Advances in Artificial Intelligence and Machine Learning; Research 5 (1) 3344-3355 Received 7-12-2024; Accepted 6-03-2025; Published 13-03-2025

Outliers Resistant Image Classification by Anomaly Detection

Anton Sergeev avsergeev@hse.ru
National Research University
"Higher School of Economics",
101000 Moscow, Russia.

Victor Minchenkov vminchenkov@hse.ru
National Research University
"Higher School of Economics",
101000 Moscow, Russia.

Aleksei Soldatov av.soldatov@hse.ru
National Research University
"Higher School of Economics",
101000 Moscow, Russia.

Vasiliy Kakurin vkakurin@hse.ru
National Research University
"Higher School of Economics",
101000 Moscow, Russia.

Yaroslav Mazikov ymazikov@hse.ru
National Research University
"Higher School of Economics",
101000 Moscow, Russia.

Corresponding Author: Anton Sergeev

Abstract

The automatic monitoring of manual assembly processes in production settings increasingly relies on advanced technologies, including computer vision models. These models are designed to detect and classify events such as the presence of components in an assembly area and the connection of these components. However, a significant challenge for detection and classification algorithms is their vulnerability to variations in environmental conditions and their unpredictable behavior when encountering objects that are not present in the training dataset. Due to the impracticality of including all potential objects into the training sample, alternative solutions are needed. This study introduces a model that combines classification with anomaly detection by leveraging metric learning to create vector representations of images in a multidimensional space, followed by classification using cross-entropy. A dataset of over 327,000 images has been prepared for this purpose. Comprehensive experiments were conducted using various computer vision models, and the performance of each approach was systematically evaluated and compared.

Keywords: Computer vision, Classification, Anomaly detection.

1. INTRODUCTION

During the manual assembly process, a variety of objects are simultaneously located in the workspace, including parts, tools and partially assembled mechanisms. In order to automatically control this process, it is essential to solve the problems of detecting and classifying various events. In particular, it is required to identify all the details, tools, partially assembled mechanisms in the workplace, and distinguish between correctly and incorrectly performed assembly steps. For example, when a screw is connected to the rest of a mechanism, it is necessary to verify whether it was attached on the correct side. Such control helps to prevent manual assembly errors that occur due to the human factor such as inattention, fatigue or lack of experience of the employee. Additionally, besides verifying the accuracy of the assembly process, the implementation of safety regulations is being monitored.

2. PROBLEM STATEMENT

The system is designed to control the process of manual assembly of some mechanism. A camera is installed above the operator's work area and transmits video to the input of the neural network. There are various objects on the workspace, including details of a mechanism, partially assembled structures, tools, as well as foreign objects. The foreign objects could be, for example, parts of other mechanisms, gloves, garbage and etc.

To solve this problem, a two-step approach was proposed: the detection model is responsible for detecting objects (including extraneous ones) in the workspace, and then the objects are being classified.

When using state-of-the-art image detection and classification models such as YOLOv5 [1], there can be problems with the classification of objects that are not present in the training data. Such models with high confidence can classify an unknown foreign object to one of the predefined classes, which may cause a false positive in the control system and, as a result, miss a potential the violation of the assembly process. As it is possible to include all possible objects in the training dataset that may appear on the workspace in the future, a modification to the formulation of the classification problem is proposed.

In order to train a classification model that is resistant to various noises and foreign objects, it is proposed to simultaneously solve two tasks during the training phase: anomaly detection and classification.

3. DATASETS

To train the models, datasets containing images of mechanism details, tools, and assembly stages of various objects were collected, including photos of people, animals and etc. During training, all images from datasets that do not represent the assembled mechanism details were labeled as extraneous. The training set contains over 32k images, the validation set 48k, and the test set 29K. Examples of the images from the datasets are shown in FIGURE 1.

