Published in final edited form as:

*Inform Med Unlocked*. 2021 ; 24: . doi:10.1016/j.imu.2021.100571.

# UMS-Rep: Unified modality-specific representation for efficient medical image analysis

**Ghada Zamzmi**[*],
**Sivaramakrishnan Rajaraman**,
**Sameer Antani**
National Library of Medicine, National institutes of Health, Bethesda, MD, USA

## Abstract

Medical image analysis typically includes several tasks such as enhancement, segmentation, and classification. Traditionally, these tasks are implemented using separate deep learning models for separate tasks, which is not efficient because it involves unnecessary training repetitions, demands greater computational resources, and requires a relatively large amount of labeled data. In this paper, we propose a multi-task training approach for medical image analysis, where individual tasks are fine-tuned simultaneously through relevant knowledge transfer using a unified modality-specific feature representation (UMS-Rep). We explore different fine-tuning strategies to demonstrate the impact of the strategy on the performance of target medical image tasks. We experiment with different visual tasks (e.g., image denoising, segmentation, and classification) to highlight the advantages offered with our approach for two imaging modalities, chest X-ray and Doppler echocardiography. Our results demonstrate that the proposed approach reduces the overall demand for computational resources and improves target task generalization and performance. Specifically, the proposed approach improves accuracy (up to ~ 9% ↑) and decreases computational time (up to ~ 86% ↓) as compared to the baseline approach. Further, our results prove that the performance of target tasks in medical images is highly influenced by the utilized fine-tuning strategy.

### Keywords

Medical image analysis; Deep learning; Disease classification; Image segmentation

## 1. Introduction

Deep learning has significantly advanced the frontiers of visual task analysis, with applications in a large number of domains including automated driving, robotics, biometrics, and biomedical imaging, to name a few. With breakthrough advances in medical image

[*]Corresponding author. National Library of Medicine, National Institutes of Health Bethesda, MD USA. alzamzmiga@nih.gov, ghadh@usf.edu (G. Zamzmi).

Declaration of competing interest
The authors declare no competing interests.

analysis, there has been a surge in research interest in health applications and biomedical research. Some common visual tasks across various medical imaging modalities include (i) region of interest (ROI) detection (e.g. Refs. [1,2], (ii) segmentation (e.g. Refs. [2–5], (iii) registration (e.g. Ref. [6], and (iv) classification (e.g., Refs. [2–4].

While these approaches have achieved promising performance, they tend to use individual pre-trained models (e.g., VGG16 and ResNet with ImageNet weights) for individual tasks. Training and deploying multiple individual models with high computational complexity for multiple tasks is feasible, but is constrained by limited computational resources and repetitive sequential training. These factors can significantly impact future advances and pose challenges in deploying applications with a large number of models in real-world clinical settings. Additionally, using as many models as there are tasks limits the potential offered by deep learning by constraining it from transferring knowledge from the source domain to a target domain or task.

These challenges can be alleviated through multi-task learning (MTL), a machine learning concept that transfers the knowledge from a shared source to multiple target tasks [7], thereby improving target tasks generalizability and aiding overall performance. The target tasks can be homogeneous with similar annotations or heterogeneous with diverse annotations. For example, two target tasks, such as semantic and instance segmentation, that use similar annotations can be jointly learned via a shared encoder and task-specific decoders. Joint learning strategy involves learning a joint loss function that takes into consideration the inter-task relationship. Other fine-tuning strategies include independent and alternating strategies [7–9]. We further discuss these strategies in Section 2.3.

MTL techniques have been widely used for natural images but few researches [[10,11]] have been carried out to apply these techniques on medical images. Inspired by the success of MTL with natural images, we propose an MTL-based training approach in medical image analysis, where individual target tasks are learned simultaneously through relevant knowledge transfer across a shared medical modality-specific feature representation. This approach enables reduced requirements for computational resources and enhances target tasks generalization.

The main contributions of this paper are as follows:

- We propose an approach that uses a shared source, called unified modality-specific representation (UMS-Rep), to simultaneously fine-tune target medical image tasks with diverse and similar annotations.
- We show that the shared source (UMS-Rep) can be constructed on a specific medical imaging modality using any learning techniques including unsupervised (ex nihilo without annotations) or supervised (with limited annotations). Also, we show that UMS-Rep can simultaneously learn pre-processing tasks such as image denoising and enhancement.
- We explore three fine-tuning strategies, namely independent, alternating, and joint, to investigate their impact on the performance of target tasks in medical images.

- We define derivable tasks, which can be any task derived from the learned target tasks. An example of a derivable task includes visual interpretation of machine learning decisions, derived from a classification target task.

Our empirical evaluations show that the proposed sequence of training each part of the network, first the shared UMS-Rep with pre-processing tasks, second the task-specific heads (target tasks) with the suitable fine-tuning strategy, and finally derivable tasks, is more efficient for medical image analysis. Particularly, our evaluations on medical image benchmarks demonstrate the superiority and efficiency of our approach, in terms of performance, resources, and computation, as compared to the baseline approach of training as many models as there are tasks.

The rest of this paper is organized as follows. Section 2 discusses different types of target tasks and fine-tuning strategies. Section 3 presents the medical image datasets used in this paper and describes our training approach for medical image analysis. Then, we present the experimental results and a comparison with the baseline approach in Section 4. Finally, we discuss the results and conclude the paper in Section 5.

## 2. Background

MTL methods have attracted attention as they show remarkable success in improving the generalizability of target tasks [7]. These target tasks can be divided based on data annotations into homogeneous tasks (similar annotation) and heterogeneous tasks (diverse annotation). Next, we present homogeneous and heterogeneous target tasks and discuss different strategies for fine-tuning.

### 2.1. Homogeneous target tasks

MTL-based methods have been widely used to jointly learn target tasks with similar annotations. For example [12], proposed a MTL-based method to jointly learn the following target tasks from a face image: pose, gender, wear glasses, and smiling. These target tasks have global image labels (presence/absence of a smile, a glass, etc.) as the ground truth (GT) annotations. The inter-task relationship or correlation was learned using a joint loss function as presented in Ref. [12]. Another method is proposed by Ref. [13] to jointly learn the category and pose of objects. Similar to Ref. [12]; both target tasks have image-level label annotations (i.e., category label and pose label) and the relationship between these tasks is learned using a joint loss function. Other methods for learning target tasks with homogeneous annotations are presented in Refs. [10,14–16].

### 2.2. Heterogeneous target tasks

[17] proposed a method for modeling three heterogeneous tasks in natural images, namely street classification, vehicle detection, and road segmentation, with diverse annotations (global, box coordinates, and pixel-level labels). Each task is trained individually, and the final loss is given as the sum of the losses for segmentation, detection, and classification. YOLO network [18] is another example of a method that learns tasks with diverse annotations. Specifically, YOLO uses a head for the bounding box (coordinates) and another

head for classification (image label). YOLO uses a loss function with a localization loss term for the bounding box task and classification loss for the classification task.

Unlike natural images, medical images are not only limited in the number of annotations, but they also exhibit different visual characteristics (e.g., color homogeneity, texture subtlety, shape variation, and highly similar appearance). An approach has been attempted by Ref. [9] for learning heterogeneous medical image tasks. They built a set of models, called Genesis, and used them as the starting point to learn classification and segmentation tasks. Both tasks are fine-tuned independently using two individual loss functions.

## 2.3. Fine-tuning strategies

We group the fine-tuning strategies for the target tasks into: joint fine-tuning, alternating fine-tuning, and independent fine-tuning.

Joint fine-tuning strategy involves optimizing the entire network to minimize a single loss function that accumulates the losses of all target tasks. This strategy allows learning relationships between target tasks. Mathematically, the joint loss function can be expressed as follows: $L_{joint} = \sum_{i=1}^{N} w_i L_T i$ where $L_T i$, $w_i$, and $N$ represent the loss of a specific task, its weight, and the total number of tasks, respectively. The weights ($w_i$) of losses can be either weighted or unweighted. In case of unweighted summation, the loss functions for all tasks receive equal weights. Otherwise, the optimal weights for tasks' losses can be hand-tuned or learned through extensive empirical experiments. Other approaches for weighting losses include homoscedastic uncertainty which was introduced by Ref. [14] and cross-task consistency as proposed by Ref. [19].

Instead of optimizing a single joint function, the alternating strategy requires each target task to have a separate loss function; i.e., $N$ loss functions for $N$ tasks [8]. The loss function for each task has two terms, the first term ($\Theta_{src}$) represents the parameters for the shared source and the second term represents the task-specific parameters ($\Theta_{TN}$) for $N$ tasks. The optimization process is performed by alternately optimizing the loss of a specific target task using a task-specific batch followed by optimizing the loss of another task using its specific batch, and so on [8]. Fig. 1 shows the optimization procedure for alternating fine-tuning strategy, which involves fine-tuning a specific task for a fixed number of batches before switching to the batches of the next task [8]. In each switch, we update the task-specific loss with its terms ($\Theta_{src}$ and $\Theta_{T_i}$). The ratio of task-specific batches can be fixed for all tasks, determined based on other factors such as performance, importance, and dataset size, or calculated using specific methods such as the method proposed in Ref. [20]. By alternating the learning among tasks and updating the weights ($\Theta_{src}$ and $\Theta_{T_{i=1:N}}$), this strategy allows to learn the similar latent representations across the target tasks.

Finally, the independent strategy involves fine-tuning target tasks independently while freezing $\Theta_{src}$. This strategy allows all target tasks to share a common representation followed by task-specific layers. Each task-specific head is fine-tuned using its own loss and optimizer. In other words, we freeze the shared representation parameters ($\Theta_{src}$) and use task-specific optimizers to minimize task-specific losses on task-specific data.

