Stochastic Filter Groups for Multi-Task CNNs: Learning Specialist and Generalist Convolution Kernels

Felix J.S. Bragman*
University College London, UK
f.bragman@ucl.ac.uk

Ryutaro Tanno*
University College London, UK
ryutaro.tanno.15@ucl.ac.uk

Sebastien Ourselin
Kings College London
sebastien.ourselin@kcl.ac.uk

Daniel C. Alexander
University College London
d.alexander@ucl.ac.uk

M. Jorge Cardoso
Kings College London
m.jorge.cardoso@kcl.ac.uk

Abstract

The performance of multi-task learning in Convolutional Neural Networks (CNNs) hinges on the design of feature sharing between tasks within the architecture. The number of possible sharing patterns are combinatorial in the depth of the network and the number of tasks, and thus hand-crafting an architecture, purely based on the human intuitions of task relationships can be time-consuming and suboptimal. In this paper, we present a probabilistic approach to learning task-specific and shared representations in CNNs for multi-task learning. Specifically, we propose "stochastic filter groups" (SFG), a mechanism to assign convolution kernels in each layer to "specialist" or "generalist" groups, which are specific to or shared across different tasks, respectively. The SFG modules determine the connectivity between layers and the structures of task-specific and shared representations in the network. We employ variational inference to learn the posterior distribution over the possible grouping of kernels and network parameters. Experiments demonstrate that the proposed method generalises across multiple tasks and shows improved performance over baseline methods.

1. Introduction

Multi-task learning (MTL) aims to enhance learning efficiency and predictive performance by simultaneously solving multiple related tasks [3]. Recently, applications of convolutional neural networks (CNNs) in MTL have demonstrated promising results in a wide-range of computer vision applications, ranging from visual scene understanding [39, 5, 33, 23, 36, 1] to medical image computing [34, 4, 2, 41].

A key factor for successful MTL neural network models is the ability to learn shared and task-specific representations [33]. A mechanism to understand the commonalities and differences between tasks allows the model to transfer information between tasks while tailoring the predictive model to describe the distinct characteristics of the individual tasks. The quality of such representations is determined by the architectural design of where model components such as features [38] and weights [32] are shared and separated between tasks. However, the space of possible architectures is combinatorially large, and the manual exploration of this space is inefficient and subject to human biases. For example, Fig. 1 shows a typical CNN architecture for MTL comprised of a shared "trunk" feature extractor and task-specific "branch" networks [41, 13, 19, 21, 36, 2]. The desired amount of shared and task-specific representations, and their interactions within the architecture are dependent on the difficulty of the individual tasks and the relation between them, neither of which are a priori known in most cases [44]. This illustrates the challenge of handcrafting an appropriate architecture, and the need for an effective automatic method to learn it from data.

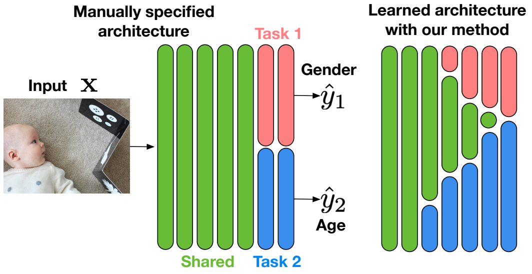

Manually specified architecture
Task 1
Input X
Gender
$\hat{y}_1$
$\hat{y}_2$
Age
Shared
Task 2
Learned architecture with our method

Figure 1: Figure on the left illustrates a typical multi-task architecture, while the figure on the right shows an example architecture that can be learned with our method. We propose *Stochastic Filter Groups*, a principled way to learn the assignment of convolution kernels to task-specific and shared groups.

In this paper, we propose *Stochastic Filter Groups* (SFGs); a probabilistic mechanism to learn the amount of task-specific and shared representations needed in each

[*] Both authors contributed equally

layer of MTL architectures (Fig. 1). Specifically, the SFGs learns to allocate kernels in each convolution layer into either “specialist” groups or a “shared” trunk, which are specific to or shared across different tasks, respectively (Fig. 2). The SFG equips the network with a mechanism to learn inter-layer connectivity and thus the structures of task-specific and shared representations. We cast the learning of SFG modules as a variational inference problem.

We evaluate the efficacy of SFGs on a variety of tasks. In particular, we focus on two multi-task learning problems: 1) age regression and gender classification from face images on UTKFace dataset [45] and 2) semantic regression (i.e. image synthesis) and semantic segmentation on a real-world medical imaging dataset, both of which require predictions over all pixels. Experiments show that our method achieves considerably higher prediction accuracy than baselines with no mechanism to learn connectivity structures, and either higher or comparable performance than a cross-stitch network [33], while being able to learn meaningful architectures automatically.

2. Related works

Our work is concerned with the goal of learning where to share neural network components across different tasks to maximise the benefit of MTL. The main challenge of such methods lies in designing a mechanism that determines how and where to share weights within the network. There are broadly two categories of methods that determine the nature of weight sharing and separation in MTL networks.

The first category is composed of methods that optimise the structures of weight sharing in order to maximise task-wise performance. These methods set out to learn a set a vectors that control which features are shared within a layer and how these are distributed across [28, 32, 33, 38]. They start with a baseline CNN architecture where they learn additional connections and pathways that define the final MTL model. For instance, Cross-Stitch networks [33] control the degree of weight sharing at each convolution layer whilst Soft-Layer Ordering [32] goes beyond the assumption of parallel ordering of feature hierarchies to allow features to mix at different layers depending on the task. Routing net [37] proposes an architecture in which each layer is a set of function blocks, and learns to decide which composition of blocks to use given an input and a task.

The second group of MTL methods focuses on weight clustering based on task-similarity [43, 17, 20, 29, 31]. For example, [29] employed an iterative algorithm to grow a tree-like deep architecture that clusters similar tasks hierarchically or [31] which determines the degree of weight sharing based on statistical dependency between tasks.

Our method falls into first category, and differentiates itself by performing “hard’ partitioning of task-specific and shared features. By contrast, prior methods are based on “soft” sharing of features [33, 38] or weights [28, 32]. These methods generally learn a set of mixing coefficients that determine the weighted sum of features throughout the network, which does not impose connectivity structures on the architecture. On the other hand, our method learns a distribution over the connectivity of layers by grouping kernels. This allows our model to learn meaningful grouping of task-specific and shared features as illustrated in Fig. 7.

3. Methods

We introduce a new approach for determining where to learn task-specific and shared representation in multi-task CNN architectures. We propose *stochastic filter groups* (SFG), a probabilistic mechanism to partition kernels in each convolution layer into “specialist” groups or a “shared” group, which are specific to or shared across different tasks, respectively. We employ variational inference to learn the distributions over the possible grouping of kernels and network parameters that determines the connectivity between layers and the shared and task-specific features. This naturally results in a learning algorithm that optimally allocate representation capacity across multi-tasks via gradient-based stochastic optimization, e.g. stochastic gradient descent.

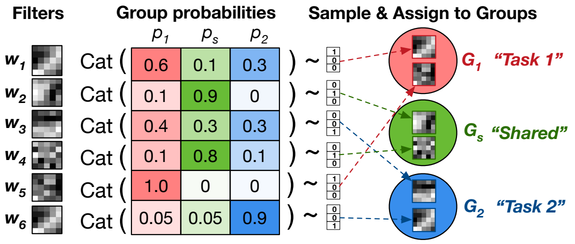


Figure 2: Illustration of filter assignment in a SFG module. Each kernel $\{\mathbf{w}_k\}$ in the given convolution layer is probabilistically assigned to one of the filter groups G_1, G_s, G_2 according to the sample drawn from the associated categorical distribution Cat(p_1, p_s, p_2).

3.1. Stochastic Filter Groups

SFGs introduce a sparse connection structure into the architecture of CNN for multi-task learning in order to separate features into task-specific and shared components. Ioannou et al. [14] introduced *filter groups* to partition kernels in each convolution layer into groups, each of which acts only on a subset of the preceding features. They demonstrated that such sparsity reduces computational cost and number of parameters without compromising accuracy. Huang et al. [12] proposed a similar concept, but differs in that the *filter groups* do not operate on mutually exclusive sets of features. Here we adapt the concept of filter groups

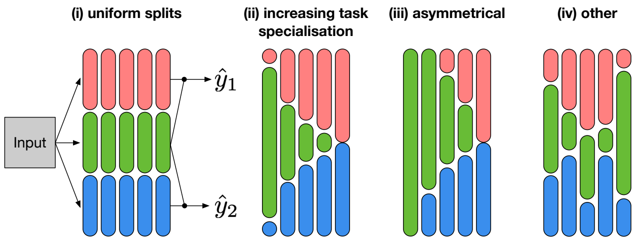


Figure 3: Illustration of possible grouping patterns learnable with the proposed method. Each set of green, pink and yellow blocks represent the ratio of filter groups G_1 (red), G_s (green) and G_2 (blue). (i) denotes the case where all kernels are uniformly split. (ii) & (iii) are the cases where the convolution kernels become more task-specific at deeper layers. (iv) shows an example with more heterogeneous splits across tasks.

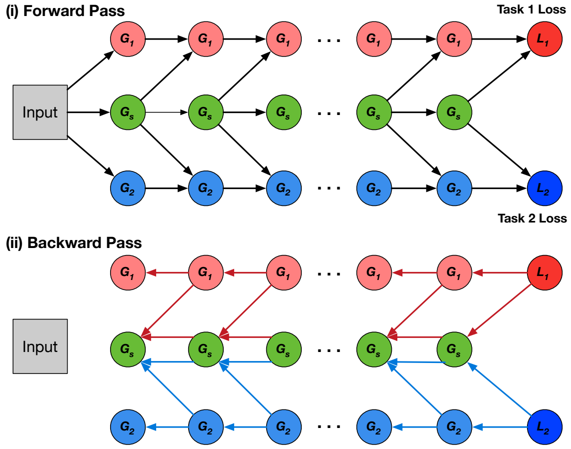


Figure 4: Illustration of feature routing. The circles G_1, G_s, G_2 denote the task-specific and shared filter groups in each layer. (i) shows the directions of routing of activations between different filter groups while (ii) shows the directions of the gradient flow from the task losses L_1 and L_2. The red and blue arrows denote the gradients that step from L_1 and L_2, respectively. The task-specific groups G_1, G_2 are only updated based on the associated losses, while the shared group G_s is updated based on both.

to the multi-task learning paradigm and propose an extension with an additional mechanism for learning an optimal kernel grouping rather than pre-specifying them.

