**LEBANESE AMERICAN UNIVESRITY**

**COMPUTER SCIENCE AND MATH DEPARTMENT, BYBLOS**

**CSC615 MACHINE LEARNING**

**ASSIGNMENT #3  
  
  
Ghadi El Hasbani, LAU I.D.: 201903122**

**Danielle Antoun Azar, PhD**

1. (10 points) Describe the vanishing gradient problem, what causes it and how it affects the learning process. Describe techniques to avoid it or overcome it.

Neural networks can be built with varying architectures and hyperparameters. Each unit has a specified activation function to relay weighted output of the previous layer while introducing non-linearities. Therefore, the choice of function is an important consideration and can affect network performance. Non-linearities are also an advantage that these functions present to enable complex feature extraction. Activation functions can be generally grouped into saturated and unsaturated functions. Saturated functions are those like the sigmoid function which is only sensitive to mid-range values while saturating close to 0 and 1. This means that it is possible for sigmoid units to cause a 0 gradient with increasing number of layers. The consequence would be loss of information (Tan & Lim, 2019). This is because at each layer, the gradient is the product of the gradient flow and the local gradient, whereby the former is influenced by deeper layers and the latter is influenced by the choice of activation. Inefficiently small gradients are then more likely with increasing depth, which increases the likelihood of a small value being propagated, and saturated activation functions. Having all local gradients between 0 and 1, such as is the case with saturated functions, will result in a vanishing gradient (Kong & Takatsuka, 2017).

Rectified linear units (ReLUs) were introduced to mitigate the issue of saturated activation functions. Unlike the sigmoid function, ReLU is sensitive to all positive inputs which made it a popular choice in deep and wide networks since it can overcome the vanishing gradient by being unsaturated. On the other hand, since ReLU’s output can be any positive value, it is prone to cause an exploding gradient where drastic weight updates are performed. It is therefore recommended to have pretrained or properly initialized weights. Moreover, an output of 0 combined with a negatively biased input will result in a 0 gradient and what is dubbed as the dying ReLU problem as neurons deactivate without possibility of reactivation (Tan & Lim, 2019). Leaky ReLU (LReLU) is a modified version of ReLU with a, usually, positive configurable slope before 0 that allows for negative output (Maas et al., 2013). The slope value is trained in parametric ReLU (PReLU; He et al., 2015). Exponential linear units (ELUs) replace the line before zero with a shifted natural exponential function (Clevert et al., 2015). Nevertheless, none of the eLU family functions maps input to a fixed interval such as the sigmoid in saturated functions which makes the output of these functions noisy. A proposed function called Hexpo maps a given input to a specific range (Kong & Takatsuka, 2017). The negative part of Hexpo resembles that of parametric ELU (PELU; Trottier et al., 2017) except that Hexpo trains the parameters. In all, Hexpo resembles saturated functions and can approximate sigmoid and tanh functions while avoiding zero-gradients through maintaining few local gradients greater than 1 at the boundaries.

Another proposed solution to the vanishing gradient problem tackles weight optimization. This is because large step size can also contribute to vanishing gradients with saturated activations like sigmoid. In this way, saturated activation functions can still be safely used in a network’s architecture. One such solution is Stochastic Diagonal Approximation Gradient Descent known as Newton’s method which approximates the gradient by a Taylor expansion up till the second order derivative approximated by the Hessian diagonal and optimizes the step-size (Tan & Lim, 2019). Newton’s method is most effective locally with non-flat search spaces. Otherwise, divergence might occur through exploding gradient due to near-zero derivatives being fed into inverse matrix operations needed to solve for the step-size. Nevertheless, since Newton’s method is designed for local optimization of parabola shaped functions it can adapt the step size better and faster than gradient descent during fine-tuning to overcome vanishing gradients.