In summary, joint fine-tuning strategy optimizes the entire network (shared source and task-specific heads) by minimizing a single joint function that accumulates the losses of all target tasks. The alternating strategy optimizes a specific-task by minimizing its loss for a fixed number of batches before switching to the next task. In each switch, both the source and task-specific parameters are updated. In the independent fine-tuning strategy, we freeze the parameters of the shared representation and independently optimize task-specific heads by minimizing their losses. It has been reported [21] that independent strategy leads to better performance when used for fine-tuning competing (or different) tasks while joint strategy leads to better performance with cooperating (or similar) tasks. This is attributed to the fact that competing tasks may lead to the transfer of irrelevant information when learned jointly (i.e., negative transfer between unrelated tasks). Further, the gradients may interfere, which makes the optimization landscape of multiple summed losses more difficult.

## 3. Materials and methods

### 3.1. Medical image datasets

Two medical imaging modalities are used to evaluate the proposed approach described in Section 3.2. These modalities are chest X-ray (CXR) and Doppler echocardiography (Doppler echo). Table 1 provides a summary of the CXR and Doppler echo datasets used in this work.

**3.1.1. CXR collections—**We used four publicly available CXR collections in our evaluation of the proposed approach. These collections are the Radiological Society of North America (RSNA) [22]; the Shenzhen TB [23]; the Montgomery TB [23]; and the Pediatric pneumonia dataset [24].

The RSNA dataset includes 26,684 normal and abnormal frontal CXRs that are provided as DICOM images with 1024 × 1024 spatial resolution. In addition to the labels, GT disease bounding boxes are made available for CXRs containing pneumonia-related opacities. The Shenzhen dataset contains 662 CXR images (336 abnormal and 326 normal image-level labels) and GT binary masks for the lungs (pixel-level labels). The size of the images in this collection varies, but it is approximately 3000 × 3000 pixels. The Montgomery dataset has 138 posterior-anterior CXRs of which 80 CXRs are normal and 58 are abnormal with TB manifestations. The images have 4020 × 4892 pixel resolution. The abnormal class for Montgomery dataset includes a wide range of abnormalities, including tuberculosis-related manifestations, effusions, and miliary patterns. Finally, the Pediatric pneumonia dataset was collected from 624 patients and labeled as described in Ref. [24]. The train set contains a total of 5232 chest X-ray images, including 3883 labeled as pneumonia (2538 bacterial and 1345 viral) and 1349 labeled as normal. The test set contains 234 normal images and 390 pneumonia images (242 bacterial and 148 viral). Fig. 2 shows labeled image examples from the aforementioned CXR datasets.

The images of all datasets were resized to 256 × 256 using bicubic interpolation (OpenCV built-in function). We also performed mean normalization to ensure that the images have a similar distribution. As it is well known, data normalization can speed up convergence while training the network.

#### 3.1.2. Doppler echo—

We used a private dataset of 2444 images showing continuous wave and pulsed wave Doppler flows collected from 100 patients who were referred for echocardiographic examination in the Clinical Center at the National Institutes of Health (NIH). The use of these de-identified images was approved by the NIH Ethics Review Board (IRB#18-NHLBI-00686). The Doppler traces of the mitral valve flow (MV), mitral annular flow (MA), and tricuspid regurgitation flow (TR) were acquired using different commercial echocardiography systems including Philips iE33, GE Vivid95, and GE Vivid9. Each Doppler image has a flow type label (TR, MV, or MA) and a segmentation mask provided by an expert technician, which separates the spectral envelope from the background. Besides, the expert technician assessed the quality of a subset of images (814 out of 2444) as low- or good-quality. All GT annotations (global labels and binary masks) provided by the expert technician were further verified by an expert cardiologist. Fig. 3 shows labeled image examples from the Doppler echo dataset.

All the images were resized to 256 × 256 using bicubic interpolation (OpenCV built-in function). We then performed mean normalization.

## 3.2. Proposed approach

The traditional approach for medical image analysis (e.g. Refs. [1,5], is shown in Fig. 4. This approach involves training $N$ individual (encoder-decoder) models for $N$ individual target tasks in isolation. Our proposed approach of simultaneously training UMS-Rep with pre-processing tasks and sharing the trained UMS-Rep among target tasks is presented in Fig. 5.

#### 3.2.1. Notations and definitions—

As shown in Fig. 5, UMS-Rep learns the visual features ($X_S$) of a source input space or specific imaging modality ($S$). The learned feature representation is then shared to simultaneously learn $N$ target tasks $T_{1:N} = \{T_1, T_2, .., T_N\}$. The source modality is defined as $S = \{X_S, P_S(X_S)\}$, where $X_S$ represents the feature space and $P_S(X_S)$ represents the probability distribution of the input modality.

UMS-Rep improves the learning and generalization of the predictive functions of up to $N$ target tasks $(F_{1:N}(\,.\,) = \{f_1(\,.\,), f_2(\,.\,),\ .., f_N(\,.\,)\})$ by transferring the shared knowledge from $S$ to $N$ to generate task-specific predictions $Y_{1:N} = \{Y_1,\ .., Y_N\}$. Since medical image analysis tasks that use a common input or modality share common low-level features, a shared modality-specific representation (source) can be trained once and used to simultaneously fine-tune target tasks with diverse or similar GT labels. For example, the features extracted from a specific imaging modality ($S$) by an autoencoder using unsupervised learning ($Y_{GT_s} = \{\}$) can be simultaneously shared by two target tasks $T_1$ and $T_2$ with different label spaces, where $P_S(X_S) \sim P_1(X_1)$ and $P_S(X_S) \sim P_2(X_2)$. A new task-specific head (or decoder) can be added on-the-fly if this constraint is satisfied.

In Fig. 5, we also define D derivable tasks $V_{1:D} = \{V_1, V_2,\ ..., V_D\}$. These tasks are the ones that use the information learned by a single or a combination of task-specific heads. An example of a derivable task is the interpretation or visual explanation generated from classification task prediction. Such a task is crucial for augmenting medical decision making

[25]. Another derivable task is automated decision recommendation, which can be generated by combining the outputs of different task-specific heads. A new derivable task can also be added on-the-fly with minimal effort as long as it uses information of previously learned task-specific heads to produce the output.

Our proposed sequence of learning each part, first the shared UMS-Rep with pre-processing tasks, second the task-specific heads, and finally the derivable tasks, allows more efficient analysis of medical images.

#### 3.2.2. UMS-rep construction—

Given that the proposed approach holds the promise for simultaneously reducing computation/resources and enhancing the generalization of $N$ target tasks, a question that arises is: what learning technique should be used to construct the shared UMS-Rep? The answer depends on the input space and the tasks at hand. For example, given a dataset with different label spaces for different tasks, UMS-Rep can be constructed using the task with the largest number of labeled data (supervised learning). Instead of using the labels of a specific task to construct the shared UMS-Rep, an autoencoder can be optimized for a given source modality ($S$) and used as a shared representation for simultaneously learning multiple target tasks (unsupervised learning).

In this paper, we experimented with both unsupervised and supervised learning techniques to construct the shared UMS-Rep. Note that other techniques (e.g., semi-supervised, residual learning) can be easily adopted to construct UMS-Rep. In both cases, we used shallow customized architectures for UMS-Rep; however, the state-of-the-arts architectures (e.g., VGG and UNet) can be used instead. While constructing UMS-Rep, a pre-processing task (i.e., noise reduction) is also learned. Using the proposed approach for simultaneously learning pre-processing tasks and solving visual tasks can prevent unnecessary repetitions, reduce computations, and enhance generalizability.

#### 3.2.3. Target tasks layers & fine-tuning strategies—

Prior to fine-tuning the target tasks, we appended task-specific layers (or decoders) to the constructed UMS-Rep. To create the task-specific layers for the segmentation task, we append a symmetrical decoder to the shared UMS-Rep. In case of classification, we append a global average pooling (GAP), fully connected (FC), dropout (D), and Softmax (SM) layers to the shared UMS-Rep. Then, we fine-tune the task-specific layers using three strategies, namely alternating, independent, and joint, to investigate the impact of the fine-tuning strategy on the performance of the target tasks.

As discussed in Section 2.3, joint fine-tuning involves learning a single loss that combines the sum of unweighted or weighted losses for all target tasks. There are different approaches (e.g. Refs. [10,12–14], for weighting losses. In this paper, we learned, through empirical experiments, the optimal weights for tasks' losses. Learning a single loss function that weighs losses of the target tasks allows learning the commonality and differences between target tasks. On the other hand, alternating strategy (see Fig. 1) learns target tasks by alternating the learning among tasks and updating the weights. This strategy allows to simultaneously 1) transfer common features from the shared source to task-specific layers and 2) learn similar latent representations across the target tasks. We determine the ratio of task-specific batches as described in Ref. [20]. After determining the ratio, we train a specific task $i$ for $n_i$ batches and switch to the next task $j$, which is trained for $n_j$ batches. Finally, we use the independent fine-tuning strategy, where we freeze the weights of the shared UMS-Rep and fine-tune each target task independently using its own loss and optimizer.

The target tasks can be optimized using different loss functions. In this paper, we optimize the target classification tasks to minimize the categorical cross-entropy (CCE) loss. For the segmentation tasks, we minimize a combination of binary cross-entropy (BCE) and Dice losses as follows:

$$L_n = w_1 L_{BCE_n} + w_2 L_{DSC_n} \tag{1}$$

where $Loss_{BCE_n}$ is the binary cross-entropy, $Loss_{DSC_n}$ is the Dice loss, $n$ denotes the batch number, and $w_1 = w_2 = 0.5$. The losses are computed for each mini-batch, and the final loss for the entire batch is determined by the mean of loss across all the mini-batches. $Loss_{BCE_n}$ and $Loss_{DSC_n}$ are expressed as follows:

$$L_{BCE_n} = [t_n log(y_n) + (1 - t_n) log(1 - y_n)] \tag{2}$$

and,

$$L_{DSC_n} = 1 - \frac{2\sum t_n \cdot y_n}{\sum t_n + \sum y_n} \tag{3}$$

where $t$ is the target and $y$ is the output from the final layer. We empirically found that the combination of $L_{BCE_n}$ and $L_{DSC_n}$ losses (equation (1)) improved the optimization due to the interplay between the global and local feature extraction capabilities of these loss functions.

