

Comparison of deep transfer learning strategies for digital pathology

Romain Mormont Pierre Geurts Raphaël Marée
University of Liège, Belgium

{r.mormont, p.geurts, raphael.maree}@uliege.be

Abstract

In this paper, we study deep transfer learning as a way of overcoming object recognition challenges encountered in the field of digital pathology. Through several experiments, we investigate various uses of pre-trained neural network architectures and different combination schemes with random forests for feature selection. Our experiments on eight classification datasets show that densely connected and residual networks consistently yield best performances across strategies. It also appears that network fine-tuning and using inner layers features are the best performing strategies, with the former yielding slightly superior results.

1. Introduction

In pathology, tissues were traditionally examined under an optical microscope after being sectioned, stained and mounted on a glass slide. During the last years, progress in scanning technologies made possible the high-throughput digitization of glass slides at high resolution. Digital pathology holds promise for biomedical research and clinical routine but raises great challenges for computer vision research [28]. First and foremost, a laboratory can produce and scan large amounts of slides per day, each of them being a multi-gigapixel image. Moreover, those slides can contain many different kinds of tissues with different staining techniques. Their quantity, variability and size therefore require efficient and versatile computer vision methods. The second challenge is the scarcity of annotated data. Indeed, annotations of digitized slides require expertise and are therefore expensive and tedious to obtain.

In parallel, deep learning has recently had an impressive impact over the computer vision field starting with work of Krizhevsky *et al.* [22], which improved previous natural image recognition performances by a large margin. More recently, researchers have been working at applying those new techniques to biomedical imaging [12, 23]. While current results are promising, improvements were not as impressive as they had been for traditional computer vision

tasks (recognition of natural scenes or objects such as ImageNet [7], face recognition...) which can likely be attributed to the lack of annotated data.

Interestingly, works such as [8, 40, 33] have shown that convolutional neural networks can be used efficiently for transfer learning, *i.e.* a network can be trained on a source task and then be reused on a target task. This technique is particularly handy when the data for the target task is scarce and one has a large dataset that can be used for training as source task. Therefore, it is not surprising that deep transfer learning has been studied by the biomedical imaging community to overcome the data scarcity problem. In their review [23], Litjens *et al.* identify two transfer learning strategies for image classification: using off-the-shelf features extracted from pre-trained networks or fine-tuning these latter networks for the task at hand. The first strategy uses features learned from the source task without re-training the network and use them to train a third-party classifier. Extracting such features is usually fast (with ad-hoc computing resources, *i.e.* graphical processing units) and this approach requires only to tune the hyper-parameters of the final classifier, which can usually be done efficiently through cross-validation. Those properties make off-the-shelf features particularly appealing for biomedical imaging given the aforementioned challenges. The second strategy consists in using a network initialized with pre-trained weights and partially re-training it on the target task. This approach is more computationally demanding than using off-the-shelf features and involves dealing with more hyper-parameters. In terms of performance, as stated in [23], both strategies have not been compared thoroughly yet in biomedical imaging and there is no consensus about whether one is better than the other.

In this work, we study thoroughly and compare several strategies that involve off-the-shelf features and network fine-tuning. We carry out several experiments over eight object classification datasets in digital histology and cytology. Different combinations of state-of-the-art networks and feature selection techniques using random forests are proposed in order to answer questions of very high practical relevance: which network provides best-performing features? How

should those features be extracted and then exploited to get the best performance? Is fine-tuning better than using off-the-shelf features? More generally, our empirical study will also contribute to confirm the interest of deep transfer learning for tackling the recurrent data scarcity problem in biomedical imaging.

2. Related work

Transfer learning has been studied for a long time [29], but its applicability in deep learning has only been discovered recently. Decaf [8] and Overfeat [32, 33] are the first published off-the-shelf feature extractor networks. The former is based on AlexNet [22], the latter is a custom architecture. Both were pre-trained on ImageNet and provide generic features for computer vision tasks. Those features can then be used by other methods such as SVM [10] or random forests [4] for final classification. Following those advances, works such as [40, 41] shed more light on the transfer learning process by empirically analyzing the extracted features for various tasks. One interesting conclusion of these works is that shallow features seem to be generic whereas deep ones are more specific to the source task.

Some of the first applications of deep transfer learning in biomedical imaging were reported in [2, 6, 39] for pulmonary nodule detection in chest x-rays and CT-scans using Decaf and OverFeat. While those works have revealed the potential of deep transfer learning in that field, the performances were not significantly better than those of previous methods. More recently, fine-tuning has been also investigated and compared with networks trained from scratch and classifiers trained on off-the-shelf features on a wide variety of biomedical imaging tasks [1, 9, 13, 31, 34, 38]. More specifically, digital pathology and microscopy are not outdone with works on tissue texture [19], cell nuclei [3] and breast cancer [15] classification from histopathology images or analysis of high-content microscopy data [21]. Most works have used networks pre-trained on the ImageNet image classification dataset. Others, like [21], have used a custom network architecture pre-trained on a medical dataset as source task.

