# AI Multimedia Lab at ImageCLEFmedical GANs 2024: Deep Learning Approaches for Analyzing Synthetic Medical Images

Notebook for the ImageCLEF Lab at CLEF 2024

Alexandra-Georgiana Andrei[1],*, Mihai Gabriel Constantin[1], Mihai Dogariu[1] and Bogdan Ionescu[1]

[1] *AI Multimedia Lab, CAMPUS Research Center, National University of Science and Technology Politehnica Bucharest, Bucharest, Romania*

**Abstract**

This paper presents the participation of AI Multimedia Lab to the 2024 ImageCLEFmedical GANs task. The 2024 ImageCLEFmedical GANs propose two sub-tasks that study security and privacy concerns related to personal medical image data in the context of generating and using artificial images in different real-life scenarios: identifying "fingerprints" left by the original training data within synthetic medical images and detecting unique "fingerprints" imprinted by different generative models on their synthetic outputs. For the first sub-task, we proposed advanced algorithms that leverage medical image segmentation, deep feature extraction, and clustering techniques to accurately identify fingerprints from the original data. For the second sub-task, we proposed robust clustering frameworks and feature extraction methods using both pre-trained deep learning models and handcrafted techniques to distinguish between synthetic images generated by various models. Our methods demonstrated promising results obtaining a maximum F1-score of 0.627 for the first sub-task and an ARI of 0.996 for the second sub-task, highlighting their potential in addressing security and privacy concerns in the context of synthetic medical images.

**Keywords**

ImageCLEFmedical GANs, Generative models, Generative Adversarial Networks, synthetic medical images, CT images

## 1. Introduction

Medical imaging plays a crucial role in the diagnosis and treatment of various conditions by providing information about the internal structures and functions of the human body. Even with current technological developments and the abundance of data available in hospitals and other private institutions, it is still difficult to make these data easily accessible to the scientific community for the purpose of developing algorithms. This difficulty arises from the necessity of maintaining patient data confidentiality. To address these impediments, generative models have been proposed to augment datasets and even improve their quality.

In its second edition, ImageCLEF2024medical GANs task [1], part of the 2024 ImageCLEF2024 [2] evaluation campaign focuses on leveraging generative models to improve the quality and utility of medical images. The task is divided in two sub-tasks that address both security and privacy concerns related to personal medical data and on understanding if different generative models imprint different discernible signatures within the synthetic images they produce.

This paper presents our methods for addressing the challenges proposed by ImageCLEFmedical GANs task. The paper is structured as follows: Section 2 presents the tasks and the datasets, Section 3 presents the proposed methods and the results are presented and discussed in Section 4. Finally, the paper closes with Section 5, where we present the conclusions.

*CLEF 2024: Conference and Labs of the Evaluation Forum, September 09–12, 2024, Grenoble, France*

*Corresponding author.

alexandra.andrei@upb.ro (A. Andrei); mihai.constantin84@upb.ro (M. G. Constantin); mihai.dogariu@upb.ro (M. Dogariu); bogdan.ionescu@upb.ro (B. Ionescu)

0 (A. Andrei)

**Table 1**
Description of training and testing datasets made available for *Identify training data “fingerprints” sub-task*.

| | Train | | Test | |
|---|---|---|---|---|
| | *Real images* | *Generated images* | *Real images* | *Generated images* |
| **Model 1** | 100 (used)<br>100 (not used) | 10,000 | 4,000 (used and not used) | 7,200 |
| **Model 2** | 3000 (used)<br>3000 (not used) | 10,000 | 4,000 (used and not used) | 5,000 |

# 2. The 2024 ImageCLEFmedical GANs Tasks

## 2.1. Identify training data “fingerprints”.

The task was introduced in the previous edition [3] and involves investigating the hypothesis that generative models produce medical images that bear similarities to the original images used for training the generative model. This addresses security and privacy concerns related to personal medical image data in the context of generating and using artificial images in various real-life scenarios.

