Hydra: Cancer Detection Leveraging Multiple Heads and Heterogeneous Datasets

Giuseppe Cuccu*, Johan Jobin*, Julien Clément*, Akansha Bhardwaj*,
Carolin Reischauer‡§, Harriet Thöny§‡, Philippe Cudré-Mauroux*

* eXascale Infolab, University of Fribourg, Switzerland – {firstname.lastname}@unifr.ch
‡ Department of Medicine, University of Fribourg, Switzerland – carolin.reischauer@unifr.ch
§ Department of Radiology, HFR Cantonal Hospital of Fribourg, Switzerland – harriet.thoeny@h-fr.ch

***Abstract*—We propose an approach combining layer freezing and fine-tuning steps alternatively to train a neural network over multiple and diverse datasets in the context of cancer detection from medical images. Our method explicitly splits the network into two distinct but complementary components: the feature extractor and the decision maker. While the former remains constant throughout training, a different decision maker is used on each new dataset. This enables end-to-end training of the feature extractor on heterogeneous datasets (here MRIs and CT scans) and organs (here prostate, lung and brain). The feature extractor learns features across all images, with two major benefits: (i) extended training data pool, and (ii) enforced generalization across different data. We show the effectiveness of our method by detecting cancerous masses in the SPIE-AAPM-NCI Prostate MR Classification data. Our training process integrates the SPIE-AAPM-NCI Lung CT Classification dataset as well as the Kaggle Brain MRI dataset, each paired with a separate decision maker, improving the AUC of the base network architecture on the Prostate MR dataset by 0.12 (18% relative increase) versus training on the prostate dataset alone. We also compare against standard end-to-end Transfer Learning over the same datasets for reference, which only improves the results by 0.04 (6% relative increase).**

***Index Terms*—Deep Learning, Transfer Learning, Prostate Cancer Detection**

I. Introduction and Motivation

According to the World Health Organization [1], “cancer caused 9.6 million deaths in 2018, making it the second leading cause of death”. Early detection and treatment dramatically increase the chances of recovery. Computer-Aided Diagnosis (CAD) could in principle automate most of the diagnostic procedures in imaging, enabling radiology specialists to focus on the most challenging cases. Furthermore, the global shortage of trained radiologists has worsened over the last decade [2], [3], while aging populations have increased their demand. Though Deep Learning (DL) methods – e.g., convolutional neural networks (CNNs) – have been used to that end in the past [4], they traditionally require the provision of large collections of annotated data for optimal training and performance.

For medical applications, this is often unattainable or subject to complex processes; privacy regulations on patient data are increasingly stringent, and using such data typically requires going through a series of lengthy steps involving patient consent, ethical committees, and data anonymization. The process of collecting high-quality data (and meta-data) is tedious, error-prone, and often led by professionals with clinical duties who need to fill this information during their down time. Furthermore, medical institutions naturally prioritize patient treatment, and do not always have the luxury to consider the long-term benefits of publishing and sharing their data. Accurate, reliable labeling also requires a high degree of expertise, while in practice the task is often relegated to residents, so that the ground truth may sometimes be questionable. Overall this makes the availability of high-quality medical data a real issue (as we experienced first-hand throughout this project).

Data availability has been one of the toughest challenges slowing the application of DL to CAD. One solution to that problem is to consider multiple heterogeneous datasets to improve the training on one task. The main technique along these lines is Transfer Learning (TL; [5], [6]), defined as “the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned”. While enabling access to a larger data pool and potentially leading to higher model generalization, this also introduces a *moving target* problem for the optimization algorithm, as the new training ends up partially forgetting previous dataset-specific properties, especially w.r.t. the decision-making process [6], [7].

We propose a method based on Transfer Learning to circumvents that limitation and allow a network to specialize on multiple datasets at the same time. Our underlying assumption is that the input data (medical images of different types and from different organs in our case) shares features that can be captured jointly (see figure 1). Our method splits the neural architecture into independent components with explicit responsibilities: a “Feature Extractor” (FE; the *body*) and multiple “Decision Makers” (DM; the *heads*, one for each dataset). Named **Hydra** after the Greek myth, our approach offers the following advantages:

- Learning generic features common across several datasets, by *leveraging* the moving target.

Authors Jobin and Clément contributed equally. The implementation was part of their Master’s thesis.

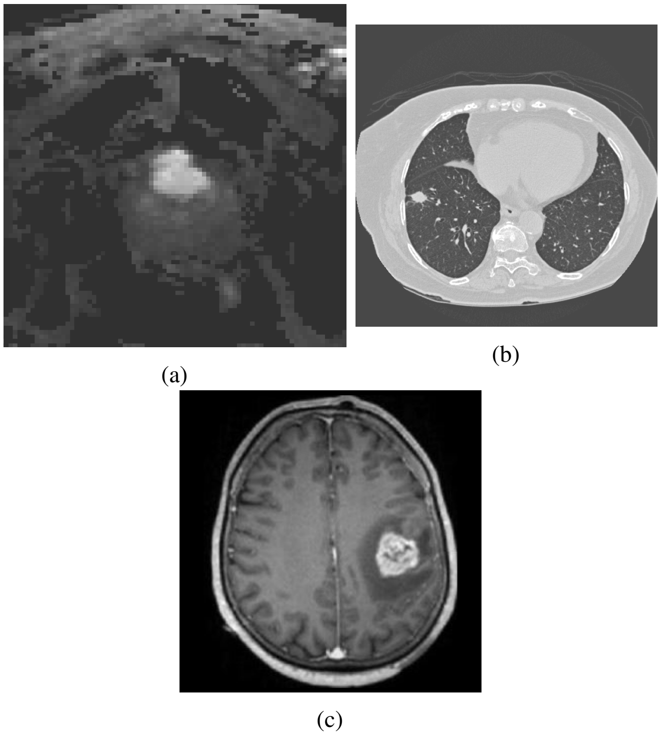


Figure 1: **Datasets.** Our Hydra framework enables training a model on multiple and diverse datasets. We present results on the PROSTATEx MRI (a) dataset, augmented by adopting the Lung CT (b) and Kaggle Brain (c) datasets for training.