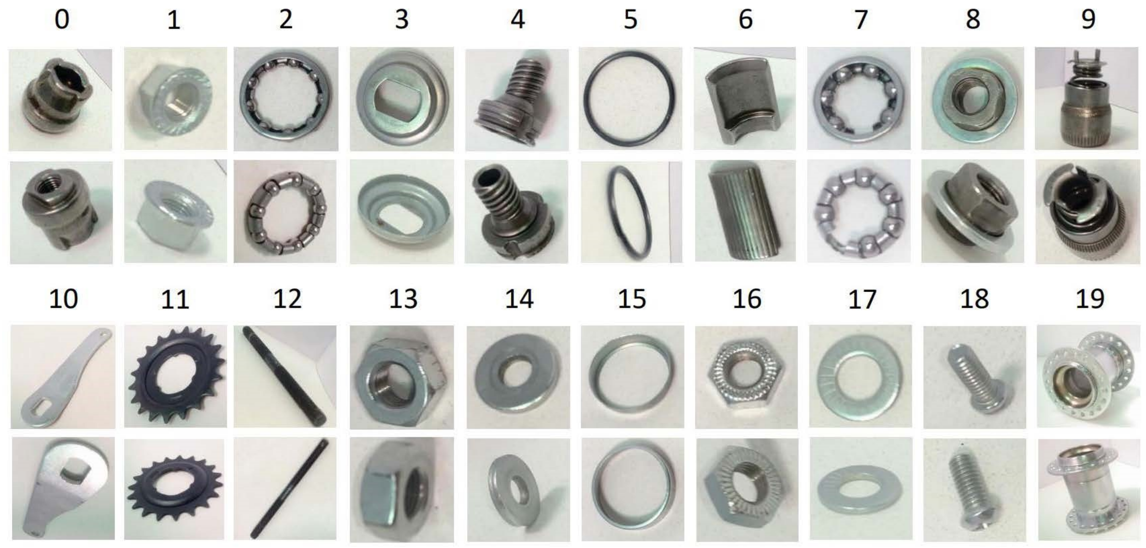

Figure 1: “Bicycle parts” dataset

4. MODEL DESCRIPTION

To build vector representations of images, various pre-trained computer vision models were utilized. For the subsequent classification of vectors, a small fully connected network was applied.

Formally, let $X \subset R^D$ represent a set of images, where $D = 3 \times 192 \times 192$ denotes the dimensionality of the images. In addition to images, let Y denote their corresponding labels, where $Y \subset \{-1,\ 0,\ \ldots C-1\}$. The class labeled as “-1” is a special category for images that are unrelated to the problem being addressed, such as those containing foreign objects.

During the learning process, the model learns the mapping $X \mapsto \hat{X} \subset R^d$, where $d \ll D$, effectively reducing the dimensionality of the feature space. Given the issue of the “curse of dimensionality” in high-dimensional vector spaces, it is not feasible to set large values for d. Furthermore, when addressing the problem of classifying vector representations using a fully connected network, the condition $d \leq C+1$ must be satisfied. Otherwise, the number of classes may become smaller than the dimensionality of the vector, potentially resulting in poorly conditioned weight matrices and significant overfitting of the network. In the models considered, the value of d was varied between 8 and 32, depending on the number of classes in the dataset.

The vector representations of objects constructed in this manner are utilized both for anomaly detection and classification tasks.

The Quadruplet Loss function minimizes intra-class distances while maximizing inter-class ones, thereby addressing the anomaly detection problem through pairwise distances. In this context, in addition to the C classes in the annotation, an extra class of unrelated images is introduced. To accommodate this, the standard Quadruplet Loss implementation has been modified. Specifically, any object from one of the C classes (excluding foreign objects) can be selected as a reference (a) and a positive (p) example. A negative example (n_1) example is then chosen from any of the $C + 1$ classes (including foreign objects), while the fourth example (n_2) is selected from any of the $C + 1$ classes, excluding the object classes ($a,\ p,\ n_1$).

To address the classification of vector representations, a fully connected neural network is implemented to perform the mapping $\hat{X} \mapsto R^{C+1}$. The output, consisting of $C + 1$ values, is subsequently transformed into probabilities representing the likelihood of membership in the corresponding classes using the SoftMax function.

4.1 Model Architecture

Several pre-trained computer vision models were evaluated, including Swin Transformer, MobileNet V3, and EfficientNetV2 [2]. A more detailed description of these models is provided in TABLE 1.

