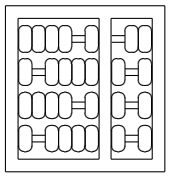

INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

Evaluating Active Learning Strategies for Image Annotation of Intestinal Parasites

Felipe Lemes Galvão *Alexandre Xavier Falcão*

Relatório Técnico - IC-PFG-16-17 - Projeto Final de Graduação

December - 2016 - Dezembro

Evaluating Active Learning Strategies for Image Annotation of Intestinal Parasites

Felipe Lemes Galvão* Alexandre Xavier Falcão†

Abstract

Manually annotating large datasets is unfeasible and, to do it automatically with a pattern classifier, it depends on the quality of a much smaller training set. Active learning techniques have been proposed to select those relevant samples from large datasets by prompting an user with label suggestions to be confirmed or corrected.

In this work we explore variations of an active learning methodology that, given an organization of the data computed beforehand and only once, allows interactive response times during the active learning process involving an expert.

We use the optimum-path forest (OPF) data clustering algorithm for the *a priori* organization and some combinations of active learning algorithms and classifiers to test the methodology. The active learning algorithms considered are the root distance-based sampling (RDS), a new variation of it that we call root path-weight-based sampling (RWS) and two additional random selection baselines. The included classifiers are the OPF-based supervised and semi-supervised learning methods, and an ensemble of logistic regression classifiers.

We tested each combination of active learning and classification algorithm against a dataset extracted from images of intestinal parasites, in which the presence of a large diverse class, namely impurities, mixed with the actual parasites poses a challenge for learning methods.

1 Introduction

Following the current advances in computing and multimedia technologies, the amount of large datasets is growing fast with the low-cost storage and acquisition of millions of samples. Information processing, organization and retrieval are important for many applications involving those datasets, and having samples annotated with labels representing their semantic content greatly facilitates such tasks. However, for large datasets, manually annotating all samples easily becomes unfeasible and more susceptible to human errors.

*Institute of Computing, University of Campinas, 13083-852, Campinas, SP
†Institute of Computing, University of Campinas, 13083-852, Campinas, SP

Therefore, research efforts have been directed at automatic and semi-automatic image annotation techniques.

The usage of active learning techniques to assist in image annotation usually consists of a loop, where a classifier is trained from supervised samples of previous iterations, the entire dataset is labeled by that classifier and organized, such that a set of the most uncertainly labeled (informative) samples are chosen for the expert's supervision through the confirmation or correction of the assigned labels. For large datasets, processing it at every iteration is costly and leads to non-interactive response times that make poor use of the expert's time.

In this work we explore variations of a recent paradigm, called DROP (Data Reduction and Organization Paradigm) [1], to solve that problem. DROP is based on a single prior organization of the dataset, followed by an active learning loop that looks at a smaller subset of the data at each iteration to provide interactive response times.

For the actual organization and active learning we use a random selection baseline, the root distance-based selection (RDS) [2] and a new variation of it called root path-weight-based selection (RWS).

The initial organization requires the usage of a data clustering algorithm for which we exclusively use the one based on optimum-path forest (OPF)[3]. Both DROP and RDS also require the usage of supervised or semi-supervised classifiers to validate and select samples, so we use one based on logistic regression, a supervised one based on OPF and a semi-supervised one also based on OPF.

We test all combinations of the aforementioned active learning and classification algorithms against a dataset related to the diagnosis of intestinal parasites to compare the performance of each combination both in the presence or absence of a large diverse class of fecal impurities.

2 Conceptual Definitions

This section briefly describes the relevant algorithms and concepts addressed in this project.

2.1 Optimum-path forest (OPF) based learning algorithms

In general, the family of OPF learning algorithms builds a graph where each training sample is a node, the arcs are defined by some adjacency relation and the arc weights are given by a distance function between sample feature vectors. Some method is used to select roots to build an optimum path forest using the IFT algorithm [4] (e.g. maxima of a *pdf* in data clustering and nearby samples from distinct classes in supervised classification) and labels are propagated from those roots.

The remainder of this section explains how exactly it can be used in each form of learning.

