Semi-supervised deep learning based on label propagation in a 2D embedded space

Bárbara C. Benato[1], Jancarlo F. Gomes[1,2], Alexandru C. Telea[3], and Alexandre X. Falcão[1]

[1] Laboratory of Image Data Science, University of Campinas, Campinas, Brazil
{barbara.benato; jgomes; afalcao}@ic.unicamp.br
[2] Faculty of Medical Sciences, University of Campinas, Campinas, Brazil
[3] Department of Information and Computing Science, Utrecht University, Utrecht, The Netherlands
a.c.telea@uu.nl

Abstract. Expert human supervision of the large labeled training sets needed by convolutional neural networks is expensive. To obtain sufficient labeled samples to train a model, one can propagate labels from a small set of supervised samples to a large unsupervised set. Yet, such methods need many supervised samples for validation. We present a method that iteratively trains a deep neural network (VGG-16) from labeled samples created by projecting the features of VGG-16's last max-pooling layer in 2D with t-SNE and propagating labels with the Optimum-Path Forest semi-supervised classifier. As the labeled set improves along iterations, it improves the network's features. We show how this significantly improves classification results on test data (using only 1% to 5% of supervised samples) of three private challenging datasets and two public ones.

Keywords: Data annotation · Label propagation · Iterative feature learning.

1 Introduction

Convolutional neural networks (CNNs) usually need large training sets (labeled images) [14, 19]. While regularization, fine-tuning, transfer learning, and data augmentation [24] can help this, manually annotating enough images (human supervision) by expert uses, as in Biology and Medicine, remains expensive.

To build a large enough training set, Lee [12] propagated labels from a small set of supervised images to a large set of unsupervised ones, as an alternative to entropy regularization. In detail, Lee trained a neural network with 100 to 3000 supervised images, assigned the class with maximum predicted probability to the unsupervised ones, and then fine-tuned a neural network with the true-plus-artificially labeled (*pseudo* labeled) samples, showing advantages over other semi-supervised learning methods. Still, this required validation sets of over 1000 supervised images for the optimization of hyperparameters; used a network with a single hidden layer; and was shown on a single dataset (MNIST).

Label propagation from supervised to unsupervised samples was recently used to build larger training sets [8, 9, 26, 13]. Amorim et al. [2] used the semi-supervised Optimum Path Forest (OPFSemi) classifier [1] for this, outperforming several existing semi-supervised techniques when training CNNs. Yet, they did not explore CNNs pre-trained with large supervised datasets for transfer learning, and still needed many supervised samples (10% of the dataset) for validation.

Graph-based semi-supervised learning has recently received increasing attention [1, 3, 6, 28]. By modeling training samples as nodes of a graph whose arcs connect adjacent samples in the feature space, one can propagate labels from supervised samples to their most strongly connected unsupervised neighbors. Benato et al. [4, 5] showed the advantages of OPFSemi for label propagation in a 2D *embedded* space created by t-SNE [15] from the latent space of an autoencoder trained with unsupervised images – which differs from [2] where propagation is done in the *feature* space. Their supervised classifiers achieved higher performance on unseen test sets when trained with large sets of truly-and-artificially labeled samples, with OPFSemi surpassing LapSVM [22] for label propagation. Yet, they have not used this strategy to train deep neural networks.

We fill the above gaps by proposing a loop (Fig. 1) that trains a deep neural network (VGG-16, [21]) with truly-and-artificially labeled samples along iterations. At each iteration, we create a 2D embedded space by the t-SNE projection (like [4], different from [2]) of VGG-16's features at the last max-pooling layer (before the MLP) and propagate labels by OPFSemi, so that the labeled set *jointly* improves with the CNN's feature space over iterations. Our method can improve classification on unseen test data of challenging datasets.

2 Proposed Pipeline

After the user supervises a small set of training images, we execute a three-step loop (*deep feature learning*, *feature space projection*, and *label propagation*; Fig. 1).

2.1 Deep Feature Learning

To minimize user effort for annotation, we use the ability of pre-trained CNNs to transfer knowledge [27] between scenarios – e.g., from natural to medical images – using few supervised samples and few epochs. We use VGG-16, pre-trained on ImageNet [19], and fine tuned with the supervised images. In the next iterations of our loop, we train VGG-16 with all true-and-artificially labeled images.