For simplicity, we describe SFGs for the case of multi-task learning with two tasks, but can be trivially extended to a larger number of tasks. At the l^{th} convolution layer in a CNN architecture with K_l kernels $\{\mathbf{w}^{(l),k}\}_{k=1}^{K_l}$, the associated SFG performs two operations:

1. **Filter Assignment:** each kernel $\mathbf{w}_k^{(l)}$ is stochastically assigned to either: i) the "task-1 specific group" $G_1^{(l)}$, ii) "shared group" $G_s^{(l)}$ or iii) "task-2 specific group" $G_2^{(l)}$ with respective probabilities $\mathbf{p}^{(l),k} = [p_1^{(l),k}, p_s^{(l),k}, p_2^{(l),k}] \in [0,1]^3$. Convolving with the respecitve filter groups yields distinct sets of features $F_1^{(l)}, F_s^{(l)}, F_2^{(l)}$. Fig. 2 illustrates this operation and Fig. 3 shows different learnable patterns.

2. **Feature Routing:** as shown in Fig. 4 (i), the features $F_1^{(l)}, F_s^{(l)}, F_2^{(l)}$ are routed to the filter groups $G_1^{(l+1)}, G_s^{(l+1)}, G_2^{(l+1)}$ in the subsequent $(l+1)^{\text{th}}$ layer in such a way to respect the task-specificity and sharedness of filter groups in the l^{th} layer. Specifically, we perform the following routing for $l > 0$:

$$\begin{aligned} F_1^{(l+1)} &= h^{(l+1)}\big([F_1^{(l)}|F_s^{(l)}] * G_1^{(l+1)}\big) \\ F_s^{(l+1)} &= h^{(l+1)}\big(F_s^{(l)} * G_s^{(l+1)}\big) \\ F_2^{(l+1)} &= h^{(l+1)}\big([F_2^{(l)}|F_s^{(l)}] * G_2^{(l+1)}\big) \end{aligned}$$

where each $h^{(l+1)}$ defines the choice of non-linear function, $*$ denotes convolution operation and $|$ denotes a merging operation of arrays (e.g. concatenation). At $l = 0$, input image $\mathbf{x}$ is simply convolved with the first set of filter groups to yield $F_i^{(1)} = h^{(1)}\big(\mathbf{x} * G_i^{(1)}\big), i \in \{1, 2, s\}$. Fig. 4(ii) shows that such sparse connectivity ensures the parameters of $G_1^{(l)}$ and $G_2^{(l)}$ are only learned based on the respective task losses, while $G_s^{(l)}$ is optimised based on both tasks.

Fig. 5 provides a schematic of our overall architecture, in which each SFG module stochastically generates filter groups in each convolution layer and the resultant features are sparsely routed as described above. The merging modules, denoted as black circles, combine the task-specific and shared features appropriately, i.e. $[F_i^{(l)}|F_s^{(l)}], i = 1, 2$ and pass them to the filter groups in the next layer. Each white circle denotes the presence of additional transformations (e.g. convolutions or fully connected layers) in each $h^{(l+1)}$, performed on top of the standard non-linearity (e.g. ReLU).

The proposed sparse connectivity is integral to ensure task performance and structured representations. In particular, one might argue that the routing of "shared" features $F_s^{(l)}$ to the respective "task-specific" filter groups $G_1^{(l+1)}$ and $G_2^{(l+1)}$ is not necessary to ensure the separation of gradients across the task losses. However, this connection allows for learning more complex task-specific features at deeper layers in the network. For example, without this routing, having a large proportion of "shared" filter group G_s at the first layer (Fig. 3 (ii)) substantially reduces the amount of features available for learning task-specific kernels in the subsequent layers—in the extreme case in which all kernels in one layer are assigned to G_s, the task-specific filter groups in the subsequent layers are effectively unused.

Another important aspect that needs to be highlighted is the varying dimensionality of feature maps. Specifically, the number of kernels in the respective filter groups $G_1^{(l)}, G_s^{(l)}, G_2^{(l)}$ can vary at each iteration of the training,

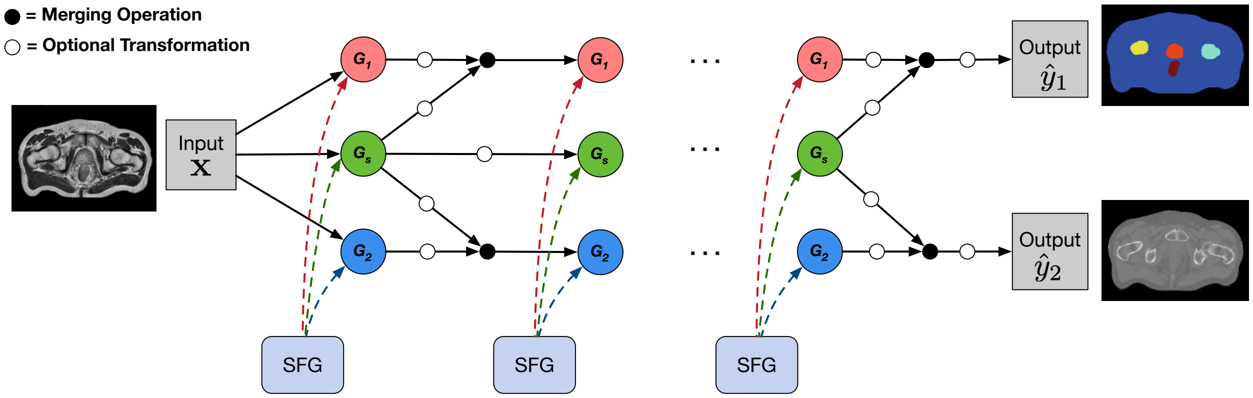


Figure 5: Schematic of the proposed multi-task architecture based on a series of SFG modules in the presence of two tasks. At each convolution layer, kernels are stochastically assigned to task-specific and shared filter groups G_1, G_s, G_2. Each input image is first convolved with the respective filter groups to yield three distinct sets of output activations, which are routed sparsely to the filter groups in the second layer layer. This process repeats in the remaining SFG modules in the architecture until the last layer where the outputs of the final SFG module are combined into task-specific predictions $\hat{y}_1$ and $\hat{y}_2$. Each small white circle denotes an optional transformation (e.g. extra convolutions) and black circle merges the incoming inputs (e.g. concatenation).

and thus, so does the depth of the resultant feature maps $F_1^{(l)}, F_s^{(l)}, F_2^{(l)}$. Instead of directly working with features maps of varying size, we implement the proposed architecture by defining $F_1^{(l)}, F_s^{(l)}, F_2^{(l)}$ as sparse tensors. At each SFG module, we first convolve the input features with all kernels, and generate the output features from each filter group by zeroing out the channels that root from the kernels in the other groups, resulting in $F_1^{(l)}, F_s^{(l)}, F_2^{(l)}$ that are sparse at non-overlapping channel indices. In the simplest form with no additional transformation (i.e. the grey circles in Fig. 5 are identity functions), we define the merging operation $[F_i^{(l)}|F_s^{(l)}], i = 1, 2$ as pixel-wise summation. In the presence of more complex transforms (e.g. residual blocks), we concatenate the output features in the channel-axis and perform a 1x1 convolution to ensure the number of channels in $[F_i^{(l)}|F_s^{(l)}]$ is the same as in $F_s^{(l)}$.

3.2. T+1 Way Concrete "Drop-Out"

Here we derive the method for simultaneously optimising the CNN parameters and grouping probabilities. We achieve this by extending the variational interpretation of binary dropout [6, 7] to the $(T+1)$-way assignment of each convolution kernel to the filter groups where T is the number of tasks. As before, we consider the case $T = 2$.

Suppose that the architecture consists of L SFG modules, each with K_l kernels where l is the index. As the posterior distribution over the convolution kernels in SFG modules $p(\mathcal{W}|\mathbf{X}, \mathbf{Y}^{(1)}, \mathbf{Y}^{(2)})$ is intractable, we approximate it with a simpler distribution $q_\phi(\mathcal{W})$ where $\mathcal{W} = \{\mathbf{W}^{(l),k}\}_{k=1,\ldots,K_l, l=1,\ldots,L}$. Assuming that the posterior distribution factorizes over layers and kernels up to group assignment, we defined the variational distribution as:

$$\begin{aligned} q_\phi(\mathcal{W}) &= \prod_{l=1}^{L}\prod_{k=1}^{K_l} q_{\phi_{lk}}(\mathbf{W}^{(l),k}) \\ &= \prod_{l=1}^{L}\prod_{k=1}^{K_l} q_{\phi_{lk}}(\mathbf{W}_1^{(l),k}, \mathbf{W}_s^{(l),k}, \mathbf{W}_2^{(l),k}) \end{aligned}$$

where $\{\mathbf{W}_1^{(l),k}, \mathbf{W}_s^{(l),k}, \mathbf{W}_2^{(l),k}\}$ denotes the k^{th} kernel in l^{th} convolution layer after being routed into task-specific $G_1^{(l)}, G_2^{(l)}$ and shared group $G_s^{(l)}$. We define each $q_{\phi_{lk}}(\mathbf{W}_1^{(l),k}, \mathbf{W}_2^{(l),k}, \mathbf{W}_s^{(l),k})$ as:

$$\mathbf{W}_i^{(l),k} = z_i^{(l),k} \cdot \mathbf{M}^{(l),k} \quad \text{for } i \in \{1, s, 2\} \tag{1}$$

$$\mathbf{z}^{(l),k} = [z_1^{(l),k}, z_2^{(l),k}, z_s^{(l),k}] \sim \text{Cat}(\mathbf{p}^{(l),k}) \tag{2}$$

where $\mathbf{z}^{(l),k}$ is the one-hot encoding of a sample from the categorical distribution over filter group assignments, and $\mathbf{M}^{(l),k}$ denotes the parameters of the pre-grouping convolution kernel. The set of variational parameters for each kernel in each layer is thus given by $\phi_{lk} = \{\mathbf{M}^{(l),k}, \mathbf{p}^{(l),k} = [p_1^{(l),k}, p_s^{(l),k}, p_2^{(l),k}]\}$.