While it is clear that training from scratch very deep networks is not viable in most cases due to data scarcity [3, 38], there is currently no consensus and best practice about how transfer learning should be applied to digital pathology and microscopy. Some recent publications in the biomedical field have shown that fine-tuning outperforms off-the-shelf features [1, 19, 31, 34]. However, as noted in [23], experiments in these papers are often carried out on a single dataset, which does not allow to draw general conclusions. Moreover, those experiments do not use current state-of-the-art networks. A short review of the networks used in biomedical imaging is given in Supplementary Table 1 and shows that the more recent and efficient resid-

ual and densely connected networks have been underused or not used at all, in particular in digital pathology and microscopy.

3. Datasets

Our experimental study uses datasets collected over the years by biomedical researchers and pathologists using the Cytomine [27] web application. Using this platform, eight image classification datasets were collected which are summarized in Table 1. These contain tissues and cells from human or animal organs (thyroid, kidney, breast, lung, ...).

For all datasets except Breast, each sample image is the crop of an annotated object extracted from a whole-slide image. The Breast dataset is composed of patches for which the label encodes the type of tissue in which the central pixel is located. Selected image samples for each dataset are shown in Figure 1.

4. Methods

4.1. Deep networks

We follow the feature extraction and classification process presented in Figure 2, which starts from a deep convolutional neural network \mathcal{N} pre-trained on a source task S . In particular, we use ImageNet as the source dataset S and, for each network, the pre-trained weights were retrieved from Keras [5]. For \mathcal{N} , we evaluate several architectures that have been or are state of the art on the ImageNet classification dataset [7] or that present interesting trade-off between computational requirements and performances: VGG16, VGG19 [35], InceptionV3 [37], ResNet50 [16], Inception-ResNetV2 [36], DenseNet201 [18], and MobileNet [17]. In the sequel, those networks will be respectively referred to as VGG16, VGG19, IncV3, ResNet, IncResV2, DenseNet, and Mobile. These networks can be used off-the-shelf or fine-tuned, as explained below.

4.2. Off-the-shelf feature extraction

Images are first resized to match the input dimension of the network. Respectively denoting by $h_{\mathcal{I}} \times w_{\mathcal{I}} \times c$ the height, width, and number of channels of the input image \mathcal{I} , we extract a square patch \mathbf{p} of (maximum) height and width $\min(h_{\mathcal{I}}, w_{\mathcal{I}})$ in the center of the image, which is then resized to the network input size $h_p \times w_p \times c$. This extraction process is parameter-free and preserves the aspect ratio of the image, since all pre-trained networks take square images as inputs (*i.e.* $h_p = w_p$).

The resized patch is then forwarded through \mathcal{N} (loaded with pre-trained weights). Given this input, the output \mathbf{a} of an arbitrary layer l (*i.e.* a set of feature maps) of dimensions $h_a \times w_a \times d$ is extracted where d is the number of feature maps and h_a and w_a are respectively their height and width. Because this tensor can be high-dimensional, one usually

Dataset	Domain	Cls	Train		Validation		Test		Total	
			Images	Slides	Images	Slides	Images	Slides	Images	Slides
Necrosis (N)	Histo	2	695	9	96	1	91	3	882	13
ProliferativePattern (P)	Cyto	2	1179	19	167	4	511	13	1857	36
CellInclusion (C)	Cyto	2	1644	21	173	2	1821	22	3638	45
MouseLba (M)	Cyto	8	1722	9	716	4	1846	7	4284	20
HumanLba (H)	Cyto	9	4051	50	346	5	1023	9	5420	64
Lung (L)	Histo	10	4881	669	562	73	888	140	6331	882
Breast (B)	Histo	2	14055	22	4206	8	4771	4	23032	34
Glomeruli (G) [25]	Histo	2	12157	91	2448	12	14608	102	29213	205

Table 1. Sizes and splits of the datasets.

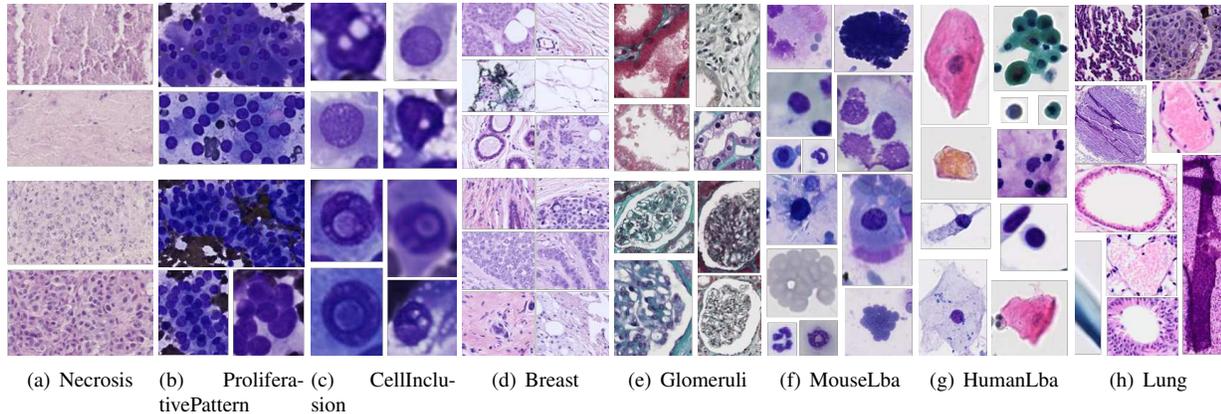


Figure 1. Overview of our eight classification datasets (the display size does not reflect actual image size). For binary classification datasets, negative and positive samples were respectively placed at the top and bottom of the figures.