- Considering a larger training pool (joint datasets), at least from the perspective of the FE.
- Sidestepping the moving target problem at the DM level, since each head retains explicit specialization on the corresponding dataset.

A. Related Work

We tested the performance of our framework in the context of Computer-Aided Diagnosis of malignant lesions. As our target dataset, we selected the SPIE-AAPM-NCI Prostate MR Classification Challenge (PROSTATEx) dataset [8]–[10], arguably the most well-known publicly available dataset for prostate lesion classification, initially published in 2017 as part of the *PROSTATEx Grand Challenge* [11]. It includes multi-sequence MRI scans from 204 patients, from multiple axes and with multiple modalities, in DICOM format.

A study by Armato et al. [11] summarizes the results of the 32 teams that took part in the challenge. The best-performing method obtained an AUC (Area Under the ROC Curve) on the challenge test set of 0.87, with the next three tied at 0.84. The median AUC for the challenge was 0.68. Armato et al. also report the ability of a less-experienced (human) radiologist at 0.81 AUC, with experts reaching 0.91.

Liu et al. [12] took part in the PROSTATEx challenge and achieved the second-best performance at the time, with a test AUC of 0.84 (validation AUC 0.92) and a network architecture called XMasNet. Their data processing pipeline stacks different combinations of MRI sequences, mimicking color channels, then process the stack with 2D convolution as if it were an RGB (red-green-blue) image.

Mehrtash et al. [13] include different processing units addressing specific MRI sequences independently. The outputs of these units are then fed into a common fully-connected layer. The final model reached an AUC of 0.80 on the official PROSTATEx challenge test set.

We base our architectural and data augmentation choices on the work by Song et al. [14], which – to the best of our knowledge – published the best PROSTATEx results so far. Their work builds on the MRI-as-color-channels stacking intuition from the input data stacks images from XMasNet, but with a novel network architecture inspired by VGG [15]. Their results claim an AUC of 0.94, though their implementation and data have not been publicly released. We re-implement their network architecture and image augmentation procedure to establish a performance baseline, before adapting it to Hydra to produce our results.

Hydra extends Transfer Learning. Abubakar and al. [16] claim that the “Transfer Learning process is used in two approaches: fine-tuning, where some modifications are made and as an off-the-shelf feature extractor where features are extracted in order to train a machine learning classifier”. The central idea behind Hydra is to combine layer freezing and fine-tuning steps alternatively. It can be seen as an asynchronous multi-task learning algorithm, where the datasets are provided in turn, switching between last layers training and whole model training. Samala and al. [17] showed that “multi-task Transfer Learning may be an effective approach for training DCNN in medical imaging applications when training samples from a single modality are limited”, which is our case. The association of a pre-trained model with multi-task learning demonstrated its efficacy [18]. Our approach keeps the essence of this architecture, but is fundamentally different in three ways: (1) we do not use a pre-trained model coming from another source like ImageNet (2) we use multiple datasets of different body parts alternatively (multiple steps) and (3) each head of the multi-task learning is trained at a different step.

In order to test our multi-dataset approach, we introduce two more datasets with different imaging modalities (CT scans and MRIs), body parts (lung and brain) and file types (PNG and JPEG): the SPIE-AAPM Lung CT Challenge dataset [19]–[21] (Lung CT), and the “Brain MRI Images for Brain Tumor Detection” dataset [22] (Kaggle Brain). A discussed above, state-of-the-art results rely on dedicated architectures (e.g. multi-channel 2D convolutions) that optimize the performance based on a single, homogeneous dataset (e.g. multi-sequence MRI). We focus on heterogeneous data types and modalities instead. To run our study, we select only one sequence from the PROSTATEx dataset (the DWI images with the highest b-value), which obviously makes our setup and results different from previous studies, but which ensures that the different datasets can be used together and that all our experiments can be

Algorithm 1 Hydra training process

```
Inputs:
    $\theta$: Target dataset, partitioned into train, test and validation: $\theta_{\text{train}}$, $\theta_{\text{val}}$, $\theta_{\text{test}}$.
    $\Sigma$: Set of support datasets $\sigma_i$, each split into train and validation (no test).
    $\mathcal{B}_0$: End-to-end model initialized with bootstrapping on $\theta_{val}$, split into feature extractor and decision maker: $\mathcal{B}_{FE}$, $\mathcal{B}_{DM}$
Initialization:
    $\mathcal{B} \leftarrow \text{train}_{FE+DM}(\mathcal{B}_0, \theta_{\text{train}})$          ▷ Train model end-to-end on target (train) dataset
Main:
    for $\delta_i$ in $\Sigma \mathbin{++} \theta$ do          ▷ $\theta$ will be used again, last, after all other $\sigma_i$
        $DM_i \leftarrow \text{init}_{\text{DM}}()$          ▷ For $\delta_i = \theta$ we can reuse the first $\mathcal{B}_{DM}$ from $\mathcal{B}_0$
        $\mathcal{B} \leftarrow \mathcal{B}_{FE} \oplus DM_i$          ▷ Build end-to-end model
        $DM_i \leftarrow \text{train}_{\textit{DM\_only}}(\mathcal{B}, \delta)$          ▷ Chosen based on results on $\delta_i^{val}$
        $\mathcal{B} \leftarrow \mathcal{B}_{FE} \oplus DM_i$          ▷ Update $DM_i$ ($FE$ has not changed)
        $\mathcal{B} \leftarrow \text{train}_{\textit{end-to-end}}(\mathcal{B}, \delta)$          ▷ Train both $FE$ and $DM_i$ this time
    return $\mathcal{B}$
```

