Classification with a domain shift in medical imaging

Alessandro Fontanella
School of Informatics
University of Edinburgh
a.fontanella@sms.ed.ac.uk

Emma Pead
VAMPIRE Project, Centre for Clinical Brain Sciences
University of Edinburgh
epead@ed.ac.uk

Tom MacGillivray
VAMPIRE Project, Centre for Clinical Brain Sciences
University of Edinburgh
T.J.MacGillivray@ed.ac.uk

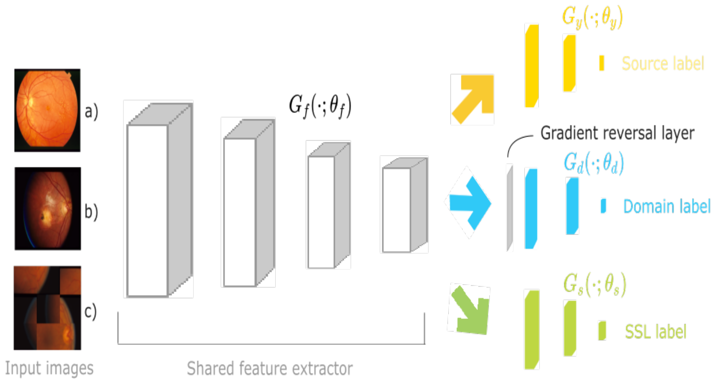
Miguel O. Bernabeu
Centre for Medical Informatics, Usher Institute
University of Edinburgh
Miguel.Bernabeu@ed.ac.uk

Amos Storkey
School of Informatics
University of Edinburgh
a.storkey@ed.ac.uk

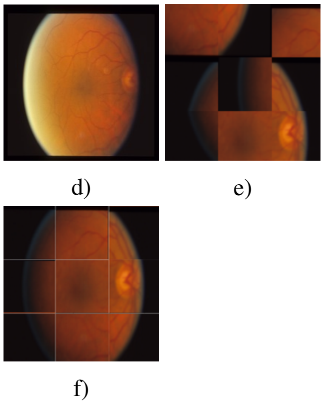
1 Introduction and Related Work

In medical imaging, the amount of unlabelled data often exceeds that of labelled data [28]. When labelled data is lacking, we may still be able to obtain training sets that are big enough for training large-scale deep models, but that are from a different data distribution from the actual data to be encountered at test time. In particular, retinal images are obtained through devices from different vendors, which have diverse characteristics. Moreover, shifts in data distribution may be found even among datasets of images collected with devices from the same vendor due to patient cooperation during the examination or interoperator variability [20, 14]. In this study, we propose a method that is able to exploit additional unlabelled datasets, possibly with a domain shift, to improve predictions on our labelled data. In order to do so, we exploit a Convolutional Neural Network (CNN) architecture with three classifiers that share a common feature backbone: (1) classifies labelled samples, (2) determines the dataset of origin of each sample, and (3) solve a self-supervised task, in particular predicting image rotations and solving Jigsaw puzzles, on unlabelled data. Classifier (2) is trained to not be able to distinguish between different datasets, so that data is projected into a common feature space, while the self supervised learner (3) learns features from unlabelled data, that are shared through the common feature backbone. Previously, in [6], Carlucci et al. tested how solving Jigsaw puzzles on unlabelled samples, while classifying labelled images, helps in learning more general representations. However, there are two key differences with respect to our work. First, their architecture does not include a domain classifier. Second, they focus on the classification performance on the unlabelled dataset, while we focus on the performance on the labelled one. The main contributions of our work are the following: 1) we proposed a new architecture that is able to exploit unlabelled data with a domain shift to improve predictions on labelled data. It works by learning features through self-supervised learning (SSL) while projecting all the data onto the same space to achieve better transfer; 2) we run a series of experiments on Office-31 dataset, to test that our method can be successfully applied to natural images; 3) we tested this method on retinal images, in particular for age-related macular degeneration (AMD) and diabetic retinopathy (DR) grading, consistently improving the results of the baselines (from 78.83% to 83.53% average test accuracy on AMD grading and from 60.00% to 67.79% average test accuracy on DR grading). We qualitatively and quantitatively verified, through saliency maps, how the proposed method is able to focus more

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.



Architecture of the method proposed. The common feature extractor projects both labelled and unlabelled samples onto the same feature space. The label prediction loss is minimised for labelled images (a), the domain classification loss for labelled and original unlabelled images (b) and the SSL loss for unlabelled images modified for the SSL task (c). A gradient reversal layer connects the feature extractor to the domain classifier, ensuring that the feature distributions of the two datasets are made similar.



SSL training procedure for Jigsaw puzzles. From original unlabelled samples (d), 9 patches are extracted and shuffled (e) before entering the feature extractor. Images are reassembled by the network (f)

closely on drusen spots in AMD images than the baselines (AUC of the precision-recall curves from 0.44 to 0.50), making it more clinically interpretable.

2 Methods and Data

The architecture proposed can be seen as an extension and revision of previous work [10] on unsupervised domain adaptation, where we aimed to optimise the accuracy on a supervised problem under domain shift, rather than the unsupervised one. As in [10], a shared feature extractor projects labelled and unlabelled images to the same space. During training, the label prediction loss is minimised for labelled samples and the domain classification loss for all samples. The domain classifier is connected to the feature extractor by a gradient reversal layer. In this way parameters of the feature extractor that maximise the loss of the domain classifier are learned, ensuring that the feature distributions of the two samples are made similar. After the shared feature backbone, we also add another classifier, solving a self-supervised task on unlabelled images, since features learned through self-supervised tasks generalize better to new domains [1]. However, such unlabelled images may have a negative impact if the domain shift between the unlabelled and labelled dataset is too big [27]. For this reason, we make sure that the two datasets are projected into the same feature space. We call the common feature extractor G_f, the label predictor for source samples G_y, the domain classifier G_d, the classifier for the self-supervised task G_s. We would like to minimise the prediction loss on labelled images optimising the parameters of the feature extractor θ_f and of the label predictor θ_y, we would like to optimise the parameters of the feature extractor to maximise the loss of the domain classifier and the parameters of the domain classifier θ_d to minimise the same loss. Finally, we would like to optimize the parameters θ_f of the feature extractor and the parameters of the self-supervised classifier θ_s to minimise the loss on the self-supervised task G_s. More formally, we would like to find the parameters delivering a saddle point of the functional:

$$\begin{aligned}
F(\theta_f,\theta_y,\theta_d,\theta_s) = &\sum_{\substack{i=1,\ldots,N\\ d_i=0}} L_y(G_y(G_f(x_i;\theta_f);\theta_y),y_i) + \beta \sum_{\substack{i=1,\ldots,N\\ d_i=1}} L_s(G_s(G_f(x_i;\theta_f);\theta_s),y_i)\\
&- \lambda \sum_{i=1,\ldots,N} L_d(G_d(G_f(x_i;\theta_f);\theta_d),y_i)\\
=& \sum_{\substack{i=1,\ldots,N\\ d_i=0}} L_y^i(\theta_f,\theta_y) + \beta \sum_{\substack{i=1,\ldots,N\\ d_i=1}} L_s^i(\theta_f,\theta_s) - \lambda \sum_{i=1,\ldots,N} L_d^i(\theta_f,\theta_d) \qquad (1)
\end{aligned}$$

where $L_y(\cdot,\cdot)$ is the loss for the labels prediction, $L_s(\cdot,\cdot)$ for the self-supervised task and $L_d(\cdot,\cdot)$ for the domain classification. L_y^i, $L_s^i$and L_d^i are the corresponding loss functions at the $i-th$ training sample, $d_i = 0$ or 1 for labelled or unlabelled data respectively, λ is the meta-parameter associated to the gradient reversal layer and β can be used to weigh the loss related to the self-supervised task differently. More details about our training procedure can be seen in Appendix A. For the experiments on AMD grading, we employed 39 control, 26 early AMD and 47 late AMD images from STARE [15] as labelled dataset and 275 images from AREDS [26] as the unlabelled one. For DR grading, 167 controls, 25 DR grade 1, 168 DR grade 2, 91 DR grade 3, 62 DR grade 4 images from IDRID [23] and 1741 samples from Messidor-2 [9] as unlabelled dataset. For the experiments on natural images, we used OFFICE-31 dataset [24], employing either dSLR or webcam as labelled and Amazon as unlabelled. These are the most challenging settings since dSLR and webcam datasets have a small domain shift, while Amazon has a bigger domain shift with the other two datasets [16].

3 Results and Discussion

In the following, our method is compared with the baseline obtained initialising a CNN with the weights learned from training on ImageNet and fine-tuning the network on the labelled samples. Using 100 samples from dSLR as labelled data and Amazon as target, our method improves average test accuracy of the fine-tuning approach from 77.83% to 80.92% using VGG-16 and from 77.10% to 78.03% using ResNet-18. With 200 labelled samples, from 93.30% to 93.96% and from 90.68% to 91.41% using VGG-16 and ResNet-18 respectively. With 70% of dSLR as training data, fine-tuning achieves 95.52% and 94.91% using VGG-16 and ResNet-18 respectively. These figures are improved by our method to 96.00% and 95.73%. Using 70% of webcam as labelled data and Amazon as unlabelled data, our approach improves the results of fine-tuning from 98.59% to 98.83% using VGG-16 and from 98.59% to 99.00% using ResNet-18. For AMD grading, we trained our models to discriminate between controls, early AMD and late AMD. The results achieved are displayed in Table 1, presenting average test accuracies and standard errors over 5 runs. We can observe how our approach improves the baseline from 78.83% to 83.53% with VGG-16 and from 75.29% to 81.18% with ResNet-18. AUC of the ROC curves improved from 0.89 to 0.96 with VGG-16 and remained the same at 0.93 with ResNet-18. We also created precision-recall curves of the saliency maps obtained with the two approaches and verified how our method achieves AUC of 0.5, while the baseline 0.44 when compared against the ground truth lables of drusen coordinates. The precision-recall curves and more results and details on AMD grading are in Appendix B. For DR grading, we trained our models to discriminate between controls and DR of grade 1 to 4, having a 5-class classification task. In Table 2, we report the results obtained. We can observe how our method is able to improve the fine-tuning baseline from 59.22% to 67.79% using VGG-16 and from 60.00% to 63.38% using ResNet-18. We were also able to improve AUC of the ROC curves from 0.88 to 0.90 with VGG-16 and from 0.87 to 0.89 with ResNet-18. More results and details on DR grading are reported in Appendix C.

Table 1: Average test accuracies and standard errors obtained grading AMD on STARE dataset with our method and with transfer learning from ImageNet followed by fine-tuning. Our method shows better results both using VGG-16 and ResNet-18

	VGG-16	ResNet-18
Fine-tuning	$78.83\% \pm 3.94\%$	$75.29\% \pm 4.53\%$
Ours - Jigsaw puzzles	$\mathbf{83.53}\% \pm \mathbf{1.05}\%$	$78.82\% \pm 3.94\%$
Ours - RotNet	$81.18\% \pm 1.97\%$	$\mathbf{81.18}\% \pm \mathbf{4.53}\%$

Table 2: Average test accuracies and standard errors obtained grading DR on IDRID dataset with our method and with transfer learning from ImageNet followed by fine-tuning. Our method shows better results both using VGG-16 and ResNet-18 as the backbone.

	VGG-16	ResNet-18
Fine-tuning	$59.22\% \pm 2.95\%$	$60.00\% \pm 2.63\%$
Ours - Jigsaw puzzles	$66.23\% \pm 2.77\%$	$\mathbf{63.38}\% \pm \mathbf{3.25}\%$
Ours - RotNet	$\mathbf{67.79}\% \pm \mathbf{2.47}\%$	$61.82\% \pm 1.93\%$

We also performed ablation studies to test whether all the components of our architecture are needed to obtain the best results and we obtained that the full architecture is necessary both on AMD and Office-31 data, especially in the low data regime. More details are presented in Appendix D. It is also worth noting that all the results described in this study were achieved without particularly focusing on hyperparameter tuning, with parameters weighing the sum of the three losses in Equation 1 fixed at 1, to prove the effectiveness of our idea in general settings. We believe that the self-supervised approach to learn more transferable features is a promising one and our work shows its potential to leverage unlabelled data, which is typically available in medical imaging, to improve the classification on labelled data. It also allows to learn more clinically relevant features, as shown through the saliency maps analysis. Our work could foster further research into this area to achieve better detection of several other pathologies.