2.2 Feature Space Projection

We project the features of the last max-pooling layer of VGG-16 by t-SNE [15] in a 2D embedded space. One may conceptually divide a deep neural network into (a) layers for feature extraction, (b) fully connected layers for feature space reduction, and (c) the decision layer (a MLP classifier). We explored features that result from (a), where the feature space is still high and sparse; a comparison

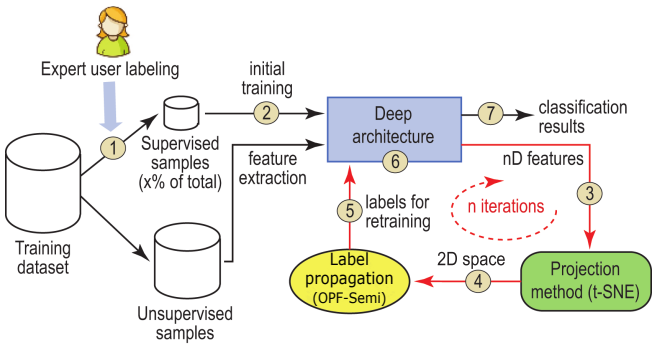

Fig. 1. Pipeline of our method. The user supervises a small fraction x of images (1). These are used to train a deep neural network (2), which extracts features from the unsupervised images (3). Features are projected in a 2D embedded space (4). A semi-supervised classifier propagates labels to the unsupervised images (5). The model is retrained by all images and their assigned labels (6), creating a new and improved feature space along iterations. Finally, the trained model is used for classification (7).

with the output of the last hidden layer is left for future work. Rauber et al. [18] showed that high classification accuracy relates to a good separation of classes in a 2D projection. Hence, if a 2D projection presents good class separation, then a good class separation can also be found in the data space.

Benato et al. [4] showed that label propagation (using two semi-supervised classifiers) in a 2D projection space leads to better classification results than using the latent feature space of an autoencoder. We also opt to investigate label propagation in a 2D projected space (created by t-SNE [15], as in [18] and [4]) to create larger training sets for deep learning.

2.3 Label Propagation

We used OPFSemi in both the 2D t-SNE projection [4] and the original feature space [2]. OPFSemi sees each sample as a node of a complete graph, setting the cost of a path between two nodes to the maximum arc weight (Euclidean distance between samples) along it. The supervised nodes seed the computing of a minimum-cost path forest – each seed propagates its label to the most closely connected unsupervised nodes of its tree.

3 Experiments and Results

3.1 Experimental Set-up

We randomly divide each dataset into supervised training samples S, unsupervised training samples U, and testing samples T. To measure the impact of annotated samples on classification, we let $S \cup U$ have 70% of samples, while T has 30%. To minimize user effort for supervision, we set $|S|$ to 1% up to 5% of the entire dataset, thus much smaller than $|U|$ For statistics, we generate three partitions of

each experiment randomly and in a stratified manner. We validate our method, called *DeepFA looping*, by three experiments:

1. *Baseline*: train VGG-16 on S, test on T, ignore U (Fig. 1 steps 1,2,6,7).
2. *DeepFA*: train VGG-16 on S; extract $S \cup U$ features from VGG-16 and project them in $2D$ with t-SNE; OPFSemi label estimation in U; train VGG-16 on $S \cup U$ and test on T (all of Fig. 1 for $n = 1$).
3. *DeepFA looping*: train VGG-16 on S; extract $S \cup U$ features from VGG-16; project them in $2D$ with t-SNE; OPFSemi label propagation on U; train VGG-16 on $S \cup U$; repeat from the projection step $n = 5$ times; test on T (all of Fig. 1 for $n > 1$).

To compare effectiveness, we compute accuracy from VGG-16's final probability. As we have unbalanced datasets, we also compute Cohen's κ coefficient, $\kappa \in [-1, 1]$, where $\kappa \leq 0$ means no possibility and $\kappa = 1$ means full possibility of agreement occurring by chance, respectively. For the experiments where OPFSemi propagates labeled samples, we also compute the label propagation accuracy in U, *i.e.*, the number of correct labels assigned in U over the size of U.

3.2 Datasets

We first use two public datasets: MNIST [11] contains handwritten digits from 0 to 9 as 28×28 grayscale images. We use a random subset of 5K samples from MNIST's total of 60K. CIFAR-10 [10] contains color images (32×32 pixels) in 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. We use a random subset of 5K images from CIFAR-10's total of 60K.

We also used three private datasets from a real-world problem (Fig. 2). These datasets contain color microscopy images (200×200 pixels) of the most common species of human intestinal parasites in Brazil, responsible for public health problems in most tropical countries [25]. These datasets are challenging, since they are unbalanced and contain an impurity class for the large majority of the samples, having samples very similar to parasites, which makes classification hard (Fig. 2). We explored two (out of three) datasets with and without the impurity class, yielding thus five total datasets: (i) *Helminth larvae*, (ii) *Helminth eggs* without impurities, (iii) *Helminth eggs* with impurities, (iv) *Protozoan cysts* without impurities, and (v) *Protozoan cysts* with impurities. The *Helminth larvae* dataset presents larvae and impurities (2 classes, 3514 images); the *Helminth eggs* dataset has several categories: *H.nana*, *H.diminuta*, *Ancilostomideo*, *E.vermicularis*, *A.lumbricoides*, *T.trichiura*, *S.mansoni*, *Taenia*, and impurities (9 classes, 5112 images); and the *Protozoan cysts* dataset has the categories *E.coli*, *E.histolytica*, *E.nana*, *Giardia*, *I.butschlii*, *B.hominis*, and impurities (7 classes, 9568 images). For more details, we refer to [17]. Table 1 presents the experimental set-up described in Sec. 3.1 for these 7 datasets.

