Report by- **Sunil Meena 170070025**
Under the guidance of- **Prof. Amit sethi**

SELF-SUPERVISED MAMMOGRAMS CLASSIFICATION

1 INTRODUCTION

Deep learning relies on ultra-large datasets, which might be a tough task in the medical domain. Due to limited data available for a particular medical domain, data privacy issues and due to the high skill resource requirement for classifying these images, creating extremely large standardized datasets for medical images is a huge challenge. It would be a lot better if a model could produce annotations and classification results with just the limited we have in the medical domain. Here, Unsupervised methods comes to our rescue. Moreover, Doersch et al. [10], Wang and Gupta [39] and Agrawal et al. have explored a novel paradigm for unsupervised learning called self-supervised learning. The main idea is to exploit different labelings that are freely available besides or within visual data, and to use them as intrinsic reward signals to learn general-purpose features. The features obtained with these approaches have been successfully transferred to classification and detections tasks, and their performance is very encouraging when compared to features trained in a supervised manner. We will be using a novel self-supervised task, the Jigsaw puzzle reassembly problem, which builds features that yield high performance when transferred to detection and classification tasks. Solving Jigsaw puzzles can be used to teach a system that an object is made of parts and what these parts are. The association of each separate puzzle tile to a precise object part might be ambiguous. However, when all the tiles are observed, the ambiguities might be eliminated more easily because the tile placement is mutually exclusive.

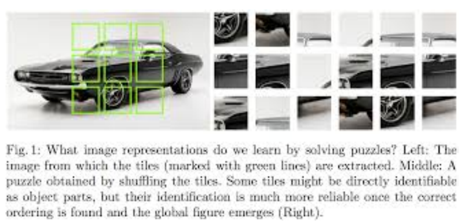
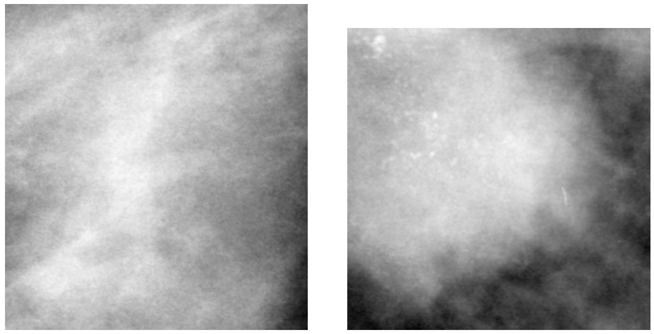


Figure 1: Image puzzle

2 DATASET AND METHODS

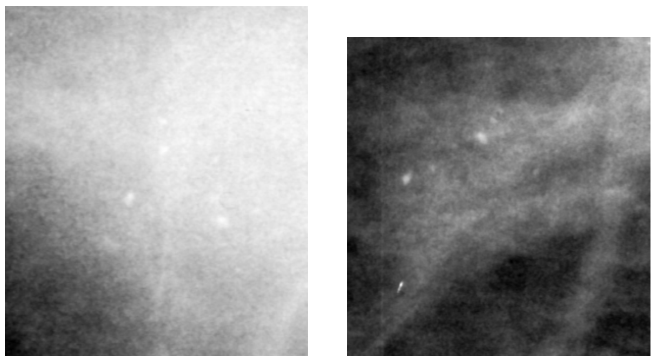
2.1 DATASET

The experimental setup is shown in Figure 4. In this project, I have used CBIS-DDSM (Curated Breast Imaging Subset of DDSM) provided by The Cancer Imaging Archive. This CBIS-DDSM is an updated and standardized version of the Digital Database for Screening Mammography (DDSM). The DDSM is a database of 2,620 scanned film mammography studies. It contains normal, benign, and malignant cases in Mass and Calcification categories with verified pathology information. The scale of the database along with ground truth validation makes the DDSM a useful tool in the development and testing of decision support systems. The CBIS-DDSM collection includes a subset of the DDSM data selected and curated by a trained mammographer. The images have been decompressed and converted to DICOM format. The image data for this collection is structured such that each participant has multiple patient IDs. For example, participant 00038 has 10 separate patient IDs which provide information about the scans within the IDs (e.g. Calc-Test_P_00038_LEFT_CC, Calc-Test_P_00038_RIGHT_CC_1). This makes it appear as though there are 6,671 patients according to the DICOM metadata, but there are only 1,566 actual participants in the cohort. Below are the samples for different categories in the dataset.