Broader Impact

Age-related macular degeneration (AMD) and diabetic retinopathy (DR) are two of the leading causes of blindness worldwide [2, 18]. Early detection of retinal changes in colour fundus photographs, such as drusen in AMD and the growth of new vessels (neovascularisation) in DR, are vital for referral of a patient to a specialised eye hospital services to receive sight preserving treatment. Deep learning is continuously producing state-of-the-art results and have been used to develop methods for automatic AMD and DR prediction and screening [5, 4, 22, 9, 8, 19]. However, a significant challenge in developing high performance retinal image deep learning models is the availability of labelled data, where the amount of unlabelled data often exceeds that of labelled data [28]. In a clinical environment images are captured on a variety of devices, have different poses (e.g. optic disc centred, macular centred), high interpatient variability (e.g. retinal pigment epithelium characteristics) and image quality (e.g. media opacity, artefact) and will exhibit a large shift in data distribution [20, 14]. For this reason, deep learning methods sometimes are not able to replicate their results in real world settings [3]. In this paper, we propose a method that is able to exploit unlabelled datasets, possibly with a domain shift, to improve predictions on the labelled data and the generalisation capability of our model. This idea could be applied for early screening and detection of several pathologies and ultimately achieve better health outcomes for the general population

Acknowledgments and Disclosure of Funding

This work was supported by the United Kingdom Research and Innovation (grant EP/S02431X/1), UKRI Centre for Doctoral Training in Biomedical AI at the University of Edinburgh, School of Informatics.

References

[1] Isabela Albuquerque, Nikhil Naik, Junnan Li, Nitish Keskar, and Richard Socher. Improving out-of-distribution generalization via multi-task self-supervised pretraining. *arXiv preprint arXiv:2003.13525*, 2020.

[2] Jayakrishna Ambati and Benjamin J Fowler. Mechanisms of age-related macular degeneration. *Neuron*, 75(1):26–39, 2012.

[3] Emma Beede, Elizabeth Baylor, Fred Hersch, Anna Iurchenko, Lauren Wilcox, Paisan Ruamviboonsuk, and Laura M Vardoulakis. A human-centered evaluation of a deep learning system deployed in clinics for the detection of diabetic retinopathy. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[4] Philippe M Burlina, Neil Joshi, Katia D Pacheco, David E Freund, Jun Kong, and Neil M Bressler. Use of deep learning for detailed severity characterization and estimation of 5-year risk among patients with age-related macular degeneration. *JAMA ophthalmology*, 136(12):1359–1366, 2018.

[5] Philippe M Burlina, Neil Joshi, Michael Pekala, Katia D Pacheco, David E Freund, and Neil M Bressler. Automated grading of age-related macular degeneration from color fundus images using deep convolutional neural networks. *JAMA ophthalmology*, 135(11):1170–1176, 2017.

[6] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[8] Jordi de La Torre, Aida Valls, and Domenec Puig. A deep learning interpretable classifier for diabetic retinopathy disease grading. *Neurocomputing*, 396:465–476, 2020.

[9] Etienne Decencière, Xiwei Zhang, Guy Cazuguel, Bruno Lay, Béatrice Cochener, Caroline Trone, Philippe Gain, Richard Ordonez, Pascale Massin, Ali Erginay, et al. Feedback on a publicly distributed image database: the messidor database. *Image Analysis & Stereology*, 33(3):231–234, 2014.

[10] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189, 2015.

[11] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[12] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6391–6400, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Jung Taeck Hong, Kyung Rim Sung, Jung Woo Cho, Sung-Cheol Yun, Sung Yong Kang, and Michael S Kook. Retinal nerve fiber layer measurement variability with spectral domain optical coherence tomography. *Korean Journal of Ophthalmology*, 26(1):32–38, 2012.

[15] AD Hoover, Valentina Kouznetsova, and Michael Goldbaum. Locating blood vessels in retinal images by piecewise threshold probing of a matched filter response. *IEEE Transactions on Medical imaging*, 19(3):203–210, 2000.

[16] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.

[17] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1920–1929, 2019.

[18] Ryan Lee, Tien Y Wong, and Charumathi Sabanayagam. Epidemiology of diabetic retinopathy, diabetic macular edema and related vision loss. *Eye and vision*, 2(1):1–25, 2015.

[19] Feng Li, Zheng Liu, Hua Chen, Minshan Jiang, Xuedian Zhang, and Zhizheng Wu. Automatic detection of diabetic retinopathy in retinal fundus photographs based on deep learning algorithm. *Translational vision science & technology*, 8(6):4–4, 2019.

[20] Jean Claude Mwanza, Mohamed G Gendy, William J Feuer, Wei Shi, and Donald L Budenz. Effects of changing operators and instruments on time-domain and spectral-domain oct measurements of retinal nerve fiber layer thickness. *Ophthalmic Surgery, Lasers and Imaging Retina*, 42(4):328–337, 2011.

[21] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[22] Yifan Peng, Shazia Dharssi, Qingyu Chen, Tiarnan D Keenan, Elvira Agrón, Wai T Wong, Emily Y Chew, and Zhiyong Lu. Deepseenet: a deep learning model for automated classification of patient-based age-related macular degeneration severity from color fundus photographs. *Ophthalmology*, 126(4):565–575, 2019.

[23] Prasanna Porwal, Samiksha Pachade, Ravi Kamble, Manesh Kokare, Girish Deshmukh, Vivek Sahasrabuddhe, and Fabrice Meriaudeau. Indian diabetic retinopathy image dataset (idrid): a database for diabetic retinopathy screening research. *Data*, 3(3):25, 2018.

[24] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] The Age-Related Eye Disease Study et al. The age-related eye disease study (areds): Design implications areds report no. 1. *Controlled clinical trials*, 20(6):573–600, 1999.

[27] Jong-Chyi Su, Subhransu Maji, and Bharath Hariharan. When does self-supervision improve few-shot learning? *arXiv preprint arXiv:1910.03560*, 2019.

[28] Youngjin Yoo, Tom Brosch, Anthony Traboulsee, David KB Li, and Roger Tam. Deep learning of image features from unlabeled data for multiple sclerosis lesion segmentation. In *International Workshop on Machine Learning in Medical Imaging*, pages 117–124. Springer, 2014.

Appendix

A Training of our architecture

We observe that the weighting parameter for the loss associated to the source samples classifier can be assumed to be 1 without loss of generality, since if we called such a weighting parameter α, it would always be possible to divide the entire expression by α (for $\alpha \neq 0$), obtaining the new hyperparamaters $\beta' = \frac{\beta}{\alpha}$, $\lambda' = \frac{\lambda}{\alpha}$.