3.3 Implementation details

We implemented VGG-16 in Python using Keras [7]. We load the pre-trained weights from ImageNet [19] and fine-tuned this model using the supervised S

Fig. 2. Datasets: (a) MNIST (b) CIFAR-10 and (c) H.eggs, with parasites (green box) and similar impurities (red box).

first, and subsequently labeled sets ($S \cup U$) for each chosen dataset. To guarantee convergence, we used 100 epochs with stochastic gradient descent with a linearly decaying learning-rate from 10^{-4} to zero over 100 epochs and momentum of 0.9.

3.4 Experimental Results

We use our results to address three joint questions:

Q1: How do more supervised samples improve the process? Per dataset, Tab. 2 first shows accuracy (mean, standard deviation) and κ for VGG-16 trained on S with 1%..5% supervised samples and tested on T (*baseline*). Accuracy and κ increase with the supervised sample count. For *H.larvae* and *H.eggs*, this trend cannot be seen for the given training-data fractions (3% to 4%). Still, VGG-16 performs better when the supervised training sample count increases.
Q2: What is OPFSemi's effect? Table 2 next shows mean and standard deviation for accuracy, κ, and propagation accuracy for VGG-16 trained with $S \cup U$, with U labeled by OPFSemi in the 2D projection (*DeepFA*). As for *baseline*, accuracy and κ increase with the fraction of supervised training samples. The propagation accuracy of OPFSemi is related to the number of supervised samples used to train VGG-16. The labeling performance of OPFSemi in the 2D projected space can be verified by the propagation accuracy. For the parasites dataset, this accuracy is over 80% even when VGG-16 was trained with *just* 1% of the data.
Q3: What is Looping OPFSemi's effect? Finally, Tab. 2 shows mean and standard deviation of 5 iterations of *DeepFA looping* for accuracy, Cohen's κ, and propagation accuracy for VGG-16 trained with $S \cup U$, with U labeled by OPFSemi in the 2D projection. As for *baseline* and *DeepFA*, we see an increase of accuracy and κ with the fractions (1% to 5%) of supervised training samples. We see

Table 1. Number of samples in each set S, U, and T considering $|S|$ for five sample percentages $x = 1, 2, \ldots, 5\%$ of supervised images in each dataset.

Dataset	H.larvae					H.eggs					P.cysts				
x	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%
$\lvert S \rvert$	35	70	105	140	175	17	35	53	70	88	38	77	115	154	192
$\lvert U \rvert$	2424	2389	2354	2319	2284	1220	1202	1184	1167	1149	2658	2619	2581	2542	2504
$\lvert T \rvert$	1055	1055	1055	1055	1055	531	531	531	531	531	1156	1156	1156	1156	1156
Total	3514	3514	3514	3514	3514	1768	1768	1768	1768	1768	3852	3852	3852	3852	3852
Dataset	H.eggs imp					P.cysts imp					MNIST / CIFAR-10				
x	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%	1%	2%	3%	4%	5%
$\lvert S \rvert$	51	102	153	204	255	95	191	287	382	478	50	100	150	200	250
$\lvert U \rvert$	3527	3476	3425	3374	3323	6602	6506	6410	6315	6219	3450	3400	3350	3300	3250
$\lvert T \rvert$	1534	1534	1534	1534	1534	2871	2871	2871	2871	2871	1500	1500	1500	1500	1500
Total	5112	5112	5112	5112	5112	9568	9568	9568	9568	9568	5000	5000	5000	5000	5000

Table 2. Results of the *baseline*, *DeepFA*, and *DeepFA looping* experiments, all datasets, five supervised sample percentages x, color-coded using a white-to-green colormap.

