[Re] DiRA: Discriminative, Restorative, and Adversarial Learning for Self-supervised Medical Image Analysis

Bruce Chidley, Mahmoud Idlbi, Mane Piliposyan

October 2023

Abstract

This study explores the application of self-supervised learning (SSL) in the medical domain, employing the innovative DiRA model. DiRA, incorporating discriminative, restorative, and adversarial learning, demonstrates reliable performance in identifying various medical conditions. The performance of DiRA in comparison to other models underscores the importance of SSL in overcoming challenges related to insufficiently labeled medical data. By leveraging a variety of image augmentation techniques, the performance of DiRA in image classification and segmentation was increased quite significantly, further highlighting its potential. DiRA's strength in transfer learning was also explored, seeing performance above what was expected. These findings pave the way for future research, offering directions to enhance DiRA's performance and adaptability in medical imaging, especially when compared to the benchmark MoCo-v2 model. Our modifications are available at: `https://github.com/Mah514/DiRA_Reproducibility`

1 Introduction

DiRA [1] is an innovative self-supervised model, that combines discriminative, restorative, and adversarial learning methods to extract useful features from unlabeled medical data. This framework can be integrated into pre-existing models, boosting performance for even those that are considered state-of-the-art (SOTA). In other words, DiRA acts as an extension that boosts the effectiveness, accuracy, and robustness of already established models making them more competitive in the medical field.

The authors claim that DiRA shows several advantages: it encourages collaborative learning among different components, resulting in more adaptable image representations; it outperforms fully supervised ImageNet models and is more robust with limited data, reducing the need for costly annotations in medical imaging; it learns detailed information, making it easier to pinpoint specific issues even with minimal annotations; and it enhances existing restorative methods, proving that DiRA is a versatile framework for unified learning of representations. It is the first SSL framework that combines the three learning techniques and sets a new SOTA for SSL models in the medical field.

The authors incorporated DiRA with three 2D SSL models (MoCo-v2, Barlow Twin, and SimSiam) and one 3D SSL SOTA model (TransVW). However, the authors have provided code for only the DiRA model based on MoCo-v2. The implementations for the other models are quite unclear due to a lack of instructions detailing how to merge the approaches. In this reproducibility report, we will focus on only $\text{Dira}_{\text{MoCo-v2}}$ for classification and segmentation tasks.

MoCo-v2 is a sophisticated SSL framework that leverages a dynamic queue of negative keys, momentum updates, and careful temperature scaling to learn high-quality visual representations from unlabeled data. The pre-trained $\text{Dira}_{\text{MoCo-v2}}$ model will be tested on classification and segmentation tasks.

2 Literature Review

Over the years, Convolutional Neural Networks (CNNs) have seen tremendous improvements. However, while these methods have proven to be quite effective in certain scenarios, CNNs alone tend to fall short when analyzing data that is poorly labeled. Ker et al. [2] not only provide a good basis for understanding the applications of CNNs in the field of medical image analysis, but also present this problem of unlabeled data. As a possible solution to some of these problems, SSL, has been a topic of much discussion in recent years in the domain of medical image analysis, as pointed out by Krishnan and Rajpurkar [3]. They discuss the problems associated with data that is not sufficiently labeled, and how much of the image data in the field of medicine unfortunately falls into this category. They suggest that a highly effective way of dealing with this problem is by using SSL, which DiRA is based around. This idea is further backed up by Azizi et al. [4], who showed that SSL, while underused in the medical field, is not only effective in image prediction with unlabeled data, but that it is also robust, and can be applied to many different topics. DiRA employs three main strategies: Discriminative Learning, Restorative Learning, and Adversarial Learning. Discriminative Learning is a cemented strategy in image recognition, helping significantly with the task of transfer learning. Taher et al. [5] outlined this method and its successful application to medical data, which DiRA drew inspiration from. Restorative Learning uses the strategy of image context restoration to improve model performance. The idea, as described in its relation to medical image analysis by Chen et al. [6], is that a model is trained on a set of images, where it is then given another set of images of the same nature that is partially incomplete, and the model will attempt to rebuild the image. By doing this, weights can be assigned to the model more accurately, and the model generally performs better. The final main strategy used in DiRA is Adversarial Learning, which was presented by Lowd and Meek [7], and acts as a way to improve robustness of models. Additionally, image preprocessing plays a large factor in the success of a model. Many techniques exist to alter images in the hopes of creating a model that holds up better under inconsistent circumstances. One of such methods is called “image cutout”, described by DeVries and Taylor [8], where models are fed images that are partially complete. DiRA uses strategies like this to improve overall performance.