In our optimisation, we are looking for the parameters $\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_s, \hat{\theta}_d$ producing a saddle point of Equation 1, that is:

$$(\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_s) = \arg\min_{\theta_f, \theta_y, \theta_s} F(\theta_f, \theta_y, \theta_s, \hat{\theta}_d) \tag{2}$$

$$\hat{\theta}_d = \arg\max_{\theta_d} F(\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_s, \theta_d) \tag{3}$$

The saddle point satisfying 2 and 3 can be found by SGD, updating the parameters with the following stochastic updates:

$$\theta_f \leftarrow \theta_f - \mu \left(\frac{\partial L_y^i}{\partial \theta_f} + \beta \frac{\partial L_s^i}{\partial \theta_f} - \lambda \frac{\partial L_d^i}{\partial \theta_f} \right) \tag{4}$$

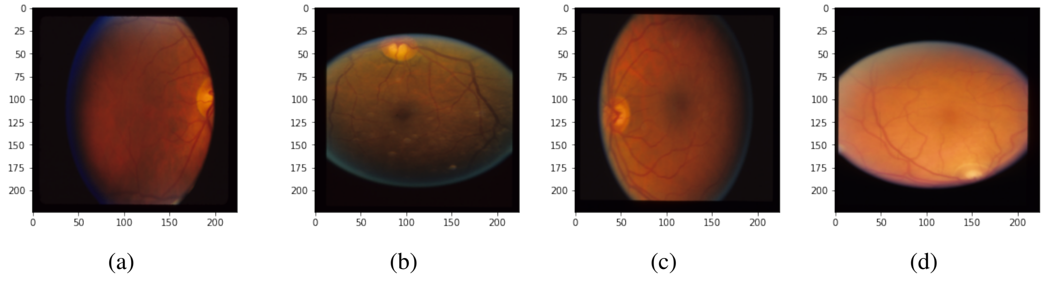
$$\theta_y \leftarrow \theta_y - \mu \frac{\partial L_y^i}{\partial \theta_y} \tag{5}$$

$$\theta_s \leftarrow \theta_s - \mu\beta \frac{\partial L_s^i}{\partial \theta_s} \tag{6}$$

$$\theta_d \leftarrow \theta_d + \mu\lambda \frac{\partial L_d^i}{\partial \theta_d} \tag{7}$$

where μ is the learning rate, determining the step size at each iteration.

As self-supervised task we tested predicting image rotations [11] and solving Jigsaw puzzles [21], since recently [12] and [17] compared different self-supervised tasks and concluded that they are among the most effective. On the other hand, we didn't choose SimCLR, even though it reached state-of-the-art results on ImageNet because it requires very large batch sizes and consequently big amounts of data [7], while we are working in the low data regime. When predicting image rotations, each image in the unlabelled dataset is rotated by 0, 90, 180 or 270 degrees (cfr. Figure 2) and the self-supervised classifier is trained on the 4-class image classification task of recognizing one of the four image rotations. To be able to recognise the rotation that was applied, the network need to understand the concept of the objects displayed in the image, such as their type, their pose and their location in the image [11].



(a) (b) (c) (d)

Figure 2: Examples of images rotated by 0° (a), 90° (b), 180° (c), 270° (d). The key idea is that it is necessary to understand the concepts of the images to predict their rotation

When solving Jigsaw puzzles we extracted tiles from the unlabelled images and shuffled them. The goal of the network is to re-assemble the image and restore each tile in the correct position. In order to do so, we first defined a set of Jigsaw puzzles permutations and assigned an index to each entry. An example permutation for 3×3 Jigsaw puzzles can be $P = (2, 9, 3, 1, 5, 4, 6, 8, 7)$. During training, we randomly picked a permutation, re-arranged the patches of the image in that order and tasked the network to output the index of the permutation used. As in [6], after extracting the patches from the image, we re-assembled them in the order given by the selected permutation, so that we could deal with a classification task on the recomposed image, which has the same dimension of the original one. In this way, we were able to exploit the same feature extractor used for the other tasks, something that would not have been possible if we followed the classic approach of dealing with different image patches separately and re-assemble them only towards the end of the learning process. In particular, we used a 3×3 grid to decompose the images in 9 patches which are then shuffled according to the permutation. However, if we considered every possible permutation of 9 elements, we would have $9! = 362,880$ of them. For this reason, we picked only 30 permutations and in order to maximise training capacity we chose these 30 permutations according to the Hamming distance algorithm in [21]. The Hamming distance is the number of different tiles locations between 2 permutations. Large Hamming distances are desirable because they make the task less ambiguous [21]. For example if the difference between two permutations is only in the position of two tiles that are very similar, the prediction task will be almost impossible. At the same we don't want the task to be too simple [21]. For this reason, we tried to remove the shortcuts that may be exploited by the network to solve the classification task. In particular, since adjacent patches have similar low-level statistics, like mean and standard deviation, that may allow the network to find the arrangement of the patches, we normalised the mean and standard deviation of each patch separately. Furthermore, we removed the continuity of edges. In order to do so, we first resized each patch to 75×75, than we extracted a centered crop of size 64×64 and resized it back to 75×75. Finally, we jittered the color channels of the images to prevent the network from estimating the position of the tiles from the chromatic aberration, i.e. the spatial shift between channels that increases from the center of the images to the borders. For the feature backbone and source labels classifier architecture, we tested VGG-16 [25] and ResNet-18 [13]. In bith cases, we modified the fully connected layers to reflect the number of classes in our experiments. The domain classifier was defined with three linear layers of decreasing dimensions with ReLUs activations, while for the classifier of the self-supervised task we followed the same architecture of the fully connected layers of the CNN employed for predicting the labels of source images. As data augmentation, we performed horizontal and vertical flips and rotations up to $45°$. Each labelled dataset was split into training, validation and test sets, each containing 70%, 15% and 15% of the data respectively.

B More results and details about AMD grading

We report the average ROC curves, obtained with VGG-16 and ResNet-18, in Figures 3 and 4 respectively. We can observe how with VGG-16 our method shows a good improvement over the fine-tuning baseline. In particular, using RotNet as self-supervised task the AUC increases from 0.89 to 0.96. On the other hand, with ResNet-18 all the methods show very similar AUC and the standard errors intervals of the ROC curves largely overlap.

To recap the results of AMD grading, in terms of accuracy our method outperformed the fine-tuning baseline with both VGG-16 and ResNet-18, while in terms of AUC it improved the baseline with VGG-16 and achieved very similar results with ResNet-18. A possible explanation for the different performance of the two CNNs could be the following: VGG-16 has 138 million parameters [25], while ResNet-18 only 11 million [13]. This means that VGG-16, being an order of magnitude wider, can learn more high dimensional and general representations. This is particularly important since target images, on which the self-supervised task is trained, have a different distribution than source images, to which the representations learned will be transferred. For this reason, the broader representation learned by VGG-16 may allow to identify more components of the images that can be later recycled for the supervised classification task. This result is also in line with the latest developments in the self-supervised literature, according to which wider networks perform better in self-supervised tasks [7, 17].