compared on a fair basis. For comparison, our method ends up using only ~15% of the imaging data available in PROSTATEx, which allows in return to highlight the applicability of Hydra to leverage multiple, diverse data types.

B. Contribution

The key contributions of this work are as follows:

- A novel framework named **Hydra**, combining fine-tuning and layer freezing and enabling training on multiple and diverse datasets (different data types and potentially different tasks) by using multiple independent decision makers (the "heads", one per dataset) on top of a common feature extractor (the "body").
- A novel learning algorithm for the Hydra framework, which alternates head specialization with end-to-end training to smooth the learning process.
- A demonstration of Hydra's applicability on a domain with high impact but bridled by data scarcity, computer-aided diagnosis based on multi-type medical imaging (MRI and CT scans over three distinct organs).
- The publication of our code-base in an open-source package for reproducibility and extensibility purposes[1]

II. Method

We propose a method that reuses features learned on a dataset to improve performance on further datasets. This immediately recalls Transfer Learning, which tackles the problem of limited data availability by training on multiple datasets sharing common features.

The main limitation of TL however lies in the fact that its training either targets *generalization* (e.g. common low-level *features*) with layer freezing or *specialization* (e.g. dataset-specific *decisions* based on the features) with fine-tuning. Both goals are typically not addressed simultaneously but separately, which does not allow to specialize different parts of the architecture to the different roles.

Instead, we explicitly split our network architecture into *feature extraction* (FE, the "body") and *decision maker* (DM, the "head"). While the common FE remains the same (as in: shared) throughout all training, each dataset is matched to a different DM head. This allows each head to specialize on the decision required for the particular dataset/task, while the body can retain its generalization capability, being trained in extracting features that are useful across all datasets.

In order to enable the applicability of standard Deep Learning end-to-end training methods on this architecture, we introduce a sequential training algorithm (see Algorithm 1 and Figure 2) providing a successful proof-of-concept of our idea and a baseline for further work. The following sections describe the architecture used in our experiments and detail our learning process.

A. Training Overview

Our training approach mainly consists of alternating training a new DM in isolation on the corresponding dataset, and end-to-end training of both FE+DM on the same dataset, totaling five steps with the proposed sequence of one target plus two supporting datasets. The process is described in Algorithm 1, illustrated in Figure 2, and discussed in detail below.

Pre-processing. At initialization, all datasets are first pre-processed into a common format. To maximize the generality of the learned features, we select data from all datasets that can be hypothesized to contain similar features. For imaging datasets, this means selecting them based on color schemes and gradients, levels of detail, and generic look.

In our case, we select DWI sequences from the MRI datasets, which visually match the CT scans of the Lung CT dataset relatively well. As a negative example, a dataset of pictures of houses or cars would not offer a good visual match, as (i) the color scheme is in principle different, (ii) the main

