Online and co-regularized algorithms for large scale learning

Tom de Ruijter
s3016269

August 10, 2012- Radboud Universiteit Nijmegen

Abstract

In this work I address the issue of large scale learning in an online setting. To tackle it, I introduce a novel algorithm that enables semi-supervised learning in an online fashion. By combining state-of-the-art online methods such as Pegasos [3] with the multi-view co-regularization framework, I achieve significantly better performance on regression and binary classification tasks. This shows that incorporation of unlabeled data is still practical even in large scale and online settings. Evaluation is done on several publicly available datasets from the UCI and LibSVM repositories. To evaluate results in a practical setting, I also consider a difficult natural language set from the BioInfer corpus [18]. In this setting the introduced algorithm outperforms current state-of-the-art methods for online learning. The main contents of this work will also be presented at the conference Discovery Science 2012 and contained in Lectures of Computer Science [21].

Contents

1 Introduction - The large scale challenge

The field of knowledge discovery, also referred to as Machine Learning, is based around modeling of problems by computers in an autonomous manner. In most cases these models are not tangible or clear for humans to understand, but have a more abstract representation. By defining algorithms that are able to generalize problems through trial and error, models are created through the notion of learning. As with humans, different flavors of learning exist and different philosophies on how to approach the 'material' do too. Methods that learn based on problem instances labeled with its solution belong to the domain of supervised methods while those that do not exclusively are non-supervised or semi-supervised. Most common learning tasks for computers are binary *classification* and *regression*. Binary classification tasks are Yes/No decisions: "does this blood sample contain a tolerable amount of substance X?". Regression tasks are similar, but the answer required is continuous instead of discrete. "What is the age of this animal, based on its physical appearance?". As becomes clear from the nature of these examples, one would want that methods are domain adaptable to provide solutions for a class of relevant problems instead of just one specific problem.

Over the last decades we have seen that as technology improves and data storage becomes cheaper, the amount of data generated also increases. Data on which statistics can be applied to discover underlying mechanisms and distributions - to discover knowledge - and to gain understanding of e.g. genetic diseases, consumer behavior or celestial bodies. As acquiring data becomes cheaper, one can assume all this data can be exploited to the fullest and so the amount of knowledge discovered should also increase. Consequently, this demands more from contemporary methods. More constrained is the case in which knowledge needs to be available in real-time, e.g. robotics or rocket science. To allow for functioning in the real world, algorithms become increasingly complex and additional sensory data is required to cope with agent or environment interactions. Naturally, the same argument applies for user agents in the world wide web.

This calls for methods that are able to process large bodies of information linearly or even sub-linearly in the amount of data available and can operate in a real-time environment. Online meaning that the model state is updatable as new information becomes available. In recent years, several different methods have been proposed that are able to process large bodies of information and are able to operate in a real-time environment, such as [1–5] or see [6] for an overview. All these methods are used in a supervised setting. Supervised meaning that each problem instance is paired with its correct answer, or label. However other sources of data are available as well, including unlabeled information. We want methods to be better, so we feed them as much data as possible. Algorithms capable of handling labeled as well as unlabeled problem instances belong to the field of semi-supervised methods. One of the semi-supervised methods we will look into more closely is the co-regularization framework proposed in [7]. It allows for unlabeled data as well as multiple representations of the same data to be added to the model.

It seems tempting to believe that inclusion of unlabeled data leads to performance synergy. It has been shown that a co-regression setting is beneficial for small datasets while remaining domain adaptable [8]. The downside is that these methods fail as data size increases. In this work I propose a novel online co-regularized learning method for solving large scale classification and regression problems. The goal of this work is to show that large scale problems also benefit from the inclusion of unlabeled data and that co-regression

functions in an online setting.

In the following section some background theory on relevant methods is explained. The next chapter explains the novel co-regularized algorithm with a short discussion. Then the algorithm is evaluated in a practical setting and finally results are discussed.

2 Background

2.1 Learning models

In practice, which model and learning process to choose is not always clear and largely depends on various properties of the problem at hand. Intuitively we want a black box that takes in data and gives us back a prediction and also how confident this prediction is. It helps to approach the situation from a more formal perspective. A supervised learning setting can be described as follows.

Consider a set $S = (X, Y)$ originating from a set $\{(\mathbf{x}_i, y_i)\}_{i=1}^m$ of data points where $X = (\mathbf{x}_1, \ldots, \mathbf{x}_m)^t \in \mathcal{X}^m$ and $Y = (y_1, \ldots, y_m)^t \in \mathbb{R}^m$. Here, X is our set of problem instances and Y are their corresponding labels. The goal is to identify a function $f \in \mathcal{H}$ that maps problem instances to their solution, where $\mathcal{H}$ is the function space containing all mappings from X to Y.