	Methods	Metrics	$x = 1\%$	$x = 2\%$	$x = 3\%$	$x = 4\%$	$x = 5\%$
H.larvae	baseline	accuracy	0.930806 ± 0.0266	0.958925 ± 0.0038	0.961138 ± 0.0066	0.960821 ± 0.0070	0.971564 ± 0.0068
		kappa	0.613432 ± 0.2334	0.824394 ± 0.0280	0.818082 ± 0.0492	0.808397 ± 0.0416	0.868460 ± 0.0361
	DeepFA	accuracy	0.962085 ± 0.0148	0.968721 ± 0.0053	0.967773 ± 0.0047	0.974092 ± 0.0112	0.977567 ± 0.0043
		kappa	0.819799 ± 0.0767	0.864692 ± 0.0251	0.854607 ± 0.0308	0.878935 ± 0.0624	0.900290 ± 0.0197
		propagation	0.961366 ± 0.0132	0.969364 ± 0.0070	0.963806 ± 0.0108	0.975193 ± 0.0135	0.979531 ± 0.0048
	DeepFA looping	accuracy	0.956398 ± 0.0202	0.974407 ± 0.0049	0.974092 ± 0.0071	0.978831 ± 0.0074	0.978515 ± 0.0031
		kappa	0.783412 ± 0.1150	0.886325 ± 0.0299	0.888211 ± 0.0298	0.904963 ± 0.0353	0.905292 ± 0.0138
		propagation	0.954860 ± 0.0153	0.976278 ± 0.0013	0.970991 ± 0.0049	0.981158 ± 0.0056	0.980887 ± 0.0015
H.eggs	baseline	accuracy	0.812932 ± 0.0599	0.925926 ± 0.0189	0.934714 ± 0.0313	0.929065 ± 0.0132	0.966101 ± 0.0032
		kappa	0.775954 ± 0.0737	0.912299 ± 0.0224	0.923002 ± 0.0367	0.916335 ± 0.0154	0.959807 ± 0.0039
	DeepFA	accuracy	0.949780 ± 0.0104	0.967985 ± 0.0082	0.962963 ± 0.0120	0.974263 ± 0.0203	0.978656 ± 0.0141
		kappa	0.940671 ± 0.0122	0.962146 ± 0.0097	0.956316 ± 0.0234	0.969631 ± 0.0239	0.974742 ± 0.0167
		propagation	0.957154 ± 0.0127	0.974400 ± 0.0055	0.971436 ± 0.0084	0.978442 ± 0.0102	0.987065 ± 0.0037
	DeepFA looping	accuracy	0.970496 ± 0.0039	0.969868 ± 0.0086	0.964846 ± 0.0237	0.974262 ± 0.0193	0.983679 ± 0.0078
		kappa	0.965144 ± 0.0046	0.964405 ± 0.0102	0.958557 ± 0.0278	0.969634 ± 0.0227	0.980694 ± 0.0093
		propagation	0.981676 ± 0.0031	0.979251 ± 0.0042	0.979520 ± 0.0140	0.983563 ± 0.0115	0.991107 ± 0.0035
P.cysts	baseline	accuracy	0.757209 ± 0.0158	0.881776 ± 0.0113	0.913783 ± 0.0079	0.914648 ± 0.0077	0.934545 ± 0.0133
		kappa	0.651933 ± 0.0232	0.837102 ± 0.0163	0.882551 ± 0.0108	0.884303 ± 0.0108	0.912294 ± 0.0177
	DeepFA	accuracy	0.847751 ± 0.0106	0.912341 ± 0.0154	0.914072 ± 0.0269	0.937428 ± 0.0018	0.950404 ± 0.0178
		kappa	0.794713 ± 0.0124	0.882537 ± 0.0213	0.885475 ± 0.0348	0.916035 ± 0.0028	0.933649 ± 0.0235
		propagation	0.832715 ± 0.0153	0.897997 ± 0.0231	0.893793 ± 0.0303	0.931627 ± 0.0042	0.937685 ± 0.0161
	DeepFA looping	accuracy	0.889562 ± 0.0030	0.925894 ± 0.0234	0.938870 ± 0.0126	0.964533 ± 0.0120	0.959919 ± 0.0088
		kappa	0.853264 ± 0.0021	0.900792 ± 0.0318	0.918280 ± 0.0163	0.952664 ± 0.0159	0.946442 ± 0.0117
		propagation	0.881800 ± 0.0130	0.925321 ± 0.0121	0.928783 ± 0.0110	0.959570 ± 0.0047	0.956726 ± 0.0044
H.eggs imp	baseline	accuracy	0.862234 ± 0.0157	0.900696 ± 0.0087	0.910256 ± 0.0167	0.931986 ± 0.0057	0.937419 ± 0.0086
		kappa	0.740861 ± 0.0287	0.815160 ± 0.0138	0.833168 ± 0.0301	0.876969 ± 0.0090	0.886231 ± 0.0159
	DeepFA	accuracy	0.928509 ± 0.0032	0.941113 ± 0.0010	0.935246 ± 0.0053	0.948501 ± 0.0111	0.956758 ± 0.0046
		kappa	0.873674 ± 0.0068	0.895627 ± 0.0014	0.885487 ± 0.0104	0.908733 ± 0.0190	0.923366 ± 0.0079
