End-to-end Training for Whole Image Breast Cancer Diagnosis using An All Convolutional Design

Li Shen
Icahn School of Medicine at Mount Sinai
New York, NY 10029
li.shen@mssm.edu, shenli.sam@gmail.com

Abstract

We develop an end-to-end training algorithm for whole-image breast cancer diagnosis based on mammograms. It requires lesion annotations only at the first stage of training. After that, a whole image classifier can be trained using only image level labels. This greatly reduced the reliance on lesion annotations. Our approach is implemented using an all convolutional design that is simple yet provides superior performance in comparison with the previous methods. On DDSM, our best single-model achieves a per-image AUC score of 0.88 and three-model averaging increases the score to 0.91. On INbreast, our best single-model achieves a per-image AUC score of 0.96. Using DDSM as benchmark, our models compare favorably with the current state-of-the-art. We also demonstrate that a whole image model trained on DDSM can be easily transferred to INbreast without using its lesion annotations and using only a small amount of training data.

Code and model availability: https://github.com/lishen/end2end-all-conv

Key Insights

- A patch classification network can be converted into an end-to-end trainable whole image network by modifying the input size and adding top layers.
- The heatmap layer generated by the patch classifier's output creates an information bottleneck in the whole image network and shall be removed to improve performance.
- The patch classifier is critical to the performance of the whole image classifier.
- All convolutional design is superior to a mix of convolutional and fully connected layers.

1 Introduction

With the rapid advancement in machine learning and especially, deep learning in recent years, there is a keen interest in the medical imaging community to apply these techniques to cancer screening. Recently, a group of researchers, along with the Sage Bionetworks and the DREAM community organized a challenge for competing teams to develop algorithms to improve breast cancer diagnosis using a large database of digital mammograms (DM) [1]. The main purpose of the challenge is to predict the probability that a patient will develop cancers within 12 months based on mammographic images. We also participated in the challenge [2] and obtained a receiver operating characteristic curve (AUC) score of 0.65, which ranked #12 on the leaderboard [3]. However, much can still be done to further improve the result.

Mammography based breast cancer diagnosis is a challenging problem that cannot be simply treated as a normal image classification task. That is because the cancerous status of a whole image is determined by a few small regions that need to be identified. For example, a mammogram is typically in the size of 4000x3000 (height by width) pixels while the cancerous lesion or the region of interest (ROI) can be as small as 100x100 pixels. Resizing a large mammogram to 224x224 – a common choice used by the image classification filed – will likely make the ROI hard to detect and/or classify. If a mammographic database comes with the ROI annotations for all images, then the diagnostic problem can be conveniently solved as an object detection and classification problem that has already been well studied in the computer vision field. For example, the region-based convolutional neural network (R-CNN) approach [4] and its variants [5]–[7] can be applied here. Many of the published works [8]–[17] assume the databases under study are fully annotated with ROIs, thus the developed models cannot be transferred to another database that lacks ROI annotations. It is unrealistic to require all databases to be fully annotated due to the time and monetary costs to obtain ROI annotations, which require expertise from radiologists. For a survey of the public databases, see [18]. If we want to build a breast cancer diagnostic system for production, we will have to consider the situation where a few datasets are annotated with ROIs while most datasets are only annotated at whole image level.

That is exactly the situation that the participants faced at the DM challenge [1] during the competitive phase. Participants were allowed to use any public databases to develop their models. However, the DM challenge database itself does not contain ROI annotations. Although the DM challenge boasts a large database with more than 640,000 images from over 86,000 women, its dataset is highly imbalanced: with only 2000 or so positive cases in the public train set, which significantly reduces the effective sample size. Therefore, directly training a classification model on such a dataset from scratch is unlikely to give good performance.

In this study, we propose an approach that utilizes a fully annotated database to train a model to recognize localized patches. After that, the model can be converted into a whole image classifier, which can be trained end-to-end without ROI annotations. The approach is based on an idea that we independently formed (see discussion in [2]) during the final stage of the DM competition, which we did not have time to implement before the competition ended. Our original entry [2] to the DM challenge used a patch classification network, built on a fully annotated public database, to predict for all the overlapping patches of a mammogram to generate a so called heatmap. A random forest classifier is then used on top of the heatmap to produce the final label. However, we observed a significant performance drop when we applied the model to the DM challenge data because the patch classifier was developed on a public database that contains a different color profile from the challenge data. It was difficult to transfer the patch classifier onto the challenge data since it does not have ROI annotations. While the whole image classifier is not end-to-end trainable because it involves a sliding window step. We later realized that end-to-end training can be achieved by using the convolutional property to change the input size from patch to whole image, create larger feature maps, and then add additional convolutional layers on top to produce the final labels. Surprisingly, we found that a similar idea was employed by the top-performing team [19] after the final results came out. Encouraged by their success, we want to continue to investigate this line of methods. We propose an all convolutional design to construct the whole image classifier by simply stacking convolutional blocks on top of the patch classifier, which provides very competitive results. We also present a pipeline to build a whole image classifier from scratch and discuss the pros and cons of some important choices.

Notice that our method is somewhat related to two recent publications [20], [21] in breast cancer diagnosis. They both claim to be an implementation of multi-instance learning (MIL) and use modified cost functions to satisfy the MIL criterion. Both studies develop whole image models that can be end-to-end trained but they completely ignore the existence of ROI annotations in databases and train models using only image level labels. It seems they focus more on the top layers to summarize predictions from local patches than the patch classifiers to make accurate predictions. However, we will show that the patch classifier is critical to the performance of the whole image classifier.

