Supplementary material: 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

Contents

*Both authors contributed equally

1. Training and implementation: additional details

1.1. Optimisation, regularisation and initialisation

All networks were trained with ADAM optimiser [9] 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 [10]. The scheduler $\tau = \max(0.10, \exp(-rt))$ recommended in [8] 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).

1.2. 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.

1.3. 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 [15].

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 [7] 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 [13]. 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.

1.4. Implementation details

We used Tensorflow and implemented our models within the NiftyNet framework [2]. 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) [12] 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 [6] 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 [8] at test-time using the temperature value τ that corresponded to the iteration in τ annealing schedule.

2. CNN architectures and details

We include schematics and details of the single-task VGG11 [14] and HighResNet [11] networks in Fig. 1. In this work, we constructed multi-task architectures by augmenting these networks with the proposed SFG modules. We used the PReLU activation function [4] in all networks. For the residual blocks used in the HighResNet networks in Fig. 1 (ii), we applied PReLU and batch-norm as pre-activation [5] to the convolutional layers. The SFG module was used to cluster the kernels in every coloured layer in Fig. 1, 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 [12] 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.

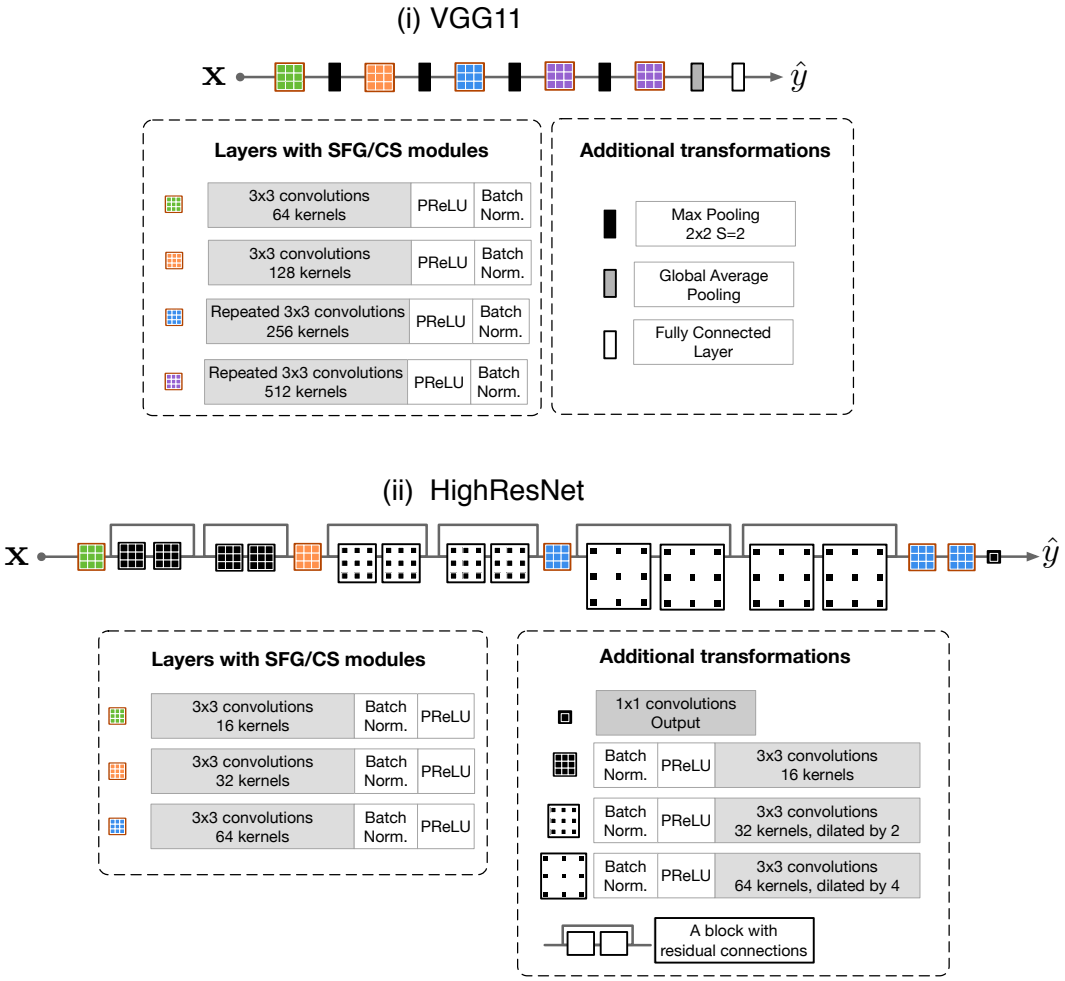

Figure 1. Illustration of the single-task architectures, (i) VGG11 and (ii) HighResNet used for UTKFace and medical imaging dataset, respectively. In each architecture, the coloured components indicate the layers to which SFG or cross-stitch (CS) modules are applied when extended to the multi-task learning 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)

.