3 Methodology

3.1 DiRA approach

As stated before, DiRA consists of three branches. The discriminative branch consists of two backbone twin encoder networks: a regular encoder network and either a momentum encoder or a regular one that shares weights with the first encoder. Before passing the input to the network for processing, image augmentation is applied to introduce noise. After this, they are given to the encoders to generate feature vectors, which are then passed through two projection heads each corresponding to each encoder. These projection heads map the inputs in a normalized space, ensuring that they lie within a hypersphere with a unit radius. To guide the learning, a discriminative loss function is used that quantifies the compatibility between these vectors.

The restorative branch, sharing an encoder with the discriminative branch, introduces a decoder in order to map the distorted images back to their original form. The loss function for this component computes the similarities between the original input and the auto-generated ones on a pixel level. Minimizing the restoration loss ensures that the distorted samples, after being processed through the encoder and decoder, are brought as close as possible to the original samples - effectively reconstructing the original content from the distorted inputs.

The adversarial branch plays a ‘min-max game’ with the encoder and decoder by trying to predict which of the inputs are real images and which are generated by the decoder. On one hand, it helps the encoder to extract more meaningful features from the inputs. On the other hand, it stimulates the decoder to reproduce more realistic outputs. This is done by introducing an adversarial discriminator, the architecture of which can be seen in Table 1.

Finally, during the joint learning, all three loss functions are used, weighted by coefficients that dictate the significance of each component within the network.

Table 1: Architecture of adversarial discriminator

#	Layers	Filters	Kernel	Stride	Output Shape	# Total Parameters
1	Input				[(224, 224)]	
2	Conv2d	64	3	2	(112, 112)	
3	Conv2d	128	3	2	(56, 56)	
4	Conv2d	256	3	2	(28, 28)	
5	Conv2d	512	3	1	(28, 28)	
6	Output	1	3	1	(28, 28)	
						1,554,433

3.2 Rational for choosing DiRA for medical domain

Choosing DiRA for the medical domain can be justified based on several factors:

- DiRA’s dual branches, discriminative and reconstructive, allow it to learn important features. Discriminative learning captures detailed patterns in medical images, while reconstructive learning preserves vital medical information.

- DiRA can be adapted to handle various medical imaging modalities, such as X-rays, MRIs, and CT scans. This flexibility allows it to effectively process different types of medical data, enhancing its applicability in research and practice.

- DiRA’s ability to learn high-level features and relationships within the data can significantly enhance the performance of different medical tasks such as segmentation and classification. Its adversarial learning mechanism helps in fine-tuning the features for specific objectives, improving the model's overall effectiveness.

- DiRA's ability to work with limited labeled data reduces the need for extensive manual annotation, which can be time-consuming and costly.

3.3 Modifications

We approached three problems: classification, segmentation, and transfer learning. Specifically, for classification, we employed two different approaches—one involving traditional classification and another utilizing pre-training and fine-tuning on a separate target dataset. For each of these, we applied different techniques to improve or evaluate performance.

For classification, we altered the image augmentation process in order to make the DiRA model more robust, and to increase performance. This was done by examining what was already in place, analyzing what was missing, and trying different configurations of augmentations until we reached the best performance possible.

For transfer learning, we executed several preprocessing steps on target dataset and incorporated data augmentations described in section 6.1.1.

For segmentation, we integrated the CLAHE filter into our preprocessing pipeline to enhance image contrast, aiming to improve the model's ability to delineate anatomical structures more precisely.