The manuscript is organized as follows. In Section 2, we present our method to convert from a patch classifier to a whole image classifier and the network training algorithm. Section 3 has two parts. In Section 3.1, we will develop whole image classifiers from scratch and evaluate them on a public database that has ROI annotations. In Section 3.2, we will transfer the whole image models developed in Section 3.1 to another database using only image level labels. In Section 4, we provide some discussion of the results and future works. We hope, by virtue of this study, we can provide further insights into this promising method for whole image classification for the medical imaging community.

2 Methods

2.1 Converting a classifier from recognizing patches to whole images

To perform classification or segmentation on large, complex images, a commonly used approach is to first develop a classifier to recognize smaller patches, and then use the classifier to scan the whole image using a sliding window. For example, this approach has been exploited to win the competition in automated detection of metastatic breast cancer [22] and to segment neuronal membranes in microscopic images [23]. However, here we want to utilize a patch classifier to initialize the weights of a whole image classification network so

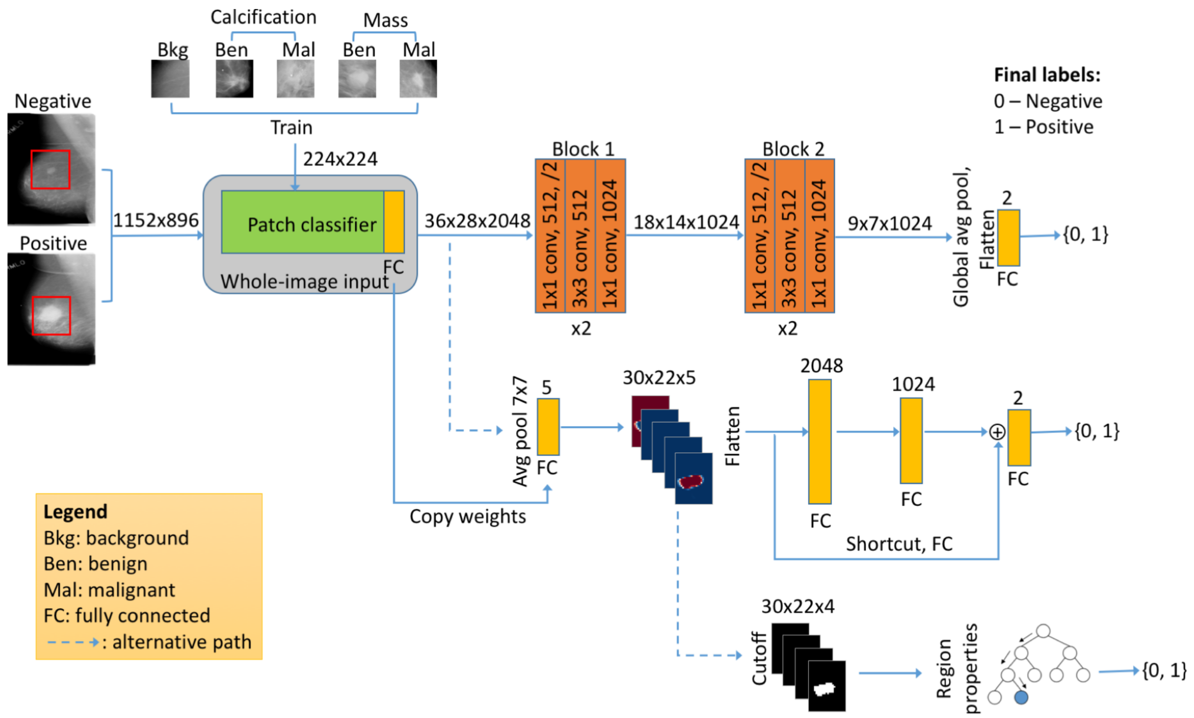

Figure 1: The deep learning structure for converting a patch classifier into a whole image classifier by adding convolutional layers on top. Shown is an example using two residual blocks with the same structure of [512-512-1024] x 2. Two alternative strategies are also presented: one is to add heatmaps and FC layers on top and the other is to add a random forest classifier on top of the heatmap.

that the latter can be trained without ROI annotations. In other words, we want the whole image classifier to be finetuned in an end-to-end fashion without the explicit reliance on a patch image set. This may not seem to be obvious on the surface but can actually be achieved by exploiting the properties of a convolutional neural network (CNN). The key idea of a CNN is to learn a set of filters with small sizes and apply them to a much larger input by using weight sharing, which greatly reduces the number of parameters to learn. Each time a new convolutional layer is added to the network, the top most layer effectively uses all previous layers to construct a set of new filters that perform more complex transformations (and also have larger effective receptive fields) than its precursors. By this reasoning, we can add on top of a patch classification network a new convolutional layer and turns the patch network itself into part of the new filters of the top layer. Although the patch network is trained to recognize patches, the properties of convolution allow us to change the input from patches to whole images. And by doing this modification, a convolutional operation in the top layer becomes equivalent to applying the patch classifier on all patches of the whole image in one forward propagation and then apply the convolutional operation on the patch classifier's outputs. Indeed, we can design any kind of operations on top of the patch classifier's output and eventually connect them with image level labels. For an illustration of this method, see Fig. 1. Notice that variable input size is a feature that is supported by most major deep learning frameworks [24]–[27] and therefore can be easily implemented. The advantage of this method is that we now have a network that takes whole images as input and image level labels as output, cutting the need for a patch image set.

This training method can find many applications in medical imaging. Use breast cancer diagnosis as an example. Databases with ROI annotations are rare and expensive to obtain. The largest public database for mammograms – The Digital Database for Screening Mammography (DDSM) [28] – contains thousands of images with pixel-level annotations, which can be exploited to construct a meaningful patch classifier. Once the patch classifier is converted into a whole image classifier, it can be finetuned on another database using only whole image level labels. This way we can significantly reduce the requirement for ROI annotations.