2.1.1 OPF for supervised learning

We briefly describe the supervised classification based on OPF presented in [5].

Given a training supervised dataset $\mathcal{Z}$ and a subset of prototypes $S \subset \mathcal{Z}$ that best represent each class, we build a complete graph with all samples $s \in \mathcal{Z}$. Each arc between samples s and t is weighted by some distance function $d(s,t)$ (e.g. Euclidean).

To run the IFT algorithm with the function f_{max}, we initialize each prototype sample as root (i.e. $f_{max}(\langle s \rangle) = 0$ for $s \in S$, and $f_{max}(\langle s \rangle) = +\infty$ for $s \in \mathcal{Z} \backslash S$) so that we obtain a partition of the graph as an optmimum-path forest rooted at the prototypes. Each training sample $s \in \mathcal{Z}$ gets a cost $C_1(s)$ based on an optimal path from a root to it so that, considering we are using f_{max}, $C_1(s)$ is the highest weight among arcs of the path. This optimum-path forest is our trained classifier.

To classify a new sample $t \notin \mathcal{Z}$, we extend paths from the optimum-path forest obtained during training using the cost function

$$C_2(t) = \min_{\forall s \in \mathcal{Z}} \{\max\{C_1(s), d(s,t)\}\} \tag{1}$$

to find $s^\star \in \mathcal{Z}$ that minimizes $C_2(t)$. We assign to t the label of $s^\star$.

2.1.2 OPF for semi-supervised learning

In this work, we follow the training method described in [6], which is an extension of the OPF-based supervised algorithm of the previous section.

In the semi-supervised scenario, during training we have two datasets $\mathcal{Z}_l \cup \mathcal{Z}_u = \mathcal{Z}$ where $\mathcal{Z}_l$ and $\mathcal{Z}_u$ are sets of labeled and unlabeled samples, respectively.

We start our training by building a complete graph G_1 from the entire training set $\mathcal{Z}$ with arcs weighted by some distance function $d(s,t)$. From this graph we compute a minimum-spanning tree (MST) (e.g. by Prim's algorithm) which will be our new graph G_{MST}.

Using all samples $p \in \mathcal{Z}_l$ as prototypes , we proceed to build an optimum-path forest from G_{MST} similar to the one described in the supervised algorithm. In this case we have $S = \mathcal{Z}_l$ as our set of roots and we limit the adjacency relation to the MST topology. Again, every sample $s \in \mathcal{Z}$ gets a cost $C_1(s)$ from the optimal path obtained with the f_{max} path-cost function.

Likewise, to classify a new sample $t \notin \mathcal{Z}$, we use equation 1 so that the sample $s^\star \in \mathcal{Z}$ which minimizes $C_2(t)$ dictates the assigned label.

2.1.3 OPF for data clustering

The data clustering based on OPF we describe is based on [3].

Given a dataset $\mathcal{Z}$ of unsupervised samples. Let $d(s,t)$ be a distance relation of two samples $s, t \in \mathcal{Z}$ in the feature space and $\mathcal{A}(s)$ be an adjacency relation defined by the k nearest neighbors of s.

We define a graph G with each sample of $\mathcal{Z}$ as node and we get the arcs from $\mathcal{A}(s)$ with weight given by $d(s,t)$. Additionally each node is weighted by the the density value

$$\rho(s) = \frac{1}{\sqrt{2\pi\sigma^2}|\mathcal{A}(s)|} \sum_{\forall t \in \mathcal{A}(s)} \exp\left(\frac{-d^2(s,t)}{2\sigma^2}\right) \tag{2}$$

where $|\mathcal{A}| = k$, $\sigma = \frac{d_f}{3}$ and d_f is the maximum arc weight in G.

We then initialize the nodes of G to obtain an OPF in the following manner. Each maxima s of the estimated *pdf* from $\rho(s)$ is defined as a prototype initialized with path cost $\rho(s)$, and all other samples t are initialized with path cost $\rho(t) - \delta$ where $\delta = \min_{\forall(s,t)\in\mathcal{A}|\rho(t)\neq\rho(s)} |\rho(t) - \rho(s)|$. Then we run an f_{min} IFT so that the prototypes compete between each other to build their respective clusters out of the remaining samples, forming an optimal-path forest rooted on *pdf* maxima.