4 Data

This experiment utilizes the ChestX-ray14 [9], DRIVE [10] and a private colorectal cancer liver metastases (CRLM) CT scan datasets. ChestX-ray14 is a collection of medical imaging assets, containing 112,120 frontal-view-xray images collected from 30,805 distinct patients. The data collection spanned from 1992 to 2015 which offers a varied and rich dataset. The images are text labeled with the following diseases: Atelectasis, Cardiomegaly, Effusion, Infiltration, Mass, Nodule, Pneumonia, Pneumothorax, Consolidation, Edema, Emphysema, Fibrosis, Pleural_Thickening, and Hernia. The labels mined from text radiological reports using Natural Language Processing techniques [11]. The DRIVE dataset is a collection of 40 color fundus images used for retinal vessel segmentation, half of which are used for training and the other half for testing. The images, including 7 with abnormal pathologies, were sourced from a diabetic retinopathy screening program in the Netherlands. Captured with a Canon CR5 non-mydriatic 3CCD camera at a resolution of 584x565 pixels, the dataset provides a ground-truth for performance evaluation through expert manual segmentations [10]. The CRLM dataset had 640 patients pre- and post-treatment CT scans. The goal is to predict treatment response, based on RECIST (Response Evaluation Criteria in Solid Tumours) criteria, reflecting the percentage change in the maximal axial diameter: complete response, partial response, stable disease, and progressive disease [12]. Original CT scans underwent segmentation to capture the region of interest which is liver area.

The original dataset consists of liver binary masks derived from pre- and post-treatment scans of 640 patients. The DiRA model was setup such that the images are automatically subject to distortion techniques including resizing, cropping, colour jittering, Gaussian blur, horizontal flipping, and their own custom distortion technique centred around adding noise to the images. Decorrelating and whitening data is not used for images as they produce substantial covariance matrices [13].

Two scripts were developed in the preprocessing and organization of medical imaging data for this research. The first script performs data transformation and structuring. Its primary objective is to transform the medical imaging test file by converting disease labels associated with each image into a binary encoded format as provided in the DiRA training text file. Initiated with the list of medical conditions, such as Atelectasis and Cardiomegaly, it processes these labels into binary representations based on the presence or absence of the disease in a predefined list. The script then parses a dataset, extracting both the image filenames and their associated disease labels, subsequently constructing dictionary. With the list of images curated, every image filename is associated with its corresponding disease labels, which are subsequently transformed into their binary encoded counterparts.

The subsequent script processes the datasets for segmentation training; where testing and validation images need to be in separate folders as per the DiRA implementation. The script distributes the images into distinct training and testing directories. It performs the process by categorizing images based on predefined lists into separate training and testing directories. By referencing these lists, the script discerns the appropriate destination for each image. Once the destination is determined, images are transferred from the main image folder to their respective folder.

4.1 Augmentations

The DiRA model uses a variety of transformations provided by the Torchvision Transforms package. The entirety of these transformations can be viewed in the "transforms.py" file provided on the DiRA GitHub page [14]. Below are the transformations they take described in detail:

- First, each image is copied twice, resulting in three identical images we will call *Image_1*, *Image_2*, and *Image_3*
- Both images are resized to 224x224 pixels, ignoring the bottom 20% of the image as it is unnecessary for making predictions (usually just empty space)
- From here, *Image_3* is converted to greyscale, and the pixel values are normalized

Image_1 and *Image_2* then go through the following transformations:

- Colour jittering is performed at a probability of 0.8
- A custom Gaussian Blur is implemented at a probability of 0.5. This is functionally the same as a regular Gaussian Blur, but it allows for certain parameters to be automatically inferred rather than them having to be explicitly stated
- Both images are flipped on the horizontal axis with a probability of 0.5

From here, the Python file checks to see what training mode the user has specified it to be. If the mode is "Di", then this indicates that we are only training a discriminative model, and *Image_1* and *Image_2* are converted to tensors and returned by themselves. If the mode is not "Di" (i.e. it is "DiR" or "DiRA"), then a random number *random* is chosen to be between 0 and 1, and more transformations are applied to *Image_1* and *Image_2* as follows:

- if *random* is less than or equal to 0.3, cut-out is performed in a loop that iterates up to 10 times. On each iteration, a new random number between 0 and 1 is chosen. If it is less than 0.1, then the loop breaks. Cut-out is performed on each loop iteration as follows:
  - Some small rectangular region of the image is randomly selected
  - The pixel values in this region are set to be 0
- If *random* is greater than 0.3 and less than or equal to 0.35, a sort of reverse cut-out is performed with the same loop structure:
  - A copy of the image is made, and all pixel values are set to be 0
  - Some small rectangular region of the original image is selected
  - The pixel values for that region are copied over to the blank image, effectively restoring a portion of the image
- If *random* is greater than 0.35 and less than or equal to 0.65, shuffling is performed 10 times:
  - A copy of the image is made
  - Two small rectangular regions of both images are selected such that the regions do not overlap
  - Region 1 of the copied image is copied to region 2 on the original image, and region 2 of the copied image is copied to region 1 on the original image
- If *random* is greater than 0.65, no transformations take place

Then, *Image_1*, *Image_2*, and *Image_3* are set to be tensors, and returned. Then, they go through the regular image training process. So, in essence, every image is copied twice, and then all three images go through some number of augments, where they are then fed into the model for training.

5 Implementation Details

DiRA (written in Python, primarily taking advantage of Torchvision) was implemented using a server provided by the Queen's University L1nna labratory's GPU computing cluster, as training a DiRA model requires the use of multiple GPUs. We decided to use the pre-processed ChestX-ray14 dataset for initial training, which was easily downloaded via the link on the main DiRA GitHub page. However, before the model could be trained on the data, we noticed that the versions were not provided for any of the packages listed in the provided requirements.txt file. Since the DiRA project was published in 2022, this meant that many of the packages had undergone some changes that caused their code to be unable to run. This was remedied by using versions of each package that would have been available around the time of initial development. We also found that on our GPU cluster, training was very slow. It was recommended that the model train for 1200 epochs, but this would have taken far too much time, and so we had to train it on only 100 epochs. Putting this aside, the DiRA model was trained by closely following the provided instructions, and the trained model was successfully saved for future testing. Next, we navigated to the Benchmark Transfer Learning GitHub page [15] for testing and comparison to other existing models. After more version control with the required packages, DiRA was tested and compared to other models.

5.1 Altering Augmentations

In the Benchmark Transfer Learning code, they implement a small number of augmentations via the Albumentations package [16]. However, these are only present for segmentation tasks, and even then, the usage is quite sparse. We examined all augmentations available through this package (which boasts the most extensive catalogue of augmentations) and implemented the ones that were applicable to our classification task. By experimenting with different values and orientations of augmentations, we settled on a number of *additions* to the existing augmentations present in the DiRA code. The following were added for *all* training modes ("Di", "DiR", and "DiRA") in the order listed:

- CLAHE (Contrast Limited Adaptive Histogram Equalization) at a probability of 0.6
  - Causes regions of the image to appear more distinct
  - Common in training medical images
- Emboss at a probability of 0.6
  - Causes specific regions of am image to pop out, similar to CLAHE
- FancyPCA at a probability of 0.6
  - Subtle sort of image distortion, focusing on minor colour distortion
- Channel Shuffle at a probability of 0.5
  - Shuffles the colour channels around
  - Implemented directly after the existing "colour jittering" augment, which adds another layer of complexity to the colour of the image
- Sharpen at a probability of 0.6
- Either Gaussian Blur, Glass Blur, Random Fog, of Zoom Blur, each at a probability of 0.6
  - All work slightly differently. Having different kinds of blurs adds a level of robustness to the model

The following destructive augments were added for *only* the "DiR" and "DiRA" modes:

- Either Gaussian Noise, ISO Noise, or Spatter, each at a probability of 0.6
  - All of these apply a similar style of noise to the entire image
  - These were all implemented in a very subtle manner in order to keep the image from becoming too noisy
- Random Gamma at a probability of 0.8
  - Slightly changes the gamma value for an image
- Random Gravel at a probability of 0.5
  - Places small circles over top of the image
- Random Rain at a probability of 0.5
  - Simulates rain drops on the image
- Random Snow at a probability of 0.5
  - Brightens small circular regions in the image
- Random Shadow at a probability of 0.5
  - Adds triangular shadows to the image
- Defocus at a probability of 0.8
  - Blurs a circular region

5.1.1 Transfer Learning

For transfer learning, we followed similar steps as the initial DiRA implementation. However, to adapt the model to our task, we created a custom data loader file for our private dataset. We then fine-tuned the DiRA model for our specific goal outlined in Section 6.1.2, adjusting parameters as needed. Afterwards, we evaluated the fine-tuned model's performance.