This mapping f is our black box and in practical situations it is hard to find a perfect mapping due to noise or non-linearity of data. Usually one is also satisfied with results that have a small deviation from the real value, e.g. wether it rains $1.0mm^2$ or $1.2\ mm^2$. A more realistic definition is the minimization of prediction differences through means of a loss function where large deviations are punished more than smaller ones. We can do so by defining a loss function $\mathcal{L}$ that penalizes prediction deviations.

Also we do not want to overfit on any specific problem instances, i.e. if the model were a line, we would want it to be smooth and not edged. We can do so by incorporating the model size as a measure of complexity in our learning problem.

Formally, we can combine the loss function and model complexity in the form of an objective function

$$J(f, S) = \mathcal{L}(f, S) + \lambda \|f\|_{\mathcal{H}}^2. \tag{1}$$

We see both the loss function and the model size incorporated. The second part is called a regularization component that prevents over-fitting of the model, in this case by using the L_2 norm in the respective function space. The parameter $\lambda \in \mathbb{R}^+$ scales this effect. Minimizing this expression is an approximation of the ideal problem solution. The type of loss function used defines the method, i.e. we obtain support vector machines [9] by choosing a hinge loss function and we obtain regularized least-squares (RLS) [10] by choosing a squared loss function. It is problem dependent which loss function works best.

In specific function spaces $\mathcal{H}$ it is sometimes possible to find a more direct notion of f. See for instance [**?**]. However such resulting expressions often contain application of matrix inverses, which are intractable to compute for larger sets.

2.2 Online Algorithms

Online algorithms are amongst the most popular approaches for large scale learning. In real-time systems, it is preferable to process data as soon as it becomes available in order to update some model state. These methods often have a linear or sub-linear time complexity in the amount of data points, making them very much fit for large scale learning. Methods such as PEGASOS [3], LASVM [1] and GURLS [11] have been successfully applied to a wide range of large scale problems leading to state-of-the-art generalization performance. Slightly overloading our notation, we reshape equation (1) to an online setting:

$$J(\mathbf{w}, S) = \sum_{i=1}^{m} \mathcal{L}(\mathbf{x}_i, y_i; \mathbf{w}) + \lambda \|\mathbf{w}\|^2, \tag{2}$$

where $\mathbf{w}$ is the model parameter to be learned.

A popular approach to tackle large scale problems is by using efficient approximation techniques such as stochastic gradient descent [6]. It is based on the idea of descending towards the closest error minimum in the model error landscape. By taking the gradient towards the model parameters, one approximates the best parameters. What makes this approach so efficient, is the amount of samples considered in the gradient calculation per time step. Formally the main part of this procedure is

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla_{\mathbf{w}} J(\mathbf{w}, S_t), \tag{3}$$

where η_t is a scaling parameter - usually referred to as learning rate - and J is any objective function to be minimized. S_t is the subset of samples used at a certain iteration over which the gradient is taken. In a standard gradient setting $|S_t| = m$, where a purely stochastic setting corresponds to $|S_t| = 1$.

Although this speed comes at a price - local minima and parameter approximation - it has been shown that stochastic gradient descent leads to state-of-the-art generalization performance [3, 12]. Given very few samples and iterations, the model parameters often converge very rapidly to an optimal region.

2.3 Co-regression

Intuitively, the most important reason to include other sources of data is because people want better performance for free. Unlabeled data is cheap and often much easier to obtain. Opposed to unlabeled data, labeling often requires human annotation, domain experts and special equipment. This makes labeling time consuming and expensive.

Another class of algorithms that is gaining more attention in recent years is the class of semi-supervised algorithms. This means that, apart from including labeled data, these models are also able to incorporate unlabeled data into their learning process. Humans also apply semi-supervised learning in life. Having seen some of pictures of cars and trucks it is easy to discern all cars from all trucks, even though one has not seen all cars and trucks in advance. Without a label, the data structure of a single example still incorporates some spatial structure when represented in its feature space, e.g. figure 1. One hopes that similar data points are somehow grouped together.

The method to incorporate unlabeled information introduced in [7] is known as the *co-regularization* framework. This framework is based on the idea of learning two models