We minimize the KL divergence between the approximate posterior $q_\phi(\mathcal{W})$ and $p(\mathcal{W}|\mathbf{X}, \mathbf{Y}^{(1)}, \mathbf{Y}^{(2)})$. Assuming that the joint likelihood over the two tasks factorizes, we have the following optimization objective:

$$\mathcal{L}_{\text{MC}}(\phi) = -\frac{N}{M}\sum_{i=1}^{M}\Big[\log p(y_i^{(1)}|\mathbf{x}_i, \mathcal{W}_i) + \log p(y_i^{(2)}|\mathbf{x}_i, \mathcal{W}_i)\Big] + \sum_{l=1}^{L}\sum_{k=1}^{K_l} \text{KL}(q_{\phi_{lk}}(\mathbf{W}^{(l),k})||p(\mathbf{W}^{(l),k})) \tag{3}$$

where M is the size of the mini-batch, N is the total number of training data points, and $\mathcal{W}_i$ denotes a set of model parameters sampled from $q_\phi(\mathcal{W})$. The last KL term regularizes the deviation of the approximate posterior from the prior $p(\mathbf{W}^{(l),k}) = \mathcal{N}(0, \mathbf{I}/l^2)$ where $l > 0$. Adapting the approximation presented in [6] to our scenario, we obtain:

$$\mathrm{KL}(q_{\phi_{lk}}(\mathbf{W}^{(l),k})||p(\mathbf{W}^{(l),k})) \propto \frac{l^2}{2}||\mathbf{M}^{(l),k}||_2^2 - \mathcal{H}(\mathbf{p}^{(l),k}) \tag{4}$$

where $\mathcal{H}(\mathbf{p}^{(l),k}) = -\sum_{i\in\{1,2,s\}} p_i^{(l),k} \log p_i^{(l),k}$ is the entropy of the grouping probabilities. While the first term performs the L2-weight norm, the second term pulls the grouping probabilities towards the uniform distribution. Plugging eq.(4) into eq.(3) yields the overall loss:

$$\mathcal{L}_{\mathrm{MC}}(\phi) = -\frac{N}{M}\sum_{i=1}^{M}\Big[\log p\Big(y_i^{(1)}|\mathbf{x}_i, \mathcal{W}_i\Big) + \log p\Big(y_i^{(2)}|\mathbf{x}_i, \mathcal{W}_i\Big)\Big] + \lambda_1 \cdot \sum_{l=1}^{L}\sum_{k=1}^{K_l}||\mathbf{M}^{(l),k}||^2 - \lambda_2 \cdot \sum_{l=1}^{L}\sum_{k=1}^{K_l}\mathcal{H}(\mathbf{p}^{(l),k}) \tag{5}$$

where $\lambda_1 > 0, \lambda_2 > 0$ are regularization coefficients.

We note that the discrete sampling operation during filter group assignment (eq. (2)) creates discontinuities, giving the first term in the objective function (eq. 5) zero gradient with respect to the grouping probabilities $\{\mathbf{p}^{(l),k}\}$. We therefore, as employed in [21] for the binary case, approximate each of the categorical variables $\mathrm{Cat}(\mathbf{p}^{(l),k})$ by the Gumbel-Softmax distribution, $\mathrm{GSM}(\mathbf{p}^{(l),k}, \tau)$ [30, 18], a continuous relaxation which allows for sampling, differentiable with respect to the parameters $\mathbf{p}^{(l),k}$ through a reparametrisation trick. The temperature term τ adjusts the bias-variance tradeoff of gradient approximation; as the value of τ approaches 0, samples from the GSM distribution become one-hot (i.e. lower bias) while the variance of the gradients increases. In practice, we start at a high τ and anneal to a small but non-zero value as in [18, 7] as detailed in supplementary materials.

4. Experiments

We tested *stochastic filter groups* (SFG) on two multi-task learning (MTL) problems: 1) age regression and gender classification from face images on UTKFace dataset [45] and 2) semantic image regression (synthesis) and segmentation on a medical imaging dataset.

UTKFace dataset: We tested our method on UTKFace [45], which consists of 23,703 cropped faced images in the wild with labels for age and gender. We created a dataset with a 70/15/15% split. We created a secondary separate dataset containing only 10% of images from the initial set, so as to simulate a data-starved scenario.

Medical imaging dataset: We used a medical imaging dataset to evaluate our method in a real-world, multi-task problem where paucity of data is common and hard to mitigate. The goal of radiotherapy treatment planning is to maximise radiation dose to the tumour whilst minimising dose to the organs. To plan dose delivery, a Computed Tomography (CT) scan is needed as CT voxel intensity scales with tissue density, thus allowing dose propagation simulations. An MRI scan is needed to segment the surrounding organs. Instead of acquiring both an MRI and a CT, algorithms can be used to synthesise a CT scan (task 1) and segment organs (task 2) given a single input MRI scan. For this experiment, we acquired 15, 3D prostate cancer scans with respective CT and MRI scans with semantic 3D labels for organs (prostate, bladder, rectum and left/right femur heads) obtained from a trained radiologist. We created a training set of 10 patients, with the remaining 5 used for testing. We trained our networks on 2D patches of size 128x128 randomly sampled from axial slices, and reconstructed the 3D volumes of size 288x288x62 at test time by stitching together the subimage-wise predictions.

4.1. Baselines

We compared our model against four baselines in addition to Cross-Stitch networks [33] trained end-to-end rather than sequentially for fair comparison. The four baselines considered are: 1) single-task networks, 2) hard-parameter sharing multi-task network (MT-hard sharing), 3) SFG-networks with constant $1/3$ allocated grouping (MT-constant mask) as *per* Fig. 3(i), and 4) SFG-networks with constant grouping probabilities (MT-constant $\mathbf{p}$). We train all the baselines in an end-to-end fashion for all the experiments.

We note that all four baselines can be considered special cases of an SFG-network. Two *single-task networks* can be learned when the shared grouping probability of kernels is set to zero. Considering Fig. 5, this would remove the diagonal connections and the shared network. This may be important when faced with two unrelated tasks which share no contextual information. A *hard-parameter sharing network* exists when all shared grouping probabilities are maximised to one leading to a scenario where all features are shared within the network up until the task-specific layers. The *MT-constant mask network* is illustrated in Fig. 3(i), where $1/3$ of kernels are allocated to the task 1, task 2 and shared groups, yielding uniform splits across layers. This occurs when an equal number of kernels in each layer obtain probabilities of $\mathbf{p}^{(l),k} = [1, 0, 0], [0, 1, 0]$ and $[0, 0, 1]$. Lastly, the *MT-constant **p*** model represents the situation where the grouping is non-informative and each kernel has equal probability of being specific or shared with probability $\mathbf{p}^{(l),k} = [1/3, 1/3, 1/3]$. Training details for these models, including the hyper-parameter settings, are provided in the supplementary document.

UTKFace network: We used VGG-11 CNN architecture [40] for age and gender prediction. The network consists of a series of 3x3 convolutional layers interleaved with max pooling layers. In contrast to the original architecture, we replaced the final max pooling and fully connected layers with global average pooling (GAP) followed by a fully connected layers for prediction. Our model's version of VGG (SFG-VGG) replaces each convolutional layer in VGG-11 with a SFG layer with max pooling applied to each feature map $F_1^{(l)}$, $F_2^{(l)}$, $F_s^{(l)}$. We applied GAP to each final feature map before the final merging operation and two fully connected layers for each task.

Medical imaging network: We used the HighResNet architecture [26] for CT synthesis and organ segmentation. This network has been developed for semantic segmentation in medical imaging and has been used in a variety of medical applications such as CT synthesis [2] and brain segmentation [26]. It consists of a series of residual blocks, which group two 3x3 convolutional layers with dilated convolutions. The baseline network is composed of a 3x3 convolutional layer followed by three sets of twice repeated residual blocks with dilated convolutions using factors $d = [1, 2, 4]$. There is a 3x3 convolutional layer between each set of repeated residual blocks. The network ends with two final 3x3 layers and either one or two 1x1 convolutional layers for single and multi-task predictions. In our model, we replace each convolutional layer with an SFG module. After the first SFG layer, three distinct repeated residual blocks are applied to $F_1^{(l=0)}$, $F_2^{(l=0)}$, $F_s^{(l=0)}$. These are then merged according the feature routing methodology followed by a new SFG-layer and subsequent residual layers. Our model concludes with 2 successive SFG-layers followed by 1x1 convolutional layers applied to the merged features $F_1^{(l=L)}$ and $F_2^{(l=L)}$.

5. Results

5.1. Age regression and gender prediction

Results on age prediction and gender classification on both datasets are presented in Tab. 1a and 1b. Our model (MT-SFG) achieved the best performance in comparison to the baselines in both data regimes. In both sets of experiments, our model outperformed the hard-parameter sharing (*MT-hard sharing*) and constant allocation (*MT-constant mask*). This demonstrates the advantage of learning to allocate kernels. In the *MT-constant mask* model, kernels are equally allocated across groups. In contrast, our model is able to allocate kernels in varying proportions across different layers in the network (Fig. 6 - SFG-VGG11) to maximise inductive transfer. Moreover, our methods performed better than a model with constant, non-informative grouping probabilities (*MT-constant* $\mathbf{p}= [1/3, 1/3, 1/3]$), displaying the importance of learning structured representations and connectivity across layers to yield good predictions.

