sum_{i=1}^{N} L(FS([g_R(I_R), g_X(I_X)]), y_i). \tag{4}
\]

During the joint-training phase, only the weights of the fusion layer and the weights of individual classification layers are optimized, keeping all other weights from individual mode intact.

IV. ANALYSIS AND RESULTS

A. Domain randomization

In this set of experiments, we evaluate the effect of domain randomization parameters on test accuracy. Our current implementation randomizes image white noise, background color, transfer function parameters, and rendering projector position and orientation. As shown in Table I in the first experiment, we introduce noise of increasing variance in the rendered image. We observe that the best accuracy of 98.1\% is obtained when \(\sigma = 1000\) with varying background, transfer function, and projector position and orientation. Also, we see a drop in the accuracy with the drop in the noise variance. The presence of noise improves convergence...
and makes training less susceptible to local minima as shown in [39]. We also observe that fixing the projector position and orientation affects accuracy the most, lowering it down to only 31.5%. This result emphasizes the use of DR as this projector position simulates different views of a 3D object, which cannot be simulated by simple 2D data augmentation. Another important factor is the transfer function, which, as discussed in Sec. III, simulates different X-ray machines and X-ray machine settings. Varying the transfer function, generalizes our data over different machine distributions. Unlike the RGB domain, we avoid pattern randomization since such variations do not exist in our test data.

### B. iCAM

iCAM is a fine grained classification method. For the sake of result section, we show the performance of iCAM on the modern smart phones (SP) classes mentioned in Section III and also CUB-200 [40] dataset. For comparisons, we evaluate the results with baseline classification networks like VGG-19 [41], ResNet-101 [42], ResNeXt-101 [43] and InceptionNet-v3 [44]. We also compare it with the SqueezeExcitation [45] with ResNet101 as the base module. All these experiments are performed on the X-ray dataset. We use average accuracy, and inference time as the metrics to compare the results for different networks as suggested in [46]. We also compare average precision and average recall of all these network architecture to our method as suggested in [47]. These results are presented in Table II. Additionally, we compare the iCAM+InceptionNet-v3 to these baseline classification networks on the CUB-200 dataset [40], a standard fine-grained classification dataset used for benchmarking image classification algorithms.

Table II shows that the iCAM block when combined with the InceptionNet-v3 base module outperforms the baseline image classification networks. The high accuracy value is attributed to the hierarchical approach iCAM takes in order to extract and combine key features from an image. Intuitively this approach is similar to looking and extracting key features at different levels of magnification and then using all the features for classifying that image. For further analysis, we calculate the heat-map using CAM as seen in Fig 6, and use them to visualize the CNN’s area of interest for similar looking classes (e.g., SP2 vs SP3). It is seen that iCAM block indeed helps in localizing key features in an X-ray image. Additionally, Table II presents that iCAM+InceptionNet-v3 has an inference time of ≈30fps, which allows the model to run in real-time with a 30fps industrial camera. Table III, shows that when iCAM is combined with InceptionNet-v3, the network outperforms InceptionNet-v3 by 10.5%. We also compare the iCAM block with the ResNet-101 feature extractor, and the result outperforms ResNet-101 [42], and other networks like ResNeXt-101 [43], SE-ResNet-101 [45], and VGG-19 [41]. This shows the efficacy of iCAM block with other extraction modules.

We also conduct a study on the number of iterations the iCAM module is run during the training time. Table IV shows the accuracy, the time taken to train 40 epochs, and the inference time for $n = 2, 3, 5$. As shown in the table, $n = 3$ shows a considerable increase in accuracy from using $n = 2$ for far less training and inference time than $n = 5$.

A comparative study is done to select the best feature aggregator strategy where 3 different feature aggregator

### TABLE I: Effect of DR parameters on accuracy. Current DR has non-fixed background, non-fixed transfer function, non-fixed simulated projector position and orientation and $\sigma = 1000$ as the variance for the white noise.

| Parameters                        | Status   | Accuracy |
|-----------------------------------|----------|----------|
| Current DR                        | Sec. III | 0.981    |
| Noise                             | $\sigma=10$ | 0.954    |
|                                   | $\sigma=100$ | 0.925    |
| Background                        | Fixed    | 0.922    |
| Transfer function                 | Fixed    | 0.655    |
| Simulated projector position and orientation | Fixed    | 0.315    |

### TABLE II: Comparison of baseline classification networks to the iCAM network on the X-ray dataset. Learning rate: 0.001, Batch Size: 16, Weight decay: 0.1 every 7 epochs, Epochs: 40. The bolded row highlights the network architecture with the best average accuracy.

| Network Architecture | Average accuracy | Precision | Recall | Infer time (ms) |
|----------------------|------------------|-----------|--------|----------------|
| InceptionNet-v3      | 0.933            | 0.907     | 0.935  | 12.2           |
| ResNet-101           | 0.953            | 0.899     | 0.956  | 11.1           |
| VGG-19               | 0.956            | 0.898     | 0.961  | 7.53           |
| ResNeXt-101 (64x4d)  | 0.929            | 0.869     | 0.948  | 20.15          |
| SE-ResNet-101        | 0.891            | 0.841     | 0.926  | 14.01          |
| iCAM+InceptionNet-v3 | 0.979            | 0.942     | 0.969  | 30.86          |