Table 1: Pre-trained models

Model weights	Accuracy@1	Accuracy@5	Params	GFLOPS
Swin_T_Weights.IMAGENET1K_V1	81.474	95.776	28.3M	4.49
MobileNet_V3_Large_Weights.IMAGENET1K_V2	75.274	92.566	5.5M	0.22
EfficientNet_V2_S_Weights.IMAGENET1K_V1	84.228	96.878	21.5M	8.37

To solve the problem of classifying vector representations having dimension d, a small fully connected network was implemented. Given that the pre-trained computer vision model generates vector representations well-suited for clustering and anomaly detection tasks (based on the optimized Quadruplet Loss function), complex architectures for the classification layer were omitted. Thus, during the optimization of the classification loss function, the pre-trained model is trained to minimize the Quadruplet Loss, while also ensuring that the resulting vectors are compatible with classification by a simple neural network.

As a result, a two-layer fully connected network with batch normalization and the LeakyReLU activation function was employed, featuring an internal dimension d.

4.2 Loss Function

The QuadrupletLoss function [3] was utilized to address the anomaly detection problem based on pairwise distances between vector representations of images. This loss function is subsequently denoted as L_{quad} . Additionally, the hard example mining method was applied, introducing an additional penalty for the most challenging object pairs for the model.

For the classification task, the FocalLoss function [4], a modification of CrossEntropyLoss, was employed. This loss function incorporates a hyperparameter that applies an additional penalty to the most challenging objects for the model and is denoted as L_c .

The final loss function, $L = L_{quad} + w_c \bullet L_c$, represents the weighted sum of the two loss functions. Given that the datasets contain a varying number of classes, ranging from 6 to 47, the coefficient w_c was introduced to normalize the classification loss function. If the model predicts equal probabilities for all C classes, the value of the loss function for a single object will be $\ln(C)$. Therefore, the normalization coefficient was set ast $w_c = \frac{1}{\ln(C)}$.

5. EXPERIMENTS

5.1 Datasets Preparation

To expand the training dataset, a series of augmentations were applied, including Athenian transformations, specular reflection, adjustments to brightness and contrast, MotionBlur, ISONoise, image resolution reduction, RandomAutocontrast, RandomEqualize, RandomPosterize, ColorJitter, Gaussian noise, and modifications to the illumination histogram. Examples of the augmented images from the training sets are presented in FIGURE 2.

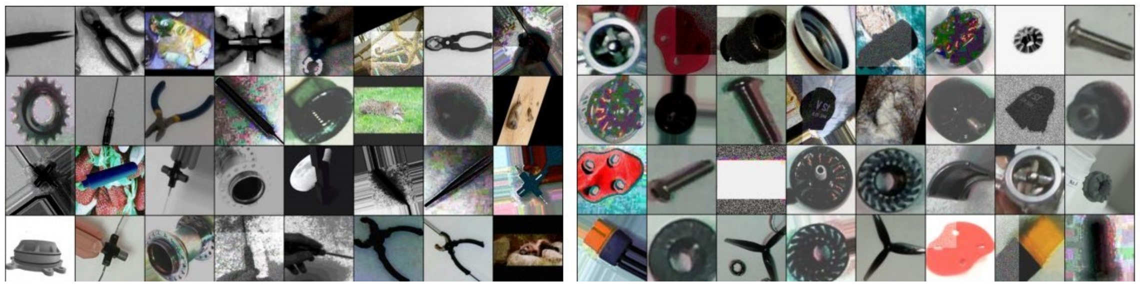

Figure 2: Examples of augmented images

5.2 Model Training

For all datasets, models were trained for 20 epochs with early stopping. To implement the hard example mining method, the proportion of the most complex examples to which an additional penalty is applied was gradually adjusted using a linear schedule. Specifically, this proportion decreased from 1.0 at epoch 2 to 0.1 at epoch 7 and remained constant thereafter.

To address class imbalance in the training dataset, weighted sampling was employed. Given the introduction of an additional class for extraneous images, the proportion of this class was experimentally set to 1/3, with the remaining 2/3 evenly distributed among the other classes.