(a) Full training data

Method	Age (MAE)	Gender (Accuracy)
One-task (VGG11) [40]	7.32	90.70
MT-hard sharing	7.92	90.60
MT-constant mask	7.67	89.41
MT-constant $\mathbf{p}$=[$1/3$,$1/3$,$1/3$]	6.34	92.10
VGG11 Cross Stitch [33]	6.78	90.30
MT-SFG (ours)	**6.00**	**92.46**

(b) Small training data

Method	Age (MAE)	Gender (Accuracy)
One-task (VGG11) [40]	8.79	85.54
MT-hard sharing	9.19	85.83
MT-constant mask	9.02	85.98
MT-constant $\mathbf{p}$=[$1/3$,$1/3$,$1/3$]	9.15	86.01
VGG11 Cross Stitch [33]	8.85	83.72
MT-SFG (ours)	**8.54**	**87.01**

Table 1: Age regression and gender classification results on UTKFace [45] with (a) the full and (b) limited training set. The best and the second best results are shown in red and blue. The mean absolute error (MAE) is reported for the age prediction and classification accuracy for gender prediction. For our model, we performed 50 stochastic forward passes at test time by sampling the kernels from the approximate posterior $q_\phi(\mathcal{W})$. We calculated the average age per subject and obtained gender prediction using the mode of the test-time predictions.

5.2. Image regression and semantic segmentation

Results on CT image synthesis and organ segmentation from input MRI scans is detailed in Tab. 2. Our method obtains equivalent (non-statistically significant different) results to the Cross-Stitch network [33] on both tasks. We have, however, observed best synthesis performance in the bone regions (femur heads and pelvic bone region) in our model when compared against all the baselines, including Cross-Stitch. The bone voxel intensities are the most difficult to synthesise from an input MR scan as task uncertainty in the MR to CT mapping at the bone is often highest [2]. Our model was able to disentangle features specific to the bone intensity mapping (Fig. 7) without supervision of the pelvic location, which allowed it to learn a more accurate mapping of an intrinsically difficult task.

5.3. Learned architectures

Analysis of the grouping probabilities of a network embedded with SFG modules permits visualisation of the network connectivity and thus the learned MTL architecture.

(a) CT Synthesis (PSNR)

Method	Overall	Bones	Organs	Prostate	Bladder	Rectum
One-task (HighResNet) [26]	25.76 (0.80)	30.35 (0.58)	38.04 (0.94)	51.38 (0.79)	33.34 (0.83)	34.19 (0.31)
MT-hard sharing	26.31 (0.76)	31.25 (0.61)	39.19 (0.98)	52.93 (0.95)	34.12 (0.82)	34.15 (0.30)
MT-constant mask	24.43(0.57)	29.10(0.46)	37.24(0.86)	50.48(0.73)	32.29(1.01)	33.44(2.88)
MT-constant $\mathbf{p}$=[${}^1/{}_3$,${}^1/{}_3$,${}^1/{}_3$]	26.64(0.54)	31.05 (0.55)	39.11 (1.00)	53.20 (0.86)	34.34 (1.35)	35.61 (0.35)
Cross Stitch [33]	27.86 (1.05)	32.27 (0.55)	40.45 (1.27)	54.51 (1.01)	36.81 (0.92)	36.35 (0.38)
MT-SFG (ours)	27.74 (0.96)	32.29 (0.59)	39.93 (1.09)	53.01 (1.06)	35.65 (0.44)	35.65 (0.37)

(b) Segmentation (DICE)

Method	Overall	Left Femur Head	Right Femur Head	Prostate	Bladder	Rectum
One-task (HighResNet) [26]	0.848(0.024)	0.931 (0.012)	0.917 (0.013)	0.913 (0.013)	0.739 (0.060)	0.741 (0.011)
MT-hard sharing	0.829(0.023)	0.933 (0.009)	0.889 (0.044)	0.904 (0.016)	0.685 (0.036)	0.732 (0.014)
MT-constant mask	0.774(0.065)	0.908 (0.012)	0.911 (0.015)	0.806 (0.0541)	0.583 (0.178)	0.662 (0.019)
MT-constant $\mathbf{p}$=[${}^1/{}_3$,${}^1/{}_3$,${}^1/{}_3$]	0.752(0.056)	0.917 (0.004)	0.917 (0.01)	0.729 (0.086)	0.560 (0.180)	0.639 (0.012)
Cross Stitch [33]	0.854 (0.036)	0.923 (0.008)	0.915 (0.013)	0.933 (0.009)	0.761 (0.053)	0.737 (0.015)
MT-SFG (ours)	0.852(0.047)	0.935 (0.007)	0.912 (0.013)	0.923 (0.016)	0.750 (0.062)	0.758 (0.011)

Table 2: Performance on the medical imaging dataset with best results in red, and the second best results in blue. The PSNR is reported for the CT-synthesis (synCT) across the whole volume (overall), at the bone regions, across all organ labels and individually at the prostate, bladder and rectum. For the segmentation, the average DICE score per patient across all semantic labels is computed. The standard deviations are computed over the test subject cohort. For our model, we perform 50 stochastic forward passes at test-time by sampling the kernels from the approximated posterior distribution $q_\phi(\mathcal{W})$. We compute the average of all passes to obtain the synCT and calculate the mode of the segmentation labels for the final segmentation.

To analyse the group allocation of kernels at each layer, we computed the sum of class-wise probabilities per layer. Learned groupings for both SFG-VGG11 network trained on UTKFace and the SFG-HighResNet network trained on prostate scans are presented in Fig. 6. These figures illustrate increasing task specialisation in the kernels with network depth. At the first layer, all kernels are classified as shared ($\mathbf{p}=[0,1,0]$) as low-order features such as edge or contrast descriptors are generally learned earlier layers. In deeper layers, higher-order representations are learned, which describe various salient features specific to the tasks. This coincides with our network allocating kernels as task specific, as illustrated in Fig. 7, where activations are stratified by allocated class per layer. Density plots of the learned kernel probabilities and trajectory maps displaying training dynamics, along with more examples of feature visualisations, are provided in supplementary materials.

Notably, the learned connectivity of both models shows striking similarities to hard-parameter sharing architectures commonly used in MTL. Generally, there is a set of shared layers, which aim to learn a feature set common to both tasks. Task-specific branches then learn a mapping from this feature space for task-specific predictions. Our models are able to automatically learn this structure whilst allowing asymmetric allocation of task-specific kernels with no priors on the network structure.

5.4. Effect of p initialisation

Fig. 3 shows the layer-wise proportion of the learned kernel groups on the UTKFace dataset for four different initilization schemes of grouping probabilities $\mathbf{p}$: (i) "dominantly shared", with $\mathbf{p} = [0.2, 0.6, 0.2]$, (ii) "dominantly task-specific", with $\mathbf{p} = [0.45, 0.1, 0.45]$, (iii) "random", where $\mathbf{p}$ is drawn from Dirichlet$(1, 1, 1)$, (iv) "start with MT-constant mask", where an equal number of kernels in each layer are set to probabilities of $\mathbf{p} = [1, 0, 0], [0, 1, 0]$ and $[0, 0, 1]$. In all cases, the same set of hyper-parameters, including the annealing rate of the temperature term in GSM approximation and the coefficient of the entropy regularizer $\mathcal{H}(\mathbf{p})$, were used during training. We observe that the kernel grouping of respective layers in (i), (ii) and (iii) all converge to a very similar configuration observed in Sec. 5.3, highlighting the robustness of our method to different initialisations of $\mathbf{p}$. In case (iv), the learning of p were much slower than the remaining cases, due to weaker gradients,

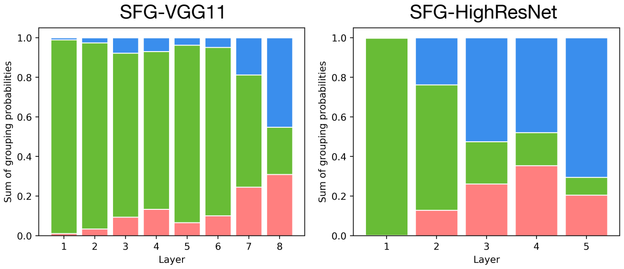


Figure 6: Learned kernel grouping in a) SFG-VGG11 network on UTKFace and b) SFG-HighResNet on medical scans. The proportions of task-1, shared and task-2 filter groups are shown in blue, green and pink. Within SFG-VGG11, task-1 age regression and task-2 is gender classification. For SFG-HighResNet, task-1 is CT synthesis and task-2 is organ segmentation.

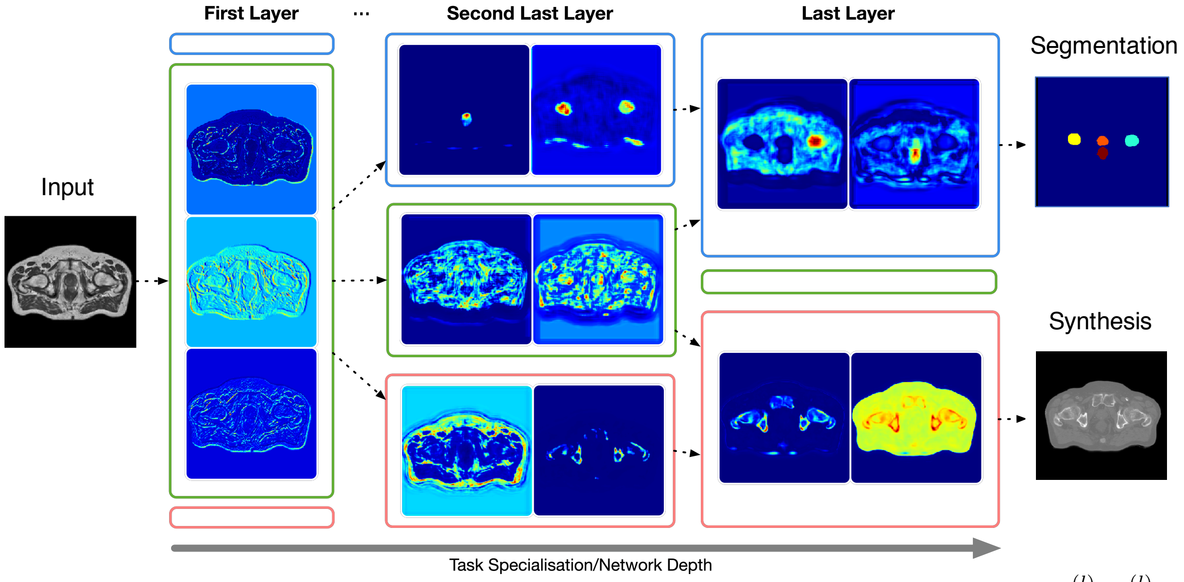


Figure 7: Activation maps from example kernels in the learned task-specific and shared filter groups, $G_1^{(l)}, G_2^{(l)}, G_s^{(l)}$ (enclosed in blue, green and pink funnels) in the first, the second last and the last convolution layers in the SFG-HighResNet model trained on the medical imaging dataset. The results from convolution kernels with low entropy (i.e. high "confidence") of group assignment probabilities $\mathbf{p}^{(l)}$ are shown for the respective layers.

and we speculate that a higher entropy regularizer is necessary to facilitate its convergence.