5.2 Segmentation

For the segmentation task we needed to ensure that the DiRA model, with its robust feature set, could be successfully transferred for use in segmentation. To accomplish this, the code within engine.py was modified to initialize a U-Net with ImageNet pre-trained weights as the backbone.

Our modification involved loading the DiRA model's weights into this U-Net architecture. Due to differences in the naming conventions and structure of the model states, we scripted a solution to parse the state dictionary of the pre-trained model, adjusting key names by stripping away unnecessary prefixes and removing any keys that were not applicable to U-Net.

After pruning the state dictionary, we loaded it into our U-Net model. This step was crucial as it allowed us to leverage the detailed feature extraction abilities learned by DiRA for the segmentation task at hand. The weights were transferred with an understanding that not every key would match perfectly, hence the choice to load the state with the strict parameter set to false.

6 Experimental design

6.1 Classification

Classification was evaluated on the ChestX-ray14 dataset. This dataset was foundational for the DiRA model, and it is also what is most documented, making it the best choice. The training process incorporated approximately 75,000 samples from the ChestX-ray14 dataset and another 11,000 for validation, making it quite a substantial dataset.

To evaluate classification, we calculated the AUC score (area under the ROC curve) over 14 trials (using different test data on each trial), as this is a reliable way of verifying the quality of a model. We compare the AUC scores between the base DiRA model, the MoCo-v2 model, and the new DiRA model with augments added per the earlier description.

6.1.1 Transfer learning on a CRLM dataset

To conduct further experimentation we pre-trained DiRA on ChestXray-14 dataset and further fine-tuned it on CRLM dataset

As mentioned in Section 4, the original dataset consists of liver binary masks derived from pre- and post-treatment scans of 640 patients. Before model training, a series of data preprocessing steps were undertaken. Firstly, tumor areas were extracted from liver segmentations, considering morphological attributes encompassing both shape and size characteristics. These attributes often reveal essential indicators related to the tumor's behavioral traits.

Subsequently, maximal axial diameters were calculated, aligning with the RECIST guidelines that categorize tumors based on this parameter. To adhere to these guidelines, tumors with diameters less than 15mm were excluded from the analysis. Following this exclusion, the dataset was refined to include 590 patients.

To enhance the model's focus on the region of interest, the images underwent cropping, ensuring square inputs for analysis. Given the variability in dimensions across patients in the NIfTI format, interpolation was applied to standardize and align the multiple slices of CT scans.

Post these preprocessing steps, the slice containing the maximal axial diameters was extracted for training purposes.

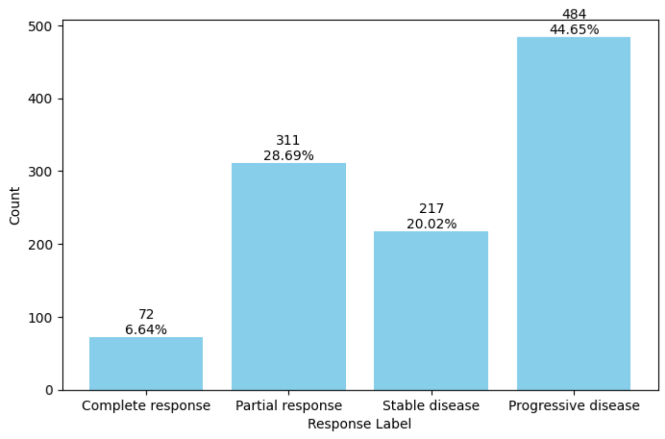

Figure 1: Response Count and Proportion

Finally, basic augmentations were applied to images to enrich the dataset. The initial dataset exhibited significant class imbalance, with only 8 patients having a complete response out of a total of 590. Nevertheless, given the authors' claim that DiRA can adeptly handle imbalanced data, we opted to intentionally preserve the dataset's inherent imbalance for further evaluation of DiRA. Figure 1 shows the final count and proportions of each class. Final dataset consisted of 1084 images which were separated to 80-10-10 splits for training, testing and validation. After preprocessing steps, DiRA was again pre-trained and fine-tuned on this dataset. We pre-trained two DiRA models: one with the original augmentations and the other one with our custom transformations. Each model was trained for 100 epochs.