3. Learned grouping probability plots

In this section, we illustrate density plots of the learned grouping probabilities $\mathbf{p}$ for each trained network (Fig. 2 and Fig. 3). 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. 2 and Fig. 3 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. 4, 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. 4, SFG layer ≥ 4). This is illustrated by high density clusters at task-specific vertices and by the trajectory plots.

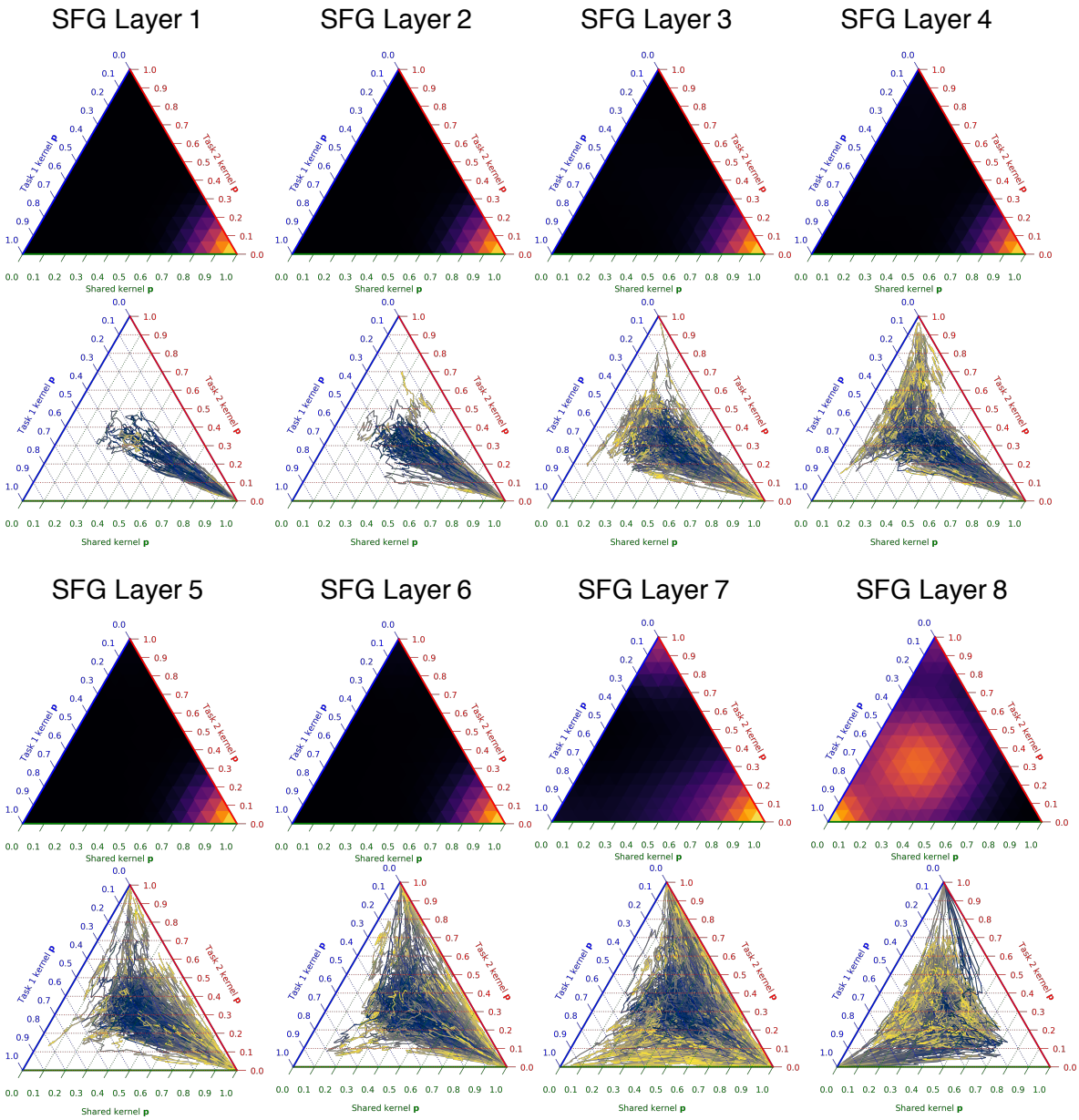

Figure 2. Density plots and trajectory plots of the learned grouping probabilities for the SFG-VGG11 architecture. The density plots represents the final learned probabilities per layer for each kernel. The trajectory plots represent how the grouping probabilities are learned during training and thus 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` [3].