		propagation	0.913639 ± 0.0068	0.931433 ± 0.0058	0.920626 ± 0.0146	0.939631 ± 0.0129	0.945314 ± 0.0018
	DeepFA looping	accuracy	0.935680 ± 0.0014	0.949370 ± 0.0063	0.944372 ± 0.0037	0.956758 ± 0.0076	0.957844 ± 0.0046
		kappa	0.885645 ± 0.0033	0.910179 ± 0.0111	0.901216 ± 0.0078	0.922875 ± 0.0130	0.925353 ± 0.0080
		propagation	0.926681 ± 0.0022	0.939352 ± 0.0066	0.937675 ± 0.0077	0.951369 ± 0.0045	0.950252 ± 0.0024
P.cysts imp	baseline	accuracy	0.850691 ± 0.0189	0.865320 ± 0.0018	0.900383 ± 0.0072	0.903634 ± 0.0129	0.916173 ± 0.0045
		kappa	0.751667 ± 0.0280	0.776938 ± 0.0031	0.832300 ± 0.0106	0.840126 ± 0.0216	0.860640 ± 0.0076
	DeepFA	accuracy	0.852084 ± 0.0066	0.848717 ± 0.0090	0.884709 ± 0.0152	0.892140 ± 0.0144	0.916405 ± 0.0074
		kappa	0.755127 ± 0.0132	0.756045 ± 0.0138	0.811239 ± 0.0231	0.823333 ± 0.0223	0.862764 ± 0.0100
		propagation	0.845055 ± 0.0065	0.840924 ± 0.0056	0.880294 ± 0.0193	0.879200 ± 0.0181	0.901996 ± 0.0094
	DeepFA looping	accuracy	0.854522 ± 0.0013	0.860908 ± 0.0292	0.900035 ± 0.0140	0.892488 ± 0.0282	0.920933 ± 0.0032
		kappa	0.763711 ± 0.0026	0.774361 ± 0.0434	0.836986 ± 0.0217	0.825604 ± 0.0450	0.869900 ± 0.0051
		propagation	0.845652 ± 0.0072	0.853915 ± 0.0256	0.897367 ± 0.0071	0.882684 ± 0.0242	0.915335 ± 0.0106
MNIST	baseline	accuracy	0.661111 ± 0.0523	0.782222 ± 0.0269	0.870445 ± 0.0050	0.876444 ± 0.0132	0.909778 ± 0.0143
		kappa	0.623148 ± 0.0582	0.757848 ± 0.0298	0.855944 ± 0.0056	0.862635 ± 0.0147	0.899686 ± 0.0159
	DeepFA	accuracy	0.766222 ± 0.0252	0.852667 ± 0.0397	0.901556 ± 0.0170	0.899778 ± 0.0096	0.932889 ± 0.0136
		kappa	0.740028 ± 0.0280	0.836263 ± 0.0440	0.890529 ± 0.0119	0.888552 ± 0.0107	0.925403 ± 0.0151
		propagation	0.750571 ± 0.0320	0.833524 ± 0.0362	0.893524 ± 0.0064	0.895333 ± 0.0114	0.923619 ± 0.0148
	DeepFA looping	accuracy	0.815778 ± 0.0212	0.862222 ± 0.0396	0.908444 ± 0.0103	0.918000 ± 0.0077	0.936444 ± 0.0224
		kappa	0.795079 ± 0.0236	0.846885 ± 0.0439	0.898190 ± 0.0114	0.908816 ± 0.0086	0.929355 ± 0.0249
		propagation	0.806000 ± 0.0230	0.856667 ± 0.0353	0.905905 ± 0.0054	0.923238 ± 0.0068	0.939905 ± 0.0156
CIFAR10	baseline	accuracy	0.266000 ± 0.0264	0.321555 ± 0.0151	0.372889 ± 0.0341	0.417111 ± 0.0413	0.455333 ± 0.0263
		kappa	0.183681 ± 0.0301	0.245770 ± 0.0166	0.303050 ± 0.0377	0.352095 ± 0.0461	0.394558 ± 0.0291
	DeepFA	accuracy	0.228445 ± 0.0435	0.310000 ± 0.0790	0.365555 ± 0.0205	0.407778 ± 0.0136	0.424889 ± 0.0093
		kappa	0.142149 ± 0.0492	0.232875 ± 0.0880	0.295078 ± 0.0230	0.341907 ± 0.0148	0.360883 ± 0.0102
		propagation	0.219048 ± 0.0428	0.288952 ± 0.0790	0.356095 ± 0.0340	0.389619 ± 0.0190	0.421143 ± 0.0126
	DeepFA looping	accuracy	0.324000 ± 0.0418	0.375333 ± 0.0436	0.402444 ± 0.0125	0.448445 ± 0.0177	0.461555 ± 0.0211
		kappa	0.248837 ± 0.0463	0.305496 ± 0.0483	0.335927 ± 0.0138	0.387059 ± 0.0199	0.401490 ± 0.0236
		propagation	0.314286 ± 0.0277	0.369334 ± 0.0235	0.411238 ± 0.0253	0.446667 ± 0.0152	0.466857 ± 0.0301