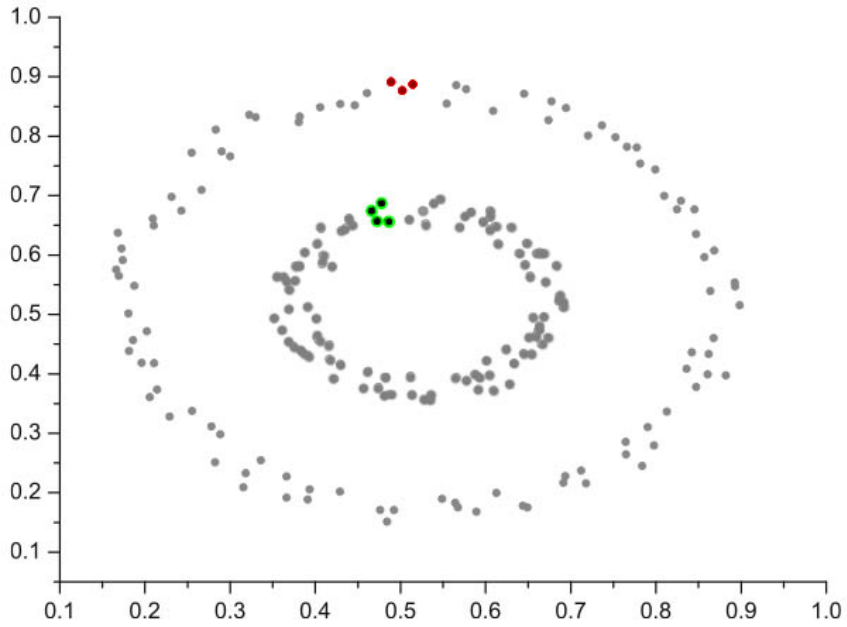

Figure 1: The artificial circles dataset. Even though only a few points from each circle are labeled, the structure become clear through the use of unlabeled information.

on the same problem and forcing those two to agree on any decisions made. Here the advantage is you can use different representations, or views, of the same instance. One for each separate model. A prediction is made by somehow aggregating model results, possibly with a weighted average.

A classical example of how this arises naturally can be found in the example of ranking web documents based on a search query. One view contains the bag of keywords found in the page contents, while the other contains the link structure to other web documents [13]. Both represent the document in some way.

The co-regularized framework can be described formally, similar to how a single supervised method is defined. Given a set $S = (X, Y)$ we consider M different hypotheses spaces $\mathcal{H}_1, \ldots, \mathcal{H}_M$ or views over unique subsets of features. Say we also have a set $\widetilde{S} = (\widetilde{X})$ with unlabeled data points $\{\mathbf{x}_{m+i}\}_{i=1}^{n}$, $\widetilde{X} = (\mathbf{x}_{m+1}, \ldots, \mathbf{x}_{m+n})^t \in \mathcal{X}^n$. Intuitively, $\widetilde{S}$ can be used to identify disagreement between models. No labels required. In the co-regularization setting we would like to identify functions $\mathbf{f} = (f_1, \ldots, f_M) \in \mathcal{H}_1 \times \ldots \times \mathcal{H}_M$ in order to minimize the objective function

$$J(\mathbf{f}) = \sum_{v=1}^{M} \mathcal{L}(f_v, S) + \lambda \sum_{v=1}^{M} \|f_v\|_{\mathcal{H}_v}^2 + \mu \sum_{v,u=1}^{M} \mathcal{L}_C(f_v, f_u, \widetilde{S}), \tag{4}$$

where $\lambda, \mu \in \mathbb{R}^+$ are regularization parameters and where $\mathcal{L}_C$ is the loss function measuring the disagreement between the prediction functions of the views on the unlabeled data. We obtain equation (1) by choosing $M = 1$.

Similar to the standard learning problem, co-regularized algorithms are usually not straightforwardly applicable to large scale learning tasks where large amounts of unlabeled as well as labeled data are available for training. Several recently proposed algorithms have complexity that is linear in the number of unlabeled data points and superlinear in the number

of labeled examples. The method proposed in [8] has cubic runtime complexity due to a matrix inverse.

In the following section, we will introduce a novel method enabling co-regression in an online environment. This makes the method linear in feature dimensionality and samples considered per iteration.

3 Online co-regularized algorithm

It is possible to extend standard online methods with a co-regularization component. In this section a stochastic gradient algorithm is modified in such manner, so that it can incorporate unlabeled information too. To do so, the method first has to be converted to the multi-view perspective. In addition, a co-regularization component that forces agreement between all pairs of views is necessary. Although the method is similar to Pegasos [3], LaSVM [1] and GURLS [11], the method introduced here is preferable in case unlabeled data points are available for learning.

Given this, lets consider (4) in an online setting similar to equation (2).

$$J(\mathbf{W}) = \sum_{v=1}^{M} \left(\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\mathbf{x}_i^v, y_i; \mathbf{w}^v) + \lambda^v \|\mathbf{w}^v\|^2 \right) + \frac{\mu}{n} \sum_{\substack{v,u=1 \\ v \neq u}}^{M} \sum_{i=m+1}^{m+n} \mathcal{L}_C(\mathbf{x}_i^v, \mathbf{x}_i^u; \mathbf{w}^v, \mathbf{w}^u), \quad (5)$$

Here the first term corresponds to equation (2) in a multi-view setting and the second term enforces agreement between all pairs of models by punishing disagreement. It could be the case that one view is naturally more important than others. Because of this, each view has a separate regularization parameter. Also both terms are divided by the amount of samples, respectively m and n, summed over. Co-regularization should be a term that compensates normal predictions. Without averaging, model training would be done purely by co-regularization for large n. We can approximate the optimal solution by minimizing (5) using stochastic gradient descent (3).