The objective is to detect “fingerprints” within synthetic biomedical image data to determine which real images were used in the training process to produce the generated images. This task involves analyzing test image datasets and assessing the likelihood that specific images of real patients were used to train the generative models. This sub-task involved investigating the hypothesis for two different generative models.The dataset provided for this tasks consists of both real and generated images, as described in Table 1. More information about the sub-task and the provided data are available in the overview paper of the task [1].

## 2.2. Detect generative models’ “fingerprints”

The task involves exploring the hypothesis that generative models imprint unique fingerprints on the synthetic images they produce. The primary focus is on determining whether different generative models or architectures leave discernible signatures within these synthetic images.

To address this, a set of synthetic images generated through various generative models is provided, with the objective of identifying and detecting the distinct "fingerprints" associated with each model. This task requires analyzing the embedded characteristics, patterns, or features in the synthetic images. This investigation contributes to a deeper understanding of the unique imprints left by generative models on their generated images, facilitating model attribution and recognition. The dataset provided for this tasks consists of generated images: 600 images generated using three different generative models for the training dataset (each model is represented by 200 images and are annotated accordingly) and a mixture of 3,000 generated images for the test dataset generated using four different generative models. More information about the sub-task and the provided data are available in the overview paper of the task [1].

# 3. Proposed Methods

## 3.1. Identify training data “fingerprints”.

For the first sub-task we propose two different approaches. The first one is based on segmenting the lung area from the CT and extracting features for different relevant regions and, finally, clustering the feature vectors such that images used for training will be grouped together and separated from the ones not used for training. The second approach relies on training an autoeconder to reconstruct the generated images. The hypothesis is that using the same autoencoder on the training images will yield lower reconstruction errors than on images outside the training set.

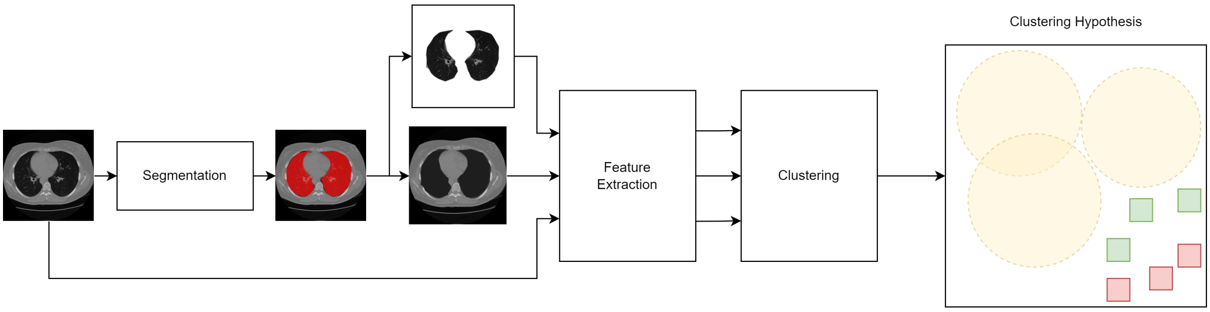

**Figure 1:** Overview of the proposed medically relevant region detection method for Identify training data "fingerprints" sub-task. We present the three stages: Segmentation, Feature Extraction, Clustering, as well as the hypothesis we use for clustering.

### 3.1.1. Analyzing medically relevant regions

Our first set of approaches for identifying training data fingerprints aims at isolating and analyzing different medically relevant regions of the target images. This approach is presented in Figure 1, and is composed of several stages as follows: (i) medical image segmentation, detecting the area of the lungs, (ii) deep feature extraction, and (iii) clustering. This results in three types of individual runs, the baseline run consisting of using the entire image for stages 2 and 3, while the second type isolates and uses only the lung regions in stages 2 and 3, which are the most medically relevant regions in the CT images. Finally, the last type of run isolates and removes the lung regions, analyzing the rest of the CT image and, while this region may be less medically relevant, it could still contain fingerprints of the original images.