2.2 Ensemble of logistic regression classifiers

We construct our ensemble of logistic regression classifiers based on [7]. To model the posterior probabilities of each class in this linear model, we use the *softmax function*, also known as the *normalized exponential*, given by

$$p(\mathcal{C}_k|\mathbf{x}_n) = y_{nk} = \frac{e^{a_{nk}}}{\sum_{j=1}^{|\mathcal{C}|} e^{a_{nj}}} \tag{3}$$

where $\mathcal{C}_k$ is the dependent variable, $\mathbf{x}_n$ is the independent variable and a_{nk} corresponds to the activation of $\mathbf{x}_n$ for $\mathcal{C}_k$

$$a_{nk} = \tilde{\mathbf{x}}_n \cdot \tilde{\theta}_k \tag{4}$$

so that θ_k is the vector of linear weights of our model corresponding to $\mathcal{C}_k$. The tilde notation that we will only use in equation 4 indicates that we are adding a dummy variable 1 to $\mathbf{x}_n$ and a corresponding bias weight to θ_k to simplify the expression $\mathbf{x}_n \cdot \theta_k + b_k$.

Consider we have a training dataset $\mathcal{Z}$ with a set of observations $\{(x_1, t_1), \ldots, (x_{|\mathcal{Z}|}, t_{|\mathcal{Z}|})\}$. Each pair (x_n, t_n) belongs to a class $\mathcal{C}_k$ and is formed by the feature vector $\mathbf{x}_n$ of size m and the binary target vector $\mathbf{t}_n$ of size $|\mathcal{C}|$, filled with zeros except for the element k. With $\mathcal{Z}$ we can write down the likelihood function

$$p(\mathbf{T}|\theta) = \prod_{n=1}^{|\mathcal{Z}|} \prod_{k=1}^{|\mathcal{C}|} p(\mathcal{C}_k|x_n)^{t_{nk}}$$

where $\mathbf{T}$ in an $|\mathcal{Z}| \times |\mathcal{C}|$ matrix formed by the target vectors $\mathbf{t}_n$, and θ is the $|\mathcal{C}| \times m$ matrix formed by the weight vectors θ_k. Our goal is to find the set of linear parameters of θ that maximizes $p(\mathbf{T}|\theta)$. To do so it is easier to minimize its negative log given by

$$l(\theta) = -\ln p(\mathbf{T}|\theta) = -\sum_{n=1}^{|\mathcal{Z}|} \sum_{k=1}^{|\mathcal{C}|} t_{nk} \ln y_{nk} \tag{5}$$

so that l is the loss function counterpart to the likelihood. This form is also known as *cross-entropy* error function. We could theoretically use other loss functions like the 0-1 loss, but equation 5 is continuous and preferable to work with.

Knowing that $l(\theta)$ is convex over the linear weights in θ and that it is strictly higher than zero (because $y_{nk} \in [0, 1]$), from calculus we know its global minimum is found by

taking the gradient of each weight and equating it to zero. For each weight θ_{ik} we can deduce analytically that

$$\nabla_{\theta_{ki}} l(\theta) = \sum_{n=1}^{|\mathcal{Z}|} x_{ni}(y_{nk} - t_{nk}) \tag{6}$$

As there is no closed form solution for $\nabla_\theta l(\theta) = 0$, we have to estimate θ by optimization. For this we used mini-batch gradient descent with the momentum method [8]. Given an initialization of θ, we repeat

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta l(\theta) \tag{7}$$

$$\theta = \theta - v_t \tag{8}$$

until some stopping condition is met. γ is the momentum parameter, η is the learning rate and $\nabla_\theta l(\theta)$ is calculated explicitly from equation 6 using a small subset o $\mathcal{Z}$ in each iteration.

With the obtained θ, we can classify new samples by evaluating equation 3 for each class and assigning the label associated with the highest calculated probability.

2.3 Active Learning

2.3.1 Data Reduction and Organization Paradigm (DROP)

The DROP methodology [1] was proposed to provide interactive response time to the users' actions during active learning.