Consider the setting in which the squared loss function is used for the co-regularization, predictions are done through a linear model and we remain using the L_2 norm for standard regularization. Linear model means that model predictions are done through the expression $\mathbf{w}^T\mathbf{x}$. The choice of squared loss for the co-regularization term is quite natural as it penalizes the differences of the prediction functions between views. For every iteration t of the algorithm, we first choose a set $A_t \subseteq S$ of size k. Similarly we choose $\tilde{A}_t \subseteq \tilde{S}$ of size l for each round t on the unlabeled dataset. We then apply the stochastic update on (5) to get the update rule

$$\begin{aligned} \mathbf{w}_{t+1}^v = (1 - \eta_t^v \lambda^v)\mathbf{w}_t^v - \quad & \frac{\eta_t^v}{|A_t|} \sum_{(\mathbf{x}, y \in A_t)} \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{x}^v, y; \mathbf{w}_t^v) \; - \\ & 4\eta_t^v \frac{\mu}{|\tilde{A}_t|} \sum_{\substack{v,u=1 \\ v \neq u}}^{M} \sum_{(\mathbf{x}, y \in \tilde{A}_t)} \left(\mathbf{w}_t^{vT} \mathbf{x}^v - \mathbf{w}_t^{uT} \mathbf{x}^u \right) \mathbf{x}^v. \end{aligned} \quad (6)$$

This may seem a bit daunting, but if we choose $A_t = S$ and $\tilde{A}_t = \tilde{S}$ on each round t we obtain the normal full gradient projection method. At the other extreme, if we choose

A_t to contain a single randomly selected example, we recover a variant of the stochastic gradient method. In general, we allow A_t to be a set of k and $\tilde{A}_t$ to be a set of l data points sampled i.i.d. from S and $\tilde{S}$, respectively. It is due to this subsampling that the update is referred to as stochastic.

In the above expression no loss function has been specified yet, leaving this to be a general method. For regression problems, the squared loss function is often used.

$$\mathcal{L}_{squared}(p, y) = \frac{1}{2} \cdot (p - y)^2,$$

where p is the model prediction and y the correct label. When derived with respect to the predicted value, this becomes

$$\nabla_p \mathcal{L}_{squared}(p, y) = (p - y).$$

The derivative is taken with respect to p. In the objective function, we consider the derivative with respect to $\mathbf{w}$. To make this work, one has to apply the chain rule. In the case of a linear model, this leads to an additional multiplication with $\mathbf{x}$.

By substituting the squared loss derivative into the second term of equation (6), we obtain the update rule for the online co-regularized algorithm with the squared loss function.

The hinge loss is usually considered as more appropriate for classification problems, although it has been demonstrated that squared loss often leads to similar performance [14] [15]. When used on classification labels $y \in \{-1, 1\}$, the hinge loss function is of the form

$$\mathcal{L}_{hinge}(p, y) = \max(0, 1 - yp),$$

where p is the model prediction and y the correct label. Since this function is not differentiable, an approximation of it, a so-called subgradient, is used when optimizing the corresponding model function.

$$\nabla_p \mathcal{L}_{hinge}(p, y) = \begin{cases} -y, & \text{if } p > 0 \\ 0 & \text{else} \end{cases}$$

We can enforce this condition in (6) by defining a set A^+ holding examples for which $\mathbf{w}^v$ obtains a non-zero loss, that is $A^+ = \{(\mathbf{x}^v, y) \in A_t : y(\mathbf{w}^{vT}\mathbf{x}^v) < 1\}$. Then by substituting the hinge loss derivative into the second term of equation (6), we obtain the update rule for the online co-regularized algorithm with the hinge loss function.

Illustrated in algorithm 1 is the Online Co-regularization Algorithm for classification using the hinge loss function. Predictions are done by computing the mean prediction value of all views.

3.1 Discussion

An important part of operationalizing this algorithm is parameter determination. With only two views, we already have 3 parameters to optimize: regularization parameters λ for each view and μ. If you regard meta parameters k and l as tunable, there are already 5 different parameters to tune. When also setting a learning parameter offset η_0 for each view, even more parameters come into play. By having so many parameters, without

Algorithm 1 Online co-regularized algorithm (OCA-k-l)

Require: Datasets S and $\tilde{S}$, regularization parameters λ, batch sizes k and l, number of views M, co-regularization parameter μ.