For medical image segmentation, we deploy a UNet [4] deep neural network, trained for lung segmentation in CT images. The detected lung regions are then prepared for the second and third stages, with one set of images containing only the lung regions, cropped according to the limits of the lung regions and with all other non-lung pixels being set to 0, while the other set of images being a copy of the original images, with the lung region pixels set to 0.

In the second stage we deploy two DNNs and extract features from their layers, namely the ResNet50 [5] and the DenseNet121 [6] networks. We use the pretrained versions of these networks, consisting of weights trained on the ImageNet dataset [7]. We extract features from the penultimate layer for each of the two networks, resulting in a vector of values of size $(2048 \times 1)$ for ResNet and a $(1024 \times 8 \times 8)$ vector of values for DenseNet, that is then averaged obtaining a vector of size $(1024 \times 1)$.

In the third stage, we test two different clustering methods: k-means and hierarchical clustering, while varying the number of clusters $k$. The hypothesis we wish to test in this set of experiments is also presented Figure 1, and it is as follows. The training process of the two generative models associated with this task will inevitably create new artificial images that can be associated with a variable number of clusters. Thus, the entire set of generated images can be expressed as a set of clusters $\mathcal{G} = G_1 \cup G_2 \cup ... \cup G_k$, where $k$ is a variable number of clusters. In theory, given a distance $d$ and two samples, $S_u$ and $S_n$, with the former used in training the generative networks and the latter not used, the used sample should be closer to the clusters than the sample that was not used during training: $d(\mathcal{G}, S_u) < d(\mathcal{G}, S_n)$. We also test two types of distances, one where only the distance to the closest cluster is taken into account $d(\mathcal{G}, S) = min(d(G_i, S))$, and one where the average of all the distances is taken into account: $d(\mathcal{G}, S) = avg(d(G_i, S))$. We also test a large number of values for the number of clusters, in total 30 possible values. We find the best values of $k$ and the best setting for the distance variation based on tests on the training set, which we then deploy on the testing set.

### 3.1.2. Anomaly detection applied to generative models

Our next approach focuses on the autoencoders' ability to capture the reconstruction ability of the training dataset. The autoencoder is composed of two main blocks, the encoder $E$, which transforms

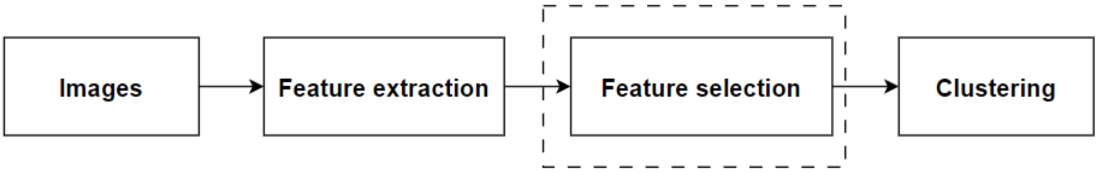

**Figure 2:** Overview of the proposed approach for Detect generative models' "fingerprints" sub-task.

an input sample into a feature vector, also called bottleneck, and the decoder $D$, which transforms the bottleneck back into the input sample's domain. The output of the decoder should, usually, match the input of the encoder. Formally, $\tilde{x} = D(E(x)) \approx x$. However, this reconstruction is almost never lossless. In order to asses the reconstruction error, we use the mean square error (MSE). Ideally, $MSE(x, \tilde{x}) = 0$. Practically, we long for MSE values as low as possible. This is an indicator that the autoencoder adapted to the training dataset's probability density function and can successfully reconstruct any (or most) of the inputs from the training dataset.

Hypothetically, an autoencoder trained on a certain dataset should yield low reconstruction errors for samples that were used for training the said autoencoder and higher reconstruction errors for samples that are outside the training dataset. In this respect, we consider that an autoencoder trained on the generated samples of a generative model should yield low reconstruction errors for samples that were used for training the generative model and higher reconstruction errors for samples outside the training dataset. This hypothesis is, of course, prone to alterations due to the large similarity between 'used' and 'not_used' subsets for the generative model's training and the low complexity of the analyzed images.