6. Discussion

In this paper, we have proposed *stochastic filter groups* (SFGs) to disentangle *task-specific* and *generalist* features. SFGs probabilistically defines the grouping of kernels and thus the connectivity of features in a CNNs. We use variational inference to approximate the distribution over connectivity given training data and sample over possible architectures during training. Our method can be considered as a probabilistic form of multi-task architecture learning [27], as the learned posterior embodies the optimal MTL architecture given the data.

Our model learns structure in the representations. The learned shared (generalist) features may be exploited either in a transfer learning or continual learning scenario. As seen in [24], an effective prior learned from multiple tasks can be a powerful tool for learning new, unrelated tasks. Our model consequently offers the possibility to exploit the learned task-specific and generalist features when faced with situations where a third task is needed, which may suffer from unbalanced or limited training data. This is particularly relevant in the medical field, where training data is expensive to acquire as well as laborious. We will investigate this in further work.

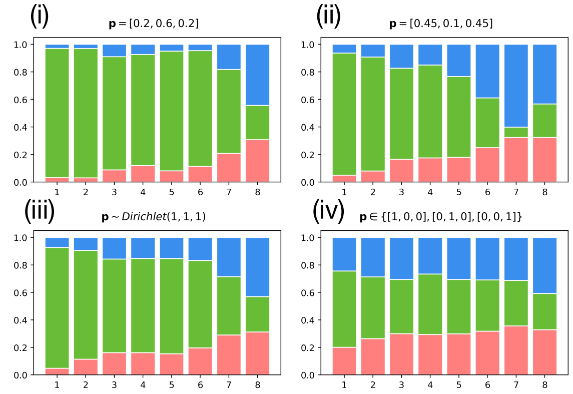


Figure 8: Effect of the initial values of grouping probabilities $\mathbf{p}$ on the learned kernel allocation after convergence.

Lastly, a network composed of SFG modules can be seen as a superset of numerous MTL architectures. Depending on the data and the analysed problem, SFGs can recover many different architectures such as single task networks, traditional hard-parameter sharing, equivalent allocation across tasks, and asymmetrical grouping (Fig. 3). Note, however, that proposed SFG module only learns connectivity between neighbouring layers. Non-parallel ordering of layers, a crucial concept of MTL models [32, 38], was not investigated. Future work will look to investigate the applicability of SFG modules for learning connections across grouped kernels between non-neighbouring layers.

Acknowledgments

FB and MJC were supported by CRUK Accelerator Grant A21993. RT was supported by Microsoft Scholarship. DA was supported by EU Horizon 2020 Research and Innovation Programme Grant 666992, EPSRC Grant M020533, R014019, and R006032 and the NIHR UCLH BRC. We thank NVIDIA Corporation for hardware donation.

References

[1] Hakan Bilen and Andrea Vedaldi. Integrated perception with recurrent multi-task neural networks. In *Advances in neural information processing systems*, pages 235–243, 2016.

[2] Felix Bragman, Ryu Tanno, Zach Eaton-Rosen, Wenqi Li, David Hawkes, Sebastien Ourselin, Daniel Alexander, Jamie McClelland, and M. Jorge Cardoso. Uncertainty in multi-task learning: joint representations for probabilistic mr-only radiotherapy planning. In *Medical Image Computing and Computer-Assisted Interventions (MICCAI)*, 2018.

[3] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[4] Sihong Chen, Dong Ni, Jing Qin, Baiying Lei, Tianfu Wang, and Jie-Zhi Cheng. Bridging computational features toward multiple semantic features with multi-task regression: A study of ct pulmonary nodules. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 53–60. Springer, 2016.

[5] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015.

[6] Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

[7] Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. In *Advances in Neural Information Processing Systems*, pages 3581–3590, 2017.

[8] Eli Gibson, Wenqi Li, Carole Sudre, Lucas Fidon, Dzhoshkun I. Shakir, Guotai Wang, Zach Eaton-Rosen, Robert Gray, Tom Doel, Yipeng Hu, Tom Whyntie, Parashkev Nachev, Marc Modat, Dean C. Barratt, Sébastien Ourselin, M. Jorge Cardoso, and Tom Vercauteren. NiftyNet: a deep-learning platform for medical imaging. *Computer Methods and Programs in Biomedicine*, 158:113–122, 2018.

[9] Marc Harper. python-ternary: Ternary plots in python. In *10.5281/zenodo.34938*, 2015.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks, 2016.

[12] Gao Huang, Shichen Liu, Laurens Van der Maaten, and Kilian Q Weinberger. Condensenet: An efficient densenet using learned group convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2752–2761, 2018.

[13] Junshi Huang, Rogerio S Feris, Qiang Chen, and Shuicheng Yan. Cross-domain image retrieval with a dual attribute-aware ranking network. In *Proceedings of the IEEE international conference on computer vision*, pages 1062–1070, 2015.

[14] Yani Ioannou, Duncan Robertson, Roberto Cipolla, Antonio Criminisi, et al. Deep roots: Improving cnn efficiency with hierarchical filter groups. 2017.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[16] Fabian Isensee, Jens Petersen, Andre Klein, David Zimmerer, Paul F. Jaeger, Simon Kohl, Jakob Wasserthal, Gregor Koehler, Tobias Norajitra, Sebastian Wirkert, and Klaus H. Maier-Hein. nnu-net: Self-adapting framework for u-net-based medical image segmentation, 2018.

[17] Laurent Jacob, Jean philippe Vert, and Francis R. Bach. Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems 21*, 2009.

[18] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[19] Brendan Jou and Shih-Fu Chang. Deep cross residual learning for multitask visual recognition. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 998–1007. ACM, 2016.

[20] Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 521–528, USA, 2011. Omnipress.

[21] Alex Kendall, Yarin Gal, and Roberto Cipolla. 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 (CVPR)*, 2018.

[22] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference for Learning Representations*, 2015.

[23] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6129–6138, 2017.

[24] Alexandre Lacoste, Boris Oreshkin, Wonchang Chung, Thomas Boquet, Negar Rostamzadeh, and David Krueger. Uncertainty in multitask transfer learning. In *arXiv:1806.07528*, 2018.

[25] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pages 6402–6413, 2017.

[26] Wenqi Li, Guotai Wang, Lucas Fidon, Sebastien Ourselin, M. Jorge Cardoso, and Tom Vercauteren. On the compactness, efficiency, and representation of 3d convolutional networks: Brain parcellation as a pretext task. 2017.

[27] Jason Liang, Elliot Meyerson, and Risto Miikkulainen. Evolutionary architecture search for deep multitask networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 466–473. ACM, 2018.

[28] Mingsheng Long and Jianmin Wang. Learning multiple tasks with deep relationship networks. In *Advances in Neural Information Processing Systems*, 2017.

[29] Yongxi Lu, Abhishek Kumar, Shuangfei Zhai, Yu Cheng, Tara Javidi, and Rogério Schmidt Feris. Fully-adaptive feature sharing in multi-task networks with applications in per-

son attribute classification. In *CVPR*, volume 1, page 6, 2017.

[30] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[31] Youssef A Mejjati, Darren Cosker, and Kwang In Kim. Multi-task learning by maximizing statistical dependence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3465–3473, 2018.

[32] Elliot Meyerson and Risto Miikkulainen. Beyond shared hierarchies: Deep multitask learning through soft layer ordering. In *International Conference on Learning Representations*, 2018.

[33] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch Networks for Multi-task Learning. In *CVPR*, 2016.

[34] Pim Moeskops, Jelmer M Wolterink, Bas HM van der Velden, Kenneth GA Gilhuijs, Tim Leiner, Max A Viergever, and Ivana Išgum. Deep learning for multi-task medical image segmentation in multiple modalities. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 478–486. Springer, 2016.

[35] L.G. Nyul, J.K. Udupa, and Xuan Zhang. New variants of a method of MRI scale standardization. *IEEE Transactions on Medical Imaging*, 19(2):143–150, 2000.

[36] Rajeev Ranjan, Vishal M Patel, and Rama Chellappa. Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1):121–135, 2019.

[37] Clemens Rosenbaum, Tim Klinger, and Matthew Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. *arXiv preprint arXiv:1711.01239*, 2017.

[38] Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. Latent multi-task architecture learning. 2019.

[39] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.

[40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014.

[41] Ryutaro Tanno, Antonios Makropoulos, Salim Arslan, Ozan Oktay, Sven Mischkewitz, Fouad Al-Noor, Jonas Oppenheimer, Ramin Mandegaran, Bernhard Kainz, and Mattias P Heinrich. Autodvt: Joint real-time classification for vein compressibility analysis in deep vein thrombosis ultrasound diagnostics. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 905–912. Springer, 2018.

[42] Nicholas J Tustison, Brian B Avants, Philip A Cook, Yuanjie Zheng, Alexander Egan, Paul A Yushkevich, and James C Gee. N4itk: Improved n3 bias correction. *IEEE Transactions on Medical Imaging*, 29(6):1310–1320, 2010.

[43] Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(Jan):35–63, 2007.

[44] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018.