Ensure: $\mathbf{w}^v = 0$

1: **for** $t = 1, 2, \ldots, N$ **do**

2: Choose $A_t \subseteq S$, where $|A_t| = k$ and $\tilde{A}_t \subseteq \tilde{S}$, where $|\tilde{A}_t| = l$

3: Set $A_t^+ = \{(\mathbf{x}^v, y) \in A_t : y(\mathbf{w}^{vT}\mathbf{x}^v) < 1\}$

4: Set $\eta_t^v = \frac{1}{\lambda t}$

5: $\mathbf{w}_{t+1}^v \leftarrow (1 - \eta_t^v \lambda)\mathbf{w}_t^v - \eta_t^v \sum_{(\mathbf{x}, y \in A_t^+)} y\mathbf{x}^v$

6: $\quad -4\eta_t^v \frac{\mu}{l} \sum_{\substack{v,u=1 \\ v \neq u}}^{M} \sum_{(\mathbf{x}, y \in \tilde{A}_t)} \left(\mathbf{w}_t^{vT}\mathbf{x}^v - \mathbf{w}_t^{uT}\mathbf{x}^u\right)\mathbf{x}^v$

7: Output $\mathbf{w}_{N+1}^v$

independency assumptions, finding the optimal parameter pair suffers from the curse of dimensionality.

To work with a high amount of parameters, one could assume simplified dependencies between views or parameters. When applying a grid search, we found it to be beneficial to pick small values for co-regularization parameter μ (< 1), possibly constraining its grid. Finding appropriate values for k and l can also be done through a simple grid search with a rough grid. This only leaves the different λ and μ to search for. If one assumes independency between λ and μ, searching for different regularization parameters λ can be done in $O(g^M)$ steps, where g is the grid size for each λ and M is the number of views.

Although the proposed algorithm is presented with the hinge loss function, the extension to logarithmic, ϵ-intensive, pairwise, and several others are relatively straightforward. In the above formulation we considered a version of the algorithm that makes use of randomly sampled subsets A_t and $\tilde{A}_t$ at every iteration. Flexibility to vary the sizes of these sets at every time step can be beneficial in some circumstances, for example when prediction functions in multiple views start to diverge significantly one can consider increasing the number of unlabeled data points in the co-regularized term.

Also, use of squared loss is associated with several computation benefits, for example it allows to precompute the co-regularized term on unlabeled data and efficiently use it in training, in case of static or low amounts of unlabeled data. We also found the algorithm to work best in a difficult setting, opposing a completely linearly separable dataset.

When dealing with sparse data, only certain fields in $\mathbf{w}$ have to be updated, while regularization operates on every field. By separating these two, we gain some computational advantage. Concretely, we can represent a weight vector $\mathbf{w}$ as a pair $(\mathbf{v}, a)$ where $\mathbf{v} \in \mathbb{R}^n$ is a vector and a is a scalar. The vector $\mathbf{w}$ is then defined as $\mathbf{w} = a\mathbf{v}$. This representation is then substituted in the algorithm leading to seperate updates for a and $\mathbf{v}$. The update for $\mathbf{v}$ is now sparse and the update for a is just one-dimensional. For numerical stability a size constraint is imposed on a. Once the size of a reaches a certain threshold, a is normalized with respect to $\mathbf{v}$. One can verify that this decreases the number of computational steps while remaining equivalent to the original solution.

4 Empirical evaluation

4.1 Experimental setup

In this section we show results of an implemented version of the newly introduced algorithm and compare it with the closely related multi-view and standard online learner.

To evaluate the performance of the proposed algorithm, we use publicly available datasets from the UCI repository[1] and the LibSVM repository[2] that are known to work well with linear models. To demonstrate the difference on a difficult problem we also use the BioInfer language corpus[3] - a real world natural language processing dataset. Typically such natural language problems are large scale in sample as well as feature size. More specific, we selected a number of standard regression and classification datasets from the repository, namely ABALONE, CADATA, HOUSING, MG, SPACE, SVMGUIDE3, GERMANNUMER, AUSTRALIAN with the BioInfer corpus in addition.

To simulate a semi-supervised learning setting, we remove part of the labels from each of the datasets. We use the classical learning setting, where 70% of the data is used for training and the remaining 30% as testing. 20% of the training data is randomly selected to be labeled, and the others are used as unlabeled data, plainly by removing their labels. The used datasets vary in size from several hundred samples to several tens of thousands and the density varies from sparse to dense. Depending on the learning task, the performance measure is either AUC for classification or RMSE for regression.

The best known performance measure for classification is accuracy: "how many questions did I answer correctly?". Though simple and proven to be successful in some cases, it has some flaws that other measures do not have. One often uses Area under Curve (AUC) for binary classification problems. It does not suffer from class distribution bias and choice of thresholding. Basically, the AUC value is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one [16]. As the name suggests, it is calculated by computing the area under the Receiving Operator Characteristic curve.

The Root Mean Squared Error (RMSE) measure is often used for regression type problems. It averages the squared differences over all predictions and labels and sequently takes the square root. This means results

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{m} (y_i - f(x_i))^2}{m}}$$

The datasets are preprocessed by applying a linear scaling to each feature to the interval $[-1, 1]$. For regression datasets we also apply a linear scaling on the labels, to the interval $[0, 100]$. When interpreting RMSE results, this grants easier interpretation for all sets, whereas with a non-normalized dataset, RMSE values completely depend on the original labeling domain. This somewhat normalizes the measure as well, making it easier to see wether a dataset is 'hard' or cross dataset interpretation of methods.