We follow 2 different approaches. During the first approach, we compute the mean of the MSE computed on the pixel level and determine a centroid value for the images that are sure to have been part of the training dataset. Similarly, we compute the mean MSE value of the reconstructions of images that are surely outside the training dataset. We, thus, obtain two central values, one for the images that were used during training and one for the images that were not used during training. Finally, we compute the distance of the MSE reconstruction loss of new (test) samples and assign them to the closest group of samples, thus labeling them as being 'used' or 'not_used' during training.

The second approach is similar to the first one. We compute the MSE loss for each input, but this time at the pixel-level. We create a reconstruction error image by averaging the pixel-level MSE errors for all images that were used for training the generative model and one for the images that were not used for training the model, thus obtaining two centroid-like images. Lastly, we compute the SSIM [8] between the reconstruction error of new samples and the two centroid-like images. The largest SSIM value yields the class of images ('used'/'not_used') the test image falls into.

## 3.2. Detect generative models' "fingerprints"

We proposed different variations of the pipeline presented in Figure 2 for detecting the "fingerprints" of the generative models. The presented methods use pattern recognition and feature extraction techniques to analyze the features embedded in the generated images. Building on the the method presented by our team in the previous edition [9], we employed two approaches for feature extraction. The first approach uses Transfer Learning, where a pre-trained model is used, while the second approach involves using a handcrafted extraction technique.

***Feature extraction*** – For the deep-learning feature extraction method, we employed different pre-trained models that were originally trained on the ImageNet dataset [10]: i) VGG-16 model [11], ii) ResNet [5], iii) MobileNetV2 [12], iv) EfficientNet [13] and v) DenseNet-121 [6]. These models were selected for feature extraction due to their proved efficacy and robustness in various computer vision tasks.

*VGG-16* generates rich hierarchical features due to its architecture comprising of 16 weight layers, including 13 convolutional layers and 3 fully connected layers. It uses $3 \times 3$ filters and it allows capture

intricate patterns and details in the input images. Same as the VGG-16, *ResNet50* captures complex patterns and hierarchical features from input images due to its architecture characterized by its residual blocks, where each block contains a series of convolutional layers with batch normalization and ReLU activation. ResNet50 is a variant of the Residual Network (ResNet) model, which is known for its powerful capabilities in deep learning applications.
*DenseNet-121*, a variant of the Densely Connected Convolutional Networks (DenseNet), is highly esteemed for its effective and efficient feature extraction capabilities. DenseNet-121 introduces an innovative architecture where each layer is directly connected to every other layer within a dense block. This dense connectivity pattern significantly enhances information flow and gradient propagation throughout the network, facilitating better learning and reducing the risk of vanishing gradients.
*MobileNetV2* is a suitable option for feature extraction in non-mobile scenarios as well, despite its initial design for mobile applications. The model's architecture, characterized by inverted residuals and linear bottlenecks, significantly reduces the number of parameters and computational complexity while maintaining high accuracy, having the ability ti provide high-quality feature representations.
*EfficientNet* is a convolutional Neural Network (CNN) capable of efficient and scalable feature extraction. Because of its scalability, it can adjust to different application needs, ranging from high-performance computer systems to situations with limited resources. Moreover, EfficientNet's architecture, which is based on squeeze-and-excitation modules and mobile inverted bottleneck convolutions guarantees that complex patterns and hierarchical features are efficiently captured.

All these models have in common the fact that the initial layers capture low-level features such as edges and textures, while the deeper layers extract complex features such as shapes and high-level patterns.