The AdamW optimizer was used with the following parameters: weight_decay = 0.01 and learning rate (lr) = 0.005.

6. RESULTS

6.1 Metrics

Table 2: Metrics of trained models on test datasets

Model	Balanced Accuracy	F1	Precision	Recall	mAP	Precision@k	mAP@k
Bicycle Parts							
Swin Transformer	0.9858	0.9777	0.9704	0.9858	0.9971	0.9814	0.9822
MobileNetV3	0.9744	0.9761	0.9784	0.9744	0.9959	0.9825	0.9830
EfficientNetV2	**0.9919**	**0.9860**	**0.9806**	**0.9919**	**0.9981**	**0.9897**	**0.9903**
Grinder Details							
Swin Transformer	0.9745	0.9749	0.9754	0.9745	0.9961	**0.9679**	0.9681
MobileNetV3	**0.9776**	**0.9802**	**0.9831**	**0.9776**	0.9958	0.9769	**0.9771**
EfficientNetV2	0.9710	0.9752	0.9796	0.9710	**0.9970**	0.9740	0.9742
Drone Tools							
Swin Transformer	0.9305	0.9282	0.9306	0.9305	0.9808	0.9426	0.9427
MobileNetV3	0.9147	0.9228	0.9378	0.9147	0.9720	0.9478	0.9481
EfficientNetV2	**0.9414**	**0.9486**	**0.9584**	**0.9414**	**0.9899**	**0.9584**	**0.9582**
Drone Details							
Swin Transformer	**0.9877**	**0.9894**	**0.9911**	**0.9877**	**0.9992**	0.9841	0.9842
MobileNetV3	0.9661	0.9766	0.9887	0.9661	0.9967	**0.9901**	**0.9902**
EfficientNetV2	0.9774	0.9824	0.9882	0.9774	0.9987	0.9819	0.9821
All Details							
Swin Transformer	**0.9919**	**0.9909**	**0.9904**	**0.9919**	**0.9978**	**0.9937**	**0.9938**
MobileNetV3	0.9816	0.9823	0.9834	0.9816	0.9932	0.9863	0.9863
EfficientNetV2	0.9717	0.9715	0.9726	0.9717	0.9945	0.9704	0.9718

The results presented in the table demonstrate that the models perform nearly optimally on the test subsets of the datasets, as indicated by the considered metrics. An analysis of the obtained results revealed that, despite the application of augmentations, the datasets predominantly consist of relatively simple examples. Below are examples illustrating the models' performance on an assembly table under conditions simulating real-world scenarios, along with the corresponding metric values.

6.2 Performance

TABLE 3 shows the number of model parameters and GFLOPs required to process a single image of size $3 \times 224 \times 224$ for all pre-trained computer vision models employed in this study.

The performance of the entire system was evaluated using the EfficientNetV2 model, which was trained for video processing. An NVIDIA GeForce RTX 3060 Mobile GPU (3840 cores, 120 TMUs, 48 ROPs, 6 GB GDDR6 memory, 192-bit bus width) and a single-threaded application utilizing OpenCV for camera video processing were employed.

Table 3: Number of parameters and GFLOPs of models

Final model	Params	GFLOPS
Swin Transformer	28.3M	4.49
MobileNetV3	5.5M	0.22
EfficientNetV2	21.5M	8.37

Object detection was performed using the YOLOv5 neural network, which had been previously trained on the dataset for this task. All objects detected within the frame were processed by the EfficientNetV2 model in a single batch.

The measurements were conducted 10 times, and the sample averages and standard deviations were computed. The results are presented in TABLE 4. The term "All time" refers to the total processing time for one frame by the entire system, while "Model time" indicates the processing time for the batch of images by the EfficientNetV2 model.