[1] See our public repository at: https://github.com/eXascaleInfolab/hydra

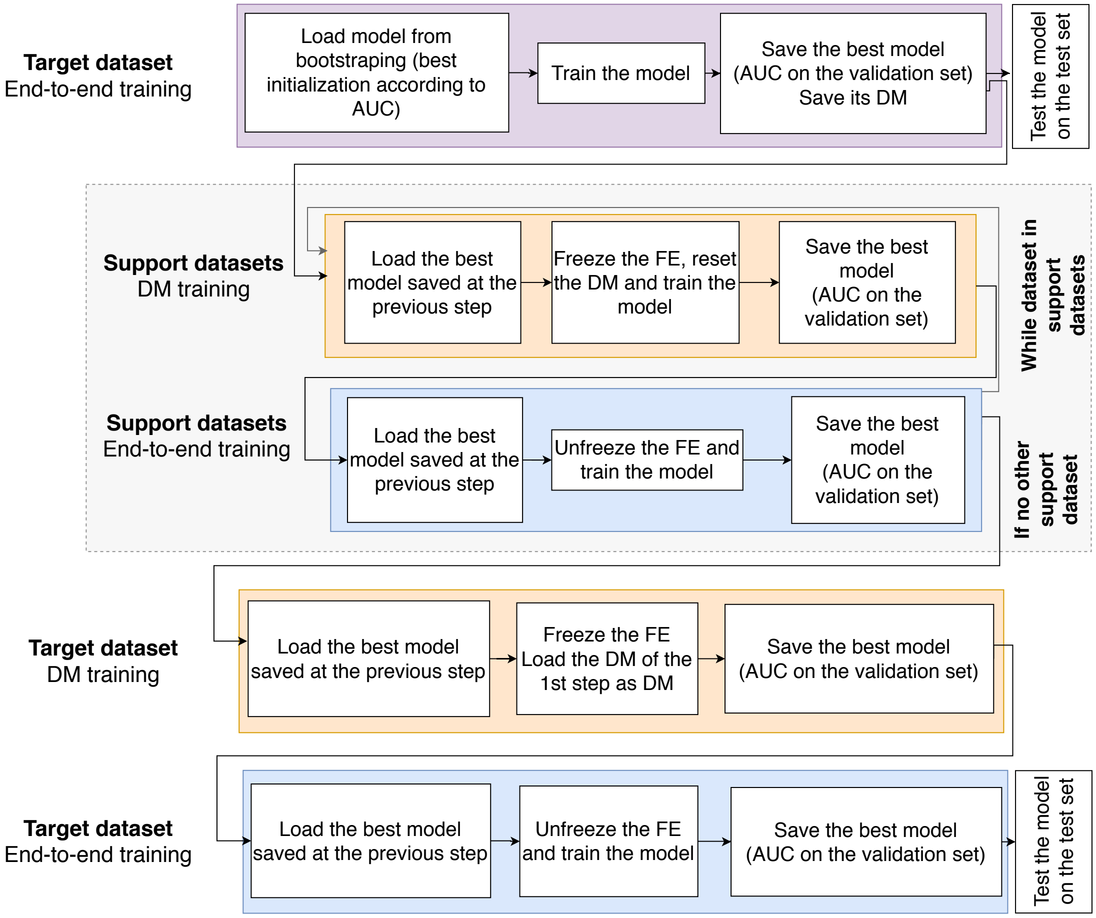

Target dataset
End-to-end training
Load model from bootstraping (best initialization according to AUC)
Train the model
Save the best model (AUC on the validation set) Save its DM
Test the model on the test set
Support datasets
DM training
Load the best model saved at the previous step
Freeze the FE, reset the DM and train the model
Save the best model (AUC on the validation set)
While dataset in support datasets
Support datasets
End-to-end training
Load the best model saved at the previous step
Unfreeze the FE and train the model
Save the best model (AUC on the validation set)
If no other support dataset
Target dataset
DM training
Load the best model saved at the previous step
Freeze the FE Load the DM of the 1st step as DM
Save the best model (AUC on the validation set)
Target dataset
End-to-end training
Load the best model saved at the previous step
Unfreeze the FE and train the model
Save the best model (AUC on the validation set)
Test the model on the test set

Figure 2: **Hydra training process.** Alternating between adapting the new head to the existing body ("DM training") and end-to-end training including the feature extractor enables a smoother transition across datasets. The first and last datasets are the same and the target of our study: the first one is to initialize the features in a more useful configuration, while the last one is used to specialize the network on the target problem.

geometries do not correspond (internal organs lack straight lines and definite angles), and (iii) the transitions are starkly different (with medical imaging showing slowly changing density gradients versus the sharp lines and color boundaries of house images).

Further pre-processing steps are independent of the proposed method and are detailed in Section III.

Model Instantiation. The common FE and first DM are generated first; to speed up the learning process, we introduce bootstrapping, by generating 200 random parametrizations and selecting the model with best validation AUC performance as our starting point. The random weights are drawn from a uniform distribution with symmetrical boundaries proportional to the square root of the number of weights in the layer.

Next, for each of the support datasets, a new independent head is generated and paired with the common body. This pairs a trained feature extractor with a random head, which constitutes an unwanted moving target for the learning. In order to meaningfully extend the training on top of the features learned so far, we first train the head alone (keeping the body "frozen" and unchanged) until it converges on the currently available features (which, as a reminder, were trained on a different dataset). This step (*DM-only*) can be interpreted as *centering* the new head on the body's features.

The following step is to release the freeze on the body, enable *end-to-end* training of both the now centered head and the common body. This process enables a smooth transition of the FE specialization from one dataset onto the next, gradually including novel information. The training on the current dataset is complete and the loop can resume on the next dataset. In the last round, the target dataset is reintroduced with the aim of both refining the learned features and training a new head that specializes on them.

Model Selection. At each epoch, the current model is evaluated on the corresponding validation set. At the end of each round (either DM-only or end-to-end), the model with the best validation result is selected to be used in the next iteration. In effect, this acts like an early stopping criterion for progressive model selection, while enabling a full study of the training trajectory for plotting purposes.

Dataset	Step name	L. rate	Dropout	Epoch	AUC
Prostate	DS1/End-to-end	1e-8	0.4	527	0.7334
Brain	DS2/DM-only	1e-7	0.3	1999	0.8483
Brain	DS2/End-to-end	1e-8	0.3	166	0.8596
Lung	DS3/DM-only	1e-5	0.3	1852	0.7755
Lung	DS3/End-to-end	1e-8	0.3	167	0.7691
Prostate	DS4/DM-only	1e-5	0.3	1914	0.7777
Prostate	DS4/End-to-end	1e-9	0.0	391	0.7749

Table I: **Hyperparameters.** Values optimized with cross-validation at each learning step (first dataset, second dataset with frozen layers, second dataset with end-to-end training...) for the Hydra algorithm. Column *"L. rate"* shows learning rates for the *Adam* optimizer. Optimal performance was consistently reached with a **batch size** of 128 for all steps. The **number of epochs** was fixed to 2000 for plot-generation purposes, but the best model (as per column *"AUC"*) was found at the epoch referenced in column *"Epoch"* for each step. It is interesting to notice how the *DM-only* training receives most benefits from the longer training, while the *end-to-end* sessions quickly cap in performance before starting to overfit.