Han et al. (2019) propose a network architecture divided into levels each with its own cost function. This method aims to tackle the vanishing gradient problem through network depth. With each few layers having their own cost function, each level will have equal impact on the gradient. This is done by calculating the cost function for each level and summing them for the combined objective function instead of calculating a single cost function value for the whole network. This is useful in input reconstruction methods which usually have similar input representations and therefore cost functions throughout the network. Nevertheless cost functions can be different across levels to adapt to other objectives.

Another example is ResNet (He et al., 2016) which resorts to double and triple-layer bypasses to connect layers at nonconsecutive depths. Similarly, a two-staged CNN is proposed for ultrasound image denoising whereby short- and long-term connections are added to connect various layers together in both components of the CNN (Chang et al., 2019). Moreover, DenseNet connects each layer with all subsequent layers (Huang et al., 2017). On the other hand, GoogleNet (Szegedy et al., 2015) introduced inception modules which have convolutional layers process the output of the previous layer in parallel with concatenated output sometimes also further processed by dimensionality reduction methods. This alleviates the depth of the network by having convolutional layers in parallel instead of sequential (Sharma & Guleria, 2022). Nevertheless, these layers do not interchange information and would not perform computations as complex as if they were connected sequentially.

1. (10 points) A common term that you encounter is “regularization”. Describe what it means. Describe two techniques and how they help achieve the objective.

Data has grown over the years, and datasets today are large in both number of samples and features. This makes analysis of such data much more complex. Moreover, for any given problem some features may be relevant while others may not. If all features are considered, irrelevant features might misguide training due to random or confounding correlations with the target output which could overfit the model. Furthermore, some algorithms are likely to overfit or not even converge when dimensionality is too large. This is because increasing number of features is accommodated by an increased number of parameters making training unstable. Finally, the more features considered, the higher the time complexity of training. All these reasons have encouraged the development of regularization methods which try to reduce the impact of certain features on the training process in order to promote model performance and generalizability and avoid overfitting. In this way, interpretability of the model is also increased through the additional availability of model feature importance information as well as customizable parameters enabling the manipulation of the regularization extent exploited in a more thorough analysis (Bickel et al., 2006).

For example, in the case of linear regression, when the number of features is greater than the number of samples, the data points are fit perfectly but the solution is not unique, meaning that new data points have more than one possible value. Ridge regression was developed whereby a penalty term, L2 or Euclidean, was added to the loss function, making the solution unique (Hoerl & Kennard, 1970). The new term panelizes the sum of squared coefficients as a factor of a regularization parameter λ. As λ increases, more weight is given to the penalty term, and high importance is attributed to lowering the sum of squared weights during the training resulting in smaller weights. This parameter can also be changed depending on the goal. It can be tuned depending on model performance, or it can be systematically varied to study its impact on the different features for feature importance analysis.

Another method, Lasso regression, uses a slightly different penalty term, L1, to minimize the sum of absolute value of coefficients to a factor of γ alongside the sum of squared errors (Tibshirani, 1996). The main difference is that Lasso regression assumes there is a sparse representation of the output as a function of the input and has the ability to reduce some coefficients to 0. In other words, Lasso regression simultaneously performs regularization and feature selection. In contrast to subset selection, Lasso performs soft thresholding characterized by continuous coefficient shrinkage to 0 as γ increases. Geometrically, the L1 penalty creates an edged constraint region which looks like a quadrilateral in 2D in contrast to the L2 penalty which creates a smooth, spherical constraint region resembling a circle in 2D. The edged constraint region of the L1 norm with vertices at the axes allows coefficients to be reduced to 0 by producing a kink at 0 in the loss function. Moreover, L1 reduces coefficients faster than L2 but is not quadratic and does not have an analytical solution. Therefore, models with L1 penalties are optimized using different efficient procedures. The disadvantage of Lasso regression is that while it may reduce some coefficients to 0, some may remain large.