To do so, given we have a separated supervised validation dataset $\mathcal{Z}_v$, we have our training dataset $\mathcal{Z} = \mathcal{Z}_u \cup \mathcal{Z}_l$, $\mathcal{Z}_u \cap \mathcal{Z}_l = \emptyset$, where $\mathcal{Z}_u$ contains unlabeled samples and $\mathcal{Z}_l$ is formed by samples already supervised by an expert.

Starting with $\mathcal{Z} = \mathcal{Z}_u$ and $\mathcal{Z}_l = \emptyset$, we proceed with some form of data organization (e.g. data clustering) which will give the structure to support an active learning loop that will follow.

At each iteration, using the organization structure, the algorithm will only look at a subset of the data to select and classify samples that will be shown to an expert, who confirms or corrects the automatically assigned labels. Those supervised samples are then transfered from $\mathcal{Z}_u$ to $\mathcal{Z}_l$ and the samples in $\mathcal{Z}_l$ are used to train a classifier that will be tested against $\mathcal{Z}_v$ for evaluation. The loop continues until the user decides to stop or some desirable classification accuracy against the validation set is achieved.

2.3.2 Root distance-based sampling (RDS)

Following the MST-BE algorithm from [1], the RDS active learning algorithm [2] was proposed to give better results in datasets with an abundant diverse class.

Following DROP described in the previous section, we first organize $\mathcal{Z}$ with a data clustering algorithm and then, for each cluster, we build a list of samples sorted by their distance to the root.

In the first iteration of the active learning loop, we want to find representative samples of our problem so we choose the root of each cluster to show to an expert. We then train a classifier with those samples and in the posterior iterations we do the following procedure to select samples.

Looking one cluster at a time, we first traverse the root distance list in forward order deciding to select or not a sample based on the following criteria: if our classifier assigns a label to the sample different from the root label, then we should select it. If a sample is selected in this manner, we start again in the next cluster. If the entire cluster is traversed without selecting any sample, we select one from the end of the root distance list and then proceed to next cluster.

The selected samples are shown to an expert and the process repeats in usual DROP fashion.

2.3.3 Root path-weight-based sampling (RWS)

In the RDS algorithm, when a data clustering method that does not assume a spheric distribution of data is used (e.g. data clustering by OPF), there is a mismatch in using the distance to the roots directly. Information of the actual cluster morphology is lost, which might impact performance negatively.

In this work we present a variation of RDS to test that hypothesis. It is called RWS and aims to improve RDS by leveraging the optimal-path forest information we obtain in the organization phase when using data clustering by OPF.

Instead of organizing samples from each cluster by their distance to the root, we sort them using the path cost $C_1(s)$ given by our data clustering by OPF described in section 2.1.3. Other than this ordering difference, RWS active learning sample selection works identically to RDS which was described in the previous section.

3 Experiments

3.1 Setup

This section describes the experimental setup associated with this project.

3.1.1 OPF

The supervised and data clustering OPF algorithms used the implementation available in LibOPF [9]. The semi-supervised version was implemented based on the same library.

We set the k_{max} parameter of the data clustering by OPF small enough so that all classes of interest are represented by the root of a cluster.

For the semi-supervised OPF algorithm, we pick for the unlabeled training set $\mathcal{Z}_u$ a random selection of samples from all available unlabeled ones until $|\mathcal{Z}_u| = 2|\mathcal{Z}_l|$ or we exhaust the unlabeled set. This is to follow the DROP methodology of not processing the entire dataset at each iteration to achieve interactive times, otherwise we could use all samples to build a stronger classifier.

3.1.2 Multi-class Logistic Regression

In this work, the multi-class logistic regression was implemented from scratch following the description in 3.1.2.

During the training phase with mini-batch stochastic gradient descent algorithm, for each epoch we keep picking a random batch of 50 samples from the entire training dataset without reposition until all training samples are used. We set a maximum of 1000 epochs to limit training time.

In the optimization step given by equation 7, the momentum parameter γ was fixed to 0.9 and the learning rate parameter η was initialized to 0.01. η was then dynamically reduced by an order of magnitude whenever the relative loss reduction between epochs $l(\theta_{epoch_t})/l(\theta_{epoch_{t-1}})$ was higher than 0.995, indicating we should refine our search. If η drops below a threshold 10^{-5}, it is used as an early stop signal.