III. Experimental Setup

As we work with multiple datasets, using a single set of common hyperparameters is likely suboptimal. To maximize training efficiency we hence search for a new set of hyperparameters for each dataset using cross-validation. Table I presents the values found at each step of the experiment.Though the validation AUCs on Lung CT and Kaggle Brain are promising, further study (including a proper test split) goes beyond the goal of this paper, and is better addressed in future work with a focus on scores optimization.

Our reference implementation is written in PyTorch and trained with the Adam optimizer. We run our experiments on a single machine with a 64-core Intel(R) Xeon(R) 6142 CPU at 2.60GHz (with 6GB of RAM per core), and 5 NVidia GeForce RTX 2080 Ti GPUs at 2.1GHz (with 10GB vRAM each). The complete training of the final Hydra model (including the generation of per-epoch statistics, but excluding the cross-validation process) took approximately 60 hours. Standard end-to-end TL in comparison took about 30 hours, as only one 2000-epoch training is done for each dataset (while Hydra does two 2000-epoch steps). Classical single-dataset end-to-end training took 7 hours. All time estimations are based on fixed full-runs of 2000 epochs. Table I suggests that the adoption of early stopping would considerably reduce these times.

Model Architecture. The network architecture used in this work is inspired by Song et al. [14], who specialized the VGG structure [15] for prostate lesion classification. A detailed description of our model is available in Figure 3. It is composed of three convolution-dropout-max-pooling blocks, followed by three fully-connected-dropout blocks. Each convolutional box (in blue in the figure) stands for a sequence of three layers, namely: convolution, batch normalization and exponential linear unit (ELU). The fully-connected layer box (in orange) is instead composed of a fully-connected layer followed by an exponential linear unit. The last fully-connected box (in purple) is again composed of a fully-connected layer, but followed this time by a softmax function for classification. Our approach introduces an explicit split of the architecture into two distinct sub-modules with specialized responsibilities: feature extractor (FE or *body*) and decision maker (DM or *head*). The split point was chosen empirically through extensive experimentation. The best performance was achieved by considering all convolutional layers in the common FE, followed by one fully-connected layer acting as a "feature composer" of sorts. The DMs are composed of multiple fully-connected layers topped by softmax activation, as common in classification applications.

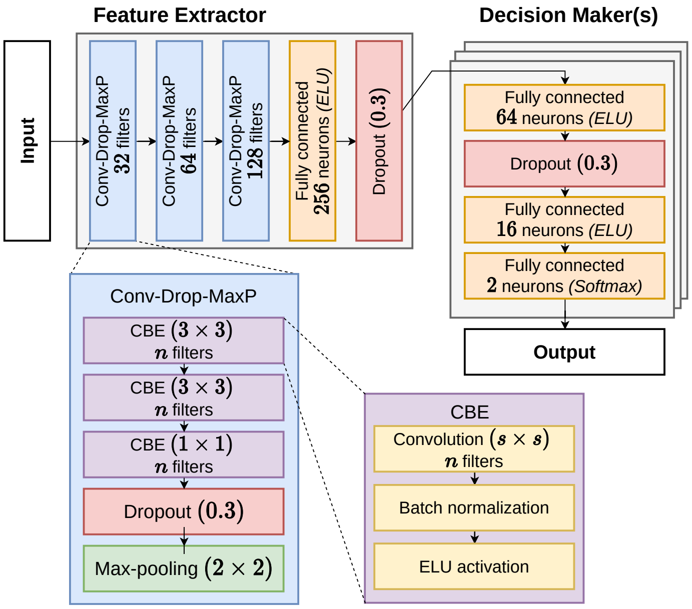

Feature Extractor
Decision Maker(s)
Input
Conv-Drop-MaxP 32 filters
Conv-Drop-MaxP 64 filters
Conv-Drop-MaxP 128 filters
Fully connected 256 neurons (ELU)
Dropout (0.3)
Fully connected 64 neurons (ELU)
Dropout (0.3)
Fully connected 16 neurons (ELU)
Fully connected 2 neurons (Softmax)
Output
Conv-Drop-MaxP
CBE (3 × 3) n filters
CBE (3 × 3) n filters
CBE (1 × 1) n filters
Dropout (0.3)
Max-pooling (2 × 2)
CBE
Convolution (s × s) n filters
Batch normalization
ELU activation

Figure 3: **Model architecture.** While basing our model on the architecture proposed by Song et al. [14], we split it explicitly into a feature extractor (top left-hand side) and a decision maker (top right-hand side), enforcing a well-defined specialization of roles. The feature extractor is mostly composed of convolutional layers, which compute increasingly complex features from the data, topped by an aggregating dense layer with ELU activation. The decision maker is a succession of fully-connected ELU layers, ending in a binary layer with softmax activation.