6.2 Segmentation

The segmentation task was evaluated on the DRIVE (Digital Retinal Images for Vessel Extraction) dataset. The DRIVE dataset was chosen due to its relevance in retinal vessel segmentation, which provides a challenging and distinct context.

Our experimental setup began with the pre-training of the DiRA model on the ChestX-ray 14 dataset. This pre-training was intended to instill a fundamental understanding of anatomical structures within the model. Subsequently, the model was fine-tuned on the DRIVE dataset to adapt its capabilities to the task of retinal vessel segmentation.

We extended our comparison to include two additional models: the MoCo-v2 and ImageNet models, which served as benchmarks for performance evaluation. For the MoCo-v2 and ImageNet models, we used the pre-trained models provided by the Segmentation Models [17] library in python.

In an effort to enhance segmentation performance, we integrated the CLAHE (Contrast Limited Adaptive Histogram Equalization) filter into our pipeline. CLAHE is known for its ability to improve the visibility of important features in medical images by enhancing contrast. We applied this filter to investigate its potential benefits on the model's ability to outline retinal vessels more accurately.

The performance of each model was assessed using the Dice score, a statistical measure of similarity between the model's predictions and the ground truth annotations. This evaluation was carried out over multiple iterations, with and without the CLAHE filter, to quantify its impact on the model's performance. The mean Dice scores across these iterations were then calculated to provide a robust indicator of each model's segmentation capabilities.

7 Results

7.1 Classification

To get a baseline for classification, a very simple ResNet-50 model was trained on the ChestX-ray14 dataset. This model performed quite poorly, having a classification accuracy of 0.2315. This is a very straightforward model, so these results are not too surprising. In contrast, the AUC values for the base DiRA model ranged from 0.6956 to 0.9055, with a mean AUC of 0.7969. For MoCo-v2, the values varied between 0.6933 and 0.9261, with a mean AUC of 0.8003. Importantly, the AUC values for the DiRA model with added augments were between 0.6892 and 0.9372 with a mean AUC of 0.8042. The specific trial results can be seen in table 1, and a summary of scores can be seen in table 2. The obtained AUC values, all being higher than 0.5, demonstrate that our models learned successfully and performed markedly better than chance. Furthermore, classification with the DiRA model with added augments resulted the highest mean AUC between the three models, showing that by tactfully inserting image augments for the DiRA training process, its performance can be increased.

Table 1: AUC values over 14 trials

Trial	$\text{DiRA}_{\text{MoCo-v2}}$	$\text{DiRA}_{\text{MoCo-v2}}$ Augs	MoCo-v2
1	0.7511	0.7616	0.7590
2	0.8811	0.8778	0.8839
3	0.8199	0.8248	0.8225
4	0.6956	0.6892	0.6933
5	0.8210	0.8223	0.8211
6	0.7386	0.7430	0.7340
7	0.7165	0.7234	0.7184
8	0.8368	0.8410	0.8369
9	0.7464	0.7530	0.7504
10	0.8428	0.8475	0.8418
11	0.8426	0.8634	0.8569
12	0.8070	0.8148	0.8093
13	0.7516	0.7593	0.7512
14	0.9055	0.9372	0.9261

Table 2: Best vs. Mean Values

Model	Best AUC	Mean AUC
$\text{DiRA}_{\text{MoCo-v2}}$	0.9055	0.7969
$\text{DiRA}_{\text{MoCo-v2}}$ Augs	0.9372	0.8042
MoCo-v2	0.9261	0.8003

7.2 Classification with transfer learning

The transfer learning results indicate that our customized DiRA model performed slightly better than both the standard MoCo-v2 and the original DiRA, as reflected in both the best and mean AUC values of 0.9484 and 0.7868 accordingly. These findings mirror our observations from the ChestX-ray14 dataset classification, where the models consistently demonstrated strong accuracy, with a slight decrease in AUC values. Overall, both models showcased robust performance in a similar range. Tables 3 and 4 present specific values for each run.