2.2 Network structures for top layers

For patch classification, we can use any networks – such as the 16-layer VGG networks (VGG16) [29] and the 50-layer residual networks (Resnet50) [30] – that are already proven to be excellent for image classifications. Therefore, we want to focus more on the structure of the top layers in this study. We propose an all convolutional design that makes no use of the fully connected (FC) layers. Briefly, we first remove the last few layers of a patch network until the last convolutional layer. We then add convolutional layers, such as the residual and VGG blocks, on top of the last convolutional layer so that the feature map reaches a proper size. Lastly, we add a global average pooling layer and an output to complete the network. Notice that we deliberately discard the output layer of the patch network. That is because the patch classifier typically has a small output, which may create an "information bottleneck" that prevents the top layers from fully utilizing the information provided by the patch network. To understand it better, let's use Resnet50 as an example. Assume it is trained to recognize a patch into one of the five categories of: background, malignant mass, benign mass, malignant calcification and benign calcification. Its last convolutional layer has a dimension of 2048. If we add a residual block of dimension of 512 after the patch classifier's output, it creates a bottleneck structure of "2048-5-512" in-between the patch network and the top layers. While a network without the patch classifier's output will have a structure of "2048-512", allowing the top layers to use more information from the patch network.

In contrast to our approach, the top performing team of the DM challenge utilizes the patch classifier's output layer to construct a so called heatmap to represent the likelihood of a patch on an input image being one of the five categories [19]. In the first version of this manuscript, we mistakenly used softmax activation on this heatmap to generate a probabilistic output for the top layers. This nonlinear transformation would certainly impede gradients flow. After confirming with the top performing team (personal communications), we remove the softmax activation from the heatmap to construct networks similar to theirs. However, as we implement the whole image classifiers, we find the totally inactivated heatmap may cause the top layers to be badly initialized, leading to divergence. We hypothesize that is because the fully unbounded values can make the top layers saturate too early. Therefore, we use relu (which truncates the negative values) on the heatmaps instead in this study and find the convergence to improve. Another important difference between our method and the top performing team's method is the use of FC layers. We argue that FC layers are a poor fit for image recognition because they require a convolutional layer's output to be flattened, which eliminates all the spatial information. Our design is fully convolutional and preserves spatial information at every stage of the network. We will compare the two different strategies in the following sections.

2.3 Network training

There are two parts in training a whole image classifier from scratch. The first part is to train a patch classification network. We compare the networks that have their weights pretrained on the ImageNet [31] database with randomly initialized ones. Pretraining may help in speedup learning as well as improving generalization of the networks. For a pretrained network, notice that the bottom layers represent primitive features that tend to be invariant across datasets. While the top layers represent higher representations that are more related to the final labels and therefore need to be trained more aggressively. This demands a higher learning rate for the top layers than the bottom layers. However, layer-wise learning rate adjustment is not available in Keras [24] – the deep learning framework that we use for this work. Therefore, we develop a 3-stage training strategy that freeze the parameter learning for all layers except the last one and progressively unfreeze the parameter learning from top to bottom and decrease the learning rate at the same time. The details of this training strategy is as follows:

1. Set learning rate to 1e-3 and train the last layer for 3 epochs.
2. Set learning rate to 1e-4, unfreeze the top layers and train for 10 epochs, where the top layer number is set to 46 for Resnet50, 11 for VGG16 and 13 for VGG19.
3. Set learning rate to 1e-5, unfreeze all layers and train for 37 epochs.

In the above, an epoch is defined as a sweep through the train set. The total number of epochs is 50 and early stopping (set to 10 epochs) is used if training does not improve the validation loss. For randomly initialized networks, we use a learning rate of 1e-3. Adam [32] is used as the optimizer and the batch size is set to 32. We also adjust the sample weights within a batch to keep the classes balanced.

The second part is to create and train a whole image classifier from the patch classifier. It is done by first altering the input size from patch to whole image, which proportionally increases the feature map size for every convolutional layer. In this study, we fix the patch size to be 224x224 and the whole image size to be 1152x896. Use Resnet50 as an example. For the patch network, the last convolutional layer has a feature map of size 7x7. While for the whole image network, the same layer's feature map size becomes 36x28. In our all convolutional design, we add on top two additional residual blocks each with a stride of 2 to reduce the feature map size to 9x7 before the global average pooling and softmax output. Alternatively, 7x7 average pooling can be applied to the convolutional layer to produce a feature map of size 30x22. The weight matrix of the output layer of the patch network can then be copied and applied onto this feature map to generate the heatmap (the same size of 30x22) but here we replace the softmax with relu to facilitate gradients flow. Similar to what the top performing team did [19], the heatmap (after max pooling) is first flattened and then two FC layers and a shortcut connection are added on top before output. Similar to the patch network training, we also develop a 2-stage training strategy to avoid unlearning the important features at the patch network. After some trial-and-errors, we decide to use smaller learning rates than usual to prevent the training from diverging. The details of the 2-stage training are as follows:

1. Set learning rate to 1e-4, weight decay to 0.001 and train the newly added top layers for 30 epochs.
2. Set learning rate to 1e-6, weight decay to 0.01 and train all layers for 20 epochs.

Due to memory constraint, we use a small batch size of 2 for whole image training. The other parameters are the same as the patch classifier training.

To make it easier to convert a patch classifier into a whole image classifier, we calculate the pixel-wise mean for the mammograms on the train set and use this value for pixel-wise mean centering for both patch and whole image training. No other preprocessing is applied. To compensate for the lack of sample size, we use the same data augmentation on-the-fly for both patch and whole image training. The followings are the random image transformations we use: horizontal and vertical flips, rotation in [-25, 25] degrees, shear in [-0.2, 0.2] radians, zoom in [0.8, 1.2] ratio and channel shift in [-20, 20] pixel values.

