Focused Crawling with Scalable Ordinal Regression Solvers

Rashmin Babaria RASHMIN@CSA.IISC.ERNET.IN
J. Saketha Nath SAKETHA@CSA.IISC.ERNET.IN
Krishnan S KRISHI@CSA.IISC.ERNET.IN
Sivaramakrishnan K R SIVARAMAKRISHNAN@GMAIL.COM
Chiranjib Bhattacharyya CHIRU@CSA.IISC.ERNET.IN
M.N.Murty MNM@CSA.IISC.ERNET.IN
Department of Computer Science and Automation, Indian Institute of Science, Bangalore.

Abstract

In this paper we propose a novel, scalable, clustering based Ordinal Regression formulation, which is an instance of a Second Order Cone Program (SOCP) with one Second Order Cone (SOC) constraint. The main contribution of the paper is a fast algorithm, CB-OR, which solves the proposed formulation more efficiently than general purpose solvers. Another main contribution of the paper is to pose the problem of focused crawling as a large scale Ordinal Regression problem and solve using the proposed CB-OR. Focused crawling is an efficient mechanism for discovering resources of interest on the web. Posing the problem of focused crawling as an Ordinal Regression problem avoids the need for a negative class and topic hierarchy, which are the main drawbacks of the existing focused crawling methods. Experiments on large synthetic and benchmark datasets show the scalability of CB-OR. Experiments also show that the proposed focused crawler outperforms the state-of-the-art.

1. Introduction

Ordinal Regression problems frequently occur in the areas of information retrieval, social science, personalized searches etc (Har-Peled et al., ; Herbrich et al., 2000; Crammer & Singer, 2002; Shashua & Levin, 2003). Given a training dataset labeled with a set of ranks, the task of Ordinal regression (OR) is to construct a function that predicts the rank of new data points. In contrast to metric regression problems, these ranks are of finite types and the metric distances between the ranks are not defined. Also the existence of the ordering information among the classes makes an OR problem different from a multiclass classification problem. Because of its wide applicability, there is considerable interest in solving large scale OR tasks. Various formulations have been proposed for solving such problems, for a review see (Chu & Keerthi, 2005). In this paper we consider the support vector formulation given by (Chu & Keerthi, 2005), which improves the formulation of (Shashua & Levin, 2003), as the baseline OR formulation.

Due to the increasing usage of Internet and growth of web, the need for fast training and prediction algorithms in the domains of information retrieval and personalized search is increasing. Existing formulations require that the number of constraints in the formulation grow with the number of data points. We propose a clustering based alternative which leads to a optimization problem where the number of constraints depend mainly on number of clusters, and not on the number of data points. Since number of clusters could be substantially smaller than the actual training data size the proposed approach has better scaling properties. Another advantage is that the time taken to infer the label of a class depends on the number of clusters making it more attractive for applications which require fast predictions. The proposed formulation turns out to be a SOCP formulation with one SOC constraint and few linear constraints. The formulation can be solved using generic SOCP solvers like SeDuMi [1]. However if the number of clusters in the training data itself is large, then these generic solvers fail to scale well.

The main contribution of the paper is a fast algorithm,

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

[1] `http://sedumi.mcmaster.ca/`

CB-OR, that can efficiently handle large number of clusters and hence very large number of data points. CB-OR exploits the fact that the proposed OR formulation has only one SOC constraint and efficiently solves the dual of the formulation. This removes the necessity of using any kind of generic optimization software like SeDuMi.

Focused crawler is an automated mechanism to efficiently find pages relevant to a topic on the web. Focused crawlers were proposed to traverse and retrieve only a part of the web that is relevant to a particular topic, starting from a set of pages usually referred to as the seed set. It makes efficient usage of network bandwidth and storage capacity. Focused crawling provides a viable mechanism for frequent updation of search engine indexes. They have been useful for other applications like distributed processing of the complete web, with each crawler assigned to a small part of the web.

Many variants of focused crawlers have been proposed in the literature and have shown good performance. But two major problems are associated with them: they require a topic taxonomy and a negative set of documents for any topic. The second major contribution of the paper is to pose focused crawling as a large scale ordinal regression problem. This removes the drawbacks of the existing focused crawling methods.

The outline of the paper is as follows, Section 2 presents the clustering based OR formulation and the CB-OR algorithm which efficiently solves its dual. The methodology of focused crawling using OR is described in Section 3. Section 4 details the experiments and discusses the results. We conclude the paper in Section 5 by describing improvements for future.

2. Ordinal Regression

In this section we give a brief introduction to the OR problem and the baseline OR formulation. We then present the clustering based OR formulation. Subsequently we present the fast CB-OR algorithm to solve the dual of the proposed formulation.