In fact, L1 and L2 are only 2 out of a family of penalty terms described as Lq in Bridge regression. For any q, the Lq penalty term is the sum of absolute values of coefficients to the power q. The ElasticNet model proposed by Zou & Hastie (2005) is a method that compromises between L1 and L2 by having a convex linear combination of both penalties included in the loss function as a factor of 1 – α and α respectively. Therefore, the parameter α regulates how much weight each penalty should be given. With α close to 1, most weight is given to the L2 penalty and the model becomes similar to Ridge regression. Similarly, ElasticNet performs Lasso regression when α is 0. Another hyperparameter λ controls the weight of the combination penalty term as compared to that of the error term and can be adaptively selected using purely data-driven methods (De Mol et al., 2009). Parameter λ controls the extent of the combined regularization. Geometrically, the constraint function can be imagined as a quadrilateral with edges curved outward and vertices at the axes. This is exactly what would be expected from an L1-L2 compromise constraint. It could be tempting to say that Lq with 1 < q < 2 would produce similar results to ElasticNet, but this only true in some aspects. The most important difference is that it is proven that only q = 1 out of all q >= 1 can reduce coefficients to 0. ElasticNet then maintains properties of Bridge with 1 < q, most notably the regularization of most coefficients, while preserving the feature selection feature of Lasso.

1. (10 points) Feature Reduction is used when we have a large number of features that hinder the learning process by either being redundant or imposing a huge training time while bringing very little information. Describe two techniques used for feature reduction.

Feature reduction is an important step that reduces the complexity of the problem to necessary dimensions depending on the evaluation of one or many scoring functions. It is based on the concept of parsimony whereby the best model with the least amount of features is chosen to solve a problem. In other words, the goal is to find a model using a subset of all available features with comparable or superior performance to the full feature set model. Some machine learning models have feature selection procedures embedded in them such as Lasso regression (Tibshirani, 1996) which can reduce feature coefficients to 0 and recursive feature elimination in support vector machines (SVM-RFE; Guyon et al., 2002) which selects features at every training iteration and eliminates features based on model feature weights.

Other methods perform feature selection as a distinct step preceding model training. Since feature selection is a combinatorial problem, time complexity is a concern and approximate search methods are usually preferred to exhaustive search methods such as best subset selection which are mostly only applicable with a reasonably-sized feature set. Approximate search methods include greedy search methods such as backward and forward best subset selection. Backward selection starts with a model containing all features and iteratively removes the one that improves the scoring function the most until a specified number of features is reached or no feature improves the scoring function by being considered as a candidate. Forward selection is similar, except that the search starts with a model containing only no features. At each iteration, the feature that improves the scoring function the most is added until similar constraints as backward selection are satisfied. Nevertheless, backward selection cannot be used in cases where the model cannot be trained when the total number of features considered is greater than the sample size. An example of this is linear regression. A mixed selection method, also called stepwise, combines advantages of both forward an backward selection. The search typically starts with no features to accommodate models such as linear regression which cannot handle the full feature set in some cases. For the starting set is of size 1, then the first iteration consists of a forward selection iteration. For the iteration having the full feature set, the iteration is that of a backward selection. All other iterations consider the addition or removal of a candidate and the best option is selected. The algorithm halts when no candidate yields an improvement of the scoring function (Zhang, 2016). These methods could also be combined with k-Fold cross-validation for robust results and rankings based on metrics such as F-test, T-test, information gain, and entropy to guide the search space (Chang et al., 2010). The greedy methods can be time-consuming, although less than exhaustive search, and are not guaranteed to yield the optimal feature subset. Greedy search can also be applied without model training by selecting appropriate metrics such as the Mutual Information method which checks for relevancy and redundancy of candidate features (Sharma & Juglan, 2018). Other methods that help mitigate these issues include meta-heuristic methods and guided search methods such as genetic algorithms. It is also possible to combine multiple feature selection methods to harness the advantages of each as well as diversify the search methods. Singh et al. (2017) present an example whereby 10 feature selection methods are each run independently on the full set of features and output voted features. For ranking methods, the top-N features are considered voted while for selection methods selected features are considered voted by any particular method. The votes for each feature are then aggregated and used to produce a ranked list which, combined with a user-defined threshold, will serve to select the top-N features. Finally, some methods bypass feature selection methods entirely and resort to dimensionality reduction methods such as principle or independent component and linear discriminant analyses (PCA, ICA, LDA) to reduce the complexity of the problem. In such cases, a certain number of components, less than the total number of features, is selected whereby each component is a weighted combination of all features.