Table 4: Frame processing time

Batch size	All time (ms) mean	All time (ms) std	Model time (ms) mean	Model time (ms) std
(65, 3, 192, 192)	256,2	34,78	16,6	2,07
(44, 3, 192, 192)	216,6	12,65	14,6	1,07
(32, 3, 192, 192)	158,1	9,56	14,9	1,20
(16, 3, 192, 192)	138,8	6,34	12,7	3,06
(14, 3, 192, 192)	160,2	13,46	12,9	2,02
(5, 3, 192, 192)	79,6	7,18	13,9	2,47
(2, 3, 192, 192)	85,1	5,97	15,6	3,84

6.3 Building Statistical Estimates

To analyze vector representations of images obtained using the Deep Metric Learning method, a t-SNE [5] vector transformation was applied, alongside the construction of box-and-whisker plots for the distributions of intra-class and inter-class distances. Additionally, the probabilities of first- and second-kind errors were estimated.

6.3.1 Intra-class distances

The box-and-whisker diagrams titled "Inter-class distances" illustrate the distributions of intra-class distances. The edges of the box represent the first and third quartiles, while the whisker borders correspond to the 0.025 and 0.975 quantiles. The median is indicated by the orange line. The Euclidean distance metric was used in all graphical representations.

6.3.2 Inter-class distances

The box-and-whisker diagrams titled "Intra-class distances" display the sample distributions of inter-class distances (i.e., distances between each class and all other classes). The same graph parameters used in the "Inter-class distances" diagrams were applied.

6.3.3 Probabilities of errors of the 1st and 2nd kind

For a given class C_i, consider a classifier based on pairwise distances with the null hypothesis H_0, which asserts that objects x of class C_i have an average distance to objects of the selected class that is smaller than the quantile q:

$$\frac{1}{|\{y : class\,(y) = C_i\}|} \sum_{y:class(y)=C_i} ||x - y|| < q, \qquad H_0 : \{class\,(x) = C_i\}$$

By considering $q = q_{1-\alpha}$ – the quantile of the order of $1 - \alpha$ of the distribution of the average distance from an object of class C_i to other objects of the same class C_i, a classifier is obtained with a first-kind error probability of approximately α. The probability of the second-kind error can be estimated by calculating the probability mass of the tail of the distribution.

Similarly, by considering $q = q_\beta$ – quantile of the order β of the distribution of the average distance from an object of class C_i to objects of other classes $C_j,\ i \neq j$, a classifier is obtained with a second-kind error probability of approximately β. The probability of the first-kind error can be estimated by calculating the probability mass of the tail of the distribution.

The values $\alpha = \beta = 0.025$ were selected.

6.4 Results on the "All Details" Dataset

The results on the "All details" dataset are presented in FIGURE 3 - FIGURE 5 and TABLE 5.

Table 5: Estimates of the probabilities of errors of the 1st and 2nd kind, dataset "All details" (all other classes have zero error values)

Class	**By $q_{1-\alpha}$**		**By q_β**	
	1st kind	**2nd kind**	**1st kind**	**2nd kind**
38	0.25	0.020	0.001	0.25
43		0.001	0.009	
44		0.000	0.001	
45		0.000	0.002	