Note that no regularization (e.g. L1, L2) was included, so the implemented model can suffer from overfitting.

3.1.3 Active learning

We implemented a modular application to test the DROP methodology where you can plug any of the implemented active learning selection and classification algorithms. The data clustering for data organization is restricted to the OPF-based one.

We list each of the implemented modules by an abbreviation of the corresponding algorithm name, which will be used as a shorthand in the results section. For the active learning selection we have:

- *Rand* - Random selection over entire dataset;
- *Rand-roots* - Random selection but selecting cluster roots in the first iteration;
- *RDS* - RDS as described in 2.3.2;
- *RWS* - RWS as described in 2.3.3.

For classification:

- *OPFSup* - Supervised classification by OPF as described in 2.1.1;
- *OPFSemi* - Semi-supervised classification by OPF as described in 2.1.2;
- *LogReg* - Multiclass logistic regression classification as described in 2.2.

3.2 Dataset

We evaluate our implementation against a dataset obtained through an automated system for the diagnosis of intestinal parasites [10]. It consists of 5948 objects, 1944 being a mix of 15 species of protozoa and helminths, and the other 4004 being fecal impurities. Each object contains 260 features and is labeled as part of one of the 16 aforementioned groups.

To compare the performance of our methods in the presence or absence of impurities (the large diverse class), we use the dataset in two forms:

- The subset including only samples from the 1944 parasites (dataset d_1);
- The entire group of 5948 samples containing parasites and impurities (dataset d_2).

3.3 Experimental methodology

For a given dataset $\mathcal{Z}_t$, we split it into $\mathcal{Z} \cup \mathcal{Z}_v = \mathcal{Z}_t$ where $\mathcal{Z}$ and $\mathcal{Z}_v$ have the same meaning of section 2.3.1. We fixed the split so that $|\mathcal{Z}| = 0.8|\mathcal{Z}_t|$ and $\mathcal{Z}_v = \mathcal{Z}_t \backslash \mathcal{Z}$, built by a random selection of samples from $\mathcal{Z}_t$.

Using that split, we test our active learning application described in 3.1.3 with each combination of active learning selection strategy and classification algorithm. We set it to a maximum of 50 active learning loops, each loop selecting $2|\mathcal{C}|$ samples to be annotated by an oracle (the known true label from the dataset), except for the first one where we select a number of samples equal the number of clusters obtained through data clustering by OPF.

The application collects the following data for each active learning iteration:

- Accuracy of the trained classifier against $\mathcal{Z}_v$;
- Percentage of samples the oracle had to correct from all shown samples;
- Time spent during the selection and re-training of the classifier;
- Number of missing classes (i.e. unseen by the oracle).

We repeat the procedure 20 times for each of the two datasets and aggregate the results with the mean and standard deviation of the collected data of any given combination of dataset, active learning selection and classifier.

4 Results

Figures 1 to 6 show the comparison between active learning methods for each fixed pair of dataset and classifier.

For both OPF classifiers, the advantage of both RDS and RWS over the random baseline is evident. The results with RDS and *OPFSup* coincide with [2], including the scenario with impurities, so that we have a validation for our implementation of the RDS algorithm.

Comparing RDS against RWS we see that both methods are nearly equivalent, except for figure 4 where RWS shows a substantial advantage in the first 20 active learning iterations.

The tests using *LogReg* show this classifier weakness when dealing with the small number of samples during the earlier iterations. The class separation hyperplanes generated by the *LogReg* classifier go out of their way to accommodate the uneven distribution of a few samples from the same class, to the point that its generalization capacity is compromised. This behavior is very evident in figure 5 where the performance drops off sharply after the first iteration, when adding more samples to the initial root set.

As the behavior of *LogReg* is clearly worse than the other two OPF classifiers, we proceed to compare *OPFSup* and *OPFSemi* separately. In figures 7 and 8 we compare the

two classifiers using RDS in both datasets. The choice of active learning method for this analysis is arbitrary as the same pattern is observed when replacing it.