applies a dimensionality reduction procedure \mathcal{R} (e.g. global average pooling, principal component analysis,...) to reduce it to k features, yielding a feature vector $\mathbf{f} \in \mathbb{R}^k$. Here, we limit our analysis to global average pooling (i.e. feature maps averaging), which, unlike principal component analysis for example, has the advantage of being parameter-free.

4.3. Fine-tuned feature extraction

For our experiments on network fine-tuning, we replace the final fully-connected layer by a fully-connected layer with as many neurons as there are classes in the current dataset. We start by freezing the network and training only the newly appended layer for 5 epochs with a learning rate of 10^{-2} . Then, we train the whole network for 45 epochs with a learning rate of 10^{-5} . We use Adam [20] as optimizer with parameters β_1 and β_2 respectively set to 0.99 and 0.999 and no weight decay as suggested in the original paper. We use the categorical cross-entropy as the loss function. The fine-tuning is done using the training set exclusively. Optionally, we use data augmentation to virtually increase the size of the training set. First, a maximum square crop is taken at a random position in the input image. Then, random flips and rotations are applied to the resized patches before they are forwarded through the network. The model is evaluated on the validation set and then saved at the

end of each epoch. When the fine-tuning is over, we select among the saved models the one that performed the best on the validation set.

4.4. Final classifier learning

When the features have been extracted for all images of a dataset (either using off-the-shelf or fine-tuned networks), they can be used for training the classifier \mathcal{C} . We use either linear support vector machines (SVM) [10] (with a one-vs-all scheme for multiclass problems), extremely randomized trees (ET) [11], or a fully-connected single layer perceptron (FC), all as implemented in `scikit-learn` [30]. SVM is the most popular method in the literature for off-the-shelf features. ET are incorporated mainly for their ability to compute feature importance scores. FC is a natural choice to mimic how the pre-trained network exploits the features. Hyper-parameters of all methods were tuned by slide-wise (or patient-wise) n -fold cross-validation on the merged training and validation sets. Namely, they are the penalty C for SVM, the maximum number of features for ET, and the learning rate and number of iterations for the single layer perceptron. Selected values for tuning are given in Supplementary Section B.

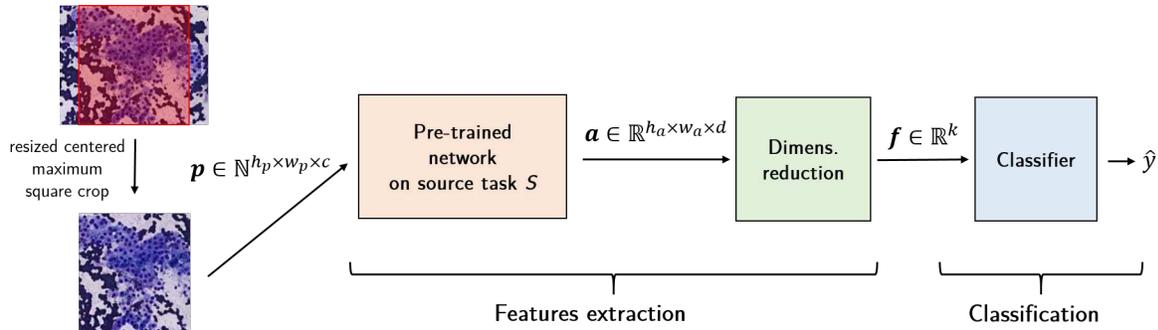


Figure 2. Feature extraction from pre-trained convolutional neural networks

4.5. Prediction

For prediction, the process is similar: patches are extracted from target images, forwarded through \mathcal{N} , compressed with \mathcal{R} and classified with the learned classifier \mathcal{C} . As for the fine-tuning, we make additional experiments using directly the fine-tuned network for classifying the images (see Section 4.3 for parameters).

5. Experiments

In this section, we propose and thoroughly compare different strategies for extracting and using features from the deep networks introduced previously. We follow a rigorous evaluation protocol described in Section 5.1. Strategies are presented and evaluated one after the other in Sections 5.2 to 5.7, then an overall comparison of strategies is discussed in Section 5.8.

5.1. Performance metrics and baseline

For performance evaluation, each dataset (presented in Section 3) is randomly splitted into training, validation and test sets. Following the guidelines in [24], image patches from the same slide are all put in the same set to avoid any positive bias (induced by overfitting the slide preparation and acquisition process) in the evaluation.