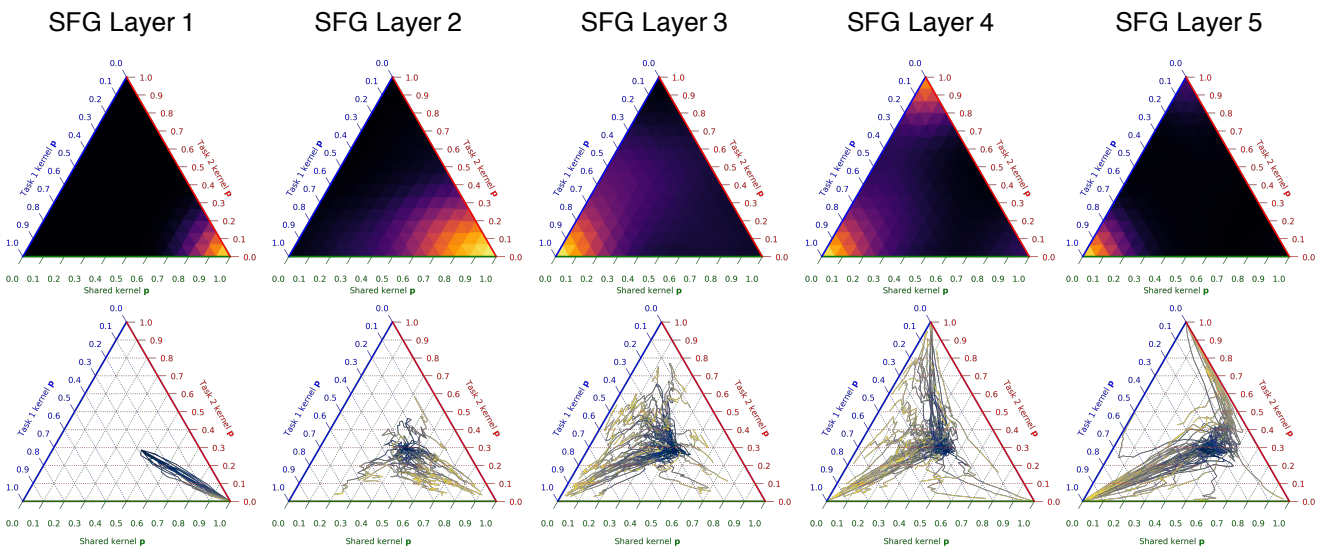

Figure 3. Density plots and trajectory plots of the learned grouping probabilities for the SFG-HighResNet architecture. The density plots represents the final learned probabilities per layer for each kernel. The trajectory plots represent how the grouping probabilities are learned during training and thus how the connectivity is determined.

4. 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. 4 and Fig. 5.

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. 4 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 [1].

Secondly, we looked at activation maps from kernels with high entropy of $\mathbf{p}$ (i.e. highly uncertain group assignment) in Fig. 5. In contrast to Fig. 4, 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.