We also show some saliency maps, through which we are able to show how our approach is able to focus more closely on drusen spots than the fine-tuning baseline and therefore is more clinically

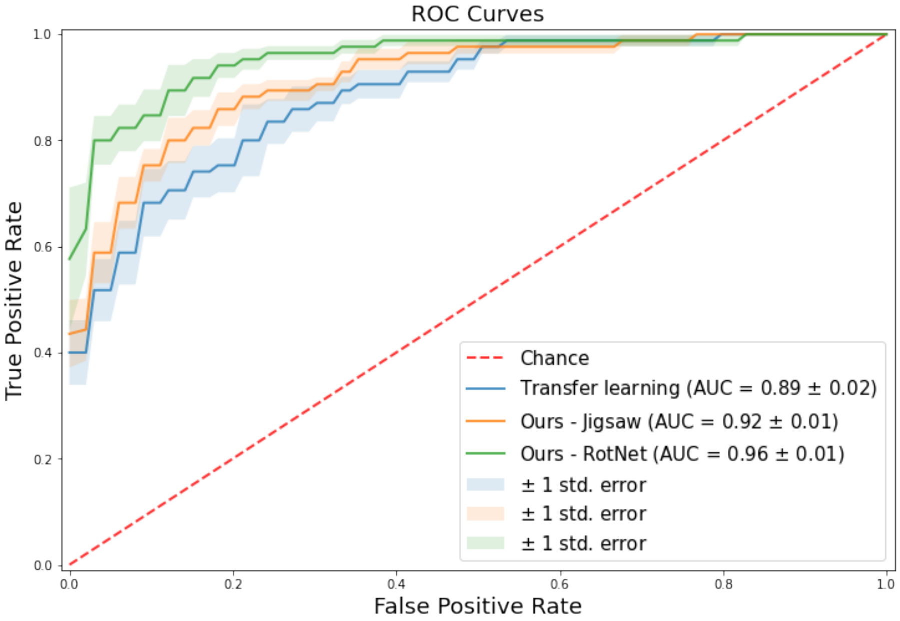


Figure 3: Average ROC curves obtained for AMD grading on STARE, using AREDS as target data and VGG-16 as CNN. Our method improves the AUC of the fine-tuning baseline, particularly when using RotNet as self-supervised task

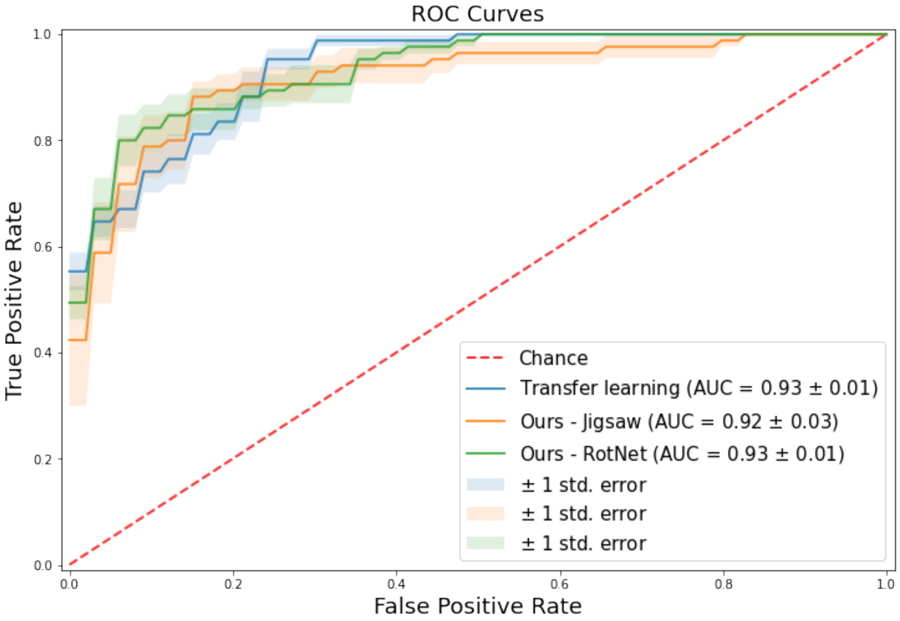


Figure 4: Average ROC curves for AMD grading on STARE, obtained using AREDS as target data and ResNet-18 as CNN. In this case our method achieves AUC comparable to the baseline.

interpretable. We created the saliency maps with VGG-16 as CNN and RotNet as self-supervised task.

We start with a qualitative analysis of saliency maps. For each image considered, we created three saliency maps: two related to the baseline method and one to our method. In particular, saliency maps created for the baseline method were obtained when fine-tuning only the fully connected layers (b) or the entire network (c) and were compared with the ones obtained with our approach (d). In

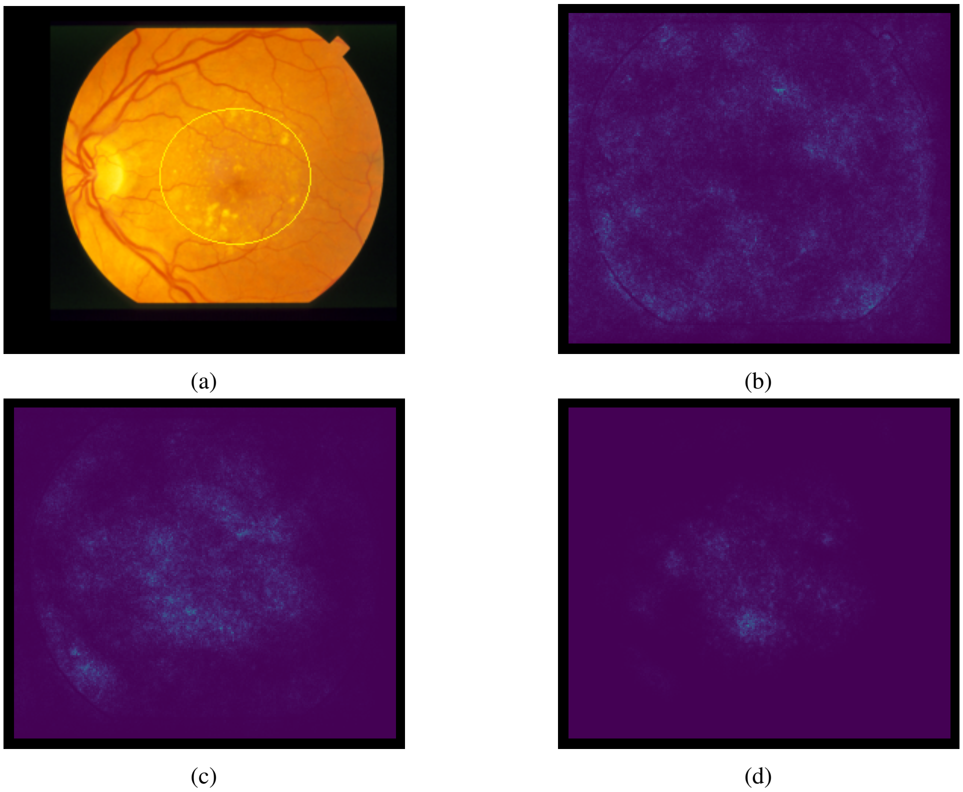


Figure 5: Original early AMD image (a) and saliency maps obtained with transfer learning, fine-tuning only the fully connected layers (b) or all the network (c) and with our approach (d). Our method is able to focus more precisely on drusen spots (circled in (a))