Data processing and augmentation. To enable a smooth transition across datasets, all images need to be pre-processed to a common format. We first converted all images into Numpy arrays (using DICOM normalization information when available), rescaled to a common resolution, and normalized with Z-score (using per-patient mean and standard deviation). Further metadata used in the processing (patient identifier, lesion coordinates, classification label, etc.) were integrated in the new filenames. The combination of (i) switching to a common, numerical data type and (ii) integrating the metadata into the filename provides a universal format that simplifies normalization and integration of multiple datasets to be processed by the same network architecture. For the sake of uniformity across datasets, the only MRI sequence selected from the PROSTATEx data was the DWI images with highest b-value under the transverse plane. This sequence was selected through visual inspection, as malignant lesions look the closest across all three dataset in terms of visual features. This limits of course the maximum score attainable on that particular dataset, but enables studying the differences in performance gained by our method in isolation, which is more in line with the goal of this paper. Improving the best score on the PROSTATEx dataset, possibly by training a decision maker on the combined features from previous work and our approach, is left for future work. The datasets were then synthetically augmented using a technique akin to Song et al. [14]. We began by cropping a relatively larger patch (e.g. 130×130 pixels for PROSTATEx) centered on the lesion. All datasets were augmented using rotation (-20° to 20°), horizontal flipping (probability of 0.5) and horizontal shifting (-1px, 0px, 1px). Each patch was then further resized to its final size of 65×65 pixels. Class imbalance was also addressed in this step, by balancing the class with fewer elements with augmented data. Finally, we split the target dataset into train, validation and test sets with a 8-1-1 split. The lower proportion for validation and test has a double effect on such a small dataset: more data is available for training, but at the same time the validation set becomes less representative of the test set, making the model selection process equivocal. The support datasets are all split 9-1 into train and validation: as we focus on one single target, there is no test set.

IV. Results

We compare the performance of Hydra against classical end-to-end Transfer Learning, maintaining the same sequence of datasets, pre-processing and augmentation, basically corresponding to Algorithm 1 minus the heads change and body freeze, which are distinctive of our contribution.

Our final results are given in Table II, which also includes a short run of Random Weight Guessing [23] on the base model to offer an estimation of the problem complexity [24] and further motivates our bootstrapping initialization.

Classical learning reaches 0.73 in AUC for validation: this network is also used for the first (initialization) step of both Hydra and TL approaches, to study the direct performance improvement of the two methods. Hydra training then reaches an AUC on validation of 0.77, corresponding to a 0.04 increase (5% relative). While not astounding *per se*, this is in line with our expectation for a method whose strength lies in higher generalization. Transfer Learning here shows the better performance, with a 0.86 validation AUC. Notably though results on the validation set are biased, since it is used in training to select the best model at each step, both for Hydra and TL.

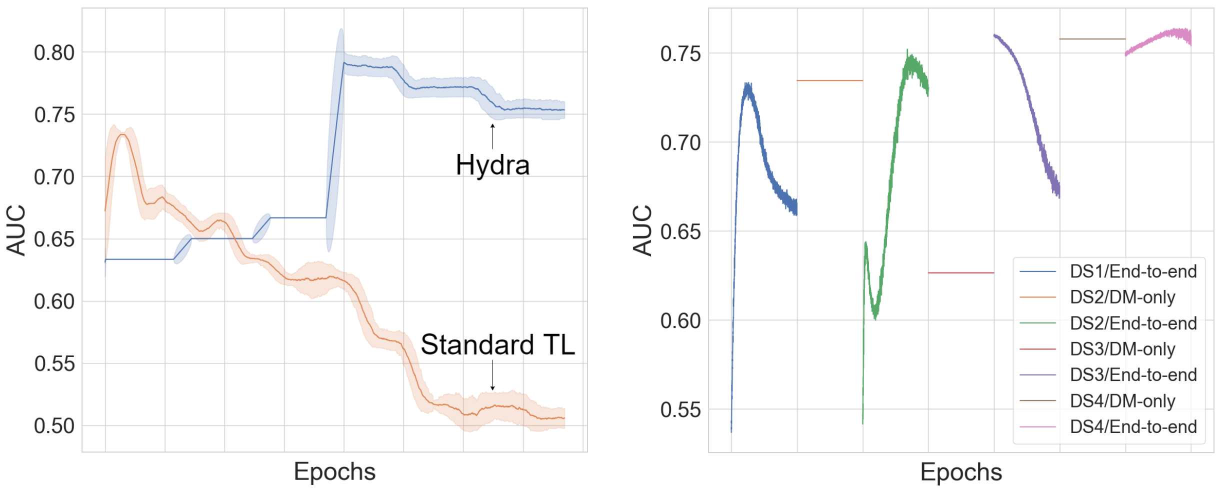
More meaningful are the results on the test set, as this data was not utilized at any point during the training process and is a more correct estimator for the final model's generalization and applicability. Here both Classical and end-to-end Transfer Learning distinctively show overfitting, with an AUC on test of 0.68 and 0.72 respectively. Hydra by contrast goes from 0.68 to 0.80, a 0.12 increment, which corresponding to an 18% increase in relative performance. This highlights the higher resilience of Hydra to overfitting, and the improved robustness when generalizing to unseen data. Furthermore, the results also support the following points:

- The AUC of canonical end-to-end training is distinctively lower on the test set than on the validation set. Together with the high RWG results, this corroborates how our choice of dataset and train-test-validation split represent a prime choice for our study on generalization.
- The AUC of Hydra on both the validation set and test set are close – actually, looking slightly better on the test. This underlines and confirms the main strength of Hydra: improved generalization, thanks to the broader training of the feature extractor.
- While standard validation-based model selection can be misleading for classical learning (loss of 0.05) and TL (loss of 0.14 AUC), the generic features learned by Hydra perform comparably well on both validation and test (*gain* of 0.2 AUC).