[45] Song Yang Zhang, Zhifei and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

A. Training and implementation details

A.1. Optimisation, regularisation and initialisation

All networks were trained with ADAM optimiser [22] with an initial learning rate of 10^{-3} and $\beta = [0.9, 0.999]$. We used values of $\lambda_1 = 10^{-6}$ and $\lambda_2 = 10^{-5}$ for the weight and entropy regularisation factors in Equation (5) in Section 3.2. All *stochastic filter group* (SFG) modules were initialised with grouping probabilities **p**=[0.2, 0.6, 0.2] for every convolution kernel. Positivity of the grouping probabilities **p** is enforced by passing the output through a *softplus* function $f(x) = \ln(1 + e^x)$ as in [25]. The scheduler $\tau = \max(0.10, \exp(-rt))$ recommended in [18] was used to anneal the Gumbel-Softmax temperature τ where r is the annealing rate and t is the current training iteration. We used $r = 10^{-5}$ for our models.

Hyper-parameters for the annealing rate and the entropy regularisation weight were obtained by analysis of the network performance on a secondary randomly split on the UTK dataset (70/15/15). They were then applied to all trained models (large and small dataset for UTKFace and medical imaging dataset).

A.2. UTKFace

For training the VGG networks (Section 4.1 - UTKFace network), we used the root-mean-squared-error (RMSE) for age regression and the cross entropy loss for gender classification. The labels for age were divided by 100 prior to training. The input RGB images (200x200x3) were all normalised channel wise to have unit variance and zero mean prior to training and testing. A batch-size of 10 was used. No augmentation was applied. We monitored performance during training using the validation set ($n = 3554$) and trained up to 330 epochs. We performed 150 validation iterations every 1000 iterations, leading to 1500 predictions per validation iteration. Performance on the validation set was analysed and the iteration where Mean Absolute Error (MAE) was minimised and classification Accuracy was maximised was chosen for the test set.

A.3. Medical imaging dataset

We used T2-weighted Magnetic Resonance Imaging (MRI) scans (3T, 2D spin echo, TE/TR: 80/2500ms, voxel size 1.46x1.46x5mm^3) and Computed Tomography (CT) scans (140 kVp, voxel size 0.98x0.98x1.5 mm^3). The MR and CT scans were resampled to isotropic resolution (1.46mm^3). We performed intensity non-uniformity correction on the MR scans [42].

In the HighResNet networks (Section 4.1 - Medical imaging network), we used the RMSE loss for the regression task and the Dice + Cross-Entropy loss [16] for the segmentation task. The CT scans were normalised using the transformation $^{\text{CT}}/_{1024} + 1$. The original range of the CT voxel intensity was $[-1024, 2500]$ with the background set to -1024. The input MRI scans were first normalised using histogram normalisation based on the 1^{st} and 99^{th} percentile [35]. The MRI scans were then normalised to zero mean and unit variance. At test time, input MRI scans were normalised using the histogram normalisation transformation obtained from the training set then normalised to have zero mean and unit variance.

All scans were of size 288x288x62. We sub-sampled random patches from random axial slices of size 128x128. We sampled from all axial slices in the volume ($n = 62$). We trained up to $200,000$ iterations using a batch-size of 10. We applied augmentation to the randomly sampled patches using random scaling factors in the range $[-10\%, 10\%]$ and random rotation angles in the range [$-10°$, $10°$]. The trained patches were zero-padded to increase their size to 136x136. However, the loss during training was only calculated in non-padded regions.

The inference iteration for the test set was determined when the performance metrics on the training set (Mean Absolute Error and Accuracy) first started to converge for at least $10,000$ iterations. In our model where the grouping probabilities were learned, the iteration when convergence in the update of the grouping probabilities was first observed was selected since performance generally increased as the grouping probabilities were updated.

A.4. Implementation details

We used Tensorflow and implemented our models within the NiftyNet framework [8]. Models were trained on NVIDIA Titan Xp, P6000 and V100. All networks were trained in the Stochastic Filter Group paradigm. Single-task networks were trained by hard-coding the allocation of kernels to task 1 and task 2 i.e. 50% of kernels per layer were allocated to task 1 and 50% were allocated to task 2 with constant probabilities **p**=[1,0,0] and **p**=[0,0,1] respectively. The multi-task hard parameter sharing (MT hard-sharing) network was trained by hard-coding the allocation of kernels to the shared group i.e. 100% of kernel per layer were allocated to the shared group with constant probability **p**=[0, 1, 0]. The cross-stitch (CS) [33] networks were implemented in a similar fashion to the single-task networks, with CS modules applied to the output of the task-specific convolutional layers. The other baselines (MT-constant mask and MT-constant **p**=[⅓, ⅓, ⅓]) were trained similarly.

We used Batch-Normalisation [15] to help stabilise training. We observed that the deviation between population statistics and batch statistics can be high, and thus we did not use population statistic at test time. Rather, we normalised using batch-statistics instead, and this consistently lead to better predictive performance. We also used the Gumbel-Softmax approximation [18] at test-time using the temperature value τ that corresponded to the iteration in τ annealing schedule.

B. CNN architectures and details

We include schematics and details of the single-task VGG11 [40] and HighResNet [26] networks in Fig. 9. In this work, we constructed multi-task architectures by augmenting these networks with the proposed SFG modules. We used the PReLU activation function [10] in all networks. For the residual blocks used in the HighResNet networks in Fig. 9 (ii), we applied PReLU and batch-norm as pre-activation [11] to the convolutional layers. The SFG module was used to cluster the kernels in every coloured layer in Fig. 9, and distinct sets of additional transformations (pooling operations for VGG and high-res blocks for HighResNet) were applied to the outputs of the respective filter groups G_1, G_2, G_s. For a fair comparison, the CS units [33] were added to the same set of layers.

For clarification, the SFG layer number n (e.g. SFG layer 2) corresponds to the n^{th} layer with an SFG module. In the case of SFG-VGG11, each convolutional layer uses SFGs. The SFG layer number thus corresponds with layer number in the network. In the case of SFG-HighResNet, not every convolutional layer uses SFGs such as those within residual blocks. Consequently, SFG layer 1 corresponds to layer 1, SFG layer 2 is layer 6, SFG layer 3 is layer 11, SFG layer 4 is layer 16 and SFG layer 5 is layer 17.

C. Learned grouping probability plots

In this section, we illustrate density plots of the learned grouping probabilities $\mathbf{p}$ for each trained network (Fig. 10 and Fig. 11). We also plot the training trajectories of grouping probabilities $\mathbf{p}$ of all kernels in each layer. These are colour coded by iteration number—blue for low and yellow for high iteration number. This shows that some grouping probabilities are quickly learned in comparison to others.

Fig. 10 and Fig. 11 show that most kernels are in the shared group at earlier layers of the network where mostly low-order generic features are learned (as illustrated in Fig. 12, SFG layer 1). They converge quickly to the shared vertex of the 2-simplex as evidenced by the colour of the trajectory plots. As the network depth increases, task-specialisation in the kernels increases (see Fig. 12, SFG layer ≥ 4). This is illustrated by high density clusters at task-specific vertices and by the trajectory plots.

D. Extra visualisation of activations

Here we visualise the activation maps of additional specialist and generalist kernels on the medical imaging dataset. To classify each kernel according to the group (task 1, task 2 or shared), we selected the group with the respective maximum assignment probability. The corresponding activation maps for various input images in the medical imaging dataset can be viewed in Fig. 12 and Fig. **??**.

We first analysed the activation maps generated by kernels with low entropy of $\mathbf{p}$ (i.e. highly confident group assignment). At the first layer, all kernels are classified as shared, and the examples in Fig. 12 support that these kernels tend to account for low-order features such as edge and contrast of the images. On the other hand, at deeper layers, higher-order representations are learned, which describe various salient features specific to the tasks such as organs for segmentation, and bones for CT-synthesis. Note that the bones are generally the most difficult region to synthesise CT intensities from an input MR scan [2].

Secondly, we looked at activation maps from kernels with high entropy of $\mathbf{p}$ (i.e. highly uncertain group assignment) in Fig. **??**. In contrast to Fig. 12, the learned features do not appear to capture any meaningful structures for both synthesis and segmentation tasks. Of particular note is the dead kernel in the top row of the figure; displaying that a high uncertainty in group allocation correlates with non-informative features.

E. Learned filter groups on duplicate tasks

We analysed the dynamics of a network with SFG modules when trained with two duplicates of the same CT regression task (instead of two distinct tasks). Fig. 13 visualises the learned grouping and trajectories of the grouping probabilities during training. In the first 3 SFG layers (layers 1, 6 and 11 of the network), all the kernels are grouped as shared. In the penultimate SFG layer (layer 16), either kernels are grouped as shared or with probability $\mathbf{p}$=[½, 0, ½], signifying that the kernels can belong to either task. The final SFG layer (layer 17) shows that most kernels have probabilities $\mathbf{p}$=[⅓, ⅓, ⅓]. Kernels thus have equal probability of being task-specific or shared. This is expected as we are training on duplicate tasks and therefore the kernels are equally likely to be useful across all groups.