## 4. Experiments and results

We evaluated our approach on two medical imaging modalities: CXR and echo Doppler. To construct UMS-Rep, we experimented with two learning paradigms: unsupervised (CXR) and supervised (echo Doppler). Then, the constructed UMS-Reps, for each modality, are appended with task-specific layers corresponding to different target tasks. These layers are fine-tuned using independent, alternating, or joint strategies. In our work, the input domains of all target tasks have similar distributions to the input domain of UMS-Reps.

To demonstrate the power of the proposed approach, we compare the performance with the traditional baseline approach presented in Fig. 4, in terms of accuracy and computations. The baseline approach for medical image analysis requires a separate model (encoder-decoder) to learn each pre-processing task in addition to $N$ encoders and $N$ decoders to learn

$N$ target tasks. On the other hand, the proposed approach requires only a single encoder and $N$ lightweight task-specific decoders for $N$ target tasks, resulting in significant savings in time and the number of training parameters.

Our experiments provide empirical answers to the following questions:

- What advantages does the proposed sequence of training offer in comparison to the baseline approach in terms of performance and computational efficiency?
- What fine-tuning strategy should be used for the target medical image tasks?

To report the performance of target tasks, we used F-score, Matthews Correlation Coefficient (MCC), and the accuracy. We also used the intersection over union (IoU) metric to report the performance of the segmentation task. To demonstrate the computational efficiency, we reported the computational times and the training parameters for all models.

We trained all models using a Windows system with the following configuration: (1) Intel Xeon CPU E3–1275 v6 3.80 GHz and (2) NVIDIA GeForce GTX 1050 Ti. Keras DL framework with Tensorflow was used for model training and evaluation. For hyperparameter optimization, Talos[a] library with Keras was used. The code of this work is made publicly available at UMS-Rep Github Page.

## 4.1. CXR imaging modality

Although the high-level (i.e., task-specific) features of CXR target tasks are relatively different, CXR images have similar low-level features (e.g., edges and blobs). Hence, UMS–Rep can be constructed to learn CXR low-level features and then shared among different CXR tasks.

**4.1.1. UMS-rep construction and pre-processing—**UMS-Rep can be trained to learn CXR low-level features and pre-processing tasks using different learning paradigms. In this paper, we used unsupervised learning to train the UMS-Rep backbone for image denoising. Noise reduction is an essential pre-processing task that improves the quality of medical images while maintaining spatial resolution. The negative impact of image noise in subsequent tasks such as classification and segmentation has been reported in several works (e.g., Refs. [26–28]. Further, the impact of different types of noise, such as Gaussian noise and speckle noise, on medical image segmentation has been reported in Ref. [28]. Therefore, we propose to train a UMS-Rep that learns this important pre-processing task to enhance the performance of the subsequent target tasks.

A convolutional denoising autoencoder (CDAE or $\text{UMS-Rep}_{CXR-Denoising}$) was trained on the RSNA CXR dataset (see Table 1). The dataset was split at the patient-level into 70%, 20%, 10% for training, validation, and testing, respectively. As depicted in Fig. 6, $\text{UMS-Rep}_{CXR-Denoising}$ has four convolutional layers (3× 3) in the encoder to compress the input to its latent space representation. We used strided convolutions instead of max-pooling layers to increase the expressive capacity of the network, which would improve

[a]https://github.com/autonomio/talos.

the overall performance without increasing the number of parameters as discussed in Ref. [29]. We used batch normalization layers to improve generalization and ReLU to speed up model training, resulting in faster convergence. The symmetrical decoder has upsampling layers to reconstruct the input from the latent space representation. We optimized CDAE ($\text{UMS-Rep}_{CXR-Denoising}$) to minimize the mean squared error (MSE) and reconstruct the input with minimal reconstruction error. The kernel size, stride, and optimizer for CDAE($\text{UMS-Rep}_{CXR-Denoising}$) are 3, 2, and RMSprop, respectively. We trained the model with 16 batch size, an initial learning rate of $1 \times 10^{-3}$ that is reduced when the loss plateau, and early stopping.CDAE parameters were selected using Talos.

To generate noisy images for training, we added Gaussian noise with different ranges of standard deviations and Poisson noise to the images and used CDAE to reconstruct clean images. The reason behind adding the noise to the images is to demonstrate that $\text{UMS-Rep}_{CXR-Denoising}$ can learn to denoise Gaussian noise or any types of noise before sharing it among target tasks. Further, adding noise to the training process of $\text{UMS-Rep}_{CXR-Denoising}$ reduces overfitting and introduces regularization. Hence, we degraded the original images by Gaussian noise with different standard deviations ($\sigma = [10;\ 20;\ 30;\ 40;\ 50]$) as well as Poisson noise ($\mu = \sigma^2$) and reported the performance in Table 2.

As shown in Table 2, we quantitatively measured the performance of reconstruction using the peak signal-to-noise ratio (PSNR), the structural similarity index (SSIM), and the multi-Scale structural similarity index (MS-SSIM). The typical range of PSNR is between 30 and 50 dB while the range of SSIM and MS-SSIM extends between −1 and +1, where higher is better [30]. The PSNR, SSIM, and MS-SSIM values in Table 2 are computed for the test images. These results indicate that the trained $\text{UMS-Rep}_{CXR-Denoising}$ faithfully reconstructed the images and learned to extract useful CXR features while ignoring noise information.

Once $\text{UMS-Rep}_{CXR-Denoising}$ is constructed, it is used as a shared representation among target CXR tasks with diverse or similar annotations as depicted in Fig. 6.

#### 4.1.2. Target tasks fine-tuning—

We used $\text{UMS-Rep}_{CXR-Denoising}$ as a shared encoder to simultaneously fine-tune target tasks with diverse and similar annotations. Examples of CXR target tasks with diverse annotations include lung segmentation (pixel-level) and abnormality classification (image-level). We used three tasks with image-level annotations as the homogeneous target tasks with similar annotations. These tasks are: (1) bacterial pneumonia vs normal classification, (2) viral pneumonia vs normal classification, and (3) bacterial vs viral pneumonia classification.

Before fine-tuning the target tasks, task-specific layers (heads) are appended to $\text{UMS-Rep}_{CXR-Denoising}$ as presented in Fig. 6. Note that in the case of the lung segmentation, we used the entire $\text{UMS-Rep}_{CXR-Denoising}$ and replaced the final convolutional layer with the one that has a single neuron to generate the binary lung masks. As for the classification tasks, only the encoder part of the optimized CDAE($\text{UMS-Rep}_{CXR-Denoising}$) was instantiated and appended with the following layers: GAP, FC, D, and SM layers.

After attaching task-specific heads to the shared UMS-Rep$_{CXR-Denoising}$, we experimented with three fine-tuning strategies to investigate their impact on the performance of the target tasks. In all experiments, the task-specific layers for the classification target tasks were fine-tuned to minimize the CCE loss while the task-specific layers for the segmentation tasks were fine-tuned to minimize the loss in equation (1). In the case of joint fine-tuning strategy, we summed the loss functions of the target tasks. The weights for task-specific losses are learned through empirical experiments (all weights are included in the code). In the case of alternating fine-tuning strategy, the loss for each target task is updated alternately as described in Section 2.3. For the independent fine-tuning strategy, we freeze the weights of UMS-Rep$_{CXR-Denoising}$ and independently minimize the loss function of the target tasks.

**Target CXR Tasks with Diverse Annotations:** These tasks were fine-tuned using the Shenzhen CXR dataset (see Table 1). We divided this dataset into different folds to perform 10-fold cross-validation. For testing, we used the Montgomery CXR dataset (see Table 1). The performance of fine-tuning lung segmentation and abnormality classification tasks based on the shared UMS-Rep$_{CXR-Denoising}$ is summarized in Table 3. As shown in the table, the independent fine-tuning strategy achieved slightly higher performance than the alternating strategy. However, the independent strategy increases the performance by a large margin as compared to the joint strategy. Statistically, the difference between the independent and alternating strategies is not significant using McNemar's test ($p < 0.05$). However, the difference between the independent and joint strategies is statistically significant (McNemar's test, $p < 0.05$). These results confirm the impact of the fine-tuning strategy on the performance and suggest that the joint strategy leads to lower performance when used with heterogeneous target tasks. This might be attributed to the negative transfer between irrelevant or conflicting tasks. Specifically, the classification task might transfer irrelevant information to the segmentation task or vice versa. Based on these results, we can conclude the superiority of independent strategy when fine-tuning heterogeneous tasks in CXR images.

We also compared the performance of the proposed approach with the baseline approach. To provide a fair comparison, CXR images are first denoised using a separate image denoising model (model 1), then we used the generated denoised images as input to the lung segmentation model (model 2) and abnormality classification model (model 3). These three models have the same architecture as in the proposed approach, but trained separately with random initialization. As shown in the last row of Table 3, the proposed approach outperformed the baseline approach in both target tasks. Statistically, the difference between the independent fine-tuning strategy and baseline approach is statistically significant (McNemar's test, $p < 0.05$). Note that the performance of the baseline segmentation model is higher than the joint fine-tuning strategy due to the negative transfer that might occur while minimizing the weighted sum of segmentation and classification tasks.

Table 4 shows the computational time and training parameters for the proposed and baseline approach. As shown in the table, the proposed approach significantly decreases the total training time and parameters as compared to the baseline approach, where separate models are used for separate tasks. Note that although the number of training parameters for the segmentation task does not change, the training time significantly decreased from 148.02 to 8.11 min. This demonstrates that the proposed approach, which uses the shared representation $\text{UMS-Rep}_{CXR-Denoising}$) and its weight, leads to faster training convergence. Precisely, the weights of the shared $\text{UMS-Rep}_{CXR-Denoising}$ resulted in improved initialization and faster learning/convergence for the target tasks.