1. (25 points) Use data from Assignment II and do regression. Add the results to the table where you describe all results from the techniques you used in Assignment I and Assignment II. Elaborate on the obtained results. Submit a link to the code and a document that includes the table and your discussion of the new results only. Have all your code on GitHub.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testing Metric** (mean% ±sd) | **C5.0** | **My ANN** | **MLPC** | **SVM** | **LR** |
| *Accuracy* | 63.46 ±1.19 | 58.53 ±8.51 | 64.05 ±0.83 | 58.00 ±0.00 | 59.73 ±0.00 |
| *Precision* | 67.83 ±5.92 | 68.35 ±1.46 | 68.36 ±0.88 | 64.40 ±0.00 | 66.01 ±0.00 |
| *Recall* | 68.49 ±1.25 | 52.06 ±27.56 | 69.90 ±3.85 | 60.30 ±0.00 | 61.81 ±0.00 |
| *Specificity* | 57.19 ±2.16 | 67.28 ±17.59 | 56.12 ±3.96 | 54.88 ±0.00 | 56.92 ±0.00 |
| *AUC* | 65.79 ±0.66 | 59.67 ±5.21 | 63.01 ±0.64 | 57.59 ±0.00 | 59.36 ±0.00 |
| *F1* | 67.97 ±2.67 | 66.62 ±1.33 | 69.06 ±1.58 | 62.28 ±0.00 | 63.84 ±0.00 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Loss** (Squared Error, mean ±sd) | **My ANN** | **MLPC** | **SVM** | **LR** |
| *Training* | 668.77 ±67.24 | 2033.08 ±64.55 | 1911.17 ±14.38 | 1835.25 ±3.97 |
| *Validation* | 110.83 ±34.63 | 225.82 ±7.33 | 206.57 ±1.87 | 203.93 ±1.01 |

The results on the testing metrics show that Logistic Regression (LR) outperforms SVM on all metrics. The simplicity and interpretability of LR makes it superior to SVM in this regard, especially since the variability in testing metrics is extremely low for both models. On the other hand, C5.0 is also a simple and interpretable model that outperforms LR on every metric at the cost of an increase in variability. Nevertheless, my ANN remains the most variable model, although it outperforms or is comparable to LR on all metrics except recall which is much higher for LR. My ANN remains the model with the highest average specificity (67.28 ±17.59) while MLPC remains the model with the highest recall (69.90 ±3.85). Although the standard deviation of my ANN’s specificity is very high, all other candidates have comparably low specificity. Nevertheless, low specificity might be preferred to a less robust but higher one. C5.0 remains the best choice if precision (67.83 ±5.92) is the goal and if simplicity and interpretability are taken into consideration. If performance as measured by precision is the primary goal, MLPC (68.36 ±0.88) is preferred. Nevertheless, LR’s specificity (56.92 ±0.00) is comparable but slightly lower to that of C5.0 (57.19 ±2.16) but is more robust. Finally, the model that performs best generally is C5.0 having highest AUC (65.79 ±0.66) as well as accuracy (63.46 ±1.19) and f1-score (67.97 ±2.67) comparable to the highest accuracy (64.05 ±0.83) and f1-score (69.06 ±1.58) of MLPC. C5.0’s variability is also reasonable compared to the highest variability in my ANN’s metrics except for precision (68.35 ±1.46) which is more variable for C5.0.