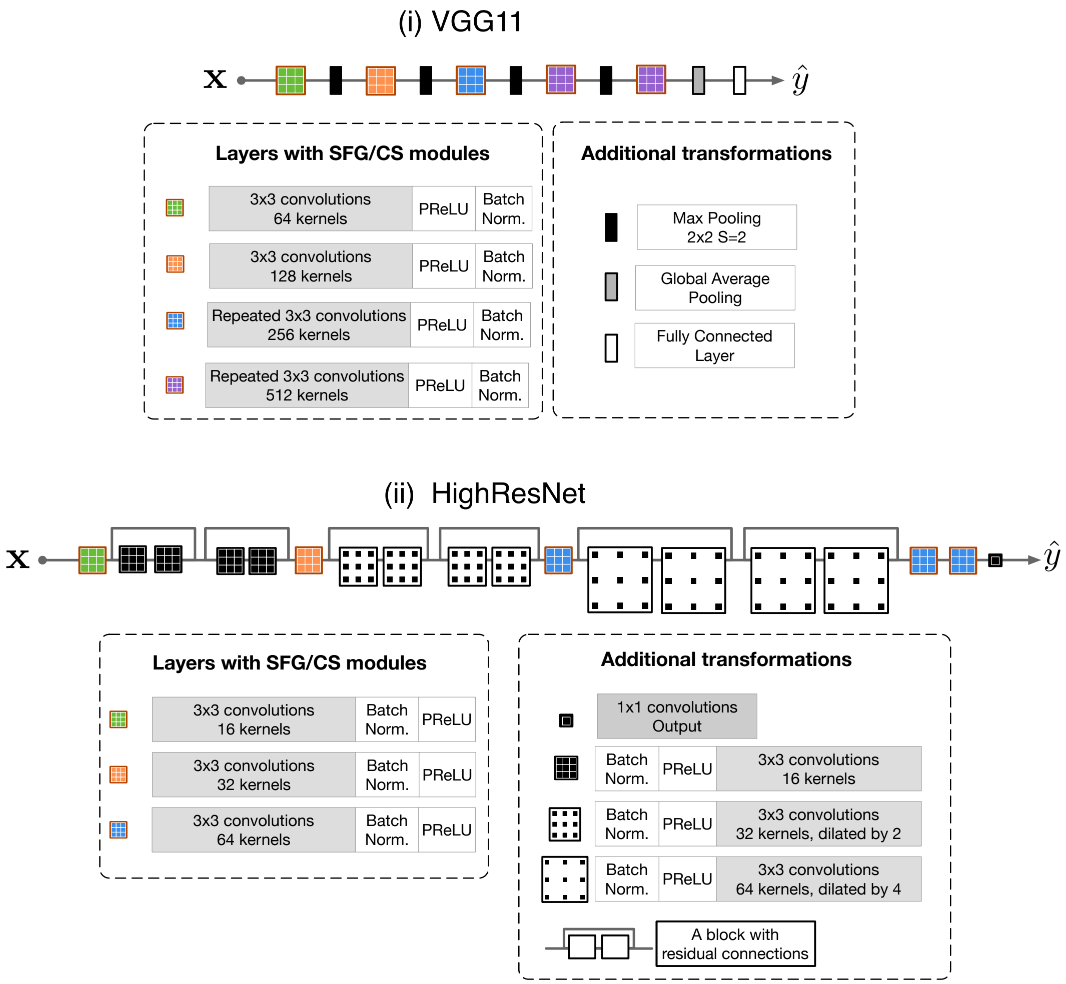

(i) VGG11
x
ŷ
Layers with SFG/CS modules
3x3 convolutions 64 kernels
PReLU
Batch Norm.
3x3 convolutions 128 kernels
PReLU
Batch Norm.
Repeated 3x3 convolutions 256 kernels
PReLU
Batch Norm.
Repeated 3x3 convolutions 512 kernels
PReLU
Batch Norm.
Additional transformations
Max Pooling 2x2 S=2
Global Average Pooling
Fully Connected Layer
(ii) HighResNet
x
ŷ
Layers with SFG/CS modules
3x3 convolutions 16 kernels
Batch Norm.
PReLU
3x3 convolutions 32 kernels
Batch Norm.
PReLU
3x3 convolutions 64 kernels
Batch Norm.
PReLU
Additional transformations
1x1 convolutions Output
Batch Norm.
PReLU
3x3 convolutions 16 kernels
Batch Norm.
PReLU
3x3 convolutions 32 kernels, dilated by 2
Batch Norm.
PReLU
3x3 convolutions 64 kernels, dilated by 4
A block with residual connections

Figure 9: Illustration of the single-task architectures, (i) VGG11 and (ii) HighResNet used for UTKFace and medical imaging dataset, respectively. In each architecture, the coloured components indicate the layers to which SFG or cross-stitch (CS) modules are applied when extended to the multi-task learning scenario, whilst the components in black denote the additional transformations applied to the outputs of respective filter groups or CS operations (see the description of black circles in the schematic provided in Fig. 5 of the main text)

.

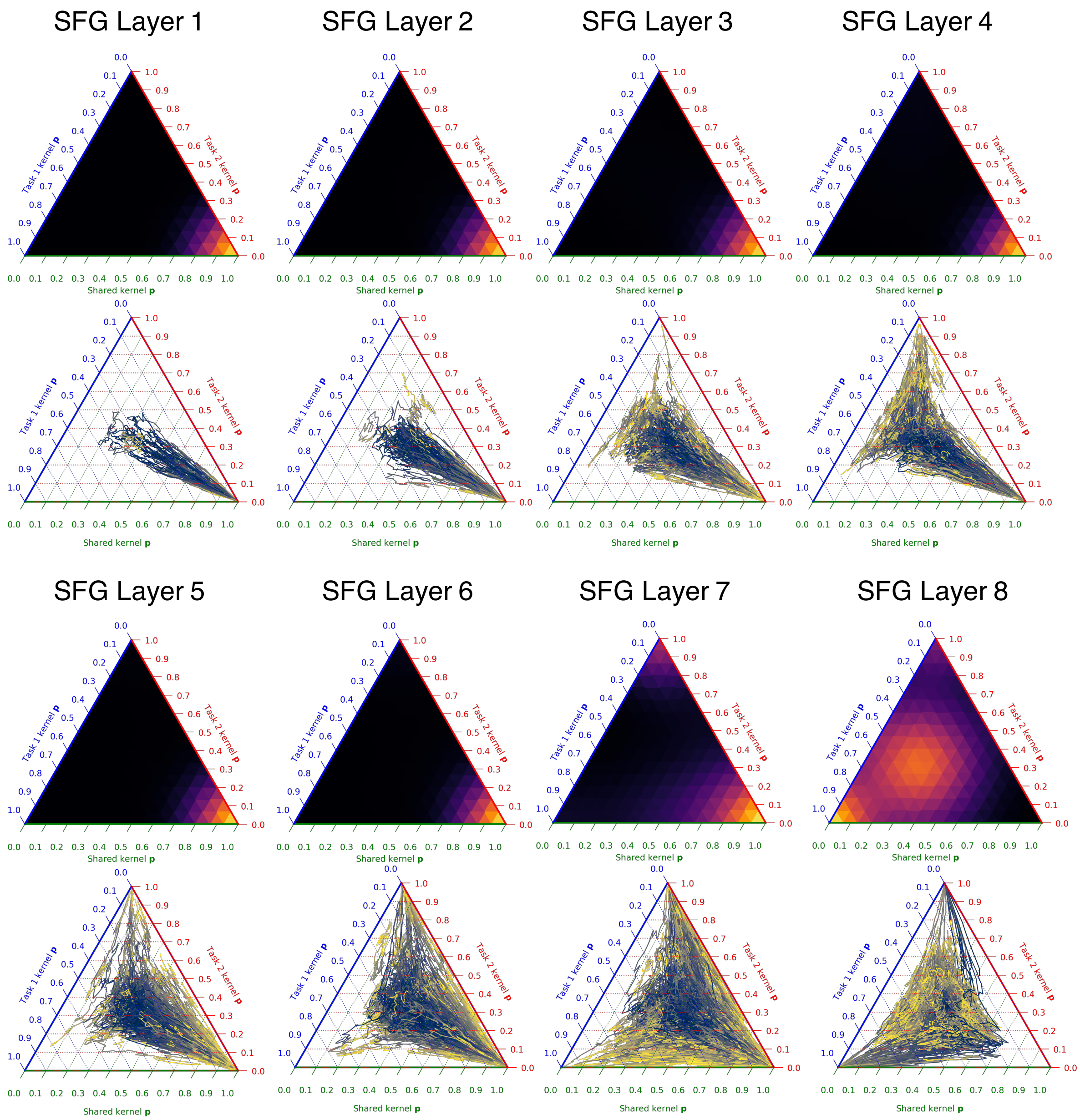


Figure 10: Density plots and trajectory plots of the learned grouping probabilities for the SFG-VGG11 architecture. The density plots represents the final learned probabilities per layer for each kernel. The trajectory plots represent how the grouping probabilities are learned during training and thus how the connectivity is determined. Histograms of the grouping probabilities were smoothed with a Gaussian kernel with $\sigma = 1$. The densities are mapped to and visualised in the 2-simplex using `python-ternary` [9].

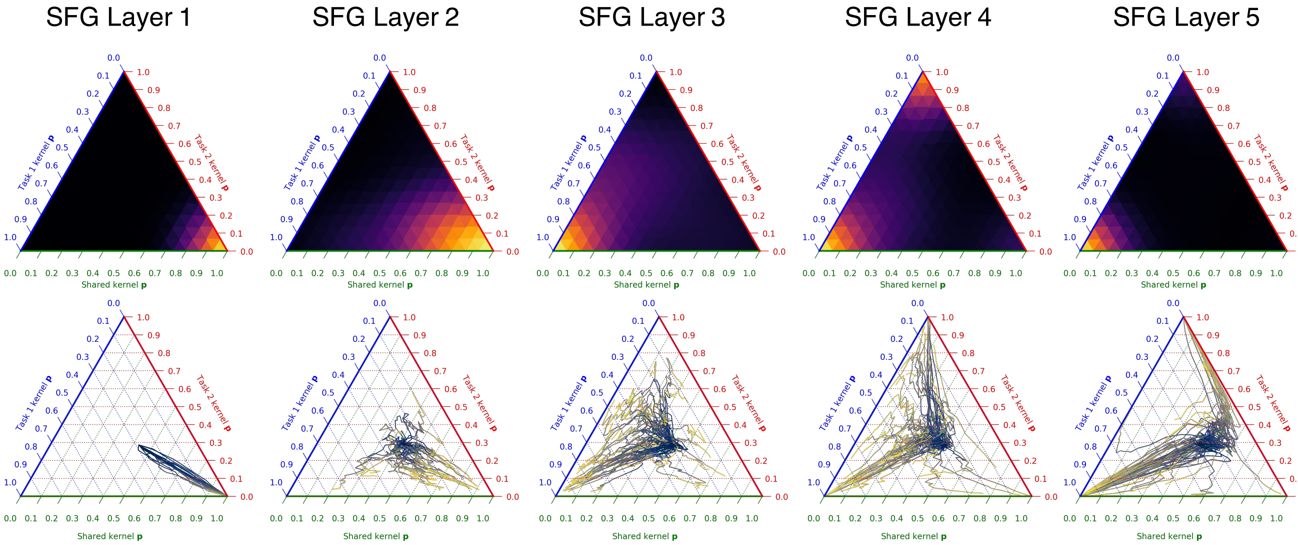


Figure 11: Density plots and trajectory plots of the learned grouping probabilities for the SFG-HighResNet architecture. The density plots represents the final learned probabilities per layer for each kernel. The trajectory plots represent how the grouping probabilities are learned during training and thus how the connectivity is determined.