3 Results

3.1 Developing patch and whole image classifiers on DDSM

3.1.1 Setup and processing of the dataset

The DDSM [28] contains digitized images from scanned films but uses a lossless-JPEG format that is obsolete. We use a modernized version of the database called CBIS-DDSM [33] which contains images that have already been converted into the standard DICOM format. Our downloaded dataset on Mar 21, 2017 from CBIS-DDSM's website contains the data for 1249 patients or 2584 mammograms, which represents a subset of the original DDSM database. It includes the cranial cardo (CC) and media lateral oblique (MLO) views for most of the screened breasts. Using both views for prediction shall provide better result than using each view separately. However, we will treat each view as a standalone image in this study due to the limitation in sample size. Our purpose is to predict the malignancy status for an entire mammogram. We perform an 85-15 split on the dataset into train and test sets based on the patients. For the train set, we further set aside 10% of the patients as the validation set. The splits are done in a stratified fashion so that the positive and negative patients have the same proportions in the train, validation and test sets. The total numbers of images in the train, validation and test sets are: 1903, 199 and 376, respectively.

The DDSM database contains the pixel-level annotation for the ROIs and their pathology confirmed labeling: benign or malignant. It further contains the type of each ROI, such as calcification or mass. Most mammograms contain only one ROI while a small portion contain more than one ROIs. We first convert all mammograms into PNG format and resize them into 1152x896. We then create several patch image sets by sampling image patches from ROIs and background regions. All patches have the same size of 224x224. The first dataset (referred to as S1) is from patches that are centered on each ROI and a random background patch from the same mammogram. The second dataset (S10) is from 10 sampled patches around each ROI with a minimum overlapping ratio of 0.9 and the same number of background patches from the same mammogram. The third dataset (S30) is created the same way as S10 but we increase the number of patches to 30 for both ROI and background per mammogram. According to the annotations of the ROIs, all patches are classified into five different categories: background, calcification-benign, calcification-malignant, mass-benign and mass-malignant.

3.1.2 Development of patch classifiers

To train a network to classify a patch into the five categories, we use three popular convolutional network structures: the Resnet50 [30] and the VGG16 and VGG19 [29]. We train the networks on the S10 and S30 sets using the 3-stage training strategy for 50 epochs in total. On the S1 set, we increase the total number of epochs to 200 and adjust the numbers for each training stage accordingly. We evaluate the models using test accuracy of the five classes. The results are summarized in Table 1. On the S1 set, both randomly initialized and pretrained Resnet50 models achieve very high accuracies but the pretrained network converges much faster: cutting down the number of epochs by half. Notice that the S1 set is intrinsically easier to classify than

Table 1: Test accuracy of the patch classifiers using the Resnet50, VGG16, and VGG19. #Epochs indicates the epoch when the best validation accuracy has been reached.

Model	Pretrained	Patch set	Accuracy	#Epochs
Resnet50	N	S1	0.97	198
Resnet50	Y	S1	0.99	99
Resnet50	N	S10	0.63	24
Resnet50	Y	S10	0.89	39
VGG16	Y	S10	0.84	25
VGG19	Y	S10	0.79	15
Resnet50	Y	S30	0.91	23
VGG16	Y	S30	0.86	22
VGG19	Y	S30	0.89	24

the S10 and S30 sets. On the S10 set, the pretrained Resnet50 performs much better than the randomly initialized Resnet50: a 0.26 difference in test accuracy. We therefore conclude that pretraining can help us train networks faster and produce better models. We use pretrained networks for the rest of the study. On both S10 and S30 sets, Resnet50 outperforms VGG16 and VGG19. VGG16 performs better than VGG19 on the S10 dataset but the order is reversed on the S30 set.

3.1.3 Converting patch to whole image classifiers

We now convert the patch classifiers into whole image classifiers by testing many different configurations for the top layers. We evaluate different models using the per-image AUC scores on the test set. We first test the conversion based on the Resnet50 patch classifiers. The results are summarized in Table 2. Notice that in the original residual network design [30], the authors increased the dimension for each residual block from the previous block by two times to compensate for the reduction in feature map size. This avoids creating "bottlenecks" in computation. However, we are not able to do the same here due to memory constraint. In our first test, we use two residual blocks with the same dimension and a bottleneck design (see [30]) of [512-512-2048] without repeating the residual units. This gives an AUC score of 0.85 for the Resnet50 trained on the S10 set and 0.63 for the Resnet50 trained on the S1 set, a large 0.22 discrepancy in score. We hypothesize that since the S10 set is 10x larger than the S1 set, it contains much more information about the variations of benign and malignant ROIs and their neighborhood regions. This information helps a lot in training a whole image classifier to locate and classify cancer-related regions for diagnosis. We then focus only on patch classifiers trained on the S10 and S30 sets for the rest of the study. We further vary the design of the residual blocks by reducing the dimension of the last layer in each residual unit to 1024, which allows us to repeat each residual unit twice without exceeding memory constraint, i.e. two [512-512-1024]x2 blocks. This design slightly increases the score to 0.86. We also reduce the dimensions of the first and second blocks and find the scores to drop by only 0.02-0.03. Therefore, we conclude that the dimensions for the newly added residual blocks are not critical to the performance of the whole image classifiers.