On dataset d_1 we observe that the usage of a random subset of our pool of unsupervised samples with the *OPFSemi* algorithm is a good choice to improve classification results given by the supervised learning of *OPFSup*, which is in accordance to [6]. However, on dataset d_2, we observe a new pattern: the presence of a large impurity class makes the performance o *OPFSemi* substantially worse when using a naive random selection of unlabeled samples.

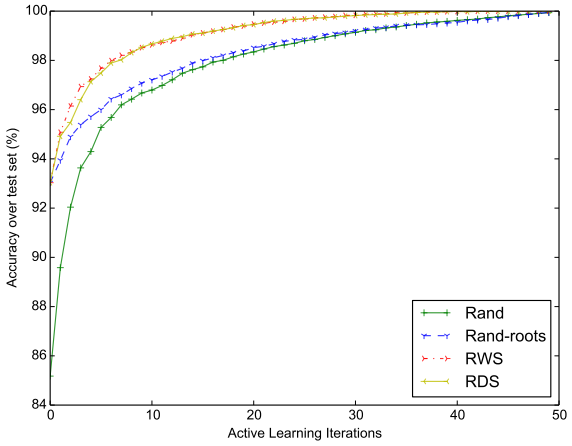

Figure 1: Comparison of active learning methods using *OPFSup* on dataset d_1

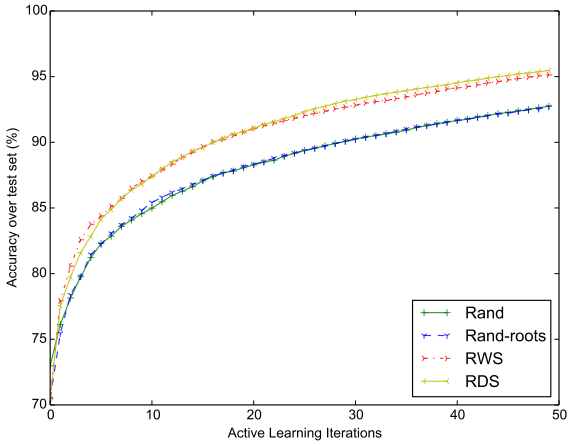

Figure 2: Comparison of active learning methods using *OPFSup* on dataset d_2

5 Discussion

We implemented a general application for the DROP [1] methodology and validated it with a few combinations of active learning and classification algorithms. We compared each

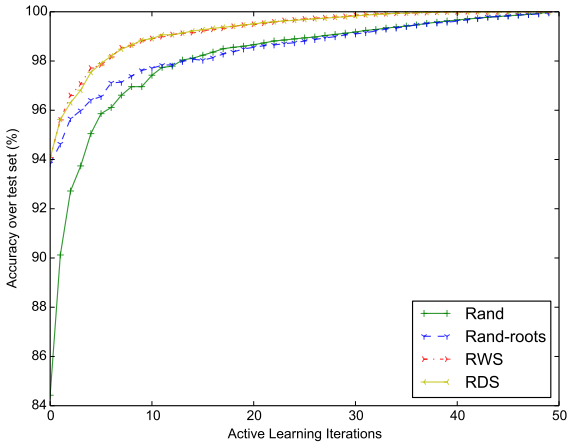

Figure 3: Comparison of active learning methods using *OPFSemi* on dataset d_1

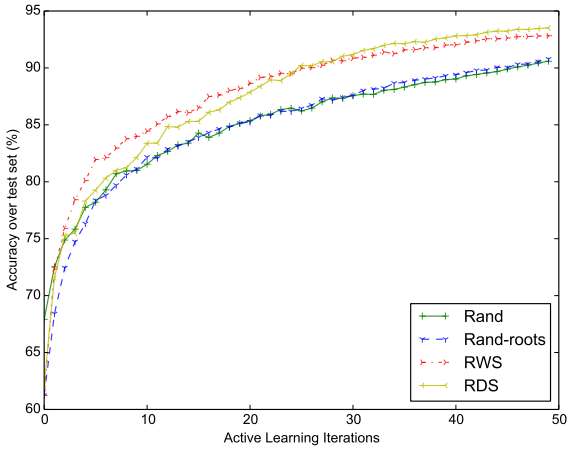

Figure 4: Comparison of active learning methods using *OPFSemi* on dataset d_2

combination using a dataset extracted from images of intestinal parasites and observed the good performance of the root distance-based sampling (RDS) [2] active learning algorithm even in the presence of a diverse class formed by impurities.