Table 3: AUC values over 4 trials

Trial	$\text{DiRA}_{\text{MoCo-v2}}$	$\text{DiRA}_{\text{MoCo-v2}}$ Augs	MoCo-v2
1	0.8997	0.9484	0.9012
2	0.6839	0.7786	0.7959
3	0.6414	0.6762	0.6041
4	0.7452	0.744	0.744

Table 4: Best vs. Mean Values

Model	Best AUC	Mean AUC
$\text{DiRA}_{\text{MoCo-v2}}$	0.8997	0.721
$\text{DiRA}_{\text{MoCo-v2}}$ Augs	0.9484	0.7868
MoCo-v2	0.9012	0.7613

7.3 Segmentation

For the purpose of image segmentation, our analysis included a comparative study of the ImageNet, DiRA, and MoCo-v2 models with and without the CLAHE filter. The ImageNet model without the CLAHE filter achieved a Dice score of 79.15%, while with CLAHE, its performance improved to 80.09%. The DiRA model, pre-trained on the ChestX-ray14 dataset and fine-tuned with the CLAHE filter on the DRIVE dataset, reached a Dice score of 79.05% with CLAHE and 79.04% without it. This was significant given the stark differences between the ChestX-ray14 and DRIVE datasets in terms of image characteristics and segmentation tasks. The ChestX-ray14 dataset, focusing on thoracic pathology, is vastly different from the retinal vessel structures in the DRIVE dataset. Despite these differences, the DiRA model showed notable adaptability and effectiveness in a new domain.

The MoCo-v2 model also underwent similar testing. With the CLAHE filter, it achieved a Dice score of 79.42%, and without CLAHE, it scored slightly lower at 79.32%. These results highlight the varying impacts of the CLAHE filter on different models and the overall effectiveness of each model in adapting to the segmentation tasks.

Table 5: Dice scores for image segmentation on the DRIVE dataset

Model	Dice Score (%)
ImageNet with CLAHE	80.09
ImageNet without CLAHE	79.15
DiRA with CLAHE	79.05
DiRA without CLAHE	79.04
MoCo-v2 with CLAHE	79.42
MoCo-v2 without CLAHE	79.32

8 Discussion

While the code for the DiRA models based on Barlow Twins and SimSiam was unavailable, the results produced from DiRA based on MoCo-v2 alone were still quite significant. Classification using base DiRA and MoCo-v2 models was able to be performed without significant difficulty - while there were inconsistencies with package versions and the documentation was unclear at times, there were no major issues with the code itself.

The two base models (DiRA and MoCo-v2) performed very well and achieved high AUC values in classification, but MoCo-v2 demonstrated a slight advantage over DiRA. In the original paper, DiRA had performed better than MoCo-v2, but this could be in part due to the lower number of epochs that were used in training due to hardware limitations. However, by adding in augmentations to the DiRA model, it ended up performing better than the base DiRA and MoCo-v2 over 100 epochs, showing that DiRA can indeed be competitive and that there are still optimizations to be made. Further testing should be done to see how this model with augmentations performs over a larger number of epochs.

The fact that including additional augmentations to the images for the training of the classification model resulted in significantly increased performance shows that there is still room for DiRA to be improved. While the augmentations added were fairly exhaustive, it is likely that new methods of this nature will be introduced in the coming years and should be incorporated to keep the model performing at its best. Extending from DiRA, the results shown here are evidence that augmentations as a whole should see more use in cutting edge models, and should ideally be implemented and trained similarly to other hyperparameters.

In the segmentation task, the performance metrics shows that DiRA, ImageNet, and MoCo-v2 performed similarly, with ImageNet attaining a slight edge at 80.09%. The DiRA model, both with and without the CLAHE filter, delivered a consistent performance, hovering around the 79% mark.

It is particularly revealing that the inclusion of the CLAHE filter resulted in a performance boost for all models, with ImageNet benefiting the most, suggesting that contrast enhancement plays a significant role in model discrimination capacity. However, the marginal differences in Dice scores across the board call for a deeper examination of the underlying factors contributing to these results.

Moreover, the DiRA model's adaptability and generalization can be put to the test by applying it to a variety of other benchmark datasets within the medical imaging field. Such cross-domain validations would not only strengthen the evidence of DiRA's versatility but also help in identifying any potential weaknesses or areas of improvement.

For future work, fine-tuning the DiRA model's hyperparameters could provide the necessary refinement for precision enhancement. Pre-training the model on more datasets could also significantly improve its performance. Additionally, experimenting with more sophisticated image preprocessing techniques, beyond the techniques discussed in this paper, might provide further improvements.