the same for propagation accuracy, which reflects the effect of 5 iterations of OPFSemi for labeling samples in the 2D projected space. For all datasets, except CIFAR-10, propagation accuracy is over 80% even when the VGG-16 feature space was trained with only 1% of data.

4 Discussion

Added-value of DeepFA looping: Figure 3 plots the average κ for our *baseline*, *DeepFA*, and *DeepFA looping* experiments, for all 7 studied datasets. *DeepFA looping* consistently obtains the best results, except for the *P.cysts with impurity* dataset. *DeepFA* shows an improvement over the *baseline* experiment, while the first one was improved by a looping addition in the method. The gain of *DeepFA*

looping is even higher when using a low number of supervised samples – relevant when one cannot, or does not want to put effort, to supervise new ones. This gain is lower for CIFAR-10 and almost zero for *P.cysts with impurities*, as these datasets are more challenging, as their lowest κ scores show.

Effectiveness of OPFSemi labeling: The positive results for VGG-16 rely on OPFSemi propagating labels accurately. Figure 3 shows this by the average propagation accuracy of OPFSemi for *DeepFA* and *DeepFA looping*, which is high for all datasets, being worst-case 50% for CIFAR-10. For CIFAR-10, the propagation accuracy *gain* of *DeepFA looping* is higher than for the other datasets. We see also the impurity class impact for H.eggs and P.cysts in propagation accuracy (roughly 5%). Propagation accuracy is high as long as the sample count increases. The *DeepFA looping* curve is on top of *DeepFA* curve for all datasets, so the effectiveness of OPFSemi label propagation consistently improves by the looping addition.

Feature space improvement: Figure 3 showed that OPFSemi improved VGG-16's effectiveness and also accurately propagated labels to unsupervised samples. The OPFSemi labeled samples also improve the VGG-16 feature space. Figure 4 shows this space projected with t-SNE for the studied datasets. Projections are colored by (i) labels (supervised samples colored by the true-label; unsupervised samples black), and (ii) OPFSemi's confidence in classifying a sample (red=low confidence, green=high confidence) [16, 20, 23]. For all datasets, we see a clear reduction of red zones from *baseline* to *DeepFA* and a good cluster formation in the projection for same-color (*i.e.*, same-class) supervised samples (Fig. 4a). From *DeepFA* to *DeepFA looping*, there is no further reduction of red zones. Yet, different-color groups get more clustered and better separated. This is clearer for CIFAR-10, which does not show good cluster separation for *DeepFA* (Fig. 4b).

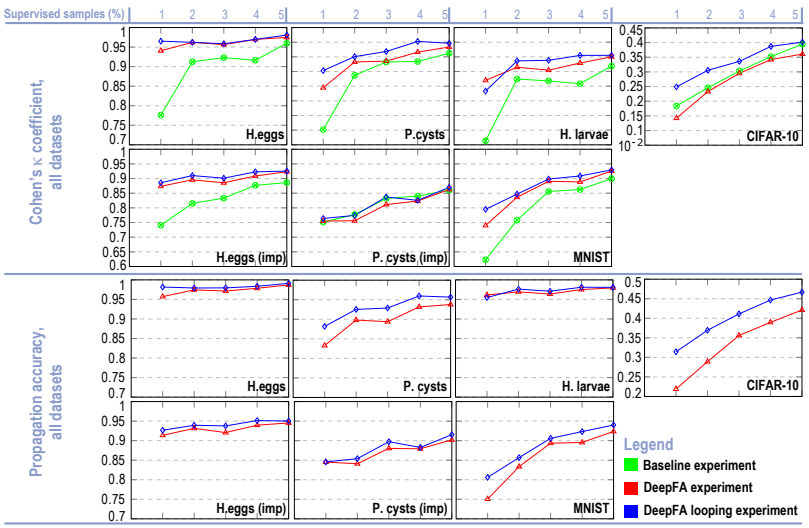

Fig. 3. Cohen's κ (top) and propagation accuracy (bottom), all datasets, for 1% to 5% supervised samples, *DeepFA* (red) *vs DeepFA looping* last iteration (blue).