We then test the conversion based on the VGG16 patch classifier trained on the S10 set. The newly added VGG blocks all use 3x3 convolutions with batch normalization (BN). The results are summarized in Table 3. We find that the VGG structure is more likely to suffer from overfitting than the residual structure. However, this can be alleviated by reducing the model complexity of the newly added VGG blocks. To illustrate this, we plot the train and validation losses for two VGG configurations during training (Fig. 2): one is two VGG blocks with the same dimensions of 512 and 3x repetitions and the other has the same dimensions but no repetition; the first VGG structure contains more convolutional layers and therefore has higher complexity than the second one. It can be seen that the first VGG network has difficulty in identifying a good local minimum and suffers badly from overfitting. While the second VGG network has smoother loss curves and smaller differences between train and validation losses. We further reduce the dimensions of the VGG blocks and find the scores to decrease by only a small margin. This is in line with the results on the Resnet50 based models. Overall, the Resnet50 based whole image classifiers perform better than the VGG16 based ones (Tables 2 & 3). In addition, the Resnet50 based models seem to achieve the best validation score earlier than the VGG16 based models. To understand whether this performance difference is caused by the patch network on the bottom or the newly added top layers, we add two residual blocks on top of the VGG16 patch network to create a "hybrid" model. It gives a score of 0.81, which is in line with the VGG16 based networks but a few points below the best Resnet50 based models. This suggests the patch network part is more important

Table 2: Per-image test AUC scores of the whole image classifiers using the Resnet50 as patch classifiers. #Epochs indicates the epoch when the best validation score has been reached. → indicates score change from using softmax to relu on heatmaps.

Add residual blocks on top					
Patch set	**Block1**	**Block2**	**Single-model AUC**	**Augmented AUC**	**#Epochs**
S1	[512-512-2048] x 1	[512-512-2048] x 1	0.63	NA	35
S10	[512-512-2048] x 1	[512-512-2048] x 1	0.85	NA	20
S10	[512-512-1024] x 2	[512-512-1024] x 2	0.86	0.88	25
S10	[256-256-256] x 1	[128-128-128] x 1	0.84	NA	25
S10	[256-256-256] x 3	[128-128-128] x 3	0.83	NA	17
S10	[256-256-512] x 3	[128-128-256] x 3	0.84	NA	48
S30	[512-512-1024] x 2	[512-512-1024] x 2	0.81	NA	49
S30	[256-256-256] x 1	[128-128-128] x 1	0.84	NA	40
Add heatmap and residual blocks on top					
S10	[512-512-1024] x 2	[512-512-1024] x 2	0.80	NA	47
S10	[64-64-256] x 2	[128-128-512] x 2	0.81	NA	41
Add heatmap and FC layers on top					
	Heatmap pool size	**FC1** **FC2**			
S10	5x5	64 32	0.67→0.73	NA	28
S10	2x2	512 256	0.68→0.72	NA	47
S10	1x1	2048 1024	0.74→0.65	NA	43
Probabilistic heatmap + random forest classifier					
S10	#trees=500, max depth=9, min samples split=300		0.73	NA	

Table 3: Per-image test AUC scores of the whole image classifiers using the VGG16 (S10) and VGG19 (S30) as patch classifiers. #Epochs indicates the epoch when the best validation score has been reached. → indicates score change from using softmax to relu on heatmaps.

Add VGG blocks on top (* is residual block)					
Patch set	**Block1**	**Block2**	**Single-model AUC**	**Augmented AUC**	**#Epochs**
S10	512 x 3	512 x 3	0.71	NA	47
S10	512 x 1	512 x 1	0.83	0.86	44
S10	256 x 3	128 x 3	0.78	NA	30
S10	256 x 1	128 x 1	0.80	NA	35
S10	128 x 1	64 x 1	0.82	NA	46
S10	*[512-512-1024] x 2	*[512-512-1024] x 2	0.81, 0.85[1]	0.88[1]	46
S30	512 x 1	512 x 1	0.75	NA	15
S30	128 x 1	64 x 1	0.75	NA	1
Add heatmap and FC layers on top					
	Heatmap pool size	**FC1** **FC2**			
S10	5x5	64 32	0.66→0.71	NA	26
S10	2x2	512 256	0.71→0.68	NA	27
S10	1x1	2048 1024	0.78→0.69	NA	50

[1]Result obtained from extended model training (See text for more details).

than the top layers to whole image classification performance.

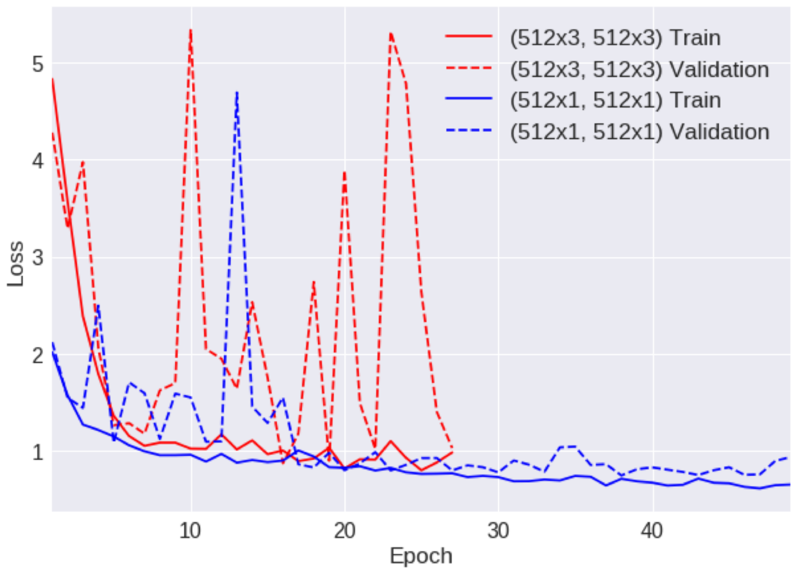

Figure 2: Train and validation loss curves of two VGG structures: one is more complex than the other and suffers from overfitting.

Encouraged by the performance leap going from the S1 to the S10 set, we choose two residual configurations and two VGG configurations and add them on top of the Resnet50 and the VGG19 patch classifiers trained on the S30 set, respectively. We expect to obtain even further performance gains. Surprisingly, the performance decreases for both Resnet50 and VGG19 based models (Tables 2 & 3). Indeed, overfitting happens for the VGG19 based model training at early stage and the validation loss stops improving further. It is clear that a good patch classifier is critical to the performance of a whole image classifier. However, merely increasing the number of sampled patches does not necessarily lead to better whole image classifiers. We leave to future work to study how to sample patches more efficiently and effectively to help building better image classifiers.