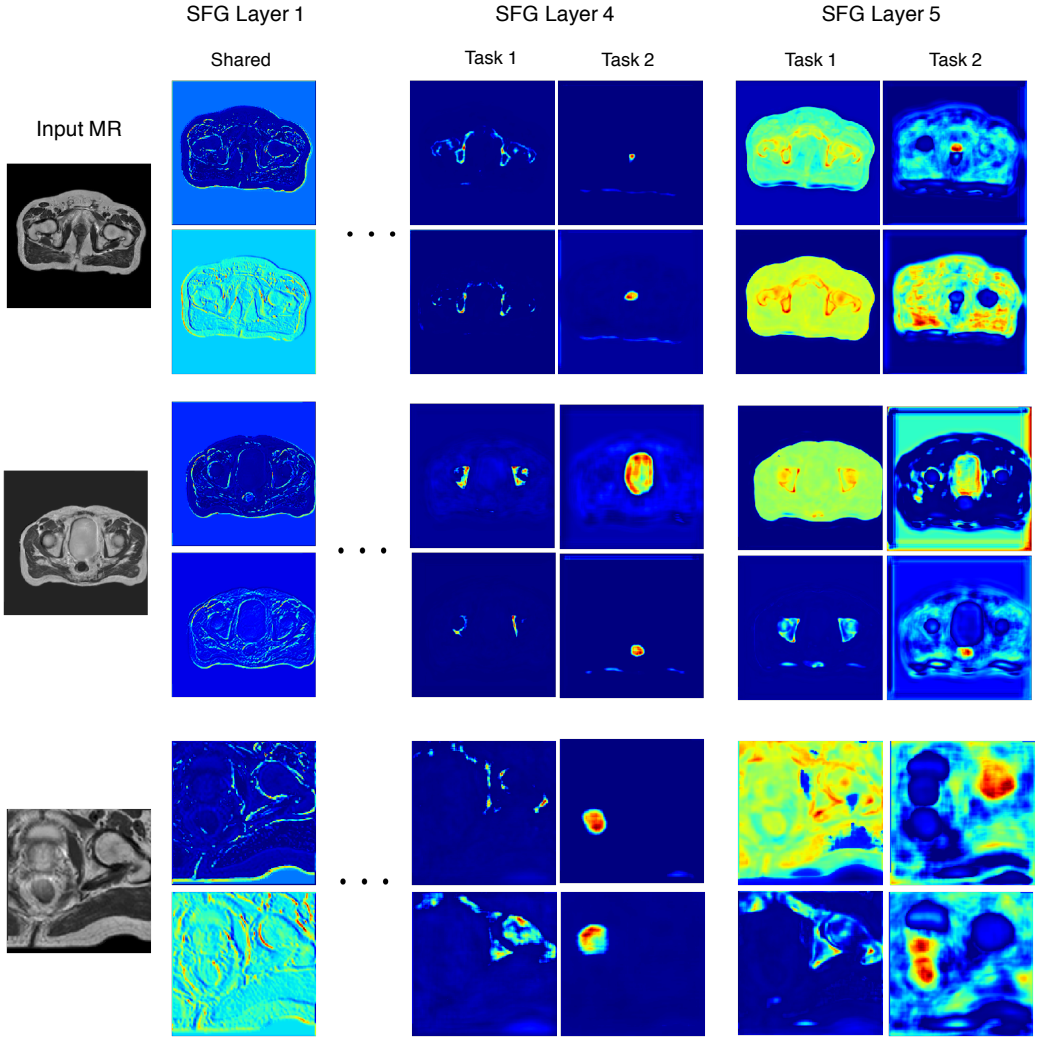

Figure 4. Example activations for kernels with low entropy of $\mathbf{p}$ (i.e. group assignment with high confidence) for three input MR slices in the SFG-HighResNet multi-task network. Columns "Shared", "Task 1" & "Task 2" display the results from the shared, CT-synthesis and organ-segmentation specific 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).

[1]The bones are generally the most difficult region to synthesise CT intensities from an input MR scan [1].