**Target CXR Tasks with Similar Annotations:** These tasks were fine-tuned using the train set of the Pediatric pneumonia dataset (Table 1), which was further divided into multiple folds to perform 10-fold cross-validation analysis. We then used the hold-out test set for reporting the performance. Recall that this dataset contains three classes: normal, bacterial pneumonia, and viral pneumonia. Examples of these classes are given in Fig. 2.

Table 5 shows the performance of fine-tuning the classification tasks based on the shared $\text{UMS-Rep}_{CXR-Denoising}$. As shown in the table, the joint fine-tuning strategy achieved higher performance than both the alternating and independent strategies in most cases. Statistically, the difference in performance between the joint and independent strategies is significant (McNemar's test, $p < 0.05$), except for bacterial vs normal. These results suggest the superiority of the joint strategy for fine-tuning medical image tasks with diverse annotations. This can be explained by the ability of the joint loss function to capture the task differences and implicitly model the task relationships; i.e., the inter-task relationships among target task 1 (bacterial vs normal), target task 2 (viral vs normal), and target task 3 (bacterial vs viral) (all use image-label) are better captured using the joint strategy. Similarly, the alternating fine-tuning strategy achieved better performance than the independent strategy as it allows, by alternately optimizing $\Theta_{src}$ and $\Theta_{T_i}$, to transfer some of the information from each task to the other. These results suggest the superiority of joint and alternating strategies for cooperative medical image tasks with similar annotations.

We also compared the performance of the proposed approach with the baseline, where separate end-to-end models with random initialization are used for separate target tasks. Specifically, we first used a separate image denoising model to provide a fair comparison with our approach. We then used the output denoised images as input to three separate models, one for each target task. These three models have the same architectures as in the proposed approach, but trained separately with random initialization. As shown in the last row of Table 5, the proposed approach outperformed the baseline approach in all target tasks. Statistically, the difference in performance between the joint fine-tuning strategy and baseline approach is significant for all tasks ($p < 0.05$). These results indicate that the proposed approach, by transferring the knowledge from a shared modality-specific source to target tasks, can improve the generalizability and lead to better performance as compared to the baseline approach. Further, the results suggest that the joint fine-tuning of similar tasks can enhance the overall performance while reducing the learning time as shown in Table 6.

Table 6 shows the computational time and training parameters for the proposed and baseline approach. As shown in the table, the proposed approach significantly decreases the total training time and parameters as compared to the baseline approach, where separate models (encoder-decoder) are used for separate tasks. Specifically, the proposed approach reduces the number of training parameters by ~ 81% as compared to the baseline approach.

Similarly, the proposed approach leads to a reduction by 86% in the computational time, and faster convergence. Finally, it is important to note that no matter the number of target tasks, our approach would lead to lower computational times and parameters as compared to the baseline approach. This is attributed to the fact that it 1) uses a single shared encoder and $N$ lightweight task-specific decoders instead of $N$ encoder-decoder for $N$ tasks and 2) the weights of the shared UMS-Rep improved initialization and leads to faster convergence for all target tasks.

**4.1.3. Derivable task—**After fine-tuning the target classification task, we can visualize the output of the target task by simply computing the gradient of the winning class with respect to the last convolutional layer of the target task as described in Ref. [31]. Then, we compute the average, weigh it against the output of this layer, and normalize it between 0 and 1 to generate the heatmap. This map visualizes discriminative areas the head looks at when classifying into normal and abnormal. As this visualization task relies on the output of the classification head, we can define it as a task derivable from the target task. Fig. 7 presents the visualization of the abnormality classification task.

## 4.2. Doppler echo imaging modality

Although the high-level (i.e., task-specific) features of Doppler echo target tasks might differ, they have relatively similar low-level features. Hence, a single UMS-Rep can be trained to learn these low-level features and then shared among different target tasks that learn task-specific features.

**4.2.1. UMS-rep construction—**In this evaluation, we used supervised learning to construct the UMS-Rep on Doppler echo dataset using the annotations of the flow classification task (see Table 1). We used 70% of the dataset to construct the UMS-Rep. As shown in Fig. 8, this UMS-Rep has six convolutional layers (3× 3) with the same padding. Dilated kernels (size 2) were used in the fourth, fifth, and sixth convolutional layers to capture wider context at a reduced computational cost. We used ReLU after each convolutional layer to speed-up model training and convergence. The output of the deepest convolutional layer from the optimized CNN was fed to the GAP and FC layers. To reduce overfitting, the output of the FC layer was fed to a dropout layer (0.5). The last FC layer has three neurons corresponding to three flow classes: TR, MV, and MA. We used Talos to select the optimal parameters from the following ranges: kernel size [3, 5, 7], dilation rate [2, 3], dropout ratio [0.1, 0.3, 0.5], optimizer [SGD, Adam, RMSprop], and batch size [16, 32, 64]. Talos outputs 3, 2, 0.5, Adam, and 16 for kernel size, dilation rate, dropout ratio, optimizer, and batch size, respectively. We used 64 epochs and optimized the model to minimize the CCE loss.

Once the optimized $\text{UMS-Rep}_{echo}$ is constructed, it is used as a shared encoder among target echo tasks with diverse and similar annotations as depicted in Fig. 8.

**4.2.2. Target tasks fine-tuning—**We used $\text{UMS-Rep}_{echo}$ as a shared backbone to simultaneously fine-tune target tasks with diverse and similar annotations. Examples of echo Doppler tasks with diverse annotations include envelope segmentation (pixel-level), flow classification (image-level), and image quality assessment (image-level). We used three classification tasks with image-level annotations as the homogeneous target tasks with similar annotations. These classification tasks are: (1) TR vs MV, (2) TR vs MA, and (3) MV vs MA.

We fine-tuned the target tasks using the remaining 30% of echo Doppler dataset, where 20% was used for training and validation (10-folds cross-validation) and 10% was used for testing. Before fine-tuning the target tasks, task-specific layers (heads) are appended to $\text{UMS-Rep}_{echo}$ as presented in Fig. 8. For the envelope segmentation task, the $\text{UMS-Rep}_{echo}$ was truncated at the deepest convolutional layer, and a symmetrical decoder was appended as shown in Fig. 8. This head was fine-tuned using Adam optimizer to optimize the loss in equation (1). To create the task specific-layers (heads) for the classification tasks, we appended GAP, FC, D, and SM layers to $\text{UMS-Rep}_{echo}$ as shown in Fig. 8. The heads for the classification tasks were fine-tuned using SGD optimizer to minimize the CCE loss. After attaching task-specific heads to the shared $\text{UMS-Rep}_{echo}$, we experimented with three fine-tuning strategies to investigate their impact on the performance of the target tasks.

**Target Echo Tasks with Diverse Annotations:** Table 7 shows the performance of fine-tuning envelope segmentation, flow classification, and quality assessment tasks. Observe that the independent strategy increases the performance by a large margin as compared to the joint strategy. Statistically, the difference between the independent and joint strategies is significant (McNemar's test, $p < 0.05$). These results suggest the superiority of independent strategy when fine-tuning echo tasks with diverse annotations. We also compared the performance of the proposed approach with the baseline approach. In the baseline approach, the segmentation model as well as the quality assessment and flow classification models have the same architecture as in the proposed approach, but trained separately with random initialization. As shown in the last row of Table 7, the proposed approach outperformed the baseline approach. The difference between the independent fine-tuning strategy and baseline approach is statistically significant (McNemar's test, $p < 0.05$). Again, the joint fine-tuning strategy does not improve the performance of the segmentation task. We believe this could be interpreted as being a result of the negative transfer among irrelevant tasks while learning a single joint function.

Table 8 shows the computational time and training parameters for the proposed and baseline approach. As shown in the table, the proposed approach significantly decreases the total computational time (↓ ~ 80%) and training parameters (↓ ~ 53%) as compared to the baseline approach, where separate models are used for separate tasks. Interestingly, although the number of training parameters for the segmentation task does not change, the training time decreased from 130.02 to 5.80 min (~ 96%) as a result of using $\text{UMS-Rep}_{echo}$ backbone. Precisely, the weights of the shared $\text{UMS-Rep}_{echo}$ resulted in improved initialization and faster convergence. These results demonstrate the benefits of using $\text{UMS-Rep}_{echo}$ as a shared backbone.

**Target Echo Tasks with Similar Annotations:** Table 9 shows the performance of fine-tuning the classification tasks based on the shared $\text{UMS-Rep}_{echo}$. We can conclude from the table that the joint fine-tuning strategy achieved higher performance in most cases. Although the difference in the performance between the joint and alternating strategies is not significant in most cases, it is statistically significant between the joint and independent (McNemar's test, $p < 0.05$). These results suggest that optimizing a single joint loss function for homogeneous target tasks leads to better overall performance. In other words, the results suggest the superiority of the joint strategy for fine-tuning medical image tasks with similar annotations as it has the ability to capture the similarities and differences among similar tasks. We also compared the performance of the proposed approach with the baseline approach. As shown in the last row of Table 9, the proposed approach outperformed the baseline approach. The statistical difference between the joint fine-tuning strategy and baseline approach is significant ($p < 0.05$). Also, the proposed approach decreases the total training parameters and computational time, as compared to the baseline approach (Table 10). Although our approach decreases the computational time in Table 10 only slightly (~ 8%), it is important to note that the reduction in the computational time will continue as we add more target tasks; e.g., adding another classification task would make the total computational time for our approach and baseline 34.94 (33.69 + 1.25) minutes and 46.02 (36.77 + 9.25) minutes respectively, and lead to ~ 24% reduction. This indicates that our approach will always have lower computations as it involves sharing a single decoder among target tasks, which enhances initialization and leads to faster convergence.