We also presented a variation called root path-weight-based sampling (RWS) that uses the path cost of an optimum-path forest (OPF) to organize samples. We expected that, combined with initial data organization based on data clustering by OPF [3], RWS would improve the results of RDS. The observed RWS performance was generally similar to RDS, but in one of our tests we have shown that the new method selection is more robust to the problem induced by the introduction of unlabeled samples from a diverse class in semi-supervised learning, which we discuss in more detail when comparing the performance of our supervised and semi-supervised classifiers.

Regardless, RDS and RWS are fundamentally very similar and share some strengths

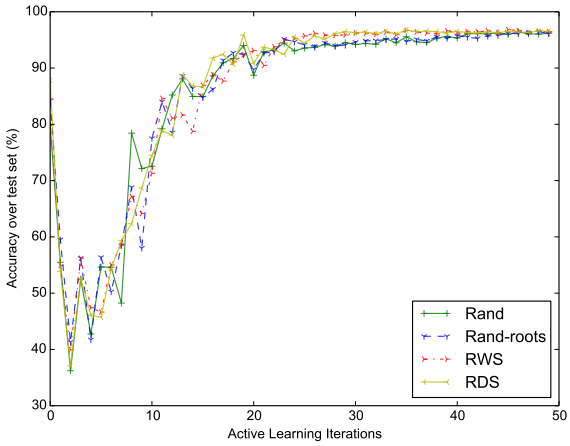

Figure 5: Comparison of active learning methods using *LogReg* on dataset d_1

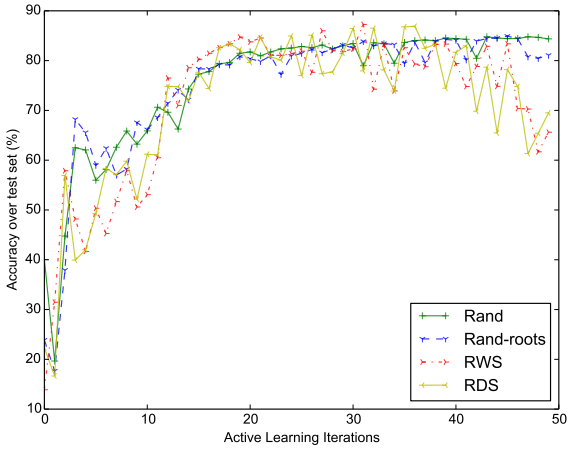

Figure 6: Comparison of active learning methods using *LogReg* on dataset d_2

and weaknesses that deserve consideration in developing future improved active learning techniques. One of the major weaknesses of both methods is that neither leverages *a priori* information of examined clusters to prioritize the ones more likely to give informative samples. That includes information such as the size of the cluster and which class it is associated with, according to its root. A choice of which clusters deserve more attention could then be guided by information such as misclassification rates against the validation set.

Given the abundance of unlabeled samples in this active learning scenario, the usage of semi-supervised classifiers to leverage them is also of interest for improved results. We compared the performance of an OPF-based supervised classifier and its semi-supervised counterpart with that goal in mind. Against our dataset without the large diverse class of fecal impurities, we obtained results similar to the existing literature [6]. However, with