To evaluate the different transfer strategies, we use two different metrics: the area under the receiving operating curve (ROC AUC) for binary classification problems and the classification accuracy for the multiclass problems. The former has the advantage of not being affected by class imbalance and it also does not require to select an operating point. For the sake of readability, a summary of the scores for all experiments and datasets is given in Table 2 while the detailed scores are only given in Supplementary Section G. In all figures, we plot instead for each method its rank among all methods compared in the same graph averaged over all eight datasets. To associate high rank with best results, we compute the rank when methods are sorted in reverse order of performance (AUC or accuracy). For example, since ten methods are compared in Figure 7, the

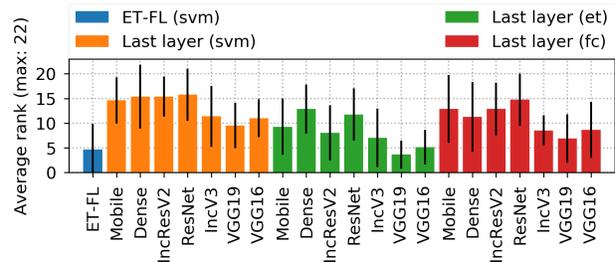


Figure 3. Average ranks of the methods for the “Last layer features” experiment. Colors encode the choice of classifier for \mathcal{C} (orange for SVM, green for extremely randomized trees and red for single layer perceptron).

maximum average rank is 10, corresponding to a method being the best one on all eight datasets, and the minimum average rank is 1, corresponding to a method always worse than all others.

In order to have a baseline for comparison, we have chosen a previously published tree-based image classification algorithm [26] because it is a fast and generic method which requires few parameters tuning. We use the algorithm variant called ET-FL, which fits an SVM model on features extracted from an ensemble of extremely randomized trees trained on random image patches. Parameters for this method are either fixed to default values or tuned by cross-validation (see in Supplementary Section B for default values and ranges).

5.2. Last layer features

Our first strategy follows a common approach in biomedical imaging where off-the-shelf features are extracted at the last layer of a pre-trained network. In our case, we take the features from the last feature maps before the first fully connected layer (the numbers of extracted features per network are given in Supplementary Table 4). For each dataset and network, we then tune and train the three types of classifiers \mathcal{C} (ET, SVM, and FC) with the extracted features, on the union of the training and validation sets, and then we

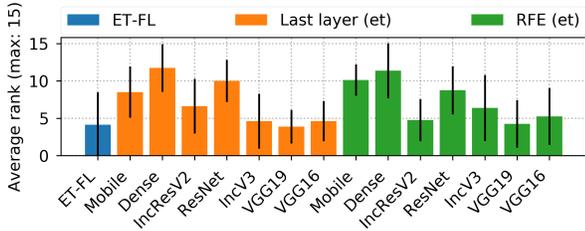


Figure 4. Average ranks for last layers' features classified with extremely randomized trees before (orange) and after (green) selection with recursive feature elimination.

evaluate them on the test set. The resulting average ranks for all classifiers and all datasets are given in Figure 3.

We observe that SVM and single layer perceptron are more efficient at classifying from deep features than extremely randomized trees, with a slight advantage to SVM. Mobile, DenseNet, IncResV2 and ResNet yield best performances when combined with SVM or single layer perceptron, while for extremely randomized trees, only DenseNet and ResNet are leading the way. Last layer features from VGG16, VGG19 and IncV3 allow most of the time to beat the baseline but they are clearly not competitive with features from the other networks whatever the classifier. Overall, the best performance is obtained by combining ResNet features with SVM.

5.3. Last layer feature subset selection

Our second strategy aims at checking whether all features extracted from the last layer of the network are important for the final classification or if a small subset of them would be sufficient to obtain optimal performance. To answer this question, we use cross-validated recursive feature elimination (RFE) [14] using importance scores derived from extremely randomized trees to rank the features (where the importance of a given feature is computed by adding up the weighted impurity decreases for all tree nodes where the feature is used, averaged over all trees). This method outputs a sequence $\{(S_t, s_t) | t = 1, \dots, T\}$ (built in reverse), where S_t are nested feature subsets of increasing sizes (with $|S_1| = 1$ and $|S_T| = k$) and s_t is the cross-validation performance of an ET model trained on S_t . From this sequence, we compute for each dataset:

$$k_{min} = \min_{\{t=1, \dots, T: s_t \geq \max_{t'} s_{t'} - l_a\}} |S_t|,$$

where l_a is a small performance tolerance (set to 0.005 in our experiment). k_{min} is thus the minimum number of features needed to reach a performance not smaller than the optimal one by more than l_a .

On average across datasets and networks, this method selected 7.5 % of the features (detailed numbers are given in Supplementary Tables 8 and 9). The models re-trained using the selected features yielded comparable performance

when using ET as classifier (see Figure 4). Feature selection even improved ET performance on the IncV3, VGG19, and VGG16 networks. Using SVM on the selected features however leads to a performance drop compared to SVM with all features. We believe that this difference is due to the fact that the selection is optimized for ET and it is thus likely to remove features that are useful for linear SVM and not for ET, which are non-linear approximators.

Our experiments show that, among the available features from the last layer, most of them are uninformative or redundant and therefore only few of them are actually needed for the prediction. This conclusion can also be drawn by observing the recursive feature elimination cross-validation curves (see Supplementary Figure 2). For all datasets and networks, the accuracy converges very abruptly to a plateau when the number of selected features increases, which indicates that removing features from the learning set does not impact negatively the predictive power of the models.