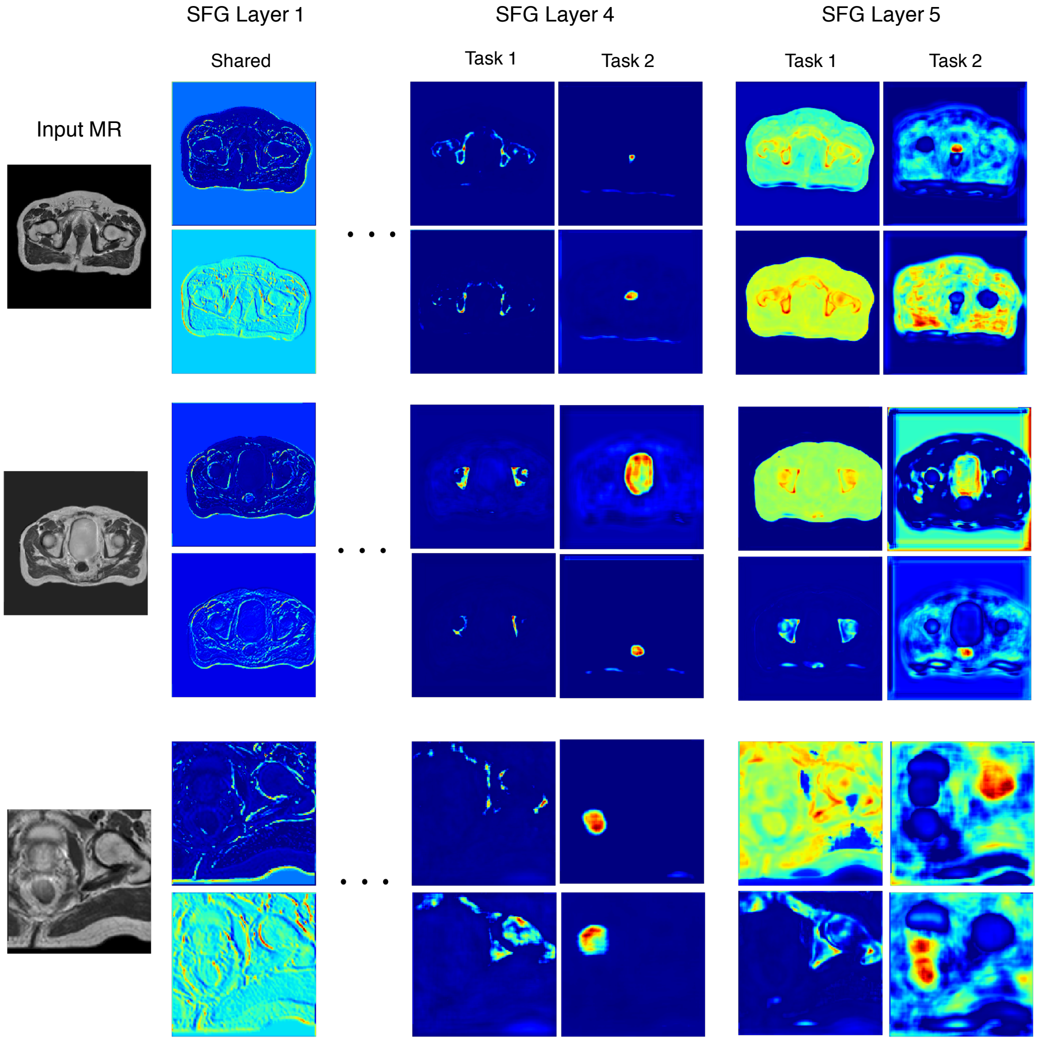
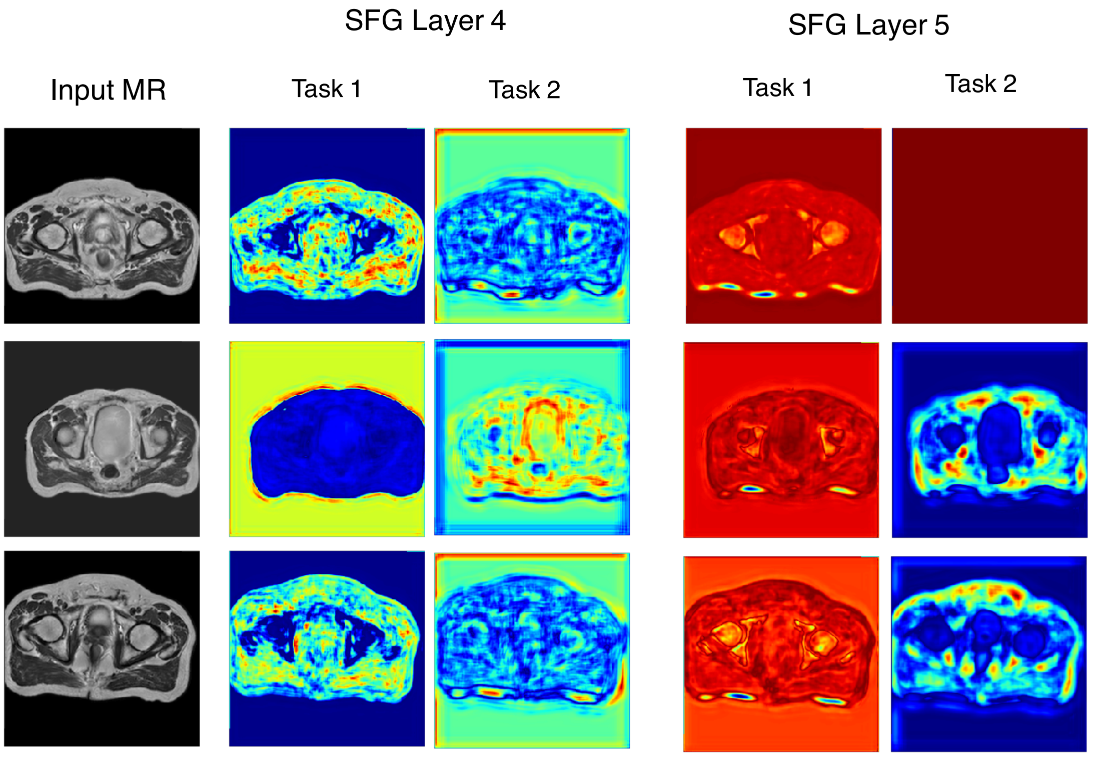


Figure 12: Example activations for kernels with low entropy of $\mathbf{p}$ (i.e. group assignment with high confidence) for three input MR slices in the SFG-HighResNet multi-task network. Columns "Shared", "Task 1" & "Task 2" display the results from the shared, CT-synthesis and organ-segmentation specific filter groups in respective layers. We illustrate activations stratified by group in layer 1 (SFG layer 1), layer 16 (SFG layer 4) and layer 17 (SFG layer 5).



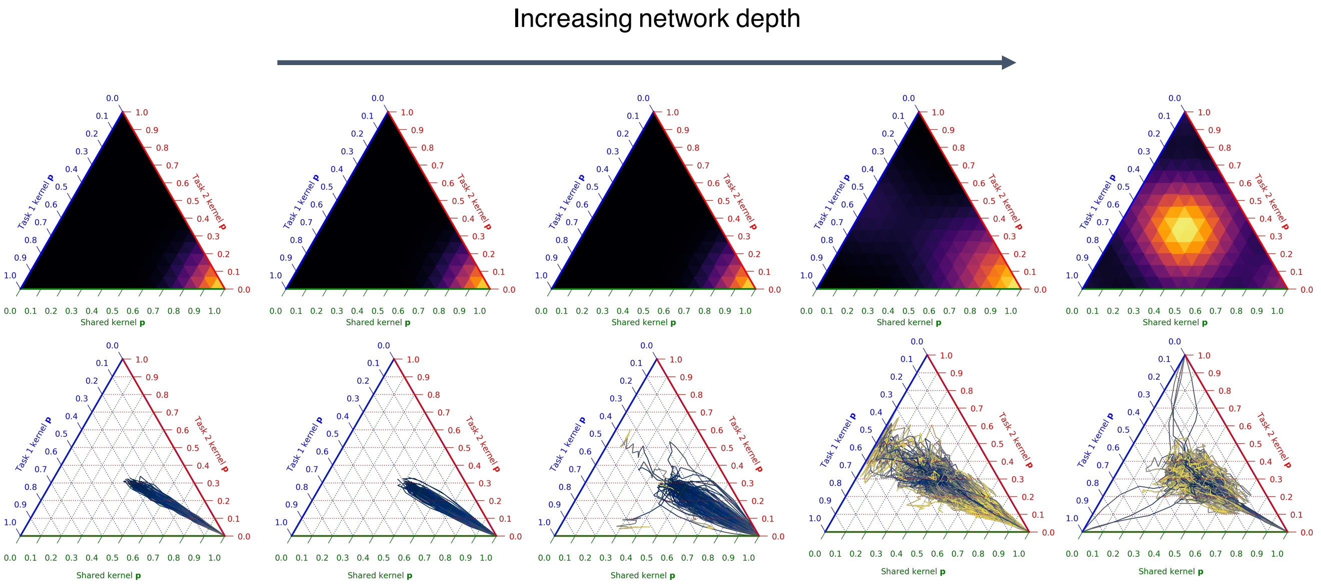


Figure 13: Top: density plots for the learned grouping probabilities at each SFG layer in a model where we trained on duplicate tasks i.e. task 1 is CT synthesis and task 2 is also CT synthesis. Bottom: trajectories of the grouping probabilities during training.