Figure 5 we present saliency maps related to an early AMD image. By observing the image, we may get the idea that saliency maps related to the baseline (b,c) tend to be more spread out and in general do not focus very well on drusen spots. On the other hand, the saliency map created after applying our method (d), gives the impression to be able to capture these areas of interest a bit more precisely. In any case, a quantitative analysis would be needed to have a more precise answer.

To this aim, we created precision-recall curves, averaged over three runs, to compare saliency maps of early AMD images with the ground truth coordinates of drusen. The result is shown in Figure 6. We can observe how the average AUC of the precision-recall curves related to the usual fine-tuning baseline where we fine-tune all the network is 0.44, while the AUC obtained fine-tuning only the fully connected layers of the CNN is 0.19, confirming how fine-tuning also the deeper layers is crucial. Furthermore, as we were presuming during our qualitative analysis, our method achieves the best results, with an average AUC of 0.50. It is also important to note that the result obtained by random chance would not be 0.5 as one may think, but a horizontal line approximately corresponding to precision $= 0.2$ (giving an AUC of around 0.2), as shown in Figure 6. 0.2 is the ratio $\frac{P}{P+N}$ between the positive examples and the total samples in our data (sum of positive and negative examples). This is true in general and can be easily proven in the case of a random classifier C assigning the i-th sample y_i to the positive class with a random probability p. In this case, the expected value for the precision is: $\mathbb{E}(\frac{TP}{TP+FP}) = \frac{pP}{pP+pN} = \frac{P}{P+N}$ by definition of precision, where P is the number of positive samples and N the number of negative ones. Similarly, the expected value for the recall is $\mathbb{E}(\frac{TP}{TP+FN}) = \frac{pP}{pP+(1-p)P} = p$.

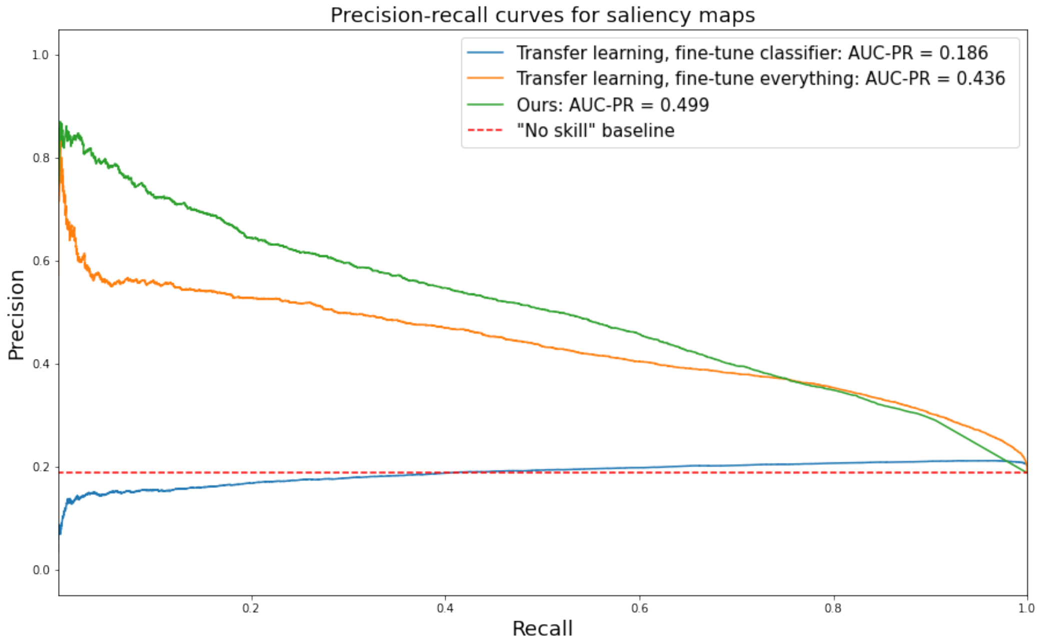


Figure 6: Precision recall curves for the saliency maps of early AMD images. Our method outperforms the fine-tuning baseline in average precision-recall AUC

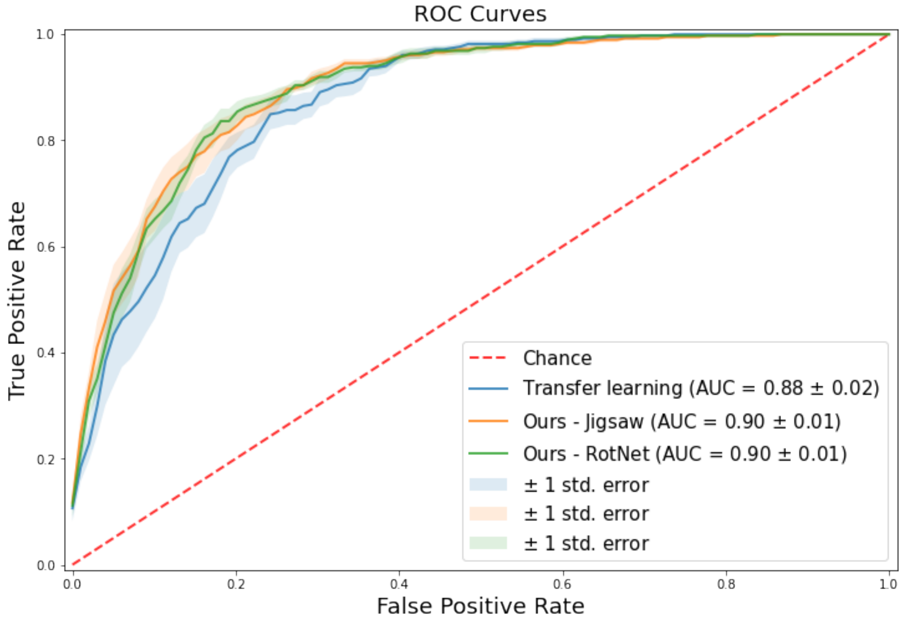


Figure 7: Average ROC curves obtained with IDRID dataset as source, Messidor-2 as target and VGG-16 as CNN. Our method improves the average AUC of the fine-tuning baseline, even if there some overlapping between standard errors intervals

C More results and details about DR grading

In Figures 7 and 8 we display the average ROC curves obtained with VGG-16 and ResNet-18 respectively. In both cases, our approach has better average AUCs than the baseline, even if there is some overlapping between standard errors intervals.