#### 4.2.3. Derivable task—

After learning the target tasks, we derived a recommendation task. This derivable task is generated based on combining the outputs of the flow classification and quality assessment tasks. This recommendation is used to decide if the image is suitable for further analysis. Currently, the echocardiographer manually excludes low-quality TR flow images with unclear envelopes from further analysis because they decrease the accuracy of measurements. Fig. 9 shows examples of the recommendation task.

### 4.3. Discussion

To resolve the challenges posed by applying the traditional training approach on medical images, we proposed a novel sequence of training that involves constructing UMS-Rep followed by using it as a shared backbone to simultaneously fine-tune target tasks with similar or diverse annotations.

Our experimental results show that the proposed approach improved the generalization, and performance (up to 9%) of target tasks by transferring the knowledge from a modality-specific shared source to these tasks. Further, using a single representation to fine-tune multiple target tasks reduced the computations by a large margin and prevented unnecessary repetitions of training individual models for individual pre-processing and visual analysis tasks; i.e., single encoder and $N$ lightweight task-specific heads instead of $N$ encoder-decoder models for $N$ tasks. Our results suggest the superiority of the independent fine-tuning strategy for heterogeneous medical image tasks, and the joint strategy (followed by alternating) for homogeneous medical image tasks. This can be explained by the ability of the joint loss function to capture the task differences and implicitly model inter-task relationships between similar tasks. Similarly, the alternating strategy allows discovering some of commonality between target tasks. As these results are consistent in two imaging modalities (CXR and echo Doppler), we may conclude that independent strategy should be used to fine-tune heterogeneous target tasks and joint (or alternating) strategy should be used to fine-tune homogeneous target tasks. Finally, it is important to note the benefit of the proposed sequence of training in reducing overfitting as reported in [32].

The proposed approach can improve the efficiency and performance of medical image tasks as follows. First, transferring the knowledge from shared modality representation (which can be trained without labels) to the segmentation task improves generalization and leads to faster convergence for the target tasks. Second, our results demonstrated that the fine-tuning strategy greatly impacts the performance of the target task, and suggested to avoid using joint strategy for fine-tuning conflicting tasks. In other words, independent fine-tuning using a shared modality-specific representation leads to better performance when used with heterogeneous tasks while joint or alternating fine-tuning leads to better performance when used with homogeneous or similar tasks.

Although we only demonstrated the feasibility of our approach to learn a single pre-processing task (i.e., noise reduction) and some relevant tasks in medical image analysis, the proposed approach is flexible and can be easily extended to integrate other pre-processing tasks (e.g., super resolution [33]) and learn any number of target tasks. For example, a bounding box regression task can be added to the UMS-Rep by appending a task-specific head that has region pooling layers for extracting region-wise features and FC layers for box classification and regression. The proposed approach can also be extended to analyze 3D images using a 3D CNN as the shared representation and task-specific layers.

## 5. Conclusions

In this paper, we proposed a unified modality-specific approach under the MTL framework where the encoder is shared across different target tasks with diverse and similar annotations. We applied the proposed approach to two medical imaging modalities, namely CXR and echo Doppler. We also explored different strategies for fine-tuning the target tasks to investigate the impact of the utilized strategy on their performance. To the best of our knowledge, the problem of learning to simultaneously transfer knowledge from shared representation to multiple target tasks has seldom been studied in medical images. Our experiments show that the proposed approach can improve the generalization and performance of target tasks, while providing computational efficiency.

## Acknowledgments

This research is supported by the Intramural Research Program of the National Library of Medicine (NLM) and the National Institutes of Health (NIH).

## References

[1]. Zhou SK, et al. Deep learning for medical image analysis. Academic Press; 2017.

[2]. Litjens G, et al. A survey on deep learning in medical image analysis. Med Image Anal 2017;42:60–88. [PubMed: 28778026]

[3]. Mahapatra D, Bozorgtabar B. Progressive generative adversarial networks for medical image super resolution. 2019. arXiv preprint arXiv:1902.02144.

[4]. Gulati T, et al. Application of an enhanced deep super-resolution network in retinal image analysis. In: Ophthalmic technologies XXX. International Society for Optics and Photonics; 2020. 112181K.

[5]. Christodoulidis A, et al. A multi-scale tensor voting approach for small retinal vessel segmentation in high resolution fundus images. Comput Med Imag Graph 2016;52:28–43.

[6]. Balakrishnan G, et al. An unsupervised learning model for deformable medical image registration. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. p. 9252–60.

[7]. Thung KH, Wee CY. A brief review on multi-task learning. Multimed Tool Appl 2018;77:29705–25.

[8]. Dong D, et al. Multi-task learning for multiple language translation. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing. vol. 1; 2015. p. 1723–32. Long Papers).

[9]. Zhou Z, et al. Models genesis: generic autodidactic models for 3d medical image analysis. In: International conference on medical image computing and computer-assisted intervention. Springer; 2019. p. 384–93.

[10]. Moeskops P, et al. Deep learning for multi-task medical image segmentation in multiple modalities. In: International conference on medical image computing and computer-assisted intervention. Springer; 2016. p. 478–86.

[11]. Kisilev P, Sason E, Barkan E, Hashoul S. Medical image description using multi-task-loss cnn. In: Deep Learning and Data Labeling for Medical Applications. Springer; 2016. p. 121–9.

[12]. Zhang Z, et al. Facial landmark detection by deep multi-task learning. In: European conference on computer vision. Springer; 2014. p. 94–108.

[13]. Elhoseiny M, et al. Convolutional models for joint object categorization and pose estimation. 2015. arXiv preprint arXiv:1511.05175.

[14]. Kendall A, et al. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2018. p. 7482–91.

[15]. Chen C, et al. Multi-task learning for left atrial segmentation on ge-mri. In: International workshop on statistical atlases and computational models of the heart. Springer; 2018. p. 292–301.

[16]. Bai W, et al. Self-supervised learning for cardiac mr image segmentation by anatomical position prediction. In: International conference on medical image computing and computer-assisted intervention. Springer; 2019. p. 541–9.

[17]. Teichmann M, et al. Multinet: real-time joint semantic reasoning for autonomous driving. In: 2018 IEEE intelligent vehicles symposium (IV). IEEE; 2018. p. 1013–20.

[18]. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 779–88.

[19]. Zou Y, et al. Df-net: unsupervised joint learning of depth and flow using cross-task consistency. In: Proceedings of the European conference on computer vision. ECCV; 2018. p. 36–53.

[20]. Luong MT, Le QV, Sutskever I, Vinyals O, Kaiser L. Multi-task sequence to sequence learning. 2015. arXiv preprint arXiv:1511.06114.

[21]. Standley T, Zamir A, Chen D, Guibas L, Malik J, Savarese S, 2020. Which tasks should be learned together in multi-task learning?, in: International Conference on Machine Learning, PMLR. pp. 9120–9132.

[22]. Shih G, Wu CC, Halabi SS, Kohli MD, Prevedello LM, Cook TS, Sharma A, Amorosa JK, Arteaga V, Galperin-Aizenberg M, et al. Augmenting the national institutes of health chest radiograph dataset with expert annotations of possible pneumonia. Radiology: Artif Intell 2019;1:e180041.

[23]. Jaeger S, Candemir S, Antani S, Wáng YXJ, Lu PX, Thoma G. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. Quant Imag Med Surg 2014;4:475.

[24]. Kermany DS, Goldbaum M, Cai W, Valentim CC, Liang H, Baxter SL, McKeown A, Yang G, Wu X, Yan F, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell 2018;172:1122–31. [PubMed: 29474911]

[25]. Rajaraman S, et al. Visualization and interpretation of convolutional neural network predictions in detecting pneumonia in pediatric chest radiographs. Appl Sci 2018;8:1715. [PubMed: 32457819]

[26]. Cheng HD, et al. Automated breast cancer detection and classification using ultrasound images: a survey. Pattern Recogn 2010;43:299–317.

[27]. Mahmood NH, et al. Comparison between median, unsharp and wiener filter and its effect on ultrasound stomach tissue image segmentation for pyloric stenosis. Int J Appl Sci Technol 2011;1.

[28]. Nyma A, et al. A hybrid technique for medical image segmentation. J Biomed Biotechnol 2012;2012:830252. 10.1155/2012/830252. Epub 2012 Jul 30. [PubMed: 22919276]

[29]. Zeiler MD, et al. Deconvolutional networks. In: 2010 IEEE Computer Society Conference on computer vision and pattern recognition. IEEE; 2010. p. 2528–35.

[30]. Welstead ST. Fractal and wavelet image compression techniques. Washington: SPIE Optical Engineering Press Bellingham; 1999.

[31]. Chen Z, et al. Gradnorm: gradient normalization for adaptive loss balancing in deep multitask networks. In: International conference on machine learning; 2018. p. 793–802.

[32]. Baxter J. A bayesian/information theoretic model of learning to learn via multiple task sampling. Mach. Learn. 1997;28:7–39.

[33]. Zamzmi Ghada, Rajaraman Sivaramakrishnan, Antani Sameer. Accelerating super-resolution and visual task analysis in medical images. In: Appl. Sci. 10. Multidisciplinary Digital Publishing Institute; 2020. p. 4282.