We now test the configurations that are inspired by the top performing team [19]: add a heatmap followed by two FC layers and a shortcut connection on top of the patch classifier (Fig. 1). We use the Resnet50 and VGG16 patch classifiers trained on the S10 set. We choose from a few different filter sizes for the max pooling layer after the heatmap and adjust FC layer sizes accordingly so that it gradually decreases the layer size until output. We also add a shortcut between the flattened heatmap and the output, implemented by an FC layer. The results are summarized in Tables 2 & 3. In the first version of the manuscript, we used heatmaps with softmax activation to generate probabilistic outputs and we found the max pooling layer to be destructive to the heatmap features and therefore harm the scores. In this version, we use relu on heatmaps instead and find the trend to be almost reversed: more pooling leads to better score. This could be related to the difficulty in training when heatmap and FC layers are used. Overall, the performance scores of the two different activations are on par with each other. Therefore, the heatmap activation is not critical to the whole image classifier's performance. Obviously, image classifiers from this design underperform the ones from the all convolutional design. Notice that this implementation is not an exact replicate of the work in [19]. They have used a modified VGG network by using 6 instead of 5 VGG blocks. Their design is more like a hybrid of the two designs compared here by using an additional convolutional block followed by FC layers.

We also want to find out whether the heatmaps are of any use or simply create a bottleneck between the patch network and the top layers. We add a heatmap (with relu) on top of the Resnet50 patch classifier and then add two residual blocks with [512-512-1024]x2 design. This network gives a score of 0.80 (Table 2), which is lower than the same design without the heatmap. To exclude the possibility that the top residual blocks may be overfit due to the small heatmap as input, we also add two residual blocks with reduced size: [64-64-256]x2 and [128-128-512]x2 on top of the heatmap, which slightly improves the score to 0.81. We conclude that heatmaps shall be removed to facilitate information flow in training whole image networks.

Finally, we test the same strategy as [2], [22]: set a cutoff to binarize the heatmaps; extract regional features; and train a random forest classifier based on the features (Fig. 1). The Resnet50 trained on the S10 set is used here as the patch classifier. For heatmaps, we use softmax to obtain probabilistic outputs to help setting cutoffs. We use four different cutoffs – 0.3, 0.5, 0.7, 0.9 and combine the features. We use a random forest

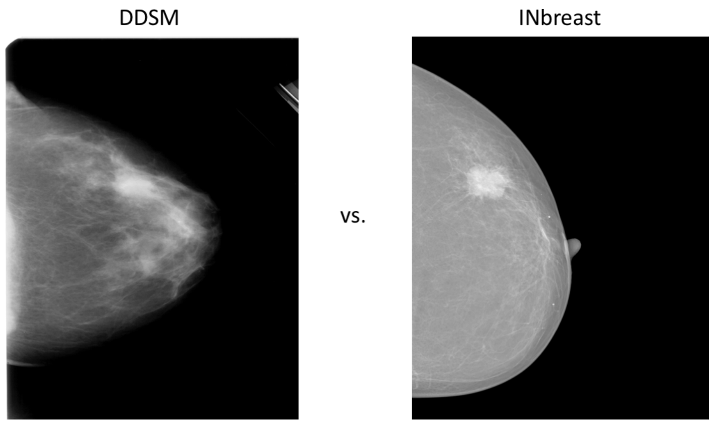

Figure 3: Comparison of two example mammograms from DDSM and INbreast.

classifier with 500 trees. This gives a test score of 0.73 (Table 2), which seems to be in line with our score of 0.65 in the DM challenge. There are two major reasons for the increase of the score in this study: 1. the patch classifier in this study has been given more training time and produces better accuracy; 2. the train and test sets are both from the same database. Clearly, this method performs worse than the end-to-end trained all convolutional networks.

3.1.4 Augmented prediction and model averaging

We pick a few models and use inference-level augmentation by doing horizontal and vertical flips to create four predictions per model and take an average. Three best performing models: Resnet50 with two [512-512-1024]x2 residual blocks, VGG16 with two 512x1 VGG blocks and VGG16 with two [512-512-1024]x2 residual blocks on top (hybrid) are selected. We perform extended training on the hybrid model and improve the single-model AUC score to 0.85. The reason of doing that will be explained in section 3.2.2. The augmented predictions for the three models improve the AUC scores from 0.86→**0.88**, 0.83→0.86 and 0.85→**0.88**, respectively (Tables 2 & 3). The average of the three augmented predictions gives an AUC score of **0.91**. Notice that in [3], they created four network models each with augmented predictions and predicted for both CC and MLO views for each breast and took an average of all predictions, which is more aggressive than the model averaging used in this study. We deliberately avoid doing too much model averaging so that we can better understand the pros and cons of different network structures.

3.2 Transfer learning for whole image classification on INbreast

3.2.1 Setup and processing of the dataset

Once a whole image classifier has been developed, we can finetune it on another database without using ROI annotations. This is a key advantage offered by an end-to-end trained classifier. The INbreast [18] dataset is another public database for mammograms. It is more recent than the DDSM and contains full-field digital images as opposed to digitized images from films. These images have different color profiles from the images of DDSM, which can be visually confirmed by looking at two example images from the two databases (Fig. 3). Therefore, this is an excellent source to test the transferability of a whole image classifier from one database to another. The INbreast database contains 115 patients and 410 mammograms including both CC and MLO views. We will treat the two views as separate samples due to sample size limitation. It includes the BI-RADS readings for the images but lacks biopsy confirmation. Therefore, we manually assign all images with BI-RADS readings of 1 and 2 as negative samples; 4, 5 and 6 as positive samples; and ignore BI-RADS readings of 3 since it has no clear designation of negative or positive class. This excludes 12 patients or 23 mammograms from further analysis. Notice that the categorization based on the BI-RADS readings makes the INbreast dataset inherently easier to classify than the DDSM dataset. This is because two mammograms with different BI-RADS readings are already visually discernible according to radiologists, which biases the labeling. We perform a 70-30 split on the dataset into train and validation sets based on the patients in a stratified fashion. The total numbers of images in the train and validation sets are 280 and 107, respectively. We use exactly the same processing steps on the INbreast images as the DDSM images.