We compare the performance of a two-view online co-regularized algorithm with several

[1] `http://archive.ics.uci.edu/ml/`

[2] `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[3] Available at `www.it.utu.fi/BioInfer`

other methods, namely the baseline - supervised - version of the algorithm, excluding the co-regularization term. This is in essence equivalent to the PEGASOS algorithm [3]. We also compare with the multi-view version of the algorithm by simply excluding the co-regularization term, termed as PEGASOS MV. For classification sets we use the hinge loss for all methods, denoted by appending HL. For regression sets we do the same with the squared loss, SL. We compare with several instantiations of the online co-regularized algorithm, termed as OCA-k-l, using various sizes of unlabeled batch examples. For the supervised learning algorithms, only the labeled part of dataset is used for training. The same set is then used for training the co-regularized model, together with the unlabeled data.

Parameter selection for each model is done by 10-fold cross-validation over the train partition of the data with a grid of possible parameter tuples. Parameters considered are the starting values for λ_1, λ_2 and μ. For the supervised models, parameters to be selected are the starting learning rate η_0 and regularization parameter λ. For the supervised and semi-supervised multi-view models we consider two views that are constructed via random partitioning of the data attributes into two unique sets. Such division of the attributes for constructing multiple views has been previously used in [8]. For the multi-view model we have to estimate the learning rate η_0, as well as the λ_1 and λ_2 parameters. The semi-supervised model has an additional parameter μ controlling the influence of the co-regularization on the views.

4.2 Results

The results of the experiments are included in the Tables 1-8. It can be observed that in all experiments except the housing dataset, the proposed co-regularized algorithm outperforms supervised learning methods. The housing dataset is also the smallest dataset considered in our empirical evaluation. We use a Wilcoxon signed-rank test [17] to estimate whether the differences in performance are statistically significant. In all cases (with the exception of the housing dataset) the OCA leads to statistically significant improvement over the standard PEGASOS algorithm. Detailed information for each dataset is reported in the caption of the corresponding table.

Table 1: Results on the Abalone dataset. The OCA-1-5 algorithm outperforms supervised learning methods. Improvement in performance is statistically significant according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is also statistically significant.

Abalone	CV PERF (RMSE)	η_0	λ_1	λ_2	μ	TEST PERF (RMSE)
PEGASOS SL	14.40	0.5	2.0	N/A	N/A	19.46
PEGASOS MV SL	11.70	0.5	16.0	0.25	N/A	15.52
OCA-1-1	11.63	0.25	16.0	0.25	0.5	14.23
OCA-1-5	11.73	0.25	16	0.25	0.5	**13.50**

Table 2: Results on the Cadata dataset. OCA-1-1 leads to a statistically significant performance improvement compared to supervised Pegasos SL and Pegasos MV SL according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is not statistically significant.

Cadata	CV perf (RMSE)	η_0	λ_1	λ_2	μ	Test perf (RMSE)
Pegasos SL	24.45	2.5	8.0	n/a	n/a	26.00
Pegasos MV SL	24.29	1.5	16.0	4.0	n/a	27.26
OCA-1-1	23.97	1.5	1.0	16.0	1.5	**25.76**
OCA-1-5	23.30	1.5	1.0	16.0	1.5	25.80

Table 3: Results on the Housing dataset. Pegasos SL outperforms other methods on the smallest dataset used in our empirical evaluations.

Housing	CV perf (RMSE)	η_0	λ_1	λ_2	μ	Test perf (RMSE)
Pegasos SL	19.34	0.0625	8.0	n/a	n/a	**16.34**
Pegasos MV SL	17.07	0.01	16.0	64.0	n/a	17.59
OCA-1-1	17.75	0.01	4.0	256	1.5	18.54
OCA-1-5	16.13	0.01	4.0	256	1.5	18.69

Table 4: Results on the MG dataset. OCA-1-1 leads to statistically significant performance improvement compared to supervised Pegasos SL according to the Wilcoxon signed rank test. The differences between the co-regularized algorithms OCA-1-1, OCA-1-5, and Pegasos MV SL are not statistically significant.

MG	CV perf (RMSE)	η_0	λ_1	λ_2	μ	Test perf(RMSE)
Pegasos SL	45.53	0.125	1.0	n/a	n/a	46.71
Pegasos MV SL	45.88	0.125	0.5	0.125	n/a	45.73
OCA-1-1	44.51	1.5	64	32	0.1	**45.57**
OCA-1-5	44.93	0.125	0.5	0.125	0.01	45.91

Table 5: Results on the Space dataset. OCA-1-1 leads to statistically significant performance improvement compared to supervised Pegasos SL and Pegasos MV SL according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is not statistically significant.