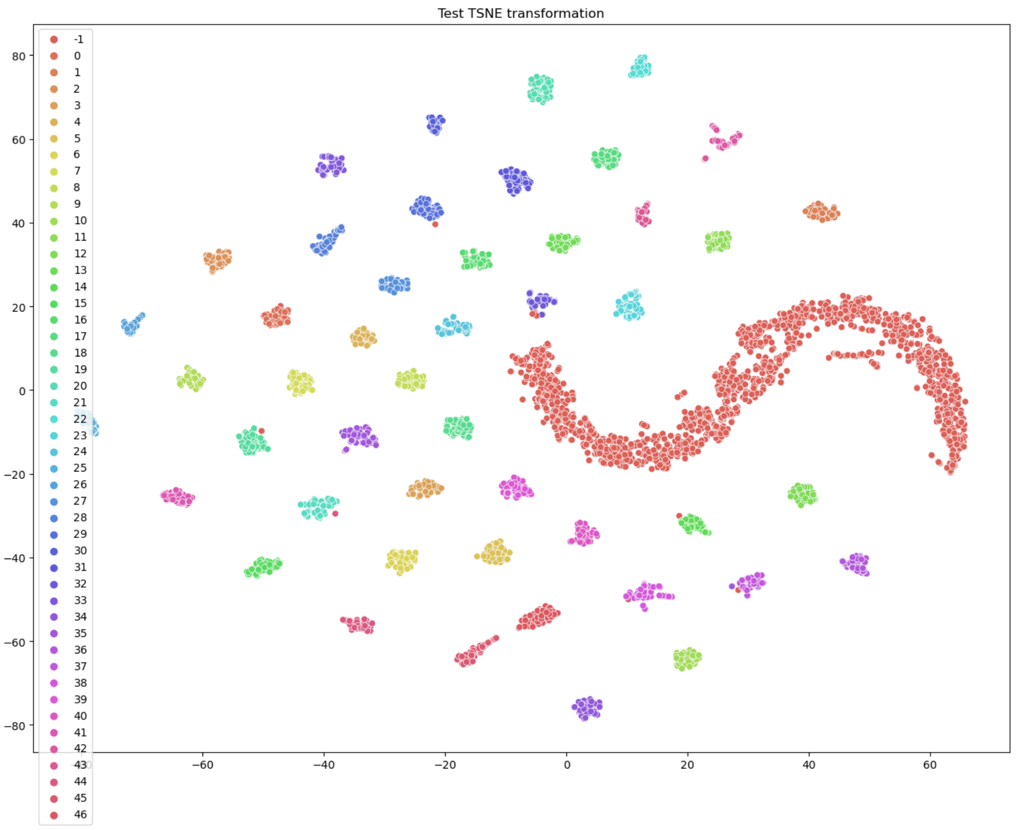

Figure 3: t-SNE conversion of vector representations of images for the "All details" dataset obtained using the Swin Transformer model

7. ANALYSIS OF THE RESULTS OF THE MODELS

In addition to evaluating the models on the test portions of the datasets, they were tested on photographs captured directly during the assembly process. This section presents the results of the models' performance on these images.

7.1 Results of the Detection Model

Object detection in the images was performed using the YOLOv5 model, which had been previously trained on a custom dataset containing images of various objects annotated with their bounding boxes.

The current model demonstrates satisfactory results when the camera is positioned perpendicularly to the table; however, its performance deteriorates significantly when the camera is placed near the table at an angle. Overall, it detects approximately half of the objects of interest. Future work will involve collecting additional data to train the detection model and exploring other models, such as YOLOv8.

