Supplementary Materials for "End-to-End Learning of Fused Image and Non-Image Features for Improved Breast Cancer Classification from MRI"

Gregory Holste[1] Savannah C. Partridge[2,3] Habib Rahbar[2,3] Debosmita Biswas[3]
Christoph I. Lee[2,3] Adam M. Alessio[1*]

[1]Michigan State University [2]University of Washington [3]Seattle Cancer Care Alliance

[*]aalessio@msu.edu

1. Methods

1.1. Non-Image Features

A full description of the 18 non-image features used alongside breast imaging can be seen in Table 1.

1.2. Fusion Model Variants

While we experimented with different fusion operations to join information from different modalities, we also conducted experiments to explore how best to optimize such a multimodal network. To this end, we constructed a variant of the *Learned Feature Fusion* architecture that added a fully-connected output layer with sigmoid activation from both the 512 learned features from the image encoder and 512 learned features from the non-image encoder. This allowed us to obtain three predictions of malignancy, one obtained from image-only features ($\hat{y}_i$), one from non-image-only features ($\hat{y}_n$), and one from fused image and non-image features ($\hat{y}_f$). The following two approaches represent two different ways of optimizing a network that fuses multimodal information.

While the original *Learned Feature Fusion* model minimized the cross-entropy (CE) loss between the predicted and known malignancy status, this version of *Learned Feature Fusion* permitted us to minimize the sum of three cross-entropy losses

$$\mathcal{L}_{sum} = CE(y, \hat{y}_i) + CE(y, \hat{y}_n) + CE(y, \hat{y}_f), \quad (1)$$

where $y \in \{0, 1\}$ is the known malignancy status of a given breast. We denote this approach *[L,L]-Fusion**.

Another training approach we employed was to optimize the three subnetworks (image encoder, non-image encoder, fusion parameters) independently. Specifically, after the forward pass, we compute each subnetwork's loss and backpropagate that loss only to the given subnetwork's learnable parameters. We denote this approach *[L,L]-Fusion*†.

1.3. Training Details

Model weights and biases were randomly initialized with PyTorch [6] default settings depending on the type of layer (e.g., convolution vs. linear/fully-connected). All models were trained with the Adam optimizer [4] with learning rate 1×10^{-4} and PyTorch's default parameters $\beta_1 = 0.9$ and $\beta_2 = 0.99$. We used a standard binary cross-entropy loss with class weights corresponding to inverse frequency in the training set (i.e., malignant cases were weighted 4.008 times more heavily than benign cases in the loss computation). We also applied label smoothing [9] with $\alpha = 0.1$ (using the formulation in [5]) to reduce confidence in predictions and encourage better model calibration. No learning rate scheduling was applied, and models were trained for up to 100 epochs. Training was terminated when the validation AUC did not increase for 25 epochs consecutively; model weights from the epoch with maximum validation AUC were then saved for later evaluation. Each model was trained on a single NVIDIA Tesla V100S GPU on the Michigan State University High Performance Computing Center.

All models that were trained on images used a ResNet50 [3] feature extractor with the first convolution operation modified to accommodate a single-channel input image (grayscale MIP). Models trained on images also used "online" data augmentation via the Albumentations library [1], where each image would be passed through the following pipeline of transformations, each occurring with probability 0.5: a horizontal flip, either a blur operation or random contrast shift, an elastic deformation [8] with $\alpha = 10$ and $\sigma = 5$, a random scaling within the range 0.8-1.2 followed by a resize back to 224×224, and a random rotation between -20 and 20 degrees. Lastly, models trained on images leveraged test-time augmentation (TTA) to improve generalization by averaging the model's predictions on five transformed versions of each test set image. The data augmentation pipeline for TTA was as follows: horizontal flip, blur or random contrast shift, and random rotation (each as

Table 1: Description of non-image features.

Non-Image Feature	Feature Type	Description
Age	Continuous	Patient's (integer) age at time of MRI study.
Breast Laterality	Binary	Whether the cropped MIP contains the left breast.
Breast Density	Ordinal	Mammographic breast density via BI-RADS assessment.
BPE	Ordinal	Breast parenchymal enhancement on the breast MRI study, as determined by a radiologist.
MRI Indication	Categorical	Clinical indication (e.g., screening or diagnostic evaluation) for MRI study.
MRI Software Version	Categorical	Vendor software used to generate 2D MIP image from DCE-MRI data.
MIP Max Intensity	Continuous	Maximum pixel intensity in cropped MIP image.
Pixel Dimensions	Continuous	Pixel Spacing DICOM field. Distance (*mm*) between adjacent pixels in original DICOM image.
MIP Height	Continuous	Height (*mm*) of cropped MIP before preprocessing.
MIP Width	Continuous	Width (*mm*) of cropped MIP before preprocessing.
Flip Angle	Continuous	Flip Angle DICOM field. Angle (degrees) to which magnetic vector is flipped from that of primary field.
Reconstruction Diameter	Continuous	Reconstruction Diameter DICOM field. Diameter (*mm*) of circular region containing all pixel data.
Precession Frequency	Continuous	Imaging Frequency DICOM field. Precession frequency (MHz) of the nucleus being targeted.
Echo Time	Continuous	Echo Time DICOM field. Time (*ms*) between the middle of excitation pulse and peak of resulting echo.
Repetition Time	Continuous	Repetition Time DICOM field. Time (*ms*) between the beginning of successive pulse sequences.
Echo Train Length	Continuous	Echo Train Length DICOM field. Lines in k-space acquired per excitation per image.
Field Strength	Binary	Magnetic Field Strength DICOM field. Field strength (Tesla) of MR magnet.
Shift Days	Continuous	Arbitrarily generated code used to anonymize the date of MRI study.