(a) Benign type (b) Malignant type

Figure 2: Mammograms classified into Mass



(a) Benign type (b) Malignant type

Figure 3: Mammograms classified into Calcification

2.2 Methods

The original images were pre-processed and resized to 255x255 and 510x510 for the study. To pass the image through the network it is divided into 9 patches. These patches are labeled form 1 to 9. Order of the patches for training is obtained from permutations of these numbers. For size 255x255, patches of 75x75 are formed. These patches are resized and transformed into 64x64 size, then passed through 9 different convolution pipes ending with a FC layer. Output from these FC layers is merged into one FC layer. After another FC layer, classification result is obtained from classifier. Classes in the classifier represents allowed permutations for the puzzle pieces i.e. the patches of the image. The number of classes can be varied to get more accurate classification results. It's downside is that for large number of classes it takes much more time to learn. After learning the features from puzzle solving, final layer is changed to 4 output FC. CrossEntropyLoss is used as the loss functions for both the parts of the training.

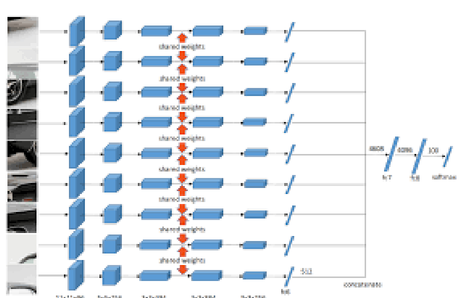


Figure 4: Jigsaw Technique architecture

3 Results and Discussion

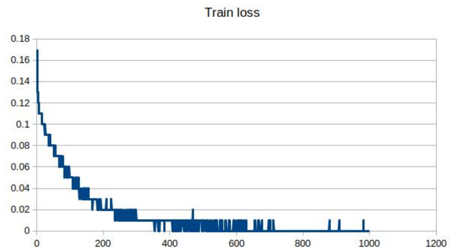
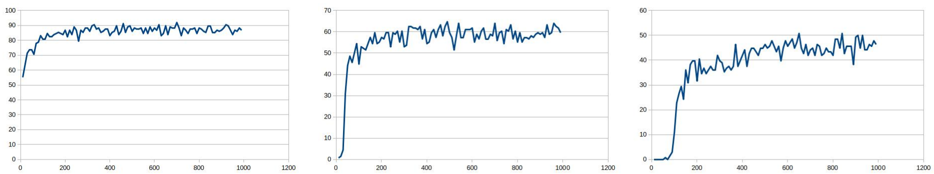


Figure 5: Training loss for class-10



(a) For class-10 (b) For class-100 (c) For class-500

Figure 6: Test accuracy for puzzle solving till 1000 epochs of training

(a) For class-10 (b) For class-100 (c) For class-500

Figure 7: Test accuracy for mammograms classification till 100 epochs of training

Training has been done for 10, 100, 500 number of classes for the puzzle solving task. In the above figure, we can see the trend in average loss per epoch. Same trend is observed for all the three cases. Loss value is higher in the initial epochs for the cases with more classes. Test accuracy should be lowest for class-500 for 1000 epochs of training as the model tries to learn more detailed features. It can be seen in figure 6 as well. Because of having more detailed features, class-500 should give better results for mammograms classification task. In figure 7, it can be observed that the Average test accuracy for (c) is closer to 50% than others. It can be improved further with more training.

4 Conclusions and Future Work

Tasks like judging the mammograms can take days as there aren't many mammographers. Self-supervised learning can make the difficult task of classification and annotation way easier even with less data. Model presented in the report isn't reliable enough yet to be used for official purposes. It needs to be improved further for classification task. For annotation, we can use Faster-RCNN which can also provide category prediction. To incorporate less data problem into the model, feature generation should be done by using convolution layer weights of the siamese network, learnt from jigsaw puzzle solving, instead of VGG-16.

REFERENCES

Code inspired from https://github.com/bbrattoli/JigsawPuzzlePytorch
Mehdi Noroozi and Paolo Favaro, Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles, cs.CV, (Aug, 2017)