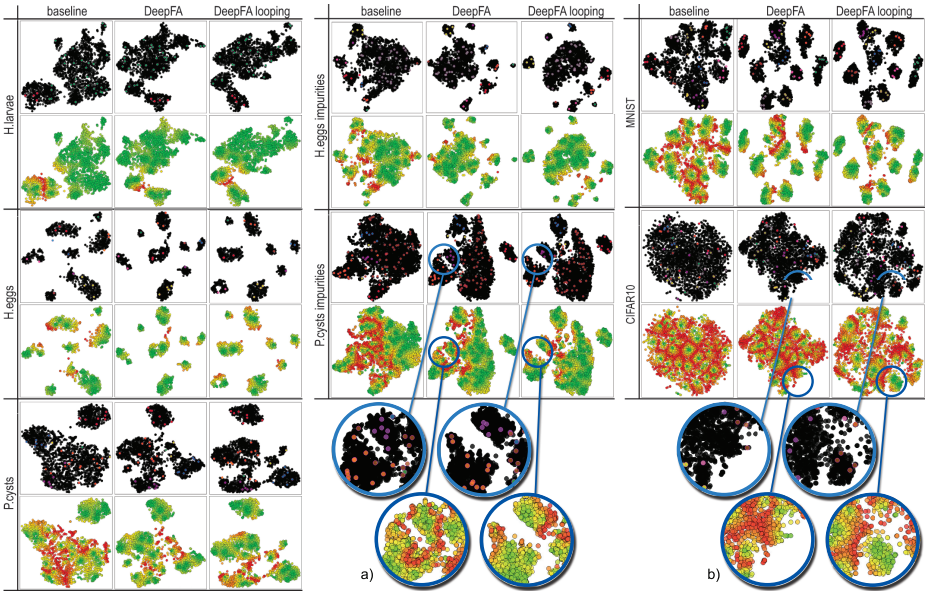

Fig. 4. 2D feature-space projections of training samples ($S \cup U$) – *baseline*, *DeepFA*, and *DeepFA looping*, 1% supervised samples. Top row per experiment: Supervised samples colored by true labels, unsupervised ones are black. Bottom row per experiment: Samples colored by OPFSemi's confidence (red=low, green=high). Insets (a,b) show details.

We conclude that OPFSemi's label propagation and the looping strategy improve VGG-16's feature space when this space is fed by those samples.

Figure 5 shows the projected space colored by class labels (unsupervised samples in black) and the OPFSemi's confidence values for 5 iterations of *DeepFA looping* on the *P. cysts* dataset with impurities, 1% supervised samples. Class separation and confidence values increase with the iterations. The red-class samples are well separated from samples of the other classes in the first iteration; some brown supervised samples get attached to them in iteration 2, creating a low-confidence region. From iteration 3 on, the problem is solved.

Limitations: Our validation used only seven datasets, one deep-learning approach (VGG-16), one semi-supervised classifier (OPFSemi), and one projection method (t-SNE). Exploring more (combinations of) such techniques would be valuable. Also, using more than 5 iterations could help understand how OPFSemi labels low-confidence regions and how it affects VGG-16's feature space.

5 Conclusion

We proposed an approach for increasing the quality of image classification and of extracted feature spaces when lacking large supervised datasets. From a few supervised samples, we create a feature space by a pre-trained VGG-16 model and use the OPFSemi technique to label unsupervised samples on a 2D t-SNE

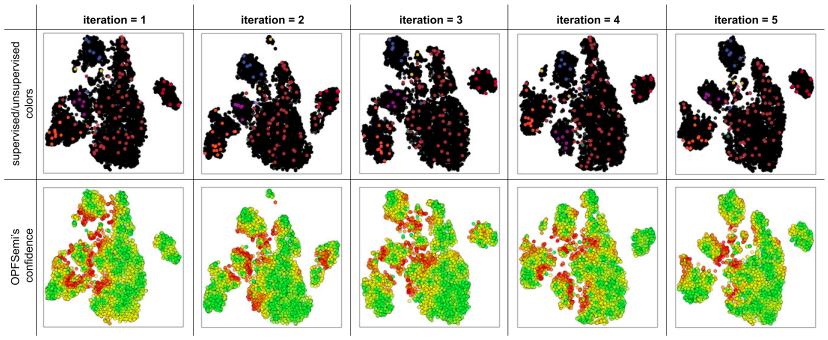

Fig. 5. 2D projections of training samples ($S \cup U$) for *DeepFA looping*, 1% supervised samples, *P. cysts impurities* dataset. Top row: Color shows class labels; unsupervised samples in black. Bottom row: Color shows OPFSemi's confidence (red=low, green=high). Class separation and confidence increase with iterations.

.

projection of that feature space. We iteratively improve labels (and the feature space) using labeled samples as input for the VGG-16 training.