It is also interesting to note that the selected features are not the same across datasets. For instance with DenseNet, for a feature to appear in the subset of best features (determined with the importances obtained during the “*Last layer features*” experiment) for all datasets, we need to consider a subset of size 1477 (*i.e.* 77% of the features). We observe similar results for the other networks (see Supplementary Table 3). On the other hand, there can be a significant overlap between features selected by RFE for specific pairs of datasets (see Supplementary Table 10). These results suggest that the best features are task-dependent and that there would be no interest in restricting a priori the subset of transferred features, even when focusing on the domain of digital pathology.

5.4. Merging features across networks

The third strategy consists in merging features from the last layer of all the studied networks. Aggregating all features results in a feature vector of size 9600. We observe in Table 2 and Figure 9 that despite the fact that features from all networks are combined, this strategy gives performance results similar to but not better than using the best single network, both with ET and SVM.

Using forest importance ranking procedure described previously, we further analyze the information brought by the last layer of each network (feature importances averaged across datasets are given in Figure 5). We observe that the features of DenseNet bring more than 25% of information on average while they only account for 20% of all the features. Moreover, the proportion of information brought by the most informative features of DenseNet is higher than for any other network. Following DenseNet, the next most informative networks are IncResV2, ResNet, IncV3 and finally the Mobile, VGG16 and VGG19 networks. Surprisingly, the importances brought by the VGG networks

Strategy	Datasets							
	C	P	G	N	B	M	L	H
Baseline (ET-FL)	0.9250	0.8268	0.9551	0.9805	0.9345	0.7568	0.8547	0.6960
Last layer	0.9822	0.8893	0.9938	0.9982	0.9603	0.7996	0.9133	0.7820
Feat. select.	0.9676	0.8861	0.9843	0.9994	0.9597	0.7438	0.8941	0.7703
Merg. networks	0.9897	0.8984	0.9948	0.9864	0.9549	0.8169	0.9155	0.7928
Merg. layers	0.9808	0.8906	0.9944	0.9964	0.9639	0.7941	0.9268	0.7977
Inner ResNet	0.9748	0.8959	0.9949	0.9964	0.9664	0.8131	0.9291	0.8113
Inner DenseNet	0.9862	0.8984	0.9962	0.9917	0.9699	0.8012	0.9268	0.7967
Inner IncResV2	0.9873	0.8948	0.9962	0.9982	0.9720	0.8137	0.9234	0.7713
Fine-tuning	0.9926	0.8797	0.9977	0.9970	0.9873	0.8727	0.9405	0.8641
Metric	Roc AUC				Accuracy (multi-class)			

Table 2. Best score for each strategy and each dataset. The best and second best scores are respectively highlighted in green and orange.

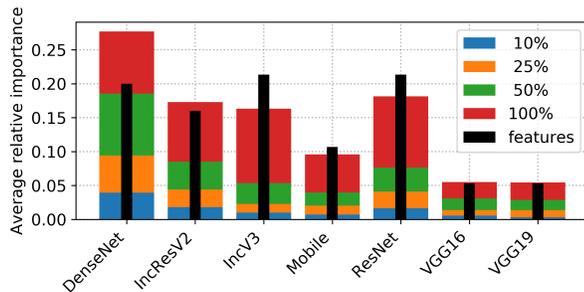


Figure 5. Average relative importances (across datasets) brought by each studied network when all their last layer features are aggregated. The black bars quantify the proportion of features of each network. The colors indicate the information brought by features of decreasing importances: blue and red features are respectively the most informative and least informative ones. Blue, orange, green and red bars regroup importances of features that respectively and cumulatively bring 10%, 25%, 50% and 100% of the information for predicting the outcome.

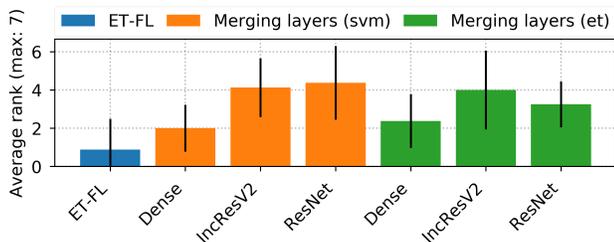


Figure 6. Average ranks for models learned on merged layers of networks compared to the baseline.

relatively to the number of features is non-negligible and higher than the one of ResNet and IncV3. This may indicate that features of those networks are redundant with features of DenseNet and IncResV2 while features of VGG16 and VGG19 are not.

5.5. Merging features across layers

This strategy aims at merging features across layers (at several depths) for a given network. Given the results in

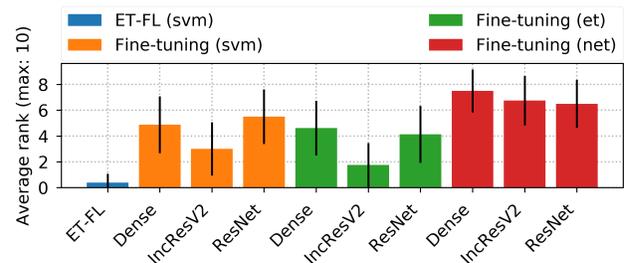


Figure 7. Average ranks for fine-tuned networks compared to the baseline. Evaluation was done either by using SVM (orange) or extremely randomized trees (green) on the fine-tuned features or by predicting the outcome using the fine-tuned fully-connected layer directly (red).