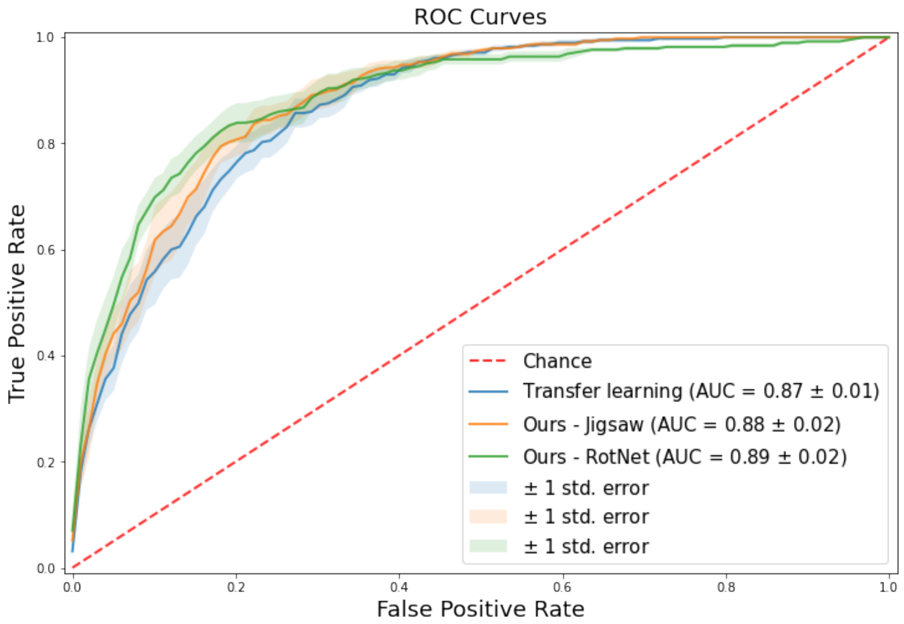


Figure 8: Average ROC curves obtained with IDRID fine-tuning baseline, even if some overlapping between standard errors intervals is present

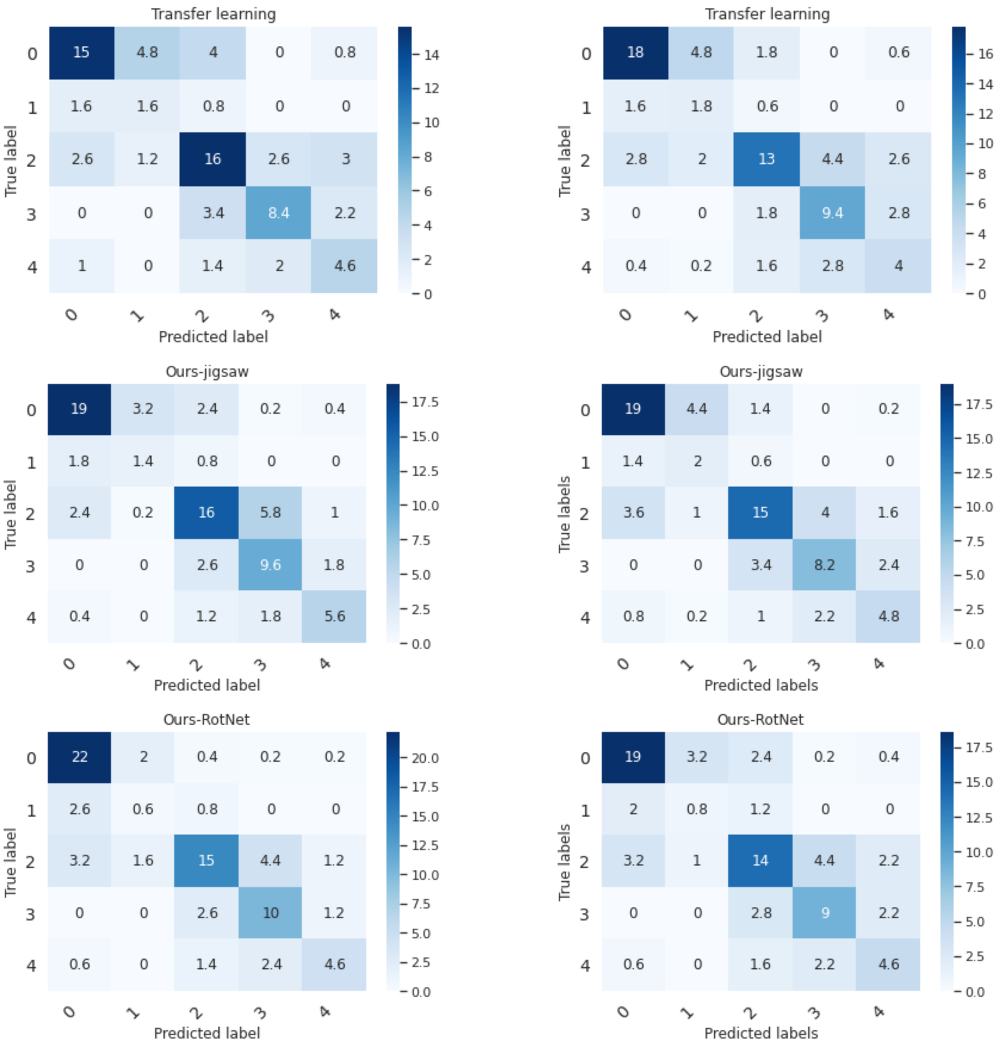
Table 3: Average numbers of images that are predicted to be more than one grade away from the ground truth (lower is better). The proposed method improves the transfer learning baseline from 12.8 to 7.2 and from 10 to 8.8 such mistakes with VGG-16 and ResNet-18 respectively

	VGG-16	ResNet-18
Fine-tuning	12.8	10
Ours - Jigsaw puzzles	8	**8.8**
Ours - RotNet	**7.2**	10.6

If we look at the confusion matrices obtained when predicting the class of images in the test set with the different methods, we can further observe how in our approach most errors happen when misclassifying an image either one grade below or one grade above, while the fine-tuning approach makes bigger mistakes on average, sometimes misclassifying images of several grades. We report the average confusion matrices in Figure 9, using VGG-16 (a) and Resnet-18 (b). The average number of images predicted to be more than one grade away from the ground truth are displayed in Table 3. The biggest difference is observed with VGG-16 as CNN, where the standard fine-tuning approach commits 12.8 such mistakes on average, while our approach 8 and 7.2, with Jigsaw and RotNet respectively as self-supervised task. On the other hand, with ResNet-18 our approach using Jigsaw puzzles as self-supervised task improves the fine-tuning baseline from 10 to 8.8 mistakes of this type. However, with RotNet, it is slightly worse with 10.6 mistakes. As reported in previous occasions, this difference in performance may be attributed to the ability of VGG-16 of learning more general representations from the unlabelled data, being a wider network.