### TABLE III: Comparison of baseline classification networks to the iCAM network on the CUB-200 [40] dataset. Same parameters as Table II. The bolded row highlights the network architecture with the best average accuracy for InceptionNet-v3 and ResNet-101 feature extractor.

| Network Architecture | Accuracy | Precision | Recall |
|----------------------|----------|-----------|--------|
| InceptionNet-v3      | 0.692    | 0.697     | 0.693  |
| iCAM+InceptionNet-v3 | 0.797    | 0.791     | 0.801  |
| ResNet-101           | 0.814    | 0.818     | 0.815  |
| ResNeXt-101 (64x4d)  | 0.829    | 0.835     | 0.830  |
| SE-ResNet-101        | 0.818    | 0.826     | 0.820  |
| iCAM+ResNet-101      | 0.834    | 0.831     | 0.838  |
| VGG-19               | 0.778    | 0.781     | 0.779  |

### TABLE IV: Comparison of accuracy, training time and inference time of the iCAM+InceptionNet-v3 network for different values of iteration N. Training was done for 40 epochs. The bolded row highlights the current n chosen for all the experiments.

| No. of CAM iteration (N) | Average accuracy | Training time (mins) | Inference-time(ms) |
|--------------------------|------------------|----------------------|-------------------|
| n=2                      | 0.962            | 325                  | 20.11             |
| n=3                      | 0.979            | 490                  | 30.86             |
| n=5                      | 0.981            | 723                  | 50.96             |
strategies are tested. The most common strategies used are mean, sum, and weighted mean of the available feature maps. All of these experiments are done for $n = 3$. We achieve an accuracy of 0.941 for the mean, 0.768 for the sum, and 0.979 for the weighted mean version of the feature aggregator. Based on this study, we conclude that weighted mean is the best feature aggregating strategy because it weighs different levels of magnification differently.

C. RGB-X classification

We evaluate the multi-modal classification approach with both the single-mode iCAM+InceptionNet-v3 architectures. For this experiment, an RGB image of a particular class is paired with a random X-ray image sampled from the same class. This is done because of the lack of a streamlined RGB + X-ray sensor setup. While pairing the images, we randomly re-orient the X-ray images to prevent any bias induced due to similar orientation of the RGB and X-ray image. The total training time for RGB-X mode is 520 mins as the RGB and X-ray modes are trained in parallel. Table V shows the average accuracy, precision, recall and the inference time of each of the mode (RGB and X-ray) as well as the combined RGB-X mode. In context of a recycling plant that recycles around 200 phones per hour [2], the 0.6 percentage gain in accuracy from RGB-X ($n = 3$) over X-ray ($n = 5$) is critical as it saves 12 phones per hour. Although this adds $11\text{ms}$ to the inference time, the total training time is lowered to 520 mins (RGB-X) from 723 mins (X-ray). The lower training time is crucial while expanding to newer products without compromising on the recycling rate of 200 phones per hour.

As mentioned in Sec.III-B.2, for joint training we combine the deepest layers and train for them, by keeping the higher CNN layers frozen. This fusion approach is termed as “fast fusion” [48]. We evaluate this approach with the “slow fusion” approach also mentioned in [48]. For slow fusion, the network architecture is similar to fast fusion, except for the fact that the weights of the higher CNN layers are also trainable. This allows the higher layers to have access to more global information.

Table VI, shows the comparisons of 2 different fusion techniques, demonstrating that fast fusion performs better than the slow fusion, as fast fusion doesn’t affect the lower level activation learned by the iCAM layers.

| Mode       | Average accuracy | Average precision | Average recall |
|------------|------------------|-------------------|----------------|
| Slow Fusion| 0.952            | 0.951             | 0.958          |
| Fast Fusion| 0.987            | 0.987             | 0.995          |

TABLE V: Comparison of single (RGB, X-ray) mode accuracy to the multi-modal (RGB-X mode) accuracy. The bolded row highlights the mode with the best average accuracy.

| Mode       | Average accuracy | Average precision | Average recall |
|------------|------------------|-------------------|----------------|
| Slow Fusion| 0.952            | 0.951             | 0.958          |
| Fast Fusion| 0.987            | 0.987             | 0.995          |

TABLE VI: Comparison of different fusion techniques. The bolded row highlights the mode with the best average accuracy.

REFERENCES

[1] Apple. Apple adds earth day donations to trade-in and recycling program. [Online]. Avail-
[44] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Re-thinking the inception architecture for computer vision. 2015.” arXiv preprint arXiv:1512.00567, 2015.
[45] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.
[46] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, “Benchmark analysis of representative deep neural network architectures,” IEEE Access, vol. 6, p. 64270–64277, 2018. [Online]. Available: http://dx.doi.org/10.1109/ACCESS.2018.2877890
[47] M. Grandini, E. Bagli, and G. Visani, “Metrics for multi-class classification: an overview,” 2020.
[48] M. B. Shaikh and D. Chai, “Rgb-d data-based action recognition: A review,” Sensors, vol. 21, no. 12, p. 4246, 2021.
[49] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 618–626.