Space	CV perf (RMSE)	η_0	λ_1	λ_2	μ	Test perf (RMSE)
Pegasos SL	58.32	0.25	0.5	n/a	n/a	58.17
Pegasos MV SL	50.42	0.125	1.0	0.125	n/a	51.90
OCA-1-1	41.95	1.0	0.125	0.5	n/a	**36.60**
OCA-1-5	42.80	0.125	1.0	0.125	0.5	36.84

4.3 Natural Language parsing

Throughout this experiment, we use the BioInfer corpus [18] which consists of 1100 manually annotated sentences.[4] For each sentence, we generate a set of candidate parses with a link grammar (LG) parser [19]. The LG parser is a full dependency parser based on a broad-coverage hand-written grammar. It generates all parses allowed by its grammar and applies a set of built-in heuristics to predict goodness of the parses. However,

[4]Available at `www.it.utu.fi/BioInfer`

Table 6: Results on the Germannumer dataset. OCA-1-1 leads to statistically significant performance improvement compared to supervised Pegasos HL and Pegasos MV HL according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is also statistically significant.

Germannumer	CV PERF (AUC)	η_0	λ_1	λ_2	μ	TEST PERF (AUC)
PEGASOS HL	0.72	10	0.03125	N/A	N/A	0.74
PEGASOS MV HL	0.76	0.75	4	0.125	N/A	0.71
OCA-1-1	0.75	0.125	0.125	0.125	0.01	**0.75**
OCA-1-5	0.75	0.125	16	0.5	0.2	0.74

Table 7: Results on the Svmguide3 dataset. OCA-1-5 leads to statistically significant performance improvement compared to supervised Pegasos HL and Pegasos MV HL according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is also statistically significant.

Svmguide3	CV PERF (AUC)	η_0	λ_1	λ_2	μ	TEST PERF (AUC)
PEGASOS HL	0.85	1	0.25	N/A	N/A	0.74
PEGASOS MV HL	0.83	1.5	0.125	0.125	N/A	0.75
OCA-1-1	0.77	1.5	0.125	0.125	0.05	0.73
OCA-1-5	0.82	0.25	0.0625	0.125	0.01	**0.76**

Table 8: Results on the Australian dataset. OCA leads to statistically significant performance improvement compared to supervised Pegasos HL and Pegasos MV HL according to the Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is not statistically significant.

Australian	CV PERF (AUC)	η_0	λ_1	λ_2	μ	TEST PERF (AUC)
PEGASOS HL	0.95	0.0625	0.03125	N/A	N/A	0.92
PEGASOS MV HL	0.95	0.25	0.0625	0.03125	N/A	0.92
OCA-1-1	0.93	0.5	0.0625	0.03125	0.001	**0.93**
OCA-1-5	0.94	1	0.0625	0.03125	0.001	**0.93**

the performance of its heuristics has been found to be poor when applied to biomedical text [20], and hence subsequent selection methods are needed. In our experiment we use the proposed online co-regularized algorithm instead of the LG parser built-in heuristics to predict goodness of the generated parse.

Our dataset consists of 3000 parses represented as sparse vectors of dimensionality 201740. We obtain a scoring for an input by comparing its parse to the hand annotated correct parse of its sentence. In order to select the parameter values, we divide the dataset into training 70% and test set 30% (we ensure that the parses that belong to the same sentence belong to a single set). Also, 20% of the training data are randomly selected to be labeled, and the rest is used as unlabeled data.

The first dataset is used for parameter estimation and the second one is reserved for the final validation. The appropriate values of the regularization parameters are determined by grid search with 10-fold cross-validation on the parameter estimation data.

Finally, the algorithm is trained on the whole training set with the selected parameter values and tested with the test parses reserved for the final validation. The results of the

Table 9: Results on the BioInfer dataset. OCA-1-5 leads to statistically significant performance improvement compared to supervised Pegasos SL and Pegasos MV SL according to Wilcoxon signed rank test. The difference between the co-regularized algorithms OCA-1-1 and OCA-1-5 is not statistically significant.

BioInfer	CV PERF (RMSE)	η_0	λ_1	λ_2	μ	TEST PERF (RMSE)
PEGASOS SL	45.36	0.5	32	N/A	N/A	63.86
PEGASOS MV SL	44.94	0.5	16	16	N/A	63.16
OCA-1-1	39.78	0.5	16	16	0.4	61.47
OCA-1-5	39.85	0.5	16	16	0.4	**61.29**

experiment are presented in Table 9. It can be observed that OCA notably outperforms both supervised methods and the improvement in performance is statistically significant according to a Wilcoxon signed rank test. Those results indicate that our algorithm is applicable to the tasks in natural language processing and other domains where sparse, high dimensional data are commonplace.