D Ablation Studies

We repeated some experiments both for AMD grading and on Office-31, considering the results obtained with all the combinations of two classifiers that follow the common feature extractor G_f, i.e. G_y and G_d, G_y and G_s, G_d and G_s. For Office-31, we start considering the case of dSLR with 100 training labels as source (with Amazon as target), since as observed this is the case where the biggest difference in performance among different methods is observed and also where the amount of labels available is similar to the case of AMD grading. Later, we will also present the results obtained with the full dSLR dataset as source. The results for AMD grading are presented in Table 4, while the



(a) (b)

Figure 9: Average confusion matrices obtained grading DR with the baseline and our approach, using VGG-16 (a) and Resnet-18 (b) as CNN. On average our method makes less prediction that are wrong by more than one grade

Table 4: Ablation study performed grading AMD images on STARE, using AREDS as target. Our method achieves better performance than each combination of two of its components

	Self-supervised task	VGG-16	ResNet-18
Ours - G_y, G_d, G_s	Jigsaw puzzles	$\mathbf{83.53\% \pm 1.05\%}$	$78.82\% \pm 3.94\%$
	RotNet	$81.18\% \pm 1.97\%$	$\mathbf{81.18\% \pm 4.53\%}$
G_y, G_d	–	$78.83\% \pm 3.57\%$	$76.47\% \pm 2.88\%$
G_y, G_s	Jigsaw puzzles	$82.36\% \pm 4.08\%$	$76.47\% \pm 1.66\%$
	RotNet	$75.30\% \pm 3.87\%$	$77.65\% \pm 3.07\%$
G_d, G_s	Jigsaw puzzles	$45.88\% \pm 8.39\%$	$77.65\% \pm 3.07\%$
	RotNet	$56.47\% \pm 6.36\%$	$78.83\% \pm 3.94\%$

first experiments on Office-31 in Table 5. As we can see from the tables, in both cases our method including all the three components achieves the best results, validating our idea that all of them are needed, at least in the low data regime.

Table 5: Ablation study performed on Office-31, using dSLR as source with 100 training labels available and Amazon as target. Our method is able to obtain better classification accuracy than each combination of two of its components

		VGG-16	ResNet-18
Ours - G_y, G_d, G_s	Jigsaw puzzles	$79.34\% \pm 1.13\%$	$\mathbf{78.03\% \pm 1.16\%}$
	RotNet	$\mathbf{80.92\% \pm 0.47\%}$	$77.30\% \pm 1.34\%$
G_y, G_d	–	$78.12\% \pm 0.55\%$	$75.99\% \pm 1.13\%$
G_y, G_s	Jigsaw puzzles	$80.79\% \pm 0.55\%$	$76.51\% \pm 1.22\%$
	RotNet	$80.79\% \pm 0.80\%$	$76.10\% \pm 0.97\%$
G_d, G_s	Jigsaw puzzles	$78.29\% \pm 0.73\%$	$76.38\% \pm 0.96\%$
	RotNet	$78.82\% \pm 0.64\%$	$76.05\% \pm 1.29\%$

Taking a closer look at the results, we can observe that with VGG-16 the second best performance is achieved by the combination of G_y and G_s for both datasets. On the other hand, with ResNet-18, the second best result is achieved by the same combination of classifiers on Office-31 images and by G_d and G_s on the AMD dataset. Interestingly, G_d and G_s also achieve the worst performance - by a large margin - with VGG-16 on the AMD dataset, from which we may assume that the classification of source images - done by G_y - is necessary to maintain stable performance across all settings. To sum up, while the best results are achieved with our method, we can suppose that the two most important components are G_y and G_s, that alone achieve the second best results in three out of four cases. G_d and G_s achieve the best performance in the remaining case - on the AMD dataset with ResNet-18 - but also show unconsistent results, obtaining very poor results with VGG-16 on AMD grading. All these considerations come with the caveat that in some cases there is some overlapping among standard errors intervals, due to the small size of the datasets available.

Table 6: Ablation study performed on Office-31, using dSLR as source with 70% of the full dataset in the training set and Amazon as target. Our method still achieve the best average performances but with ties or small margin and overlapping between standard errors intervals

		VGG-16	ResNet-18
Ours - G_y, G_d, G_s	Jigsaw puzzles	$95.73\% \pm 1.16\%$	$94.93\% \pm 1.27\%$
	RotNet	$\mathbf{96.00\% \pm 1.13\%}$	$\mathbf{95.73\% \pm 1.38\%}$
G_y, G_d	–	$\mathbf{96.00\% \pm 0.84\%}$	$94.40\% \pm 1.57\%$
G_y, G_s	Jigsaw puzzles	$94.93\% \pm 1.16\%$	$94.93\% \pm 1.27\%$
	RotNet	$95.73\% \pm 0.88\%$	$94.93\% \pm 1.03\%$
G_d, G_s	Jigsaw puzzles	$95.47\% \pm 0.16\%$	$95.2\% \pm 1.11\%$
	RotNet	$94.67\% \pm 1.00\%$	$\mathbf{95.73\% \pm 0.88\%}$

In Table 6 we display the results obtained using the full dSLR dataset as source, with 70% of the data in the training set, and again Amazon as target. In this case, our approach is still able to achieve the best average accuracy, but the results are less clear since there are ties with results achieved with some combination of two components of the architecture and in general all the results appear very close. At the same, the combinations of two components tying the best performance achieved by the full architecture are different for different CNNs and thus employing all the components of our method would still seem the best choice. We can conclude that while our method is able to outperforms standard fine-tuning on labelled samples for natural images also with more labels available, the biggest improvements are displayed in the low data regime. Moreover, it is less clear whether our approach is better than any combination of two of its components when more training data is available. This could be explained considering that with more training data, a model is able to generalise better and the benefit achievable by transferring features from external datasets is less signficant.