OPFSemi shows low label-propagation errors and leads VGG-16 to good classification results for several tested datasets, thereby improving the VGG-16 training and hence the feature space. The small gain yielded by looping tells that OPFSemi can stagnate, its label-propagation errors lowering classification quality. To help OPFSemi during label propagation, we plan next a bootstrapping strategy to avoid propagation in low certainty regions. We also aim to include user knowledge to support OPFSemi's label propagation and to understand the VGG-16 training process and feature space generation. This co-training approach involving a bootstrapping strategy and two classifiers (OPFSemi and VGG-16) can lead to higher quality, and more explainable, deep-learning methods.

Acknowledgments

The authors are grateful to FAPESP grants #2014/12236-1, #2019/10705-8, CAPES grants with Finance Code 001, and CNPq grants #303808/2018-7.

References

1. Amorim, W., Falcão, A., Papa, J., Carvalho, M.: Improving semi-supervised learning through optimum connectivity. Patt Recogn **60**, 72–85 (2016)
2. Amorim, W., Rosa, G., Rogério, Castanho, J., Dotto, F., Rodrigues, O., Marana, A., Papa, J.: Semi-supervised learning with connectivity-driven convolutional neural networks. Pattern Recognition Letters **128**, 16 – 22 (2019)
3. Belkin, M., Matveeva, I., Niyogi, P.: Regularization and semi-supervised learning on large graphs. In: Learning Theory. pp. 624–638. Springer (2004)
4. Benato, B.C., Telea, A.C., Falcão, A.X.: Semi-supervised learning with interactive label propagation guided by feature space projections. In: Proc. SIBGRAPI. pp. 392–399 (2018)
5. Benato, B.C., Gomes, J.F., Telea, A.C., Falcão, A.X.: Semi-automatic data annotation guided by feature space projection. Pattern Recognition **109**, 107612 (2021)

6. Chapelle, O., Scholkopf, B., Zien, A.: Semi-supervised learning. IEEE TNN **20**(3), 542–542 (2009)
7. Chollet, F., et al.: Keras. https://keras.io (2015)
8. Gong, M., Yang, H., Zhang, P.: Feature learning and change feature classification based on deep learning for ternary change detection in SAR images. J Photogram Remote Sensing **129**, 212 – 225 (2017)
9. Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Label propagation for deep semi-supervised learning. In: Proc. IEEE CVPR (2019)
10. Krizhevsky, A., Nair, V., Hinton, G.: CIFAR-10 dataset www.cs.toronto.edu/ kriz/cifar.html
11. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), yann.lecun.com/exdb/mnist
12. Lee, D.H.: Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. In: Proc. ICML-WREPL (2013)
13. Li, Z., Ko, B., Choi, H.J.: Naive semi-supervised deep learning using pseudo-label. P2P Netw Appl **12**, 1–11 (2018)
14. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Proc. ECCV. pp. 740–755 (2014)
15. van der Maaten, L.: Accelerating t-SNE using tree-based algorithms. JMLR **15**(1), 3221–3245 (2014)
16. Miranda, P.A.V., Falcão, A.X.: Links between image segmentation based on optimum-path forest and minimum cut in graph. JMIV **35**(2), 128–142 (Oct 2009)
17. Peixinho, A.Z.: Learning image features by convolutional networks under supervised data constraint. Master's thesis, University of Campinas (2017)
18. Rauber, P., Falcão, A., Telea, A.: Projections as visual aids for classification system design. Information Visualization (2017)
19. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. IJCV **115**(3), 211–252 (Dec 2015)
20. Silva, A.T., Santos, J.A., Falcão, A.X., Torres, R.S., Magalhães, L.P.: Incorporating multiple distance spaces in optimum-path forest classification to improve feedback-based learning. CVIU **116**(4), 510 – 523 (2012)
21. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014), arxiv.org/abs/1409.1556
22. Sindhwani, V., Niyogi, P., Belkin, M.: Beyond the point cloud: From transductive to semi-supervised learning. In: Proc. ICML. pp. 824–831 (2005)
23. Spina, T., Miranda, P., Falcão, A.: Intelligent understanding of user interaction in image segmentation. IJPRAI **26**(02), 126–001 (2012)
24. Sun, C., Shrivastava, A., Singh, S., Gupta, A.: Revisiting unreasonable effectiveness of data in deep learning era. In: Proc. ICCV. pp. 843–852 (2017)
25. Suzuki, C., Gomes, J., Falcão, A., Shimizu, S., J.Papa: Automated diagnosis of human intestinal parasites using optical microscopy images. In: Proc. Symp. Biomedical Imaging. pp. 460–463 (April 2013)
26. Wu, H., Prasad, S.: Semi-supervised deep learning using pseudo labels for hyperspectral image classification. IEEE TIP **27**(3), 1259–1270 (2018)
27. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Proc. NIPS. pp. 3320–3328 (2014)
28. Zhu, X.: Semi-supervised learning literature survey. Comput Sci, University of Wisconsin-Madison **2** (07 2008)