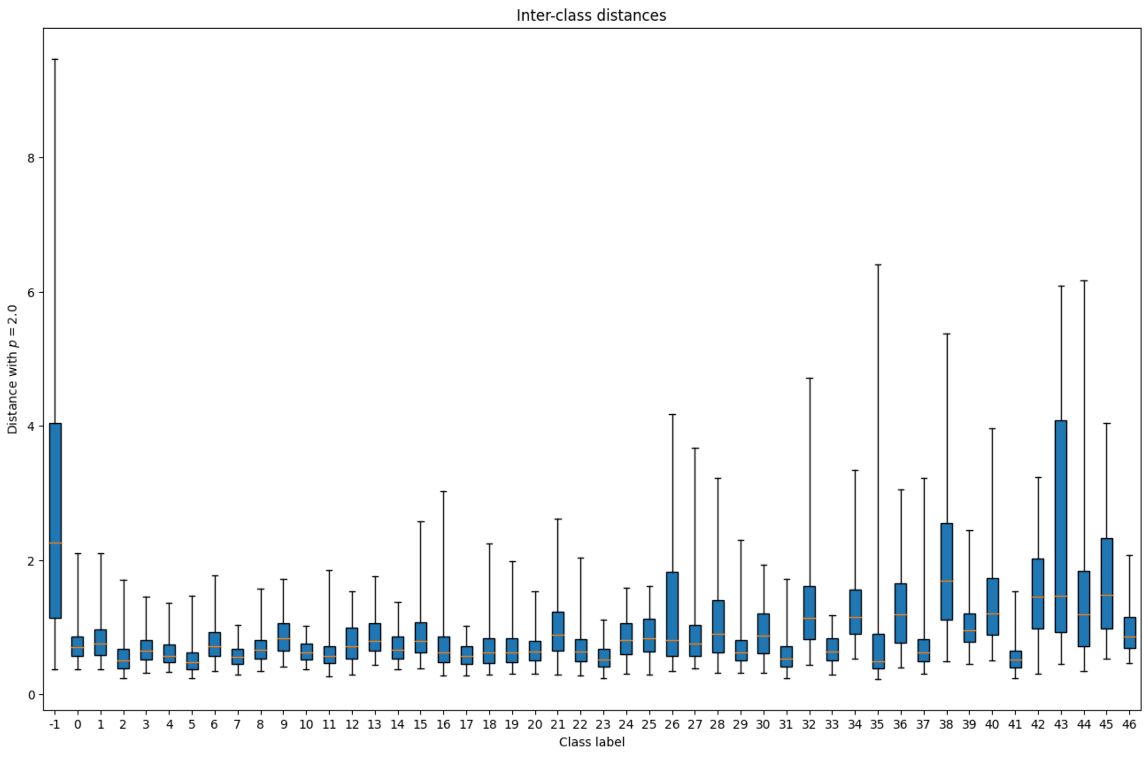

Figure 4: Box-and-whiskey diagrams for intra-class distances, dataset “All details”

7.2 Results of the Detection Model

The results were compared with the works of Deep Quadruplet Network [3], QuadNet [6] and FIDI [7], in which the authors solve the re-identification problem, also using various combined loss functions. Their authors used the CUHK, VIPeR, Market1501, DukeMTMC datasets created to solve the person re-identification task, as well as the MVB dataset for the baggage re-identification task.

Because different approaches employ diverse model architectures, loss functions, and metrics, the accuracy metric was chosen for comparison, as it is reported in all articles across all datasets.

The results of the trained model sre shown in the TABLE 6. They are comparable according to the selected accuracy metric with the results from the work [3, 6, 7]. Thus, a classification model capable of processing extraneous objects by assigning them to a special class has been developed, while maintaining performance quality for the other classes.

The results shown in TABLE 6 differ for the worse from the results of the models on the test part of the datasets (TABLE 2). This is a consequence of the lack of complicated examples in the custom datasets. To improve the performance of the model during the build process, it is planned to expand the custom datasets.