Basic description of all 18 non-imaging features used. Feature type describes the quality of each feature before preprocessing; after preprocessing, all continuous variables remained continuous and all other variables were converted to categorical variables and one-hot-encoded. MIP = maximum intensity projection, DICOM = Digital Imaging and Communications in Medicine.

described earlier). While these are random operations, all random number-generating seeds were fixed so that each model that used TTA was evaluated on the exact same set of (transformed) test images.

1.4. Statistical Tests

As described in the pROC [7] documentation for the *roc.test* function, we used a simple nonparametric test for differences in both AUC and specificity at 95% sensitivity. To compare models A and B by test AUC, we first establish their performance on the original test set with AUC_A and AUC_B, respectively. We then

1. Draw a stratified bootstrap sample of the test set (maintaining the exact numbers of malignant and benign cases observed in the test set),
2. Compute AUC for each model on the bootstrapped test set,
3. Repeat steps 1-2 for 5,000 iterations,
4. Compute test statistic $D = \frac{AUC_A - AUC_B}{s}$, where s is the standard deviation of the 5,000 differences in bootstrapped AUC between models A and B,
5. Compare D to a standard normal distribution to obtain a (two-tailed) significance level.

This procedure was chosen over the DeLong test because it can be applied to metrics other than AUC. While results are not shown, we found negligible differences in AUC hypothesis tests between the DeLong test and the bootstrap method described above.

1.5. Permutation Importance

As first described in the context of interpreting random forests [2], we used a permutation-based feature importance method to understand which non-image features are most influential to model predictions. First establishing an observed AUC on the original test set AUC_{base}, we would then

1. Randomly permute (shuffle) the values of only feature k in the test set,
2. Find AUC_{perm} by computing AUC on the permuted test data,
3. Compute feature importance

$$I = \left(\frac{-\left(AUC_{perm} - AUC_{base}\right)}{AUC_{base}} \right) * 100,$$

the percent decrease in test AUC upon permuting feature k,

4. Repeat steps 1-3 for 30 iterations,
5. Repeat steps 1-4 for all non-image features $k = 1, 2, \ldots, 18$.

Table 2: Breast cancer prediction results of *Learned Feature Fusion* model with different optimization approaches.

Model	Best Run		Five-Run Ensemble	
	AUC	**Specificity at 95% Sensitivity (%)**	**AUC**	**Specificity at 95% Sensitivity (%)**
[L,L]-Fusion	0.898 [0.885, 0.909]	49.1 [38.8, 55.3]	0.903 [0.891, 0.914]	50.3 [44.2, 59.0]
[L,L]-Fusion*	0.900 [0.888, 0.911]	49.6 [44.1, 58.5]	0.906 [0.895, 0.917]	55.2 [46.6, 60.2]
[L,L]-Fusion†	0.894 [0.882, 0.906]	46.7 [40.3, 53.9]	0.905 [0.894, 0.916]	53.2 [46.5, 58.0]

Values represent the specified performance metric, and values in brackets represent 95% bootstrapped confidence intervals obtained on the test set (N=4,909). Each model was trained five separate times; "Best Run" refers to the single model realization with maximum validation AUC, and "Five-Run Ensemble" refers to an ensemble of the five realizations of each model.

* Variant of *[L,L]-Fusion* trained on the sum of three subnetwork losses, as described in Section 1.2.

† Variant of *[L,L]-Fusion* with each subnetwork optimized independently, as described in Section 1.2.

This procedure produces 30 measures of feature importance for each of the 18 features; however, we considered the absolute median of the 30 iterations to be representative of that feature's importance for the purposes of ranking feature saliency in the main results. The intuition behind this procedure is that shuffling a particularly important feature, effectively "mismatching" patients to their data and destroying interaction effects between the permuted feature and the others, would cause a notable change in model performance, whereas shuffling a noisy feature would not drastically impact model performance.

2. Results

2.1. Optimization Method

Table 2 shows results of the *Learned Feature Fusion* model variants, described in Section 1.2, compared to the original *[L,L]-Fusion*. All three models use concatenation as the fusion operation. The two approaches of minimizing the sum of the three subnetwork losses (*[L,L]-Fusion**) and of optimizing the three subnetworks independently (*[L,L]-Fusion*†) were both competitive with the original *[L,L]-Fusion*, especially upon ensembling. The five-run deep ensembles of both *[L,L]-Fusion** and *[L,L]-Fusion*† reached both higher AUC and specificity at 95% sensitivity than the original *[L,L]-Fusion*, though these differences were not found to be statistically significant ($P > 0.05$ for each test).

References

[1] E. Khvedchenya V. I. Iglovikov A. Buslaev, A. Parinov and A. A. Kalinin. Albumentations: Fast and Flexible Image Augmentations. *ArXiv e-prints*, 2018. _eprint: 1809.06839. 1

[2] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. Publisher: Springer. 2

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. ISSN: 1063-6919. 1

[4] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[5] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When Does Label Smoothing Help? In *Advances in Neural Information Processing Systems*, pages 4694–4703, 2019. 1

[6] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems*, 32:8026–8037, 2019. 1

[7] Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti, Frédérique Lisacek, Jean-Charles Sanchez, and Markus Müller. pROC: An Open-source Package for R and S+ to Analyze and Compare ROC Curves. *BMC Bioinformatics*, 12(1):77, Mar. 2011. 2

[8] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*, pages 958–963, Aug. 2003. 1

[9] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, June 2016. ISSN: 1063-6919. 1