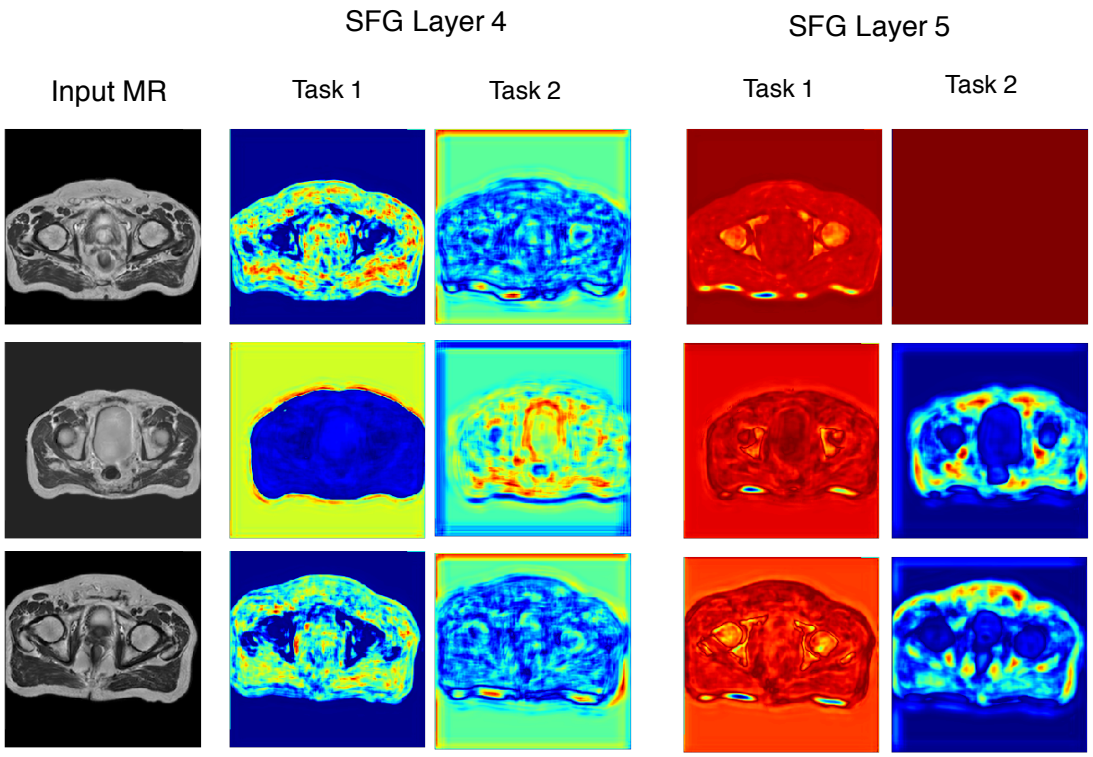

Figure 5. Example activations for kernels with high entropy (i.e. group assignment with low confidence) for three input MR slices in the SFG-HighResNet multi-task network. Columns "Shared", "Task 1" & "Task 2" display the results from the shared, CT-synthesis and organ-segmentation specific filter groups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

5. 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. 6 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.

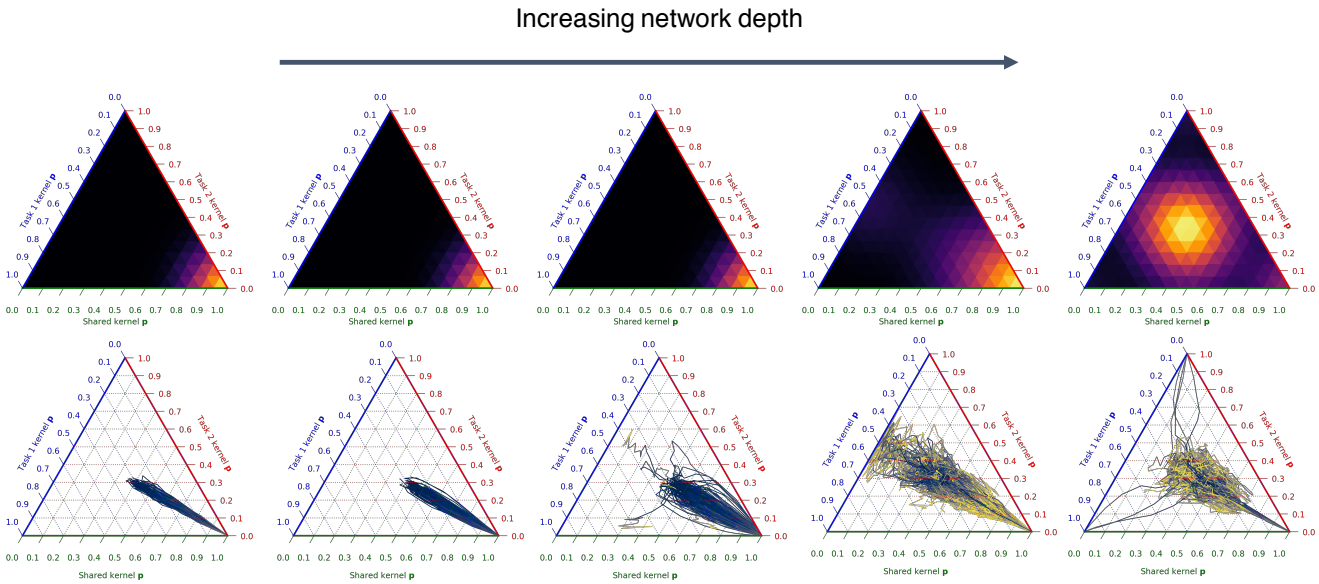

Figure 6. Top: density plots for the learned grouping probabilities at each SFG layer in a model where we trained on duplicate tasks i.e. task 1 is CT synthesis and task 2 is also CT synthesis. Bottom: trajectories of the grouping probabilities during training.

References

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

[2] 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.

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

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

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

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

[7] 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.

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

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

[10] 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.

[11] 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.

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

[13] 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.

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

[15] 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.