Section 5.2, we limit our analysis to the IncResV2, ResNet and DenseNet networks. Those three networks have complex structures and many layers which yield plenty of possible cut points for feature extraction. To reduce the number of possibilities, we limit the extraction to bottlenecks of the networks. Details about the layer selection are given in Supplementary Section D.

The average ranks for each layer of all studied networks are given in Figure 6. One first observation is that there is no significant difference between SVM and ET in terms of performance, unlike when we use the last layer only. Surprisingly, DenseNet is not performing well with respect to the other network while it was competitive with using the last layer only. Merging the layers actually leads to a drop of performance with respect to using only the last layer for DenseNet, while it leads to a small improvement for the other two networks (see Section 5.8).

As in the previous section, we use feature importances to identify most informative features. Detailed importances plots for each network can be found in Supplementary Figure 1. These plots clearly show that the most informative features are spread over all layers. We also observe that the relative importance of features in the early layers is higher than the ones in deeper layers. One possible reason is that

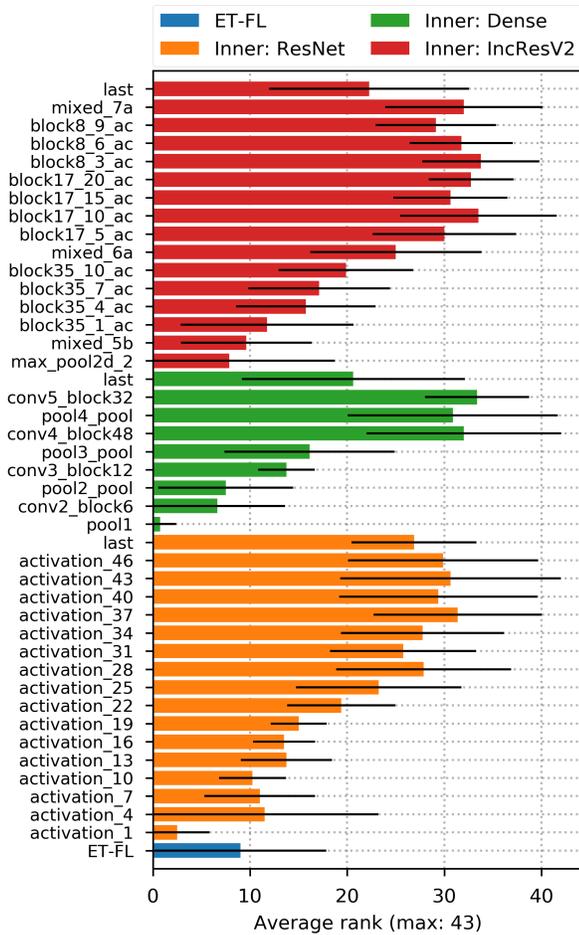


Figure 8. Average ranks for the baseline and the models trained using features from layers inside the networks. For each network, the layers are sorted by decreasing depth (from top to bottom).

last layers actually have more features than earlier ones and as shown in Section 5.3, most of those features are either irrelevant or redundant. Therefore, the extremely randomized trees actually discard most of them during training.

Merging features from several layers results in large feature vectors for describing the images. Those large vectors make this method less attractive as it results in longer classifier training time. This is especially true for extremely randomized trees. Unlike in the previous strategy however, feature extraction does not increase computation requirements as only one forward pass through the network is needed to extract all the features.

5.6. Inner layers features

For this strategy, we assess features extracted from each layer separately. The motivation is to determine if there is a layer that, taken alone, yields better performance than the others, and in particular, the last one. Using the same cut

points as in the previous section to define the layers, we learn as many SVM classifiers as there are layers, each using the features of a single layer.

Average ranks for each inner layer and each network are given in Figure 8. In all cases, the last layer features are always outperformed by features taken from an inner layer of the network. The optimal layer is however always located rather at the end of the network, while the first layers are clearly never competitive. Unfortunately, we have not found that a specific layer was better for all datasets, so in practice the choice of the layer should be determined by internal cross-validation as we did. Interestingly, the baseline either outperforms the early layers of the networks or yield comparable results which tends to indicate that the features provided by ET-FL are somewhat low-level.

5.7. Fine-tuned features

All previous experiments explored strategies using off-the-shelf features. In this last strategy, we investigate fine-tuning as described in Section 4.3. We focus on the same three networks as in the previous sections (ResNet, IncResV2 and DenseNet).

The average ranks for the different fine-tuning methods are given in Figure 7. With the three networks, best performances are obtained by making predictions directly from the fine-tuned fully connected layer. SVM and ET trained on the features extracted from the fine-tuned networks are clearly inferior, in particular with IncResV2. Note that, for the other two, last layer features extracted from the fine-tuned network are nevertheless better than last layer features from the original network, when used as inputs to SVM (see Figure 9). Fine-tuning is thus globally improving the quality of the features for these two networks. Overall, the best performance is obtained with fine-tuned DenseNet.