9 Contributions

By leveraging image augmentation techniques, we were able to improve the performance of all models in classification and segmentation tasks, including DiRA - a SOTA SSL model. The fact that these augmentations increased the performance of the models points to a greater problem of the under-utilization of augmentations. Augmentations are seeing much development, and new tools that are highly effective are becoming readily available. Thus, their implementation should be considered and explored in great detail for all models operating in the field of image recognition. We use DiRA as an example of this, and hope to extend this mindset moving forward both in our future work, and in the work of others in the field of machine learning.

10 Conclusion

DiRA was successfully reproduced, with classification and segmentation performing roughly how the initial paper had described. We were able to improve upon the model's performance in both classification and segmentation tasks by leveraging modern image augmentation methods. This highlights the possibility that even further work could be done to improve upon this model, and that avenues such as hyperparameter tuning should be explored in more depth. DiRA's performance compared to MoCo-v2 showcases its strength in handling medical images, and is a step towards overcoming difficulties associated with SSL.

References

[1] Fatemeh Haghighi, Mohammad Reza Hosseinzadeh Taher, Michael B Gotway, and Jianming Liang. Dira: Discriminative, restorative, and adversarial learning for self-supervised medical image analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20824–20834, 2022.

[2] Justin Ker, Lipo Wang, Jai Rao, and Tchoyoson Lim. Deep learning applications in medical image analysis. *IEEE Access*, 6:9375–9389, 2018.

[3] Rayan Krishnan and Pranav Rajpurkar. Self-supervised learning in medicine and healthcare. *Natural Biomedical Engineering*, 6:1346–1352, Aug 2022.

[4] Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zachary Beaver, Jan Freyberg, Jonathan Deaton, Aaron Loh, Alan Karthikesalingam, Simon Kornblith, Ting Chen, Vivek Natarajan, and Mohammad Norouzi. Big self-supervised models advance medical image classification, 2021.

[5] Mohammad Reza Hosseinzadeh Taher, Fatemeh Haghighi, Ruibin Feng, Michael B. Gotway, and Jianming Liang. A systematic benchmarking analysis of transfer learning for medical image analysis, 2021.

[6] Liang Chen, Paul Bentley, Kensaku Mori, Kazunari Misawa, Michitaka Fujiwara, and Daniel Rueckert. Self-supervised learning for medical image analysis using image context restoration. *Medical Image Analysis*, 58:101539, Dec 2019.

[7] Daniel Lowd and Christopher Meek. Adversarial learning. KDD '05, page 641–647. Association for Computing Machinery, 2005.

[8] Terrance DeVries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout, 2017.

[9] Le Lu Zhiyong Lu Mohammadhadi Bagheri Xiaosong Wang, Yifan Peng and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. page 2097–2106, Dec 2017.

[10] J. Staal, M.D. Abramoff, M. Niemeijer, M.A. Viergever, and B. van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, 2004.

[11] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M. Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. `https://paperswithcode.com/dataset/chestx-ray14`, 2017. Accessed: 2023-09-05.

[12] Lawrence H Schwartz, Saskia Litière, Elisabeth De Vries, Robert Ford, Stephen Gwyther, Sumithra Mandrekar, Lalitha Shankar, Jan Bogaerts, Alice Chen, Janet Dancey, et al. Recist 1.1—update and clarification: From the recist committee. *European journal of cancer*, 62:132–137, 2016.

[13] Andrej Karpathy. Cs231n winter 2016: Lecture 5: Neural networks part 2, 2016.

[14] Fatemeh Haghighi, Mohammad Reza Hosseinzadeh Taher, Michael B Gotway, and Jianming Liang. Dira. `https://github.com/fhaghighi/DiRA`, 2023.

[15] Mohammad Reza Hosseinzadeh Taher. Benchmarktransferlearning. `https://github.com/MR-HosseinzadehTaher/BenchmarkTransferLearning`, 2023.

[16] Mikhail Druzhinin. Albumentations. `https://github.com/albumentations-team/albumentations`, 2023.

[17] Pavel Iakubovskii. Segmentation models. `https://github.com/qubvel/segmentation_models`, 2019.