1. (25 points) Search the literature for a paper that uses Transfer Learning and write a critique of it. Submit the paper along with the assignment. Name the file “<Yourname>Q5Paper.pdf”.

Transfer learning is a popular method in deep learning that takes advantage of pre-trained models on large-scale benchmark datasets or any large data on a specific task A and extends this model on a different, most of the times similar, task B. Transfer learning is usually used when data for the task at hand is limited or sparse, and training with the data at hand would not yield satisfactory performance. This is usually because the task at hand is complex. The pre-trained model can then be transferred as is or fine-tuned with the limited data at hand that corresponds to the given task B. Transfer learning is particularly popular in image processing. The availability of large object detection datasets such as ImageNet (Deng et al., 2009) and state-of-the-art object recognition models such as GoogleNet (Szegedy et al., 2015), ResNet (He et al., 2016), VGG-family (Simonyan & Zisserman, 2014) and others enable an opportunity to compensate for sparse image datasets such as in medical imaging. Moreover, the use of deep learning models over static feature extraction methods makes the feature extraction dynamic. Meng et al. (2017) propose a transfer-learning-based method for staged liver fibrosis classification using a dataset of only 279 region of interest (ROI) ultrasound (US) images consisting of 79 healthy, 89 early-stage (S1-S3) fibrosis, and 111 late-stage (S4) fibrosis. Each sample is accompanied by a label given by a clinician indicating the corresponding group. The authors select VGGNet trained on the ImageNet dataset (Deng et al., 2009) as the transferred model. Using the last feature maps of VGGNet, the authors build heatmap representations of the input from each group and plot the intensity distributions of each stage. Although obvious discriminating features are present in the distributions that are consistent with the literature on texture and fibrosis, the authors claim that VGGNet’s representations are then sufficient for correct classification as compared to the original input. Nevertheless, the distributions reported also have similarities, especially for the early- and late-stage fibrosis groups. Moreover, it is unclear if these histograms correspond to only 1 sample per group. It would perhaps be more useful to average the histograms across samples per group for dataset-wide insight. Moreover, VGGNet is selected as opposed to other models because it is shallower. The justification given behind this choice is the sample size which could cause divergence and overfitting with deeper networks. To take advantage of the complex processing of deeper architectures while avoiding overfitting, the feature maps are then forwarded to a predictive component dubbed FCNet which is trained separately. In this way, training does not preside over all layers at once. Nevertheless, the same training set is used to both fine-tune VGGNet and train FCNet which will increase the chance of overfit and unreliable results. It is preferred to train each component of the model on a separate set while testing on a third, independent holdout set.

The VGGNet component of the model is fine-tuned using head training procedure. The learning rate starts as 0.1 times the VGGNet learning rate on the ImageNet dataset (Deng et al., 2009) and decay 10-fold with each iteration. The learning rate for the trainable softmax layer is set 10 times higher. The VGGNet layers are not trained otherwise, but investigating this option could have been worthwhile. Depending on the problem, fine-tuning the first or last few layers could help adjust feature representations to the given task depending on whether distinctive features of the task are simple or complex. Since only one model is used, a cross-validated tuning procedure for the learning rate could also be useful to prevent overfitting. It could additionally be helpful to use an ensemble to reduce variability, especially given the small sample size and noisy nature of US images. The FCNet component consists of three fully connected layers each followed by a dropout layer. The first fully connected layer is followed by ReLU activation. The reason behind using ReLU as an activation function is to avoid vanishing gradients. Nevertheless, dying ReLU is still a possibility and LReLU and PReLU could be better choices. Moreover, vanishing gradient is still possible if saturated functions are used at the second fully connected layer since the 0 gradient is propagated backward. ReLU would be more useful at deeper layers. The FCNet is also only 3 layers deep, which means it is unclear what impact the choice of activation function will have on performance. This is because vanishing gradients are a bigger concern with deeper networks. In general, the activation function, dropout fraction, and number of hidden units could be tuned using a cross-validated procedure.