***Feature selection*** – Using pre-trained networks for features extraction, resulted in a substantial number of feature, as presented in Table 2 for the training dataset. To manage the high dimensionality of these features, Principal Component Analysis (PCA) was applied to reduce them to the minimum number of components required to capture the variance in the original data. During this process, different values for the number of components were tested to determine the optimal balance between dimensionality reduction and information retention. This approach ensured that the selected number of components was neither too few, which would risk losing significant information, nor too many, which would undermine the benefits of dimensionality reduction.

We have also employed a handcrafted feature extraction technique for extracting the Local Binary Pattern (LBP). It captures information about the local spatial patterns and the grayscale contrast of the images.

***Clustering*** – Knowing the number of classes/clusters to which the images belong (3 for the development/test dataset and 4 for the test dataset), we used two different clustering methods: i) k-means and ii) hierarchical clustering. Both k-means and hierarchical clustering aim to group data points into clusters based on similarity, but they do so in fundamentally different ways. We chose to use both k-means and hierarchical clustering to identify patterns and group CT slices with similar features within the provided dataset. By comparing the outcomes, we can validate that the feature extraction method effectively captures the data's intrinsic properties.

# 4. Results and Discussion

## 4.1. Identify training data "fingerprints".

For the first task, the best results for identifying the training data "fingerprints" were obtained with different methods on the two datasets. On the first dataset, the method focusing on analyzing the medically relevant regions obtained the highest F1 score, 0.5385. On the second dataset, the method relying on detecting anomalies with by computing SSIM on reconstruction errors obtained the highest F1 score, 0.62753. We present both methods in detail, next.

**Table 2**
Performance results of the proposed methods on the training dataset for *Detect generative models' "fingerprints"* task. The best results on the testing set are presented in bold

| Feature extraction | Clustering method | # of features | ARI |
|---|---|---|---|
| VGG-16 | k-means | 25088 | 0.8740 |
| VGG-16 | hierarchical clustering | 25088 | 0.8740 |
| VGG-16 + PCA (95%) | k-means | 351 | 0.9949 |
| VGG-16 + PCA (95%) | hierarchical clustering | 351 | 0.8740 |
| VGG-16 + PCA (90%) | k-means | 235 | 0.9752 |
| VGG-16 + PCA (90%) | hierarchical clustering | 235 | 0.8740 |
| Hand-crafted – LBP | k-means | 1 | 0.3726 |
| Hand-crafted – LBP | hierarchical clustering | 1 | 0.4435 |
| ResNet50 | k-means | 2048 | 0.9850 |
| ResNet50 | hierarchical clustering | 2048 | 0.1 |
| ResNet50 + PCA (95%) | k-means | 152 | 0.9850 |
| ResNet50 + PCA (95%) | hierarchical clustering | 152 | **1** |
| ResNet50 + PCA (90%) | k-means | 78 | 0.9850 |
| ResNet50 + PCA (90%) | hierarchical clustering | 78 | **1** |
| MobileNetV2 | k-means | 1280 | **0.9949** |
| MobileNetV2 | hierarchical clustering | 1280 | **0.9900** |
| EfficientNet | k-means | 1280 | 0.2166 |
| EfficientNet | hierarchical clusterin | 1280 | 0.3608 |
| EfficientNet + PCA (95%) | k-means | 3 | 0.2166 |
| EfficientNet PCA (95%) | hierarchical clusterin | 3 | 0.3159 |
| DenseNet-121 | k-means | 1024 | **1** |
| DenseNet-121 | hierarchical clusterin | 1024 | **1** |

### 4.1.1. Analyzing medically relevant regions

Overall, the performances of hierarchical clustering on the training set were significantly superior to those of the k-means approach in all preliminary tests. We therefore decided to proceed and test only predictions associated with the hierarchical clustering approaches. Table 3 presents the results associated with this approach, as well as the features and the clustering setup defined as (distance type, number of clusters) for each of the six submitted runs. The best result on the testing set is achieved with a ResNet-based feature extractor, processing outer regions not containing the lung segments, and using the minimum distance variation with a number of 10 clusters for the hierarchical clustering method. This approach achieved an F1 score of 0.5295 for the first dataset, 0.5475 for the second dataset, and an average F1 value of 0.5385.