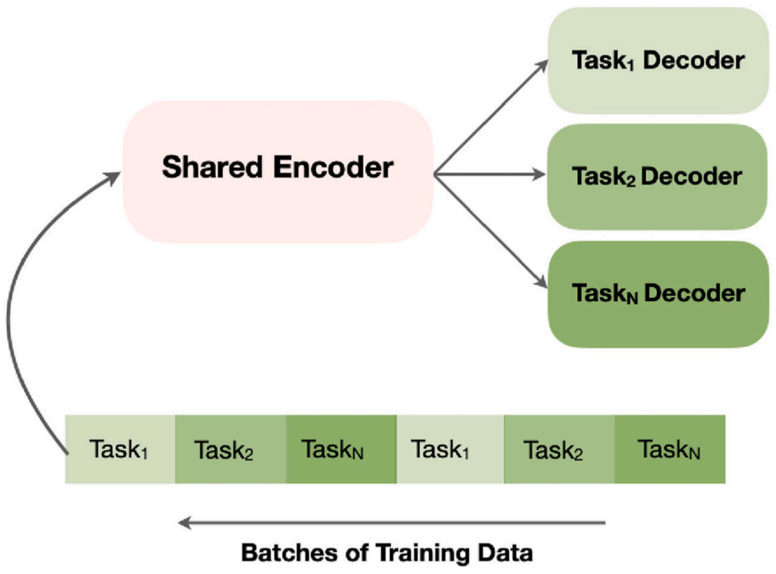

**Fig. 1.**
The fine-tuning approach in alternating strategy.

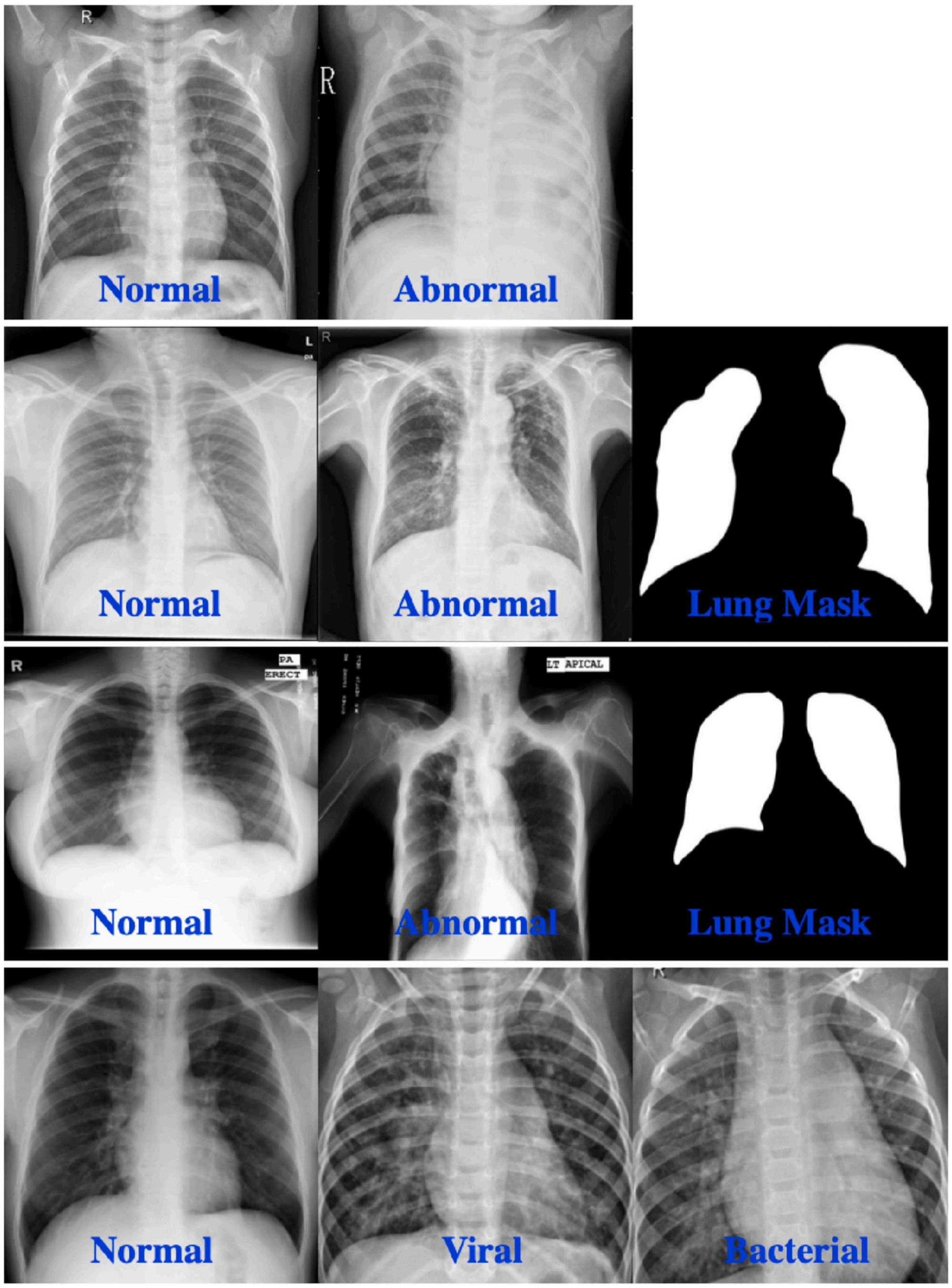

**Fig. 2.**
Samples from CXR datasets and their annotations; 1st row: RSNA, 2nd row: Shenzhen TB CXR, 3rd row: Montgomery TB CXR, and 4th row: Pediatric pneumonia CXR.

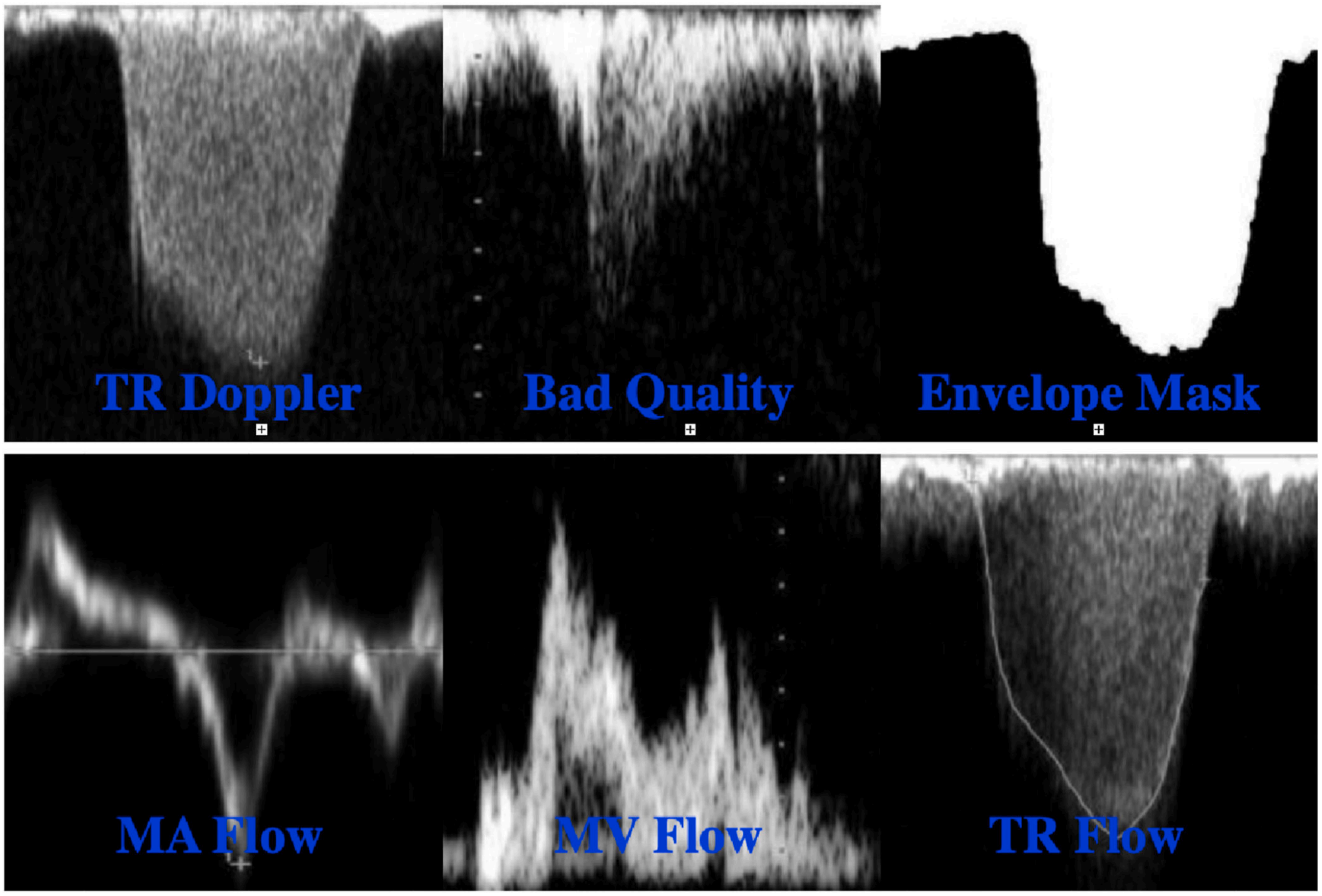

**Fig. 3.**
Samples from echo Doppler datasets and their annotations; 1st row: Doppler tasks and 2nd row: Doppler flows.

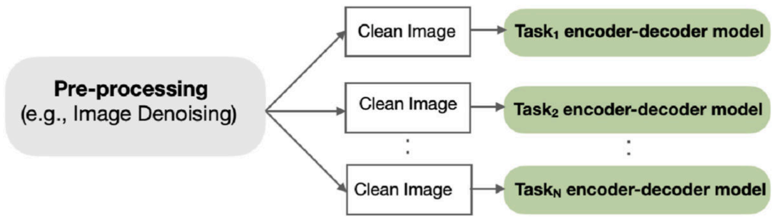
Pre-processing
(e.g., Image Denoising)
Clean Image
Clean Image
Clean Image
Task₁ encoder-decoder model
Task₂ encoder-decoder model
Taskₙ encoder-decoder model

**Fig. 4.**
The traditional sequence of training in medical image analysis involves using individual models for individual tasks. For example, denoising medical images prior to the analysis requires using a separate image denoising model to generate clean images followed by using these images as input to individual models.