In this paper, the original images consist set A while augmented versions through cropping and flipping comprise a larger set B. The authors train models either on 70% of set A or B and reporting testing metrics on the other 30% of the same set, the other set, and both combined. For example, if the model was trained on 70% of set B, it will be tested on the remaining 30% of set B, all of set A, as well as all of set A and the remaining 30% of set B combined. In all those experimental variations, the only one exists whereby the model is tested on a completely independent holdout set from the one on which it was trained. In this scenario, the model is trained on 70% of set A and tested on the remaining 30%. This is because any model trained on set B does not guarantee that the remaining 30% contains augmentations of images not shared with the other 70%. This is because an augmented version of the original image cannot be considered independent of the original image, especially since the augmentations used are simple and mostly leave images intact. This is also very evident given the reported scores. The scenario which tests on a completely independent set (30% of A) after training on another (70% of A) presents the lowest performance of 63.28% compared to all other scenarios with scores above 80%. The other two scores for models trained on 70% of set A are lower (around 80%) than those trained on 70% of set B (all above 90%). This is most probably due to the fact that overlap between training and testing set is reduced when training with part of set A, since this ensures that at least some images (all those in the remaining 30%) are never used for training. In contrast, when training with 70% of set B, most images have been exposed many times to the model through training and are likely to be repeated in the testing sets.

The proposed VGG-FCNet is also compared to AlexNet (Krizhevsky et al., 2017), CaffeNet (Jia et al., 2014), VGGNet, each VGGNet feature map combined with either Random Forest (RF), support vector machine (SVM), gradient boosted decision trees (GBDT), or multilayer perceptron (MLP). It would have also been useful to compare scores to each VGGNet feature map combined with FCNet to ensure that all intermediate representations are relevant and that performance increases with the addition of each layer. Interestingly, the proposed method outperforms all other methods or is tied with the best one in all experimental scenarios except where 70% of set A is used for training and the remaining 30% for testing. In the latter scenario, 11 out of 19 other methods outperform VGG-FCNet with VGGNet and the first feature map of VGGNet (VGGNet-A) combined with RF tied for best performance. In other words, using only the first feature map of VGGNet combined with RF achieves better generalizability than the proposed method. This might have been evident if VGGNet-A combined with FCNet was tested amongst other methods. Perhaps some combination of VGG feature map and FCNet might have outperformed VGGNet and VGGNet-A + RF. This would have indicated possible overfit supported by the reported scores. Finally, no standard deviations are reported since there was no indication that experiments were run more than a single time, and no metrics other than accuracy were reported. Since the dataset is not very balanced between groups, metrics such as balanced accuracy could provide additional insight.

1. (25 points) Search the literature for a paper that uses Ensemble Learning and write a critique of it. Name the file “<Yourname>Q6Paper.pdf”.

Ensemble Learning (EL) is a useful method that combats model variance. Variance is usually increased with depth of the network and small sample sizes. In the case of ultrasound (US) image processing methods, problems are usually complex and data availability is sparse. Therefore, ensemble methods are used to gain different perspectives on the problem at hand using diverse submodels with a single, integrated output. Ensemble pruning methods can also be useful to select an optimal subset of models to boost prediction. Misra et al. (2021) propose an ensemble transfer learning (TL) approach combining B-mode breast US (B-US) and strain elastography breast US (SE-US) images to discriminate between benign and malignant breast tumors. True labels are from biopsy diagnoses of a total of 261 (130 benign, 131 malignant) images from 85 (42 benign, 43 malignant) patients. For each patient, there are corresponding B-US and SE-US images that contribute structural and functional information, respectively. Nevertheless, they are not spatially aligned. It is stated that pooling layers will serve to eventually align the images. Nevertheless, it could improve performance if integration of images is done by a former, separate step using CNNs. For example, images could serve as input to separate convolutional layers which then map to a common convolutional layer. The singular output can then be fed into TL models. Therefore, the image integration as well as the extracted features would be dynamic. Moreover, as mentioned by the authors, one limitation is fine-tuning models trained on a natural dataset such as ImageNet (Deng et al., 2009) instead of medical images which could improve performance due to task similarity.