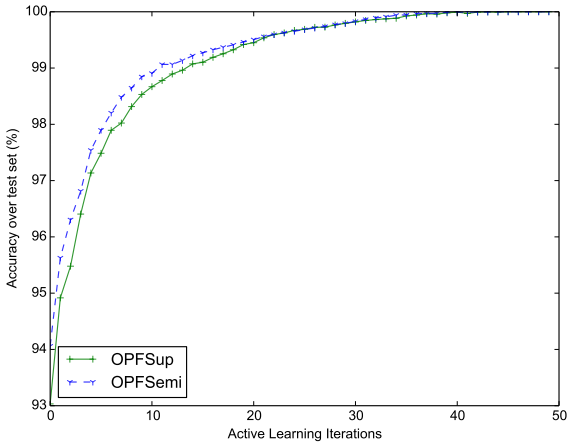

Figure 7: Comparison of OPF classifiers using *RDS* on dataset d_1

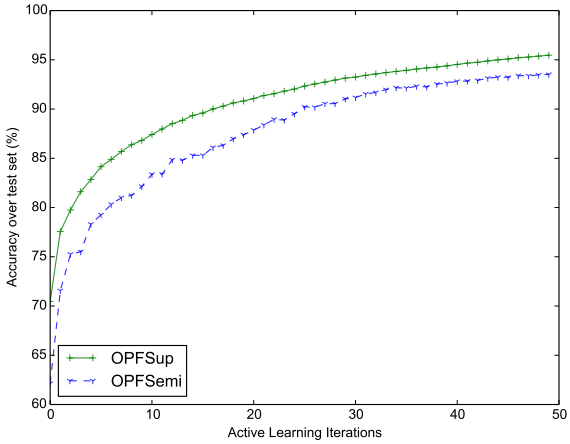

Figure 8: Comparison of OPF classifiers using *RDS* on dataset d_2

the inclusion of those impurities, we made a novel observation that the usage of random unsupervised samples actually worsens the performance of our semi-supervised algorithm.

This result motivates that, in the presence of a large diverse class, we need some form of selection intelligence when choosing unsupervised samples for semi-supervised learning methods. Our future work will try to analyze implementations of such a selection intelligence, which can be guided by the degree of certainty we have about the classification of those unsupervised samples, and by ensuring a balanced class diversity during selection. The latter can be done using the same organization we use in active learning, given we know the root labels in advance, we select samples that our classifier assigns the same label of the root.

Furthermore, regardless of our choice of active learning and classifier algorithms, the substantial difference in performance observed when introducing a large diverse class sug-

gests that future work should pay special attention to dealing with that diverse class. In the context of our parasitology images, we can first consider the binary problem of separating impurity and non-impurity objects, and then explore descriptors and classifiers better suited to this scenario.

References

[1] Saito, P.T., de Rezende, P.J., Falcao, A.X., Suzuki, C.T., Gomes, J.F.: An active learning paradigm based on a priori data reduction and organization. Expert Systems with Applications **41**(14) (2014) 6086–6097

[2] Saito, P.T., Suzuki, C.T., Gomes, J.F., de Rezende, P.J., Falcão, A.X.: Robust active learning for the diagnosis of parasites. Pattern Recognition **48**(11) (2015) 3572–3583

[3] Rocha, L.M., Cappabianco, F.A., Falcão, A.X.: Data clustering as an optimum-path forest problem with applications in image analysis. International Journal of Imaging Systems and Technology **19**(2) (2009) 50–68

[4] Falcão, A.X., Stolfi, J., de Alencar Lotufo, R.: The image foresting transform: Theory, algorithms, and applications. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**(1) (2004) 19–29

[5] Papa, J.P., Falcao, A.X., Suzuki, C.T.: Supervised pattern classification based on optimum-path forest. International Journal of Imaging Systems and Technology **19**(2) (2009) 120–131

[6] Amorim, W.P., Falcão, A.X., Papa, J.P., Carvalho, M.H.: Improving semi-supervised learning through optimum connectivity. Pattern Recognition **60** (2016) 72–85

[7] Bishop, C.M.: Pattern recognition. Machine Learning **128** (2006)

[8] Qian, N.: On the momentum term in gradient descent learning algorithms. Neural networks **12**(1) (1999) 145–151

[9] LibOPF: (visited in 12/16/2016) `http://www.ic.unicamp.br/~afalcao/libopf/`.

[10] Suzuki, C.T., Gomes, J.F., Falcao, A.X., Papa, J.P., Hoshino-Shimizu, S.: Automatic segmentation and classification of human intestinal parasites from microscopy images. IEEE Transactions on Biomedical Engineering **60**(3) (2013) 803–812