3.2.2 Effectiveness and efficiency of transfer learning

Table 4: Transfer learning efficiency with different train set sizes. Shown are per-image validation AUC scores.

#Patients	#Images	Resnet50	VGG16	VGG-Resnet Hybrid
20	79	0.78	0.87	0.89
30	117	0.78	0.90	0.90
40	159	0.82	0.90	0.93
50	199	0.80	0.93	0.93
60	239	0.84	0.95	0.91

Although the INbreast database contains ROI annotations, we simply ignore them to test the transferability of a whole image classifier. We directly finetune the whole image networks on the train set and evaluate the model performance using per-image validation AUC scores. Two best models – the Resnet50 + two [512-512-1024]x2 residual blocks and the VGG16 + two 512x1 VGG blocks – are used for transfer learning. We use Adam [32] as the optimizer and set the learning rate at 1e-6; the number of epochs at 200 and the weight decay to be 0.01. The finetuned Resnet50 based model achieves a score of 0.84. Surprisingly, the finetuned VGG16 based model achieves a score of 0.92, better than the Resnet50 based models. Yaroslav argued in [19] that the VGG structure is better suited than the residual structure for whole image classification because the residual networks reduce the feature map sizes too aggressively to damage the ROI features at the first few layers. Based on this, the underperformance of the Resnet50 based models is likely due to the few bottom most layers, not the top residual blocks. To validate that, we use the hybrid model, which has the VGG16 patch classifier at the bottom and two residual blocks on top, to finetune on the INbreast dataset. This hybrid model achieves a very high score of **0.95**, which proves the point. However, if the VGG structure is indeed better than the residual structure as the bottom layers, how does Resnet50 beat VGG16/19 on the DDSM data? We reason that is because the VGG networks need to be trained longer than the residual networks to reach their full potentials. This is in line with the observation that the VGG16 based whole image classifiers achieve the best validation scores on DDSM at later stage than the Resnet50 based ones (Tables 2 & 3). Our choice of 50 epochs on DDSM is mainly driven by computational resource limitation. Since the residual networks are powered by BN and shortcuts to speed up training, they are able to converge better than the VGG networks within the same 50 epochs. To prove that, we perform another run of model training for the hybrid model on the DDSM data with 200 additional epochs. The model improves the test score from 0.81 to 0.85 (Table 3), which is as good as the best Resnet50 based models. We did not perform additional training for the other VGG16 based models due to computational constraint.

We also want to find out how much data is required to finetune a whole image classifier to reach satisfactory performance. This has important implications in practice since obtaining labels, even at the whole image level, can be expensive. We sample a subset with 20, 30, …, 60 patients from the train set for finetuning and evaluate the model performance on the same validation set (Table 4). With as little as 20 patients or 79 images, the VGG16 based and the hybrid models can achieve scores of 0.87 and 0.89, respectively. The scores seem to quickly saturate as we increase the train set size. We hypothesize that the "hard" part of the learning is to recognize the textures of the benign and malignant ROIs while the "easy" part is to adjust to different color profiles. This quick adjustment can be a huge advantage for the end-to-end trained whole image networks. In future works, a whole image classifier can be finetuned to make predictions on another database with only a small amount of training data. This greatly reduce the burden of train set construction.

Finally, with augmented prediction, the VGG16 based model improves the validation score from 0.92→0.94 and the hybrid model improves the score from 0.95→**0.96**. The average of the two augmented models gives a score of **0.96**. Including the Resnet50 based model in model averaging does not improve the score.

4 Discussion

We have shown that accurate whole-image breast cancer diagnosis can be achieved with a deep learning model trained in an end-to-end fashion that is independent from ROI annotations. The network can be based on an all convolutional design that is simple yet powerful. It can be seen that high-resolution mammograms are critical to the accuracy of the diagnosis. However, large image size can easily lead to an explosion of memory requirement. If more GPU memory becomes available in the future, we shall return to this problem and train our models with larger image sizes or even use the original resolution without downsizing. This will provide much more details of the ROIs and can potentially improve the performance.

The decrease of the scores from the models based on the S10 to the S30 set is a surprise. Yet it indicates the intricacy of training a whole image classifier. More research is needed to make the whole image training more robust against divergence and overfitting, especially when the train set is not large. Patch sampling can also be made more efficient by focusing more on the difficult cases than the easy ones.

Our result supports Yaroslav's argument [19] that the VGG networks are more suitable than the residual networks for breast cancer diagnosis. However, we also demonstrate the superiority of the residual networks to the VGG networks in several aspects. The only problem with the residual structure is the first few layers that may destroy the fine details of the ROIs. In future works, we can modify the original residual network to make the first few layers less aggressive in reducing the feature map sizes. That shall lead to improved performance for the residual networks.

Computational environment

The research in this study is carried out on a Linux workstation with 8 CPU cores and a single NVIDIA Quadro M4000 GPU with 8GB memory. The deep learning framework is Keras 2 with Tensorflow as the backend.

Acknowledgements

We would like to thank Gustavo Carneiro and Gabriel Maicas for providing comments on the manuscript; Quan Chen for discussion on the use of INbreast data.