Figure 4 highlights the training trajectory of Hydra versus TL. We present in (a) results on the test set of the current model, for each epoch, as it is trained on the training set. While TL shows a positive trend in the first epochs, it progressively suffers from overfitting, which lowers the model performance at each epoch. Hydra on the other hand shows two distinct trends. In the first half of training, performance is continuous improving as the DM is tuned on the features learned by the FE so far. The second half shows the switch to end-to-end training, which (akin to TL) after a short but steep improvement phase immediately begins declining towards overfitting. In this case though the change is noticeably slower, which hints at higher robustness and generalization capabilities. The right-hand side

Metric	RWG	Classical	Transfer	Hydra
AUC PROSTATEx Validation	0.80	0.73	0.86	0.77
AUC PROSTATEx Test	0.75	0.68	0.72	**0.80**

Table II: **Final results.** Performance comparison between Random Weight Guessing (RWG; [23]) classical learning, Transfer Learning and Hydra. RWG is presented here as a baseline of the problem complexity: these results come from generating 1000 random networks with standard initialization, evaluating them on the target validation (no training), and running the best performing (validation) network again on the test set. Its performance suggests the task is deceiving towards overfitting, which makes it hard for classical learning. The bootstrapping for both classical and Transfer Learning only uses 200 random initializations, accounting for a lower starting score.



(a)

(b)

Figure 4: **(a) Performance comparison.** Evolution of model performance (AUC) on the target test set, as training progresses. While the results on the validation sets are biased (as they are used for model selection), the test set is never accessed during training and thus represents a better proving ground for the model's generalization and applicability. The discontinuity (and peak) of Hydra's path at the center of the graph corresponds to its switching from DM-only training to end-to-end. This also means that Hydra's training took 2000+2000 epochs, while the Transfer Learning only took 2000 (only end-to-end), though running it for 4000 epochs would be unlikely to improve the final result as the method converges into overfitting. Curves show moving averages over 100 epochs for smoothing, with standard deviation areas. **(b) Hydra learning trajectories.** Validation results for the reference decision maker on the target dataset, as the feature extractor is trained on multiple datasets. This shows how progressively learning features from the other datasets into the common FE impacts the performance on the prostate lesion classification dataset. Each training step generates a different training curve. The blue curve corresponds to standard end-to-end training of the initial model (FE + DM1) on the first dataset (our target: PROSTATEx). DM1 is saved as a *reference*, and used to produce all the remaining points in the plot. It remains unchanged until the last iteration, while the common FE is trained on the support datasets paired with their corresponding DM. This is why every second curve is flat: another DM is being trained on a support dataset while the FE is "frozen". Notably, the maximum AUC reached by each trajectory (the height of the crest of each curve in turn) increases monotonically as the learning sequence progresses.

of the figure (Figure 4 (b)) illustrates the learning trajectory of our system as it is trained on multiple datasets. The figure shows how progressively learning features from other datasets impacts the performance on the prostate lesion classification dataset.

V. Conclusion

Broader Impact. A global shortage has impacted the profession of radiologist for the past twenty years [2], [3]. Computer-Aided Diagnosis could become a powerful tool to support the available medical personnel, but the application of state-of-the-art Deep Learning techniques is currently hampered by a severe shortage of very large datasets across most medical imaging domains. While medical image datasets are often small in size, mostly due to ethical and operational constraints, thousands of different datasets (though diverse and often incompatible) have been made publicly available over the past decades. Aggregating such bountiful information using a unifying learning framework is however nontrivial, as medical images span a broad range of imaging techniques, body parts (hence tasks), and data configurations (e.g. resolution, format, etc.). End-to-end training on a single dataset is often insufficient, while standard end-to-end Transfer Learning across a large number of datasets suffers from catastrophic forgetting due to the moving target of the varying task.

This work addresses these limitations by providing a new neural network framework and learning algorithm (Hydra), capable of supporting both generalization and specialization at the same time by maintaining a common feature extractor (the *body*) and a set of independent decision makers (heads, one per dataset). This enables the application of established end-to-end training algorithms, while both specializing multiple decision makers and sharing a common feature extractor. Given the widespread lack of training data in the medical domain, we propose this method as an alternative to (and generalization of) Transfer Learning, with potentially broad applicability.

Results. We provide experimental results on the widely used PROSTATEx dataset, where Hydra achieves a test-set AUC increase of 0.12 from 0.68 to 0.80, which corresponds to an 18% relative improvement. This result is comparable to a less-experienced (human) radiologist (0.81) [11], which we find promising in a proof-of-concept study. By contrast, standard end-to-end Transfer Learning following the same training pattern only improves by 0.04 (6% relative).

Future work. We are currently exploring the full potential of our framework through a series of further experiments, including sequence optimization (i.e., how to identify the best training sequence for a given list of datasets), parallel training (i.e., randomize at each epoch which dataset/head is trained), and improved model selection (i.e., how to select the best model to be used in the next step).

Acknowledgments

This work received support from the Swiss NSF under grant numbers 407540_167320 and 32003B_176229/1.

References

[1] W. H. Organization, "Fact Sheet about Cancer," 2020. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/cancer

[2] J. H. Sunshine, Y. S. Cypel, and B. Schepps, "Diagnostic radiologists in 2000: basic characteristics, practices, and issues related to the radiologist shortage," *American Journal of Roentgenology*, vol. 178, no. 2, pp. 291–301, 2002.

[3] J. H. Sunshine, C. D. Maynard, J. Paros, and H. P. Forman, "Update on the diagnostic radiologist shortage," *American Journal of Roentgenology*, vol. 182, no. 2, pp. 301–305, 2004.

[4] J. Gao, Q. Jiang, B. Zhou, D. Chen, 1 College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China., and 2 Shanghai University of Medicine &amp; Health Science, Shanghai 201308, China, "Convolutional neural networks for computer-aided detection or diagnosis in medical image analysis: An overview," *Mathematical Biosciences and Engineering*, vol. 16, no. 6, pp. 6536–6561, 2019.