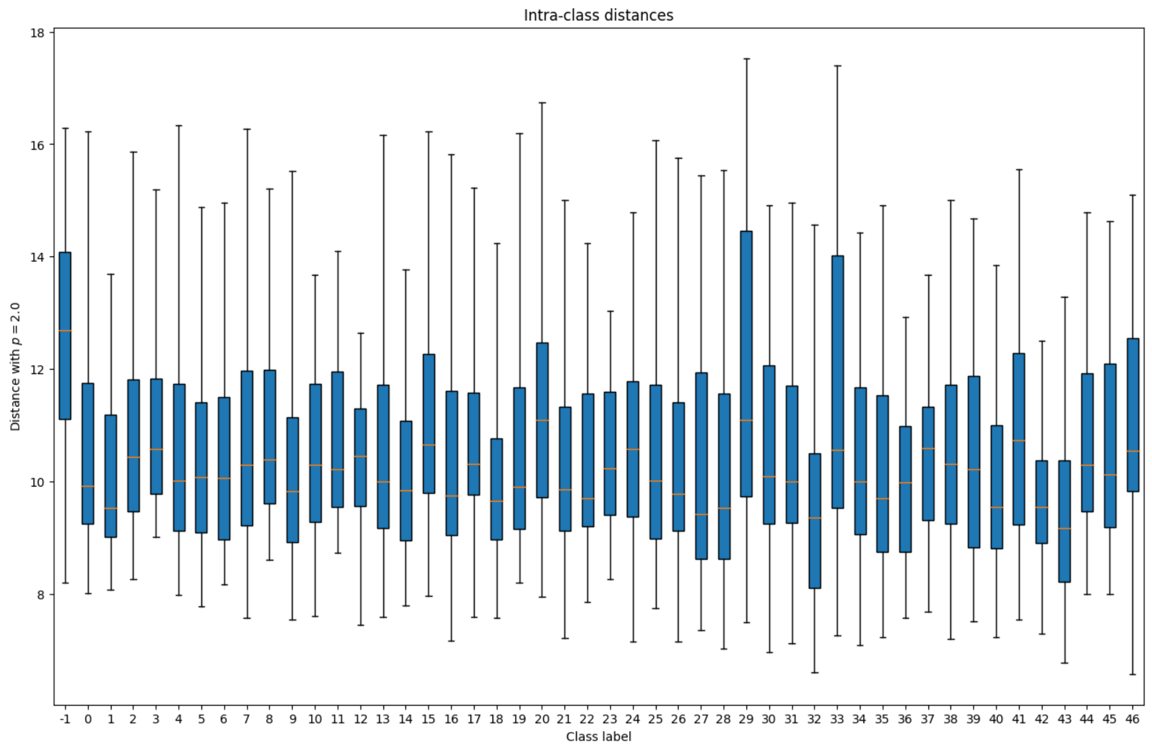

Figure 5: Box-and-whiskey diagrams for inter-class distances, dataset "All details"

Table 6: Comparing our models with other works

Model	Dataset	Accuracy
Deep Quadruplet Network	CUHK03	74.47 %
	CUHK01(p=486)	62.55%
	CUHK01(p=100)	79.00%
	VIPeR	48.42%
QuadNet	MVB	80.30%
FIDI	Market1501	94.50%
	DukeMTMC	88.10%
	CUHK03-D	72.10%
	CUHK03-L	75.00%
EfficientNetV2 (Custom)	Bicycle Parts (Custom)	70.14%
MobileNetV3 (Custom)	Grinder Details (Custom)	98.13%
Swin Transformer (Custom)	All Details (Custom)	66.81%

7.3 The Resulting Vector Representations

To visually assess the constructed vector representations of images, which are subsequently used to address the classification task, t-SNE transformation graphs of these vectors were generated. For all datasets, the majority of objects form distinct, separable clusters. Objects located at the boundaries of their clusters or classified as outliers may be incorrectly assigned to alternative classes based on clustering. To estimate the proportion of such objects and assess their proximity to other classes, box-and-whisker diagrams were constructed, and estimates of the first and second kind error probabilities of a simple classifier were also calculated.

Analysis of the box-and-whisker diagrams reveals that for certain classes, the constructed vector representations exhibit a significantly larger spread of pairwise distances compared to other classes. Notably, this is observed for the "-1" class (representing extraneous objects), which is attributed to the high diversity of objects within this class. For instance, both a photo of a box and a photo of an animal may be categorized under this class in the custom datasets. In contrast, for other classes, a wide spread of pairwise distances may indicate suboptimal quality of the image vectors corresponding to these classes.

The error estimates for the first and second kinds provide a means of evaluating the quality of the constructed vector representations. Although a more complex classifier is employed in the model than the decision rule based on quantile distributions of pairwise distances, the obtained estimates facilitate the identification of problematic classes. These classes may require additional data collection to improve the accuracy of the vector representations.

8. ACKNOWLEDGEMENTS

The article was prepared within the framework of the Basic Research Program at HSE University.

References

[1] https://github.com/ultralytics/yolov5

[2] https://pytorch.org/vision/stable/models.html#table-of-all-available-classification-weights

[3] Chen W, Chen X, Zhang J, Huang K. Beyond Triplet Loss: A Deep Quadruplet Network for Person Re-Identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. New York: IEEE. 2017:403-412.

[4] Lin TY, Goyal P, Girshick R, He K, Dollár P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE international conference on computer vision. 2017:2980-2988.

[5] Van der Maaten L, Hinton G. Visualizing Data Using T-Sne. J Mach Learn Res. 2008;9.

[6] Yang H, Chu X, Zhang L, Sun Y, Li D, et. al. Quad Net: Quadruplet Loss for Multi-View Learning in Baggage Re-Identification. Pattern Recognition. 2022;126:108546.

[7] Yan C, Pang G, Bai X, Liu C, Ning X, et. al. Beyond Triplet Loss: Person Re-Identification With Fine-Grained Difference-Aware Pairwise Loss. IEEE Trans Multimedia. 2022;24:1665-1677.