All data splits are performed patient-wise because it was reported that the model could predict different classes for images of the same patient. Moreover, patient-wise analysis was also performed on the model, so metrics have to be informative in that regard. This is accomplished by a Voting classifier that will perform patient-wise predictions. Another importance that was not pointed out is that if the model were to be deployed in real-life clinical settings, as is stated as one of the future goals of the authors, it is important to be informed of a model’s generalizability to new patients independent of training, as is usually the case in clinical settings.

Images were used for training either with the original size or cropped size. Cropping is performed manually with guidance from a radiologist. Images were also augmented using random cropping and horizontal flipping at each training epoch. Submodels chosen were AlexNet (Krizhevsky et al., 2017) and ResNet-18 (He et al., 2016) trained on ImageNet (Deng et al., 2009). The models were chosen for their good performance and shallow structures for computational efficiency as well as ResNet’s ability to combat vanishing gradients with skip connections. Shallow structures help combat overfitting with small sample sizes. Nevertheless, as noted by the authors in the limitations of this work, more TL models can be considered in the model to boost performance. Furthermore, it could be helpful to also optimize ensemble members using pruning methods although this might increase computational time. Nevertheless, with more candidate TLs, this could be worthwhile to explore. For fine-tuning, the authors freeze the first few layers since they usually represent simpler features and fine-tune a certain number of deeper layers. Grid-search cross-validation was used to tune how many of the last layers need to be re-trained for each submodel separately. It could also have been more optimal to consider fine-tuning middle layers and not only from a layer onward. Although this could also increase computational time, optimization methods could be used to guide the search. Since B-US images are greyscale and SE-US images are RGB, combined images have 4 channels and were then resized to 224x244. Resizing could instead have been done as downsizing to the minimum size or adjusting to the average size so that the general ratios are not severely changed.

The classification layer is then dropped from each model and the output of both networks are concatenated. The concatenated output is finally passed to a softmax classification layer. Adam optimizer, since it is efficient, and cross-entropy loss were used for weight optimization with a starting learning rate of 0.0001 and an early stopping protocol after 200 epochs of inefficient or no improvement. The best epoch weights are then considered. There was no indication that different sets were used to train each submodels and final classification layer, but this might be helpful for performance generalizability. Moreover, since the concatenated output contains 4608 features, it might be helpful to add fully connected layers to process these features before the final classification layer. Dropout strategy can also be used to prevent overfitting as in FCNet (Meng et al., 2017). No hyperparameters were tuned since the authors aimed to compare methods under the same conditions. Although tuning all compared methods to the data could give a better comparison of performances, it would be a time-consuming procedure.

The training data consisted of 65 (80%) of the patients with the rest reserved for testing using accuracy, precision, specificity, sensitivity, and f1-score. The metrics are chosen to reflect diverse aspects of model performance. Moreover, scores were evaluated image-wise and patient-wise for the full or cropped images. For patient-wise comparison, majority voting is considered over all images of the given patient as aforementioned. All experiments were additionally run for each imaging type alone and for the combined 4-channel images. All metrics are also cross-validated. The proposed method is then compared to AlexNet, ResNet, VGG, manual and assist strain ratios (MSR and ASR; Barr & Managuli, 2019), SqueezeNet (Iandola et al., 2016), and DenseNet (Huang et al., 2017) individually. The proposed ensemble achieves superior or relatively comparable performance in all metrics and all experimental scenarios as compared to other methods. Whenever the ensemble performs comparably, AlexNet is the competing method. Moreover, the patient recognition rate, defined as the mean proportion of correctly classified images per patient is comparable to AlexNet except using B-US images alone where the ensemble is superior. Since AlexNet is one of the two models in the ensemble, this indicates that the ensemble’s performance is majorly dominated by AlexNet’s performance. This supports the previous suggestion to explore more models and optimize which combination to pick. Nevertheless, the mean predictive probability value of the ensemble is significantly higher than that of AlexNet and ResNet with a 95% confidence interval. There was no indication to how many times this experiment was repeated. Finally, Grad-CAM saliency maps (Selvaraju et al., 2017) were then constructed to demonstrate model interpretability.