References

[1] A. D. Trister, D. S. M. Buist, and C. I. Lee, "Will Machine Learning Tip the Balance in Breast Cancer Screening?," *JAMA Oncol*, May 2017.

[2] L. Shen, "Breast cancer diagnosis using deep residual nets and transfer learning," 2017. [Online]. Available: https://www.synapse.org/#!Synapse:syn9773182/wiki/426912. [Accessed: 23-Aug-2017].

[3] "The Digital Mammography DREAM Challenge - Final Ranking of Validation Round." [Online]. Available: https://www.synapse.org/#!Synapse:syn4224222/wiki/434546. [Accessed: 23-Aug-2017].

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, Washington, DC, USA, 2014, pp. 580–587.

[5] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[7] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks," *arXiv:1605.06409 [cs]*, May 2016.

[8] A. R. Jamieson, K. Drukker, and M. L. Giger, "Breast image feature learning with adaptive deconvolutional networks," in *Proc. SPIE*, 2012, vol. 8315, pp. 831506-831506–13.

[9] J. Arevalo, F. A. González, R. Ramos-Pollán, J. L. Oliveira, and M. A. G. Lopez, "Convolutional neural networks for mammography mass lesion classification," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 797–800.

[10] G. Carneiro, J. Nascimento, and A. P. Bradley, "Unregistered Multiview Mammogram Analysis with Pre-trained Deep Learning Models," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 652–660.

[11] N. Dhungel, G. Carneiro, and A. P. Bradley, "Automated Mass Detection in Mammograms Using Cascaded Deep Learning and Random Forests," in *2015 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2015, pp. 1–8.

[12] M. G. Ertosun and D. L. Rubin, "Probabilistic visual search for masses within mammography images using deep learning," in *2015 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2015, pp. 1310–1315.

[13] A. Akselrod-Ballin, L. Karlinsky, S. Alpert, S. Hasoul, R. Ben-Ari, and E. Barkan, "A Region Based Convolutional Network for Tumor Detection and Classification in Breast Mammography," in *Deep Learning and Data Labeling for Medical Applications: First International Workshop, LABELS 2016, and Second International Workshop, DLMIA 2016, Held in Conjunction with MICCAI 2016, Athens, Greece, October 21,*

2016, Proceedings, G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, J. P. Papa, J. C. Nascimento, M. Loog, Z. Lu, J. S. Cardoso, and J. Cornebise, Eds. Cham: Springer International Publishing, 2016, pp. 197–205.

[14] J. Arevalo, F. A. González, R. Ramos-Pollán, J. L. Oliveira, and M. A. Guevara Lopez, "Representation learning for mammography mass lesion classification with convolutional neural networks," *Computer Methods and Programs in Biomedicine*, vol. 127, pp. 248–257, Apr. 2016.

[15] Daniel Lévy and A. Jain, "Breast Mass Classification from Mammograms using Deep Convolutional Neural Networks," *arXiv preprint arXiv:1612.00542*, 2016.

[16] N. Dhungel, G. Carneiro, and A. P. Bradley, "The Automated Learning of Deep Features for Breast Mass Classification from Mammograms," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham: Springer International Publishing, 2016, pp. 106–114.

[17] A. S. Becker, M. Marcon, S. Ghafoor, M. C. Wurnig, T. Frauenfelder, and A. Boss, "Deep Learning in Mammography: Diagnostic Accuracy of a Multipurpose Image Analysis Software in the Detection of Breast Cancer," *Invest Radiol*, Feb. 2017.

[18] I. C. Moreira, I. Amaral, I. Domingues, A. Cardoso, M. J. Cardoso, and J. S. Cardoso, "INbreast: Toward a Full-field Digital Mammographic Database," *Academic Radiology*, vol. 19, no. 2, pp. 236–248, Feb. 2012.

[19] Yaroslav Nikulin, "DM Challenge Yaroslav Nikulin (Therapixel)." [Online]. Available: https://www.synapse.org/#!Synapse:syn9773040/wiki/426908. [Accessed: 23-Aug-2017].

[20] W. Zhu, Q. Lou, Y. S. Vang, and X. Xie, "Deep Multi-instance Networks with Sparse Label Assignment for Whole Mammogram Classification," *arXiv:1705.08550 [cs]*, May 2017.

[21] Yoni Choukroun, Ran Bakalo, Rami Ben-Ari, Ayelet Askelrod-Ballin, Ella Barkan, and Pavel Kisilev, "Mammogram Classification and Abnormality Detection from Nonlocal Labels using Deep Multiple Instance Neural Network," presented at the Eurographics Workshop on Visual Computing for Biology and Medicine, 2017.

[22] D. Wang, A. Khosla, R. Gargeya, H. Irshad, and A. H. Beck, "Deep Learning for Identifying Metastatic Breast Cancer," *arXiv:1606.05718 [cs, q-bio]*, Jun. 2016.

[23] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 2843–2851.

[24] F. Chollet and others, *Keras*. GitHub, 2015.

[25] Martín Abadi *et al.*, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015.

[26] Y. Jia *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," *arXiv:1408.5093 [cs]*, Jun. 2014.

[27] T. Chen *et al.*, "MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems," *arXiv:1512.01274 [cs]*, Dec. 2015.

[28] Michael Heath, Kevin Bowyer, Daniel Kopans, Richard Moore, and W. Philip Kegelmeyer, "The Digital Database for Screening Mammography," in *Proceedings of the Fifth International Workshop on Digital Mammography*, 2001, pp. 212–218.

[29] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556 [cs]*, Sep. 2014.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015.

[31] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *Int J Comput Vis*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[32] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Dec. 2014.

[33] R. S. Lee, F. Gimenez, A. Hoogi, and D. Rubin, "Curated Breast Imaging Subset of DDSM," *The Cancer Imaging Archive*, 2016.