When analyzing the differences between the expected performance (F1 values on the training set) and real performance (F1 values on the testing set), a noticeable difference can be observed between the two datasets. For the first dataset, the decrease in performance is between 8.21% and 20.53% across all six runs, while for the second dataset, only one of the six runs registered a lower performance on the

**Table 3**
Results of the "Analyzing medically relevant regions" methods on the training and testing dataset for Identify training data "fingerprints" subtask. The best results on the testing set are presented in bold.

| | | | | F1 - training set | | | F1 - testing set | | |
|---|---|---|---|---|---|---|---|---|---|
| # run | Image Crop | Features | Clustering Setup | DB1 | DB2 | average | DB1 | DB2 | average |
| #1 | full image | DenseNet | (min, 50) | 0.6 | 0.5163 | 0.55815 | 0.4805 | 0.519 | 0.49975 |
| #2 | full image | ResNet | (min, 8) | 0.55 | 0.506 | 0.528 | 0.4765 | 0.528 | 0.50225 |
| #3 | lung regions | ResNet | (avg, 4) | 0.6 | 0.504 | 0.552 | 0.496 | 0.5345 | 0.51525 |
| #4 | lung regions | DenseNet | (avg, 20) | 0.56 | 0.5073 | 0.53365 | 0.514 | 0.506 | 0.51 |
| #5 | outer regions | ResNet | (min, 10) | 0.59 | 0.5036 | 0.5468 | **0.5295** | **0.5475** | **0.5385** |
| #6 | outer regions | DenseNet | (min, 60) | 0.66 | 0.517 | 0.5885 | 0.5245 | 0.5295 | 0.527 |

**Table 4**
Results of the "Anomaly detection applied to generative models" methods on the testing dataset for Identify training data "fingerprints" subtask.

| # run | MSE reduction | DB1 | | | | DB2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Acc | Prec | Rec | F1 | Acc | Prec | Rec | F1 | mean F1 |
| #7 | average MSE | 0.50075 | 1.0 | 0.0015 | 0,002995 | 0.49775 | 0.49747 | 0.4425 | 0.46837 | 0.23568 |
| #8 | SSIM | 0.49925 | 0.4996 | 0.9775 | 0,661254 | 0.52325 | 0.51725 | 0.697 | 0.59382 | **0.62753** |

testing set, with a 0.2% decrease in performance, with the other five runs surprisingly registering higher performance on the testing set compared with the training set, an increase between 0.52% and 8.71%. While this different behaviour must be more thoroughly studied in future experiments, at this moment we believe this to be the outcome of the number of samples in the training set for the two dataset. The first dataset is composed of only 200 real images, while the second one has 6000. This would result in a sub-optimal search for the parameters of the clustering method (the type of distance and the number of clusters), as the low number of real images in the first dataset may not be representative enough for the entire dataset, or a better method of clustering is needed in cases with lower number of samples in the training set.

Finally, the cropping and masking method seems to have an important effect on the final results, seemingly regardless of the dataset analyzed, feature extractor, or clustering setup. The original uncropped images show a maximum F1 performance of 0.50225 across both datasets, while the lung region and outer region images show a maximum F1 performance of 0.51525 and 0.5385 respectively. This is another interesting phenomenon, that must be further studied, but, at this point, we theorize that analyzing smaller patches of the original images may allow the feature extractors to concentrate more on specific areas and features and to concentrate their responses to those specific areas. Another point of discussion here is if the nature of the patches has any importance on the improvement of the results. In our approaches, we specifically targeted regions of the image that have a medical significance (regions containing only lungs and the rest of the body and CT scan). However, in our future studies we should test if any random patch of a large enough size of the images would improve the result compared with a full image analysis.