We begin by formally defining the OR problem. Given a data set $\mathcal{D} = \{({\mathbf{x}_i}^j, y_i) \mid \mathbf{x}_i^j \in \mathbb{R}^m, i = 1, \cdots, r,\ j = 1, \cdots, n_i\}$ where y_i is the rank of the data point, r is the number of ranks, n_i is the number of data points having rank 'i' and $n = \sum_{i=1}^{r} n_i$ is the total number of data points, the task of OR is to compute a function $f : \mathbb{R}^m \to \{1, \ldots, r\}$ such that $f(x_i^j) = y_i$. Such formulations find ready appeal in many problems such as ranking (Herbrich et al., 2000).

The baseline OR formulation (Chu & Keerthi, 2005) finds a set of hyperplanes $\mathbf{w}^\top \mathbf{x} - b_i = 0,\ i = 1, \ldots, r-1$, which separate the data points of different classes. In other words, one can define $b_0 = -\infty, b_r = \infty$ and constrain that the data points of class 'i' must lie between $\mathbf{w}^\top \mathbf{x} - b_{i-1} = 0$ and $\mathbf{w}^\top \mathbf{x} - b_i = 0$. The authors also provide a SMO type algorithm for solving the ordinal regression problem efficiently. The formulation can be written as (Chu & Keerthi, 2005):

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b},\xi_i^j,\xi_i^{*j}} \quad & \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{r}\sum_{j=1}^{n_i} \xi_i^j + \xi_i^{*j} \\ \text{s.t.} \quad & \mathbf{w}^\top \mathbf{x}_i^j - b_i \le -1 + \xi_i^j,\ \xi_i^j \ge 0, \\ & \mathbf{w}^\top \mathbf{x}_i^j - b_{i-1} \ge 1 - \xi_i^{*j},\ \xi_i^{*j} \ge 0,\ \forall\, i, j \\ & b_i - b_{i-1} > 0,\ i = 2, \ldots, r-1 \end{aligned} \tag{1}$$

where $\mathbf{b}$ is the vector containing $b_i, i = 1, \ldots, r-1$. The above formulation can be shown to be equivalent to:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b},\xi_i^j,\xi_i^{*j}} \quad & \sum_{i=1}^{r}\sum_{j=1}^{n_i} \xi_i^j + \xi_i^{*j} \\ \text{s.t.} \quad & \mathbf{w}^\top \mathbf{x}_i^j - b_i \le -1 + \xi_i^j,\ \xi_i^j \ge 0,\ \xi_i^{*j} \ge 0, \\ & \mathbf{w}^\top \mathbf{x}_i^j - b_{i-1} \ge 1 - \xi_i^{*j},\ \forall\, i, j,\ \|\mathbf{w}\|_2 \le W, \\ & b_i - b_{i-1} > 0, i = 2, \ldots, r-1 \end{aligned} \tag{2}$$

Experiments in (Chu & Keerthi, 2005) also show that the above formulation achieves good generalization in practise. However, the size of the optimization problem problem that needs to be solved is dependent on the data size. In order to make the formulation scalable to large datasets we propose a clustering based large scale ordinal regression formulation that is independent of problem size.

2.1. Large Scale Clustering Based OR

Let Z_i be the random variable that generates the data points of rank 'i'. Assume that the distributions of Z_i can be modeled using mixture models. Let k_i be the number of components of Z_i where each component distribution has spherical covariance. Let $X_i^j,\ j = 1, \cdots, k_i$ be the random variable generating the j^{th} component of Z_i whose second order moments are given by $(\mu_i^j, {\sigma_i^j}^2 I)$. Any good clustering algorithm will correctly estimate the second order moments of the components. BIRCH (Zhang et al., 1996) is one such clustering algorithm, that scales well for large datasets. Given these estimates of second order moments, an optimal regressor that generalizes well can be built.

As mentioned above, the data points that belong to class 'i' must lie between the hyperplanes $\mathbf{w}^\top \mathbf{x} - b_{i-1} =$

0 and $\mathbf{w}^\top \mathbf{x} - b_i = 0$ with high probability. This can be mathematically expressed as: $P(\mathbf{w}^\top Z_i - b_i \leq -1 + \xi_i^j) \geq \eta,\ P(\mathbf{w}^\top Z_i - b_{i-1} \geq 1 - \xi_i^{*j}) \geq \eta$ where η is user defined parameter. η lower bounds the classification accuracy. Following the arguments in (Nath et al., 2006), and using the formulation (2), one can derive the following clustering based large margin OR formulation:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b},\xi_i^j,\xi_i^{*j}} \quad & \sum_{i=1}^{r}\sum_{j=1}^{k_i} \xi_i^j + \xi_i^{*j} \\ \text{s.t.} \quad & \mathbf{w}^\top \mu_i^j - b_i \leq -1 + \xi_i^j - \kappa\sigma_i^j W, \\ & \mathbf{w}^\top \mu_i^j - b_{i-1} \geq 1 - \xi_i^{*j} + \kappa\sigma_i^j W, \\ & \xi_i^j \geq 0,\ \xi_i^{*j} \geq 0,\ \forall\ i,j,\ \|\mathbf{w}\|_2 \leq W, \\ & b_i - b_{i-1} > 0,\ i = 2,\ldots,r-1 \end{aligned} \tag{3}$$