5.8. Discussion

To allow comparison of all strategies, the best scores per strategy and dataset are summarized in Table 2 and the average ranks of all methods evaluated in the previous experiments are given in Figure 9.

Concerning the networks, ResNet and DenseNet often yield the best performing models whatever the way they are exploited. They are followed by the IncResV2, Mobile, and IncV3 networks. Performances obtained with the VGG networks are below those of the others.

Concerning the methods, fine-tuning (and predicting with the network) usually outperforms all other methods whatever the network. Especially, this strategy yields significant improvements for the multi-class datasets. For binary datasets, the improvement is often not as impressive, but on three of these datasets, the performances of all methods are already very high (greater than 0.9).

Moreover, for each dataset, there is at least one inner

layer that yields the best or second best scores and this best layer is never the last one. This is confirmed by the rank plot that shows that the ranks of the models learned on last layer features are below those using inner layer features. This might be explained by the fact that last layer features are too specific to the source task (natural images).

Merging features across networks and layers yield results similar to using last layer features but they are outperformed by the best inner layers and also by fine-tuning. The inferior performances of these methods could be attributed to the fact that important features are lost among many redundant and uninformative ones and they are thus suffering from overfitting. One way to improve these methods could be to perform feature selection. However, given that these methods are already more computationally demanding, they are definitely less interesting than fine-tuning and selecting the best inner layer. In particular, merging features across networks requires a forward-pass through all the selected networks. Although only one pass is needed when merging the layers, it still yields a large feature vector which makes further training and tuning slower, especially for ET.

Throughout the experiments, we have also gained more insights about the extracted features. By performing feature selection, we have discovered that very few features are actually useful for learning efficient classifiers on our datasets and the best features are task-dependent.

6. Conclusion

In this work, we have empirically investigated various deep transfer learning strategies for recognition in digital pathology and microscopy. We have observed that residual and densely connected networks often yielded best performances across the various experiments and datasets. We have also observed that fine-tuning outperformed features from the last layer of off-the-shelf networks. It also appeared that using one network’s inner layer features yielded performances slightly superior to using those of the last layer and inferior to fine-tuning but with the advantage of not having to re-train the network. We hope our thorough study and our results will help practitioners to devise best practices for efficient usage of existing deep networks.

In the future, we want to study further deep transfer learning (e.g. combining inner layers and fine-tuning strategies). We also aim at collecting and merging larger annotated biomedical datasets to train networks using a large-scale source dataset closer to our target tasks. We also plan to integrate these strategies into the Cytomine [27] web application to ease their application on novel datasets.

Acknowledgments

R. Marée is supported by grant “IDEES” (Wallonia / ERDF). Computing resources have been provided by Prof.

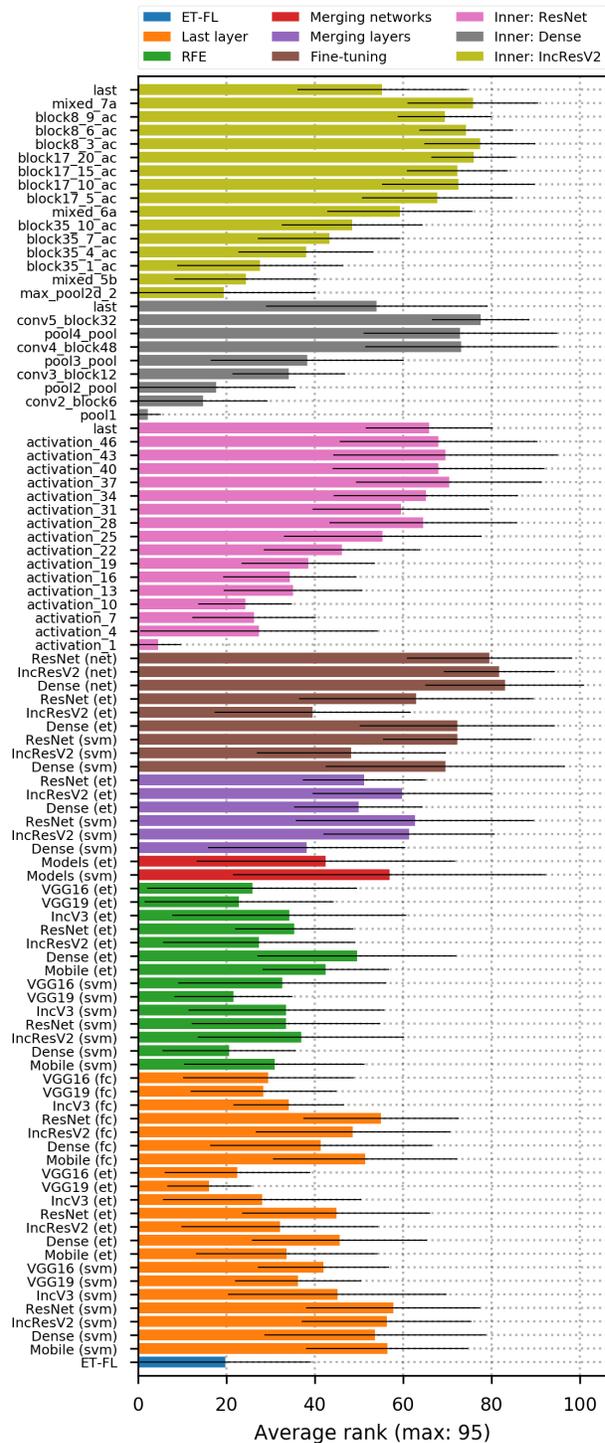


Figure 9. Average ranks for all the evaluated methods.