**References**

Barr, R. G., & Managuli, R. A. (2019). A clinical study comparing the diagnostic performance of assist strain ratio against manual strain ratio in ultrasound breast elastography. *Ultrasound quarterly*, *35*(1), 82-87.

Bickel, P. J., Li, B., Tsybakov, A. B., van de Geer, S. A., Yu, B., Valdés, T., ... & van der Vaart, A. (2006). Regularization in statistics. *Test*, *15*(2), 271-344.

Chang, C. Y., Chen, S. J., & Tsai, M. F. (2010). Application of support-vector-machine-based method for feature selection and classification of thyroid nodules in ultrasound images. *Pattern recognition*, *43*(10), 3494-3506.

Chang, Y., Yan, L., Chen, M., Fang, H., & Zhong, S. (2019). Two-stage convolutional neural network for medical noise removal via image decomposition. *IEEE Transactions on Instrumentation and Measurement*, *69*(6), 2707-2721.

Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248-255). Ieee.

De Mol, C., De Vito, E., & Rosasco, L. (2009). Elastic-net regularization in learning theory. *Journal of Complexity*, *25*(2), 201-230.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, *46*(1), 389-422.

Han, X. H., Zheng, Y., & Chen, Y. W. (2019). Multi-level and multi-scale spatial and spectral fusion CNN for hyperspectral image super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (pp. 0-0).

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026-1034).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55-67.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4700-4708).

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. *arXiv preprint arXiv:1602.07360*.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., ... & Darrell, T. (2014, November). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia* (pp. 675-678).

Kong, S., & Takatsuka, M. (2017, May). Hexpo: A vanishing-proof activation function. In *2017 International Joint Conference on Neural Networks (IJCNN)* (pp. 2562-2567). IEEE.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90.

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml* (Vol. 30, No. 1, p. 3).

Misra, S., Jeon, S., Managuli, R., Lee, S., Kim, G., Lee, S., ... & Kim, C. (2021). Ensemble Transfer Learning of Elastography and B-mode Breast Ultrasound Images. *arXiv preprint arXiv:2102.08567*.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision* (pp. 618-626).

Sharma, S., & Guleria, K. (2022, April). Deep learning models for image classification: comparison and applications. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)* (pp. 1733-1738). IEEE.

Sharma, V., & Juglan, K. C. (2018). Automated classification of fatty and normal liver ultrasound images based on mutual information feature selection. *Irbm*, *39*(5), 313-323.

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

Singh, B. K., Verma, K., Thoke, A. S., & Suri, J. S. (2017). Risk stratification of 2D ultrasound-based breast lesions using hybrid feature selection in machine learning paradigm. *Measurement*, *105*, 146-157.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

Tan, H. H., & Lim, K. H. (2019, June). Vanishing gradient mitigation with deep learning neural network optimization. In *2019 7th international conference on smart computing & communications (ICSCC)* (pp. 1-4). IEEE.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267-288.

Trottier, L., Giguere, P., & Chaib-Draa, B. (2017, December). Parametric exponential linear unit for deep convolutional neural networks. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 207-214). IEEE.

Zhang, Z. (2016). Variable selection with stepwise and best subset approaches. *Annals of translational medicine*, *4*(7).

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, *67*(2), 301-320.

Also:

<https://online.stat.psu.edu/stat508/lesson/5/5.4>

<https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote07.html>