where $\kappa = \sqrt{\frac{\eta}{1-\eta}}$. The ordinal regression formulation (3) is an SOCP problem. This problem can be solved using generic SOCP solvers like SeDuMi to obtain the optimal values of $\mathbf{w}$ and $\mathbf{b}$. Note that the number of constraints for each rank 'i' in (3) is k_i, compared to n_i in (1). Thus the clustering based OR formulation scales better than baseline formulation (1). The overall training scheme is to cluster the data points using any scalable clustering algorithm like BIRCH, which provides the second order moments of all the clusters and then solve the SOCP problem (3) using solvers like SeDuMi.In the following section we extend the clustering based formulation (3) to the non-linear case and propose a fast algorithm, CB-OR, to solve the dual of the resulting formulation.

2.2. CB-OR Algorithm

We start by presenting an extension to the proposed clustering based OR formulation for the non-linear case. As discussed in (Nath et al., 2006), the geometric interpretation of the inequalities in the formulation (3) is that of separating the spheres centered at μ_i^j and radius $\kappa\sigma_i^j$. Now suppose that the non-linear mapping ϕ maps the means μ_i^j to $\phi(\mu_i^j)$. Assume the mapping has the property that "closer data points remain close and farther data points remain far". The mapping implicitly achieved by Gaussian kernel $K(\mathbf{x}_1, \mathbf{x}_2) = \exp\left\{-s\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2\right\}$, where s is the Gaussian kernel parameter, is in fact one such mapping. Though the following discussion holds for any such mapping, we restrict ourselves to the case of Gaussian kernel, in order to keep the equations simple. One can easily verify that if $\|\mathbf{x} - \mu_i^j\|_2 \leq \kappa\sigma_i^j$, then $\|\phi(\mathbf{x}) - \phi(\mu_i^j)\|_2 \leq r_i^j$ where $r_i^j = \sqrt{2\left(1 - \exp\left\{-s\left(\kappa\sigma_i^j\right)^2\right\}\right)}$. Using this, one can rewrite the OR formulation (3) as:

$$\begin{aligned} \min_{\mathbf{w},\mathbf{b},\xi_i^j,\xi_i^{*j}} \quad & \sum_{i=1}^{r}\sum_{j=1}^{n_i} \xi_i^j + \xi_i^{*j} \\ \text{s.t.} \quad & \mathbf{w}^\top \phi(\mu_i^j) - b_i \leq -1 + \xi_i^j - r_i^j W, \\ & \mathbf{w}^\top \phi(\mu_i^j) - b_{i-1} \geq 1 - \xi_i^{*j} + r_i^j W, \\ & \xi_i^j \geq 0,\ \xi_i^{*j} \geq 0,\ \forall\ i,j,\ \|\mathbf{w}\|_2 \leq W, \\ & b_i - b_{i-1} > 0,\ i = 2,\ldots,r-1 \end{aligned} \tag{4}$$

The dual of the above formulation turns out to be

$$\begin{aligned} \max_{\alpha,\alpha^*,\rho} \quad & \mathbf{d}^\top(\alpha + \alpha^*) - \rho W \\ \text{s.t.} \quad & \sqrt{(\alpha^* - \alpha)^\top \mathbf{K}(\alpha^* - \alpha)} \leq \rho, \\ & 0 \leq \alpha \leq 1, 0 \leq \alpha^* \leq 1 \\ & S_i^* \leq S_i,\ \forall\ i = 1,\ldots,r-2, S_{r-1}^* = S_{r-1} \end{aligned} \tag{5}$$

where, α is the vector containing the Lagrange multipliers α_i^j for the inequalities $\mathbf{w}^\top\phi(\mu_i^j) - b_i \leq -1 + \xi_i^j - r_i^j W$, α^* is the vector containing the Lagrange multipliers α_i^{*j} for the inequalities $\mathbf{w}^\top\phi(\mu_i^j) - b_{i-1} \geq 1 - \xi_i^{*j} + r_i^j W$, ρ is the Lagrange multiplier for the inequality $\|\mathbf{w}\|_2 \leq W$, $\mathbf{d}$ is the vector containing $1 + r_i^j W$ as its entries, $\mathbf{K}$ is the matrix containing dot products of $\phi(\mu_i^j)$ with each other, $S_i = \sum_{k=1}^{i}\sum_{j=1}^{n_k}\alpha_k^j$ and $S_i^* = \sum_{k=2}^{i+1}\sum_{j=1}^{n_k}\alpha_k^{*j}$ (please refer Appendix for a short derivation of the dual).