#### 4.1.2. Anomaly detection applied to generative models

The anomaly detection approaches seem to yield good results for the second dataset, whereas it fails on the first dataset. On the first dataset, the 'used' and 'not_used' centroid-like features that were computed on the development data and adequately tuned to ensure a good segregation between the data that was used to train the generative model and the external one does not seem to generalize to the test dataset, outputting a single label for the entire test set. This might happen due to a large difference between the dataset used to train the development generator and the dataset used to train the test generator.

For the second dataset, however, the proposed approaches seem to have a better fitting, suggesting that the second model has a stronger correlation between the training dataset and the test dataset.

Between the two applied methods it is clear that the SSIM approach is superior to the vanilla one. This is somewhat to be expected, since for images it is also important where the reconstruction error is located and not only its absolute value. SSIM takes into account the entire structure (pixel error placement) instead of averaging over all points, obtaining an F1 score of 0.5938 for DB2, as opposed to 0.4683, as per Table 4.

### 4.2. Detect generative models' "fingerprints".

Analyzing the results obtained on the testing set presented in Table 2, it is observed that the best results, achieving the highest possible ARI score of 1, were obtained using DenseNet-121 for feature extraction. This result was consistent across both clustering methods. The next best results were obtained using the

**Table 5**
Results of the proposed methods on the test dataset for Detect generative models' "fingerprints" task.

| # run | Method | ARI |
|---|---|---|
| #1 | DenseNet-121 + hierarchical clustering | **0.9965** |
| #2 | DenseNet-121 + k-means | 0.9347 |
| #3 | LBP + hierarchical clustering | 0.3293 |
| #4 | LBP + k-means | 0.5036 |
| #5 | MobileNetV2 + hierarchical clustering | 0**.9971** |
| #6 | MobileNetV2 + k-means | 0.9008 |
| #7 | VGG−16 + PCA (95%) + hierarchical clustering | 0.5527 |
| #8 | VGG−16 + PCA (95%) + k-means | 0.6540 |
| #9 | ResNet 50 + hierarchical clustering | 0.7229 |
| #10 | ResNet50 + k-means | 0.6454 |

MobileNetV2 network for feature extraction, which achieved an ARI of 0.9949 with the k-means method and an ARI of 0.99 with hierarchical clustering. For this method, PCA was also applied to reduce the number of features, but the results were consistent. Significant ARI values were also achieved using the VGG-16 network for feature extraction. Moreover, applying PCA for feature reduction in conjunction with VGG-16 feature extraction and k-means classification resulted in an increased ARI value.

Based on the evaluation of various methods tested on the development dataset, we selected the top-performing approaches to be applied for the test dataset and we obtained the results presented in Table 5. Specifically, we chose DensNet-121, MobileNetV2, ResNet and VGG-16 which demonstrated superior ARI on the development dataset. In addition to this, we included the method that employs hand-crafted feature – LBP – to provide a comparative analysis between automated feature extraction through deep learning and traditional hand-crafted feature extraction.

When comparing the expected performance, measured by ARI on training set, with the actual performance obtained on the training set, a noticeable difference can be observed between the results obtained using VGG-16 and ResNet CNNs for feature extraction. The decrease in performance is between 22.8% and 33% for all runs that used VGG-16 and ResNet for feature extraction. This divergence in performance can be attributed to differences in the composition of the training and testing datasets from the perspective of the number of generative models employed. It is plausible that the features extracted by VGG-16 and ResNet from the training set do not generalize well to the testing set due to variations in the distribution or characteristics of the generative models represented in each dataset. As a result, the models trained on VGG-16 and ResNet features may struggle to effectively classify instances in the testing set, leading to a notable decline in performance compared to the training set. Similar performances were obtained using MobileNetV2 and DenseNet-121 for feature extraction. The higher ARIs on the testing set of 0.9971 respectively 0.9965 were obtained using DensNet and MobileNet for feature extraction and hierarchical clustering. Similar results were also obtained using the hand-crafted feature extraction technique we use to extract the LBP feature.