Marc Van Droogenbroeck and by the GIGA Institute. This work is partially supported by the Belspo project INSIGHT (BRAIN-be). We thank our collaborators for providing images and annotations (see Supplementary Section H).

References

- [1] J. Antony, K. McGuinness, N. E. O'Connor, and K. Moran. Quantifying radiographic knee osteoarthritis severity using deep convolutional neural networks. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 1195–1200. IEEE, 2016.
- [2] Y. Bar, I. Diamant, L. Wolf, S. Lieberman, E. Konen, and H. Greenspan. Chest pathology detection using deep learning with non-medical training. In *Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on*, pages 294–297. IEEE, 2015.
- [3] N. Bayramoglu and J. Heikkilä. Transfer learning for cell nuclei classification in histopathology images. In *European Conference on Computer Vision*, pages 532–539. Springer, 2016.
- [4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
- [5] F. Chollet et al. Keras. <https://github.com/keras-team/keras>, 2015.
- [6] F. Ciompi, B. de Hoop, S. J. van Riel, K. Chung, E. T. Scholten, M. Oudkerk, P. A. de Jong, M. Prokop, and B. van Ginneken. Automatic classification of pulmonary peri-fissural nodules in computed tomography using an ensemble of 2d views and a convolutional neural network out-of-the-box. *Medical image analysis*, 26(1):195–202, 2015.
- [7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.
- [8] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.
- [9] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542(7639):115, 2017.
- [10] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.
- [11] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.
- [12] H. Greenspan, B. van Ginneken, and R. M. Summers. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging*, 35(5):1153–1159, 2016.
- [13] V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, D. Wu, A. Narayanaswamy, S. Venugopalan, K. Widner, T. Madams, J. Cuadros, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama*, 316(22):2402–2410, 2016.
- [14] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.
- [15] Z. Han, B. Wei, Y. Zheng, Y. Yin, K. Li, and S. Li. Breast cancer multi-classification from histopathological images with structured deep learning model. *Scientific reports*, 7(1):4172, 2017.
- [16] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [18] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, page 3, 2017.
- [19] B. Kieffer, M. Babaie, S. Kalra, and H. Tizhoosh. Convolutional neural networks for histopathology image classification: Training vs. using pre-trained networks. *arXiv preprint arXiv:1710.05726*, 2017.
- [20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [21] O. Z. Kraus, B. T. Grys, J. Ba, Y. Chong, B. J. Frey, C. Boone, and B. J. Andrews. Automated analysis of high-content microscopy data with deep learning. *Molecular systems biology*, 13(4):924, 2017.
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [23] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.
- [24] R. Marée. The need for careful data collection for pattern recognition in digital pathology. *Journal of pathology informatics*, 8, 2017.
- [25] R. Marée, S. Dallongeville, J.-C. Olivo-Marin, and V. Meas-Yedid. An approach for detection of glomeruli in multisite digital pathology. In *Biomedical Imaging (ISBI), 2016 IEEE 13th International Symposium on*, pages 1033–1036. IEEE, 2016.
- [26] R. Marée, P. Geurts, and L. Wehenkel. Towards generic image classification using tree-based learning: An extensive empirical study. *Pattern Recognition Letters*, 74:17–23, 2016.
- [27] R. Marée, L. Rollus, B. Stévens, R. Hoyoux, G. Louppe, R. Vandaele, J.-M. Begon, P. Kainz, P. Geurts, and L. Wehenkel. Collaborative analysis of multi-gigapixel imaging data using cytomine. *Bioinformatics*, 32(9):1395–1401, 2016.
- [28] M. McCann, C. Castro, J. Ozolek, B. Parvin, and J. Kovacevic. Automated histology analysis: opportunities for signal processing. *IEEE Signal Processing*, 2014.
- [29] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

- [30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [31] H. Ravishankar, P. Sudhakar, R. Venkataramani, S. Thiruvankadam, P. Annangi, N. Babu, and V. Vaidya. Understanding the mechanisms of deep transfer learning for medical images. In *Deep Learning and Data Labeling for Medical Applications*, pages 188–196. Springer, 2016.
- [32] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.
- [33] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.
- [34] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.
- [35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [36] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.
- [37] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.
- [38] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE transactions on medical imaging*, 35(5):1299–1312, 2016.
- [39] B. van Ginneken, A. A. Setio, C. Jacobs, and F. Ciompi. Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans. In *Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on*, pages 286–289. IEEE, 2015.
- [40] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.
- [41] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.