5 Conclusions

This work presents an online co-regularized algorithm for regression and classification tasks. The research goal is met by demonstrating that large scale problems benefit from the inclusion of unlabeled data and that co-regression can function in an online section by demonstrating so in a practical setting. The algorithm is computationally efficient and is naturally suited for learning tasks in which large amounts of unlabeled and labeled data are available for training The method is related to online methods such as PEGASOS [3], LASVM [1] and GURLS [11] and unlike many co-regularized algorithms has computational complexity independent of the number of training data points when using subsamples of data. In the empirical evaluation we demonstrate that our method consistently performs well on publicly available datasets as well as notably outperforms supervised learning algorithms on the BioInfer corpus from the natural language processing domain. Last but not least, we make available an efficient implementation of our algorithm coded in Python.

The algorithm can be extended to be applicable to various learning tasks. For instance, it can be adapted for the task of large scale preference learning and ranking. Large scale learning to rank has recently received notable attention and while several supervised learning algorithms have been proposed [4], taking into account large amount of unlabeled data (naturally abundant in IR domain) can help to even further improve predictive performance of the method. Thus, an interesting future research direction is to adapt and apply the online co-regularized algorithm to large scale learning to rank tasks.

The body of knowledge introduced in this work will be published later this year in Discovery Science 2012 [21].

5.1 Acknowledgements

Significant parts of this work were done in cooperation with Evgeni Tsivitsivadze [5], who provided me with ideas and valuable discussion material.

References

[1] Bottou, L., Bordes, A., Ertekin, S.: Lasvm (2009) `http://mloss.org/software/view/23/`.

[2] Shalev-Shwartz, S., Srebro, N.: Svm optimization: inverse dependence on training set size. In: Proceedings of the 25th international conference on Machine learning. ICML '08, New York, NY, USA, ACM (2008) 928–935

[3] Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In: Proceedings of the 24th international conference on Machine learning, ACM (2007) 807–814

[4] Sculley, D.: Large Scale Learning to Rank. In: NIPS 2009 Workshop on Advances in Ranking. (2009) 1–6

[5] Hazan, E., Koren, T., Srebro, N.: Beating sgd: Learning svms in sublinear time

[6] Yuan, G.x., Ho, C.h., Lin, C.j.: Recent advances of large-scale linear classification. Proceedings of the IEEE (3) (2011) 1–15

[7] Sindhwani, V., Niyogi, P., Belkin, M.: A co-regularization approach to semi-supervised learning with multiple views. In: Proceedings of ICML Workshop on Learning with Multiple Views. (2005)

[8] Brefeld, U., Gärtner, T., Scheffer, T., Wrobel, S.: Efficient co-regularised least squares regression. In: Proceedings of the 23rd International Conference on Machine learning, New York, NY, USA, ACM (2006) 137–144

[9] Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)

[10] Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. In: Advances in Learning Theory: Methods, Model and Applications, Amsterdam, IOS Press (2003) 131–154

[11] Tacchetti, A., Mallapragada, P., Santoro, M., Rosasco, L.: GURLS: a toolbox for large scale multiclass learning. In: NIPS 2011 workshop on parallel and large-scale machine learning. `http://cbcl.mit.edu/gurls/`.

[12] Bottou, L., Bousquet, O.: The tradeoffs of large scale learning. In Platt, J., Koller, D., Singer, Y., Roweis, S., eds.: Advances in Neural Information Processing Systems. Volume 20. NIPS Foundation (http://books.nips.cc) (2008) 161–168

[13] Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on Computational learning theory, New York, NY, USA, ACM (1998) 92–100

[5] http://learning-machines.com

[14] Rifkin, R., Yeo, G., Poggio, T.: Regularized least-squares classification. In Suykens, J., Horvath, G., Basu, S., Micchelli, C., Vandewalle, J., eds.: Advances in Learning Theory: Methods, Model and Applications. Volume 190 of NATO Science Series III: Computer and System Sciences. IOS Press, Amsterdam (2003) 131–154

[15] Zhang, P., Peng, J.: SVM vs regularized least squares classification. In: Proceedings of the International Conference on Pattern Recognition. ICPR '04 (2004) 176–179

[16] Cortes, C., Mohri, M.: Auc optimization vs. error rate minimization. In: IN ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, MIT Press (2003)

[17] Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research **7** (2006) 1–30

[18] Pyysalo, S., Ginter, F., Heimonen, J., Björne, J., Boberg, J., Järvinen, J., Salakoski, T.: BioInfer: A corpus for information extraction in the biomedical domain. BMC Bioinformatics **8**(50) (2007)

[19] Sleator, D.D., Temperley, D.: Parsing english with a link grammar. Technical Report CMU-CS-91-196, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA (October 1991)

[20] Pyysalo, S., Ginter, F., Pahikkala, T., Boberg, J., Järvinen, J., Salakoski, T.: Evaluation of two dependency parsers on biomedical corpus targeted at protein-protein interactions. Recent Advances in Natural Language Processing for Biomedical Applications, special issue of the International Journal of Medical Informatics **75**(6) (2006) 430–442

[21] de Ruijter, T., Tsivtsivadze, E.: Online co-regularized methods. In: Lecture Notes in Computer Science. Volume 7569., Springer (2012)