[5] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[6] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[7] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *NIPS 2005 workshop on transfer learning*, vol. 898, 2005, pp. 1–4.

[8] G. Litjens, O. Debats, J. Barentsz, N. Karssemeijer, and H. Huisman, "ProstateX Challenge data," *The cancer imaging archive*, 2017. [Online]. Available: https://wiki.cancerimagingarchive.net/display/Public/SPIE-AAPM-NCI+PROSTATEx+Challenges

[9] ——, "Computer-aided detection of prostate cancer in MRI," *IEEE Transactions on Medical Imaging*, vol. 33, no. 5, pp. 1083–1092, May 2014.

[10] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, L. Tarbox, and F. Prior, "The Cancer Imaging Archive (TCIA): Maintaining and operating a public information repository," *Journal of Digital Imaging*, vol. 26, no. 6, pp. 1045–1057, Dec. 2013.

[11] S. G. Armato, H. Huisman, K. Drukker, L. Hadjiiski, J. S. Kirby, N. Petrick, G. Redmond, M. L. Giger, K. Cha, A. Mamonov, J. Kalpathy-Cramer, and K. Farahani, "PROSTATEx Challenges for computerized classification of prostate lesions from multiparametric magnetic resonance images," *Journal of Medical Imaging*, vol. 5, no. 04, p. 1, Nov. 2018.

[12] S. Liu, H. Zheng, Y. Feng, and W. Li, "Prostate cancer diagnosis using deep learning with 3D multiparametric MRI," in *Medical Imaging 2017: Computer-Aided Diagnosis*, S. G. A. III and N. A. Petrick, Eds., vol. 10134, International Society for Optics and Photonics. SPIE, 2017, pp. 581 – 584.

[13] A. Mehrtash, A. Sedghi, M. Ghafoorian, M. Taghipour, C. M. Tempany, W. M. W. III, T. Kapur, P. Mousavi, P. Abolmaesumi, and A. Fedorov, "Classification of clinical significance of MRI prostate findings using 3D convolutional neural networks," in *Medical Imaging 2017: Computer-Aided Diagnosis*, S. G. A. III and N. A. Petrick, Eds., vol. 10134, International Society for Optics and Photonics. SPIE, 2017, pp. 589 – 592.

[14] Y. Song, Y.-D. Zhang, X. Yan, H. Liu, M. Zhou, B. Hu, and G. Yang, "Computer-aided diagnosis of prostate cancer using a deep convolutional neural network from multiparametric MRI: PCa classification using CNN from mp-MRI," *Journal of Magnetic Resonance Imaging*, vol. 48, no. 6, pp. 1570–1577, Dec. 2018.

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556 [cs]*, Apr. 2015.

[16] A. Abubakar, M. Ajuji, and I. Yahya, "Comparison of deep transfer learning techniques in human skin burns discrimination," *Applied System Innovation*, vol. 3, p. 20, 04 2020.

[17] R. K. Samala, H.-P. Chan, L. M. Hadjiiski, M. A. Helvie, K. H. Cha, and C. D. Richter, "Multi-task transfer learning deep convolutional neural network: application to computer-aided diagnosis of breast cancer on mammograms," *Physics in Medicine & Biology*, vol. 62, no. 23, pp. 8894–8908, nov 2017.

[18] W. Zhang, R. Li, T. Zeng, Q. Sun, S. Kumar, J. Ye, and S. Ji, "Deep model based transfer and multi-task learning for biological image analysis," *IEEE Transactions on Big Data*, vol. 6, no. 2, pp. 322–333, 2020.

[19] S. G. Armato, K. Drukker, F. Li, L. Hadjiiski, G. D. Tourassi, R. M. Engelmann, M. L. Giger, G. Redmond, K. Farahani, J. S. Kirby, and L. P. Clarke, "LUNGx Challenge for computerized lung nodule classification," *Journal of Medical Imaging*, vol. 3, no. 4, p. 044506, Dec. 2016.

[20] S. G. Armato, L. Hadjiiski, G. D. Tourassi, K. Drukker, M. L. Giger, F. Li, G. Redmond, K. Farahani, J. S. Kirby, and L. P. Clarke, "Guest Editorial: LUNGx Challenge for computerized lung nodule classification: reflections and lessons learned," *Journal of Medical Imaging*, vol. 2, no. 2, p. 020103, Jun. 2015.

[21] S. G. Armato, L. Hadjiiski, G. D. Tourassi, M. L. Giger, F. Li, G. Redmond, K. Farahani, J. Kirby, and L. P. Clarke, "SPIE-AAPM-NCI Lung Nodule Classification Challenge Dataset," *The cancer Imaging Archive*, 2017. [Online]. Available: https://wiki.cancerimagingarchive.net/display/Public/SPIE-AAPM+Lung+CT+Challenge#b38bc90c1f4c498fbcb2acb3495cd9d8

[22] "Brain MRI images for brain tumor detection," 2019. [Online]. Available: https://kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection

[23] J. Schmidhuber, S. Hochreiter, and Y. Bengio, "Evaluating benchmark problems by random guessing," *A Field Guide to Dynamical Recurrent Networks, ed. J. Kolen and S. Cremer*, pp. 231–235, 2001.

[24] D. Oller, T. Glasmachers, and G. Cuccu, "Analyzing reinforcement learning benchmarks with random weight guessing," in *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20*, 2020, pp. 975–982.