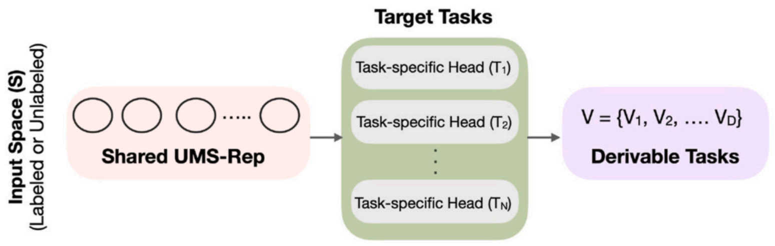

**Fig. 5.**
Our proposed sequence of simultaneously training UMS-Rep with pre-processing tasks (pink box). UMS-Rep is then shared among target tasks with heterogeneous or homogeneous annotations. $T_{1:N} = \{T_1, T_2, .., T_N\}$ and $V_{1:D} = \{V_1, V_2, ..., V_D\}$ represent target tasks (green box) and derivable tasks (purple task), respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

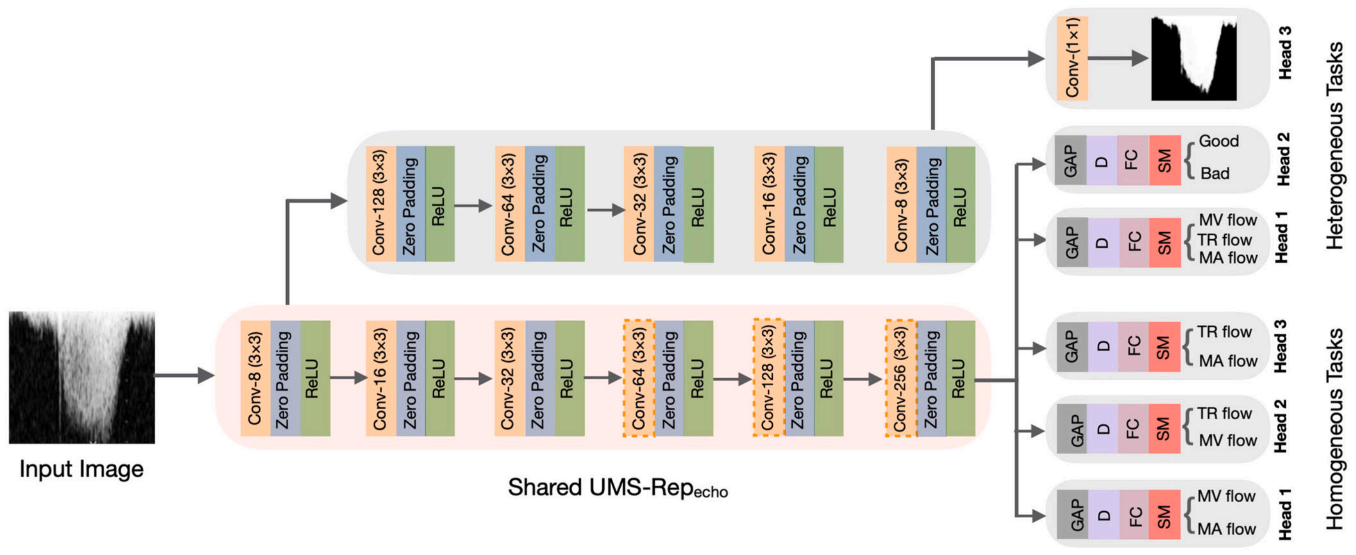

**Fig. 6.**

UMS-Rep$_{CXR-Denoising}$ shared backbone and task-specific heads for heterogeneous and homogeneous tasks. GAP, D, FC, SM indicate global average pooling, dropout, fully connected, and softmax layers, respectively. Dashed orange boxes indicate convolutional layers with dilation. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

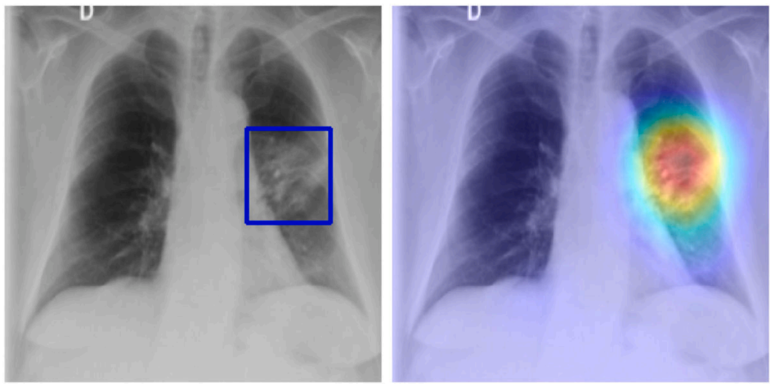

**Fig. 7.**
Visual explanation task. Left: original image with GT annotation (bounding box in blue). Right: visualization on a testing image. High activation (red pixels) is observed in the affected ROIs. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

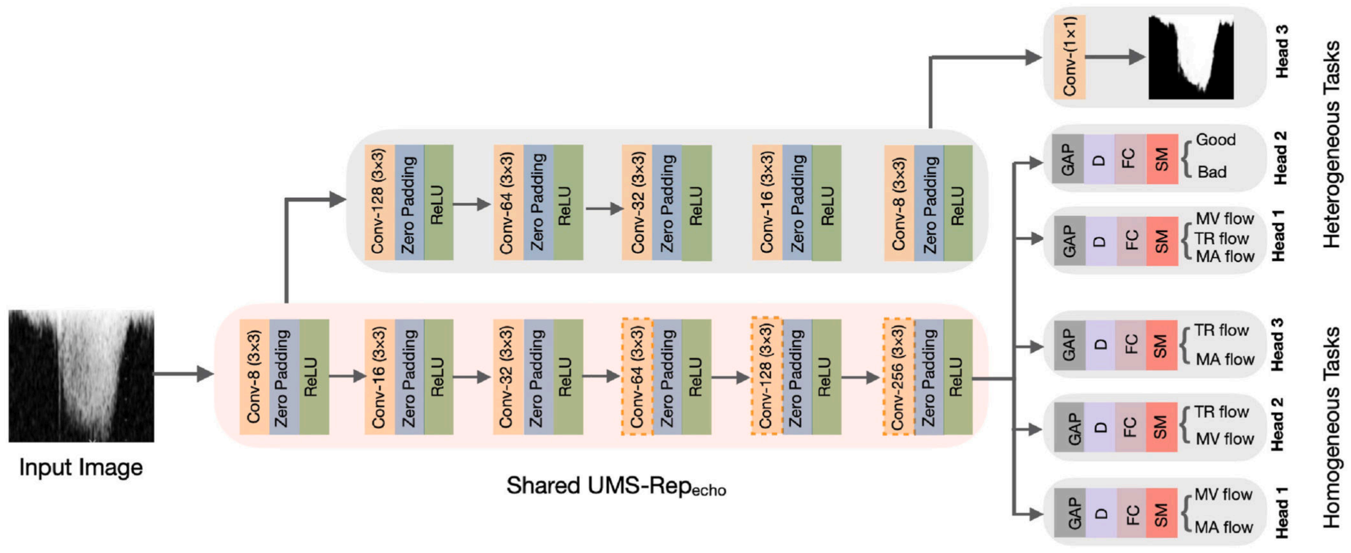

**Fig. 8.**

UMS-Rep$_{echo}$ (pink box) is shared among target tasks. GAP, D, FC, SM indicate global average pooling, droput, fully connected, and softmax layers, respectively. Dashed orange boxes indicate convolutional layers with dilation. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

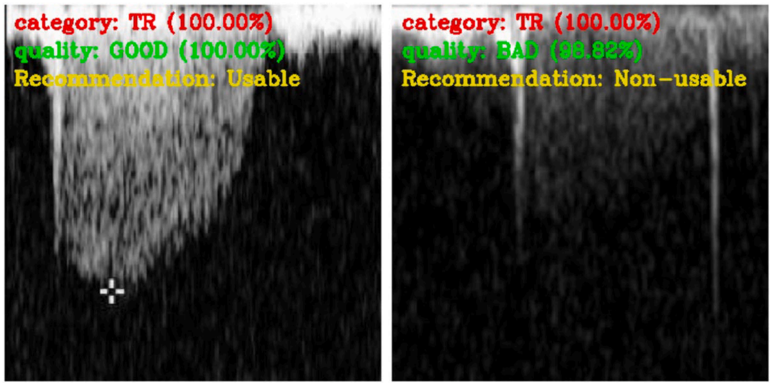

**Fig. 9.**

Derivable recommendation task. Examples of merging the outputs of tasks to select only good quality TR images.