The achieved ARI value of 0.9971 indicates an exceptionally high level of agreement between our clustering results and the ground truth. This result underscores the effectiveness of the proposed method (feature extraction using MobileNetV2 and hierarchical clustering) in accurately capturing the inherent patterns and relationships within the images generated using different generative models.

## 5. Conclusions

This paper presents the methods developed by AI Multimedia Lab for 2024 ImageCLEFmedical GANs task, focusing on identifying training data "fingerprints" and detecting generative models' "fingerprints" within medical images. We provided various methods, including analyzing medically relevant regions, anomaly detection using autoencoders, and employing deep learning-based feature extraction together with clustering techniques. Our results demonstrate promising capabilities in detecting the images used for training a generative models, obtaining an F1-score of 0.627. Using feature extracting and clustering methods, we managed to achieve almost a maximum ARI in clustering synthetic images based on the generative model that generated them. Further work can be conducted to optimize the approaches for broader applicability across diverse medical imaging scenarios.

## Acknowledgments

The contribution to this task is supported under project AI4Media, A European Excellence Centre for Media, Society and Democracy, H2020 ICT-48-2020, grant #951911. Mihai Dogariu's work was supported by a grant from the National Program for Research of the National Association of Technical Universities - GNAC ARUT 2023.

## References

[1] A. Andrei, A. Radzhabov, D. Karpenka, Y. Prokopchuk, V. Kovalev, B. Ionescu, H. Müller, Overview of 2024 ImageCLEFmedical GANs Task – Investigating Generative Models' Impact on Biomedical Synthetic Images, in: CLEF2024 Working Notes, CEUR Workshop Proceedings, CEUR-WS.org, Grenoble, France, 2024.

[2] B. Ionescu, H. Müller, A. Drăgulinescu, J. Rückert, A. Ben Abacha, A. Garcıa Seco de Herrera, L. Bloch, R. Brüngel, A. Idrissi-Yaghir, H. Schäfer, C. S. Schmidt, T. M. Pakull, H. Damm, B. Bracke, C. M. Friedrich, A. Andrei, Y. Prokopchuk, D. Karpenka, A. Radzhabov, V. Kovalev, C. Macaire, D. Schwab, B. Lecouteux, E. Esperança-Rodier, W. Yim, Y. Fu, Z. Sun, M. Yetisgen, F. Xia, S. A. Hicks, M. A. Riegler, V. Thambawita, A. Storås, P. Halvorsen, M. Heinrich, J. Kiesel, M. Potthast, B. Stein, Overview of ImageCLEF 2024: Multimedia retrieval in medical applications, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction, Proceedings of the 15th International Conference of the CLEF Association (CLEF 2024), Springer Lecture Notes in Computer Science LNCS, Grenoble, France, 2024.

[3] A.-G. Andrei, A. Radzhabov, I. Coman, V. Kovalev, B. Ionescu, H. Müller, Overview of imageclefmedical gans 2023 task: identifying training data "fingerprints" in synthetic biomedical images generated by gans for medical image security, in: Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2023), volume 3497, 2023.

[4] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, Springer, 2015, pp. 234–241.

[5] B. Koonce, B. Koonce, Resnet 50, Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization (2021) 63–72.

[6] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, K. Keutzer, Densenet: Implementing efficient convnet descriptor pyramids, arXiv preprint arXiv:1404.1869 (2014).

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., Imagenet large scale visual recognition challenge, International journal of computer vision 115 (2015) 211–252.

[8] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE transactions on image processing 13 (2004) 600–612.

[9] A.-G. Andrei, B. Ionescu, Aimultimedialab at imageclefmedical gans 2023: determining "fingerprints" of training data in generated synthetic images, in: CLEF2023 Working Notes, CEUR Workshop Proceedings, Thessaloniki, Greece, 2023.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255. doi:10.1109/CVPR.2009.5206848.

[11] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4510–4520.

[13] B. Koonce, B. Koonce, Efficientnet, Convolutional neural networks with swift for Tensorflow: image recognition and dataset categorization (2021) 109–123.