**Table 1**

Summary of CXR and Doppler echo Datasets.

| Modality | Dataset | Total Images | GT Labels | Resolution |
|---|---|---|---|---|
| CXR | RSNA | 26,684 | Normal<br>Abnormal | 1024 × 1024 |
| | Shenzhen | 662 | Normal (336)<br>Abnormal (326)<br>Lung mask (566) | 3000 × 3000 |
| | Montgomery | 138 | Normal (80)<br>Abnormal (58)<br>Lung mask (138) | 4020 × 4892 |
| | Pediatric Pneumonia | Train: 5232 | Bacterial (2,538)<br>Viral (1345)<br>Normal (1,349) | – |
| | | Test: 624 | Bacterial (242)<br>Viral (148)<br>Normal (234) | |
| Doppler Echo | Doppler Tasks | 2444 | Doppler Flow (2444)<br>Image Quality (814)<br>Envelope mask (2444) | – |
| | Doppler Flows | | MV Flow (855)<br>MA Flow (490)<br>TR Flow (1099) | |

**Table 2**

Performance of customized $\text{UMS-Rep}_{CXR-Denoising}$ for image denoising.

| Noise | PSNR | SSIM | MS-SSIM |
|---|---|---|---|
| Gaussian ($\sigma = 10$) | 39.05 dB | 0.96 | 0.99 |
| Gaussian ($\sigma = 20$) | 35.38 dB | 0.91 | 0.98 |
| Gaussian ($\sigma = 30$) | 32.75 dB | 0.89 | 0.95 |
| Gaussian ($\sigma = 40$) | 30.54 dB | 0.85 | 0.91 |
| Gaussian ($\sigma = 50$) | 28.35 dB | 0.79 | 0.88 |
| Poisson ($\mu = \sigma^2$) | 33.37 dB | 0.93 | 0.97 |

## Table 3

Performance of CXR target tasks with diverse annotations. Recall that we used three strategies to fine-tune the heads attached to UMS-Rep$_{CXR-Denoising}$. Bold values indicate highest performance.

| Method | Tuning Strategy | Target Task | Accuracy | F-score | IoU | MCC |
|---|---|---|---|---|---|---|
| | Independent | Lung Segmentation | **0.99** | **0.96** | **0.98** | - |
| | Alternating | Lung Segmentation | 0.99 | 0.95 | 0.96 | - |
| | Joint | Lung Segmentation | 0.92 | 0.92 | 0.94 | - |
| UMS-Rep$_{CXR-Denoising}$ | | | | | | |
| | Independent | Abnormality Classification | **0.86** | **0.83** | - | **0.68** |
| | Alternating | Abnormality Classification | 0.85 | 0.82 | - | 0.66 |
| | Joint | Abnormality Classification | 0.83 | 0.80 | - | 0.63 |
| Baseline | Separate Model | Lung Segmentation | 0.97 | 0.91 | 0.95 | - |
| | Separate Model | Abnormality Classification | 0.80 | 0.80 | - | 0.60 |

**Table 4**

Heterogeneous CXR target tasks: summary of computational time and training parameters for the proposed approach and the baseline approach.

| Method | Task | Computational Time | Training Parameters |
|---|---|---|---|
| Proposed | Shared Source (UMS-Rep$_{CXR-Denoising}$) | 302.20 min | 800,067 |
| | Lung Segmentation Head | 8.11 min | 786,497 |
| | Abnormality Classification Head | 2.51 min | 295,715 |
| | Shared Source & 2 lightweight Heads | 312.82 min | 1,882,279 |
| Baseline | Separate Denoising Model | 302.20 min | 800,067 |
| | Separate Lung Segmentation Model | 148.02 min | 786,497 |
| | Separate Abnormality Classification Model | 9.25 min | 1,573,506 |
| | 3 Separate end-to-end Models for 3 Tasks | 459.47 min | 3,160,070 |

## Table 5

Performance of homogeneous CXR target tasks using the proposed method and the baseline. Recall that we used three strategies to fine-tune the heads attached to UMS-Rep$_{CXR-Denoising}$. Bold values indicate best performance.

| Method | Tuning Strategy | Target Task | Accuracy | F-score | MCC |
|---|---|---|---|---|---|
| | Independent | Classification (Bacterial vs Normal) | 0.95 | 0.95 | 0.90 |
| | Alternating | Classification (Bacterial vs Normal) | **0.96** | **0.96** | **0.92** |
| | Joint | Classification (Bacterial vs Normal) | 0.96 | 0.95 | 0.91 |
| | Independent | Classification (Viral vs Normal) | 0.95 | 0.92 | 0.88 |
| UMS-Rep$_{CXR-Denoising}$ | Alternating | Classification (Viral vs Normal) | 0.96 | 0.93 | 0.90 |
| | Joint | Classification (Viral vs Normal) | **0.97** | **0.94** | **0.92** |
| | Independent | Classification (Bacterial vs Viral) | 0.81 | 0.76 | 0.56 |
| | Alternating | Classification (Bacterial vs Viral) | 0.82 | 0.77 | 0.58 |
| | Joint | Classification (Bacterial vs Viral) | **0.83** | **0.80** | **0.62** |
| Baseline | Separate Model | Classification (Bacterial vs Normal) | 0.94 | 0.94 | 0.89 |
| | Separate Model | Classification (Viral vs Normal) | 0.93 | 0.90 | 0.86 |
| | Separate Model | Classification (Bacterial vs Viral) | 0.80 | 0.75 | 0.54 |

**Table 6**

Homogeneous CXR target tasks: summary of computational time and training parameters for the proposed and baseline approaches.

| Method | Task | Computational Time | Training Parameters |
|---|---|---|---|
| Proposed | Classification Head 1 (Bacterial vs Normal) | 1.90 min | 295,715 |
| | Classification Head 2 (Viral vs Normal) | 1.75 min | 295,715 |
| | Classification Head 3 (Bacterial vs Viral) | 1.16 min | 295,715 |
| | Shared Source & 3 lightweight Heads | 4.81 min | 887,145 |
| Baseline | Separate Model (Bacterial vs Normal) | 12.75 min | 1,573,506 |
| | Separate Model (Viral vs Normal) | 9.21 min | 1,573,506 |
| | Separate Model (Bacterial vs Viral) | 12.19 min | 1,573,506 |
| | Separate end-to-end Models for 4 Tasks | 34.15 min | 4,720,518 |

**Table 7**

Performance of heterogeneous echo target tasks using the proposed method and the baseline. Bold values indicate best performance.

| Method | Tuning Strategy | Target Task | Accuracy | F-score | IoU | MCC |
|---|---|---|---|---|---|---|
| | Independent | Envelope Segmentation | **0.97** | **0.95** | **0.98** | – |
| | Alternating | Envelope Segmentation | 0.96 | 0.94 | 0.98 | – |
| | Joint | Envelope Segmentation | 0.91 | 0.93 | 0.94 | – |
| | Independent | Quality Assessment | **0.97** | **0.96** | – | **0.92** |
| UMS-Rep$_{echo}$ | Alternating | Quality Assessment | 0.95 | 0.91 | – | 0.87 |
| | Joint | Quality Assessment | 0.93 | 0.88 | – | 0.83 |
| | Independent | Flow Classification | **0.93** | **0.91** | – | **0.85** |
| | Alternating | Flow Classification | 0.91 | 0.90 | – | 0.82 |
| | Joint | Flow Classification | 0.92 | 0.90 | – | 0.84 |
| Baseline | Separate Model | Envelope Segmentation | 0.93 | 0.92 | 0.95 | – |
| | Separate Model | Quality Assessment | 0.88 | 0.85 | – | 0.78 |
| | Separate Model | Flow Classification | 0.91 | 0.88 | – | 0.81 |

**Table 8**

Heterogeneous echo target tasks: summary of time and training parameters for the proposed approach and baseline.

| Method | Task | Computational Time | Training Parameters |
|---|---|---|---|
| Proposed | Shared UMS-Rep$_{echo}$ | 28.19 min | 460,115 |
| | Envelope Segmentation Head | 5.80 min | 786,497 |
| | Quality Assessment Head | 3.00 min | 295,715 |
| | Flow Classification Head | 1.78 min | 295,715 |
| | Shared UMS-Rep$_{echo}$ & 3 lightweight Heads | 38.77 min | 1,838,042 |
| Baseline | Separate Envelope Segmentation Model | 130.02 min | 786,497 |
| | Separate Quality Assessment | 29.34 min | 1,573,506 |
| | Separate Flow Classification Model | 36.27 min | 1,573,506 |
| | 3 Separate end-to-end Models for 3 Tasks | 195.63 min | 3,933,509 |

## Table 9

Performance of homogeneous echo target tasks using the proposed method and the baseline. Bold values indicate the best performance.

| Method | Tuning Strategy | Target Task | Accuracy | F-score | MCC |
|---|---|---|---|---|---|
| | Independent | Classification (MV vs MA) | 0.92 | 0.91 | 0.83 |
| | Alternating | Classification (MV vs MA) | 0.94 | 0.94 | 0.87 |
| | Joint | Classification (MV vs MA) | **0.95** | **0.95** | **0.89** |
| | Independent | Classification (MV vs TR) | 0.95 | 0.94 | 0.88 |
| UMS-Rep$_{echo}$ | Alternating | Classification (MV vs TR) | 0.96 | 0.96 | 0.91 |
| | Joint | Classification (MV vs TR) | **0.97** | **0.96** | **0.93** |
| | Independent | Classification (MA vs TR) | 0.97 | 0.96 | 0.92 |
| | Alternating | Classification (MA vs TR) | 0.98 | 0.96 | 0.93 |
| | Joint | Classification (MA vs TR) | **0.98** | **0.98** | **0.95** |
| Baseline | Separate Model | Classification (MV vs MA) | 0.91 | 0.91 | 0.81 |
| | Separate Model | Classification (MV vs TR) | 0.94 | 0.93 | 0.86 |
| | Separate Model | Classification (MA vs TR) | 0.98 | 0.96 | 0.93 |

**Table 10**

Homogeneous echo target tasks: computational time and training parameters for the proposed and baseline approaches.

| Method | Task | Computational Time | Training Parameters |
|---|---|---|---|
| Proposed | Shared Source (UMS-Rep$_{echo}$) | 28.19 min | 460,115 |
| | Classification Head 1 (MV vs MA) | 2.42 min | 295,715 |
| | Classification Head 2 (MV vs TR) | 1.25 min | 295,715 |
| | Classification Head 3 (MA vs TR) | 1.83 min | 295,715 |
| | 1 Shared Source & 3 lightweight Heads | 33.69 min | 1,347,260 |
| Baseline | Separate Model (MV vs MA) | 16.67 min | 1,573,506 |
| | Separate Model (MV vs TR) | 10.85 min | 1,573,506 |
| | Separate Model (MA vs TR) | 9.25 min | 1,573,506 |
| | 3 Separate end-to-end Models for 3 Tasks | 36.77 min | 4,720,518 |