Clearly (5) is an SOCP problem and can be solved using SeDuMi. However it is interesting to note that the dual is not any SOCP, but a special case where the number of SOC constraints is one. Algorithms which exploit this fact, in general, work faster than generic solvers like SeDuMi (Erdougan & Iyengar, 2006). We further exploit the special structure of the dual problem and propose a very fast, easy to implement, SMO (Platt, 1999) kind of algorithm, CB-OR, to solve the dual.

The constraints in (5) imply a lower bound on ρ and objective implies minimizing ρ. Hence at optimality, $\rho = \sqrt{(\alpha^* - \alpha)^\top \mathbf{K}(\alpha^* - \alpha)}$. Using this condition we can rewrite the dual as:

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & W\sqrt{(\alpha^* - \alpha)^\top \mathbf{K}(\alpha^* - \alpha)} - \mathbf{d}^\top(\alpha + \alpha^*) \\ \text{s.t.} \quad & 0 \leq \alpha \leq 1, 0 \leq \alpha^* \leq 1 \\ & S_i^* \leq S_i,\ \forall\ i = 1,\ldots,r-2, S_{r-1}^* = S_{r-1} \end{aligned} \tag{6}$$

From the KKT conditions (14) and the optimal value of ρ (assuming $\rho \neq 0$), one can calculate the value of $\mathbf{w}$ as $\frac{W}{\rho}\sum_{i=1}^{r}\sum_{j=1}^{n_i}\left(\alpha_i^{*j} - \alpha_i^j\right)\phi(\mu_i^j)$. Thus the decision function can be written as:

$$f(\mathbf{x}) \equiv \mathbf{w}^\top \mathbf{x} - b = \frac{W}{\rho}\mathbf{K}_{\mathbf{x}}^\top(\alpha^* - \alpha) \tag{7}$$

where $\mathbf{K_x}$ is the vector of dot products of $\phi(\mathbf{x})$ and $\phi(\mu_i^j)$. Thus neither for solving the dual (5), nor for calculating $f(\mathbf{x})$, ϕ is explicitly needed; dot products are enough. The optimal values of $b_1, \ldots, b_{r-1}$ can be computed using the KKT conditions (14):

$$\begin{aligned}
\alpha_i^j = 0 \quad & f(\phi(\mu_i^j)) + 1 + r_i^j W \leq b_i \\
0 < \alpha_i^j < 1 \quad & f(\phi(\mu_i^j)) + 1 + r_i^j W = b_i \\
\alpha_i^j = 1 \quad & f(\phi(\mu_i^j)) + 1 + r_i^j W \geq b_i \\
\alpha_{i+1}^{*j} = 0 \quad & f(\phi(\mu_{i+1}^j)) - 1 - r_i^j W \geq b_i \\
0 < \alpha_{i+1}^{*j} < 1 \quad & f(\phi(\mu_{i+1}^j)) - 1 - r_i^j W = b_i \\
\alpha_{i+1}^{*j} = 1 \quad & f(\phi(\mu_{i+1}^j)) - 1 - r_i^j W \leq b_i
\end{aligned} \tag{8}$$

Let ${b_{low}}^i, {b_{up}}^i$ denote the greatest lower bound, least upper bound on b_i. Hence we have the conditions ${b_{low}}^i \leq b_i \leq {b_{up}}^i, i = 1, \ldots, r-1$. The KKT conditions (14) also indicate that for $i = 2, \ldots, r-1$, $b_{i-1} \leq b_i$ and if $S_{i-1} > S_{i-1}^*$ then $b_{i-1} = b_i$. Thus the overall optimality conditions can be written as ${B_{low}}^i \leq b_i \leq {B_{up}}^i$ where

$${B_{low}}^i = \begin{cases} \tilde{B}_{low}^{i+1} & \text{if } S_i > S_i^* \\ \tilde{B}_{low}^i & \text{otherwise} \end{cases} \tag{9}$$

and

$${B_{up}}^i = \begin{cases} \tilde{B}_{up}^{i-1} & \text{if } S_{i-1} > S_{i-1}^* \\ \tilde{B}_{up}^i & \text{otherwise} \end{cases} \tag{10}$$

and $\tilde{B}_{low}^i = \max\{{b_{low}}^k : k = 1, \ldots, i\}$, $\tilde{B}_{up}^i = \min\{{b_{up}}^k : k = i, \ldots, r-1\}$. Thus b_{low}, b_{up} represent the conditions at every hyperplane due to neighboring class data points; whereas B_{low}, B_{up} represent the conditions over all hyperplanes.

The proposed CB-OR algorithm starts with some feasible solution. Then at every iteration, ${B_{low}}^i, {B_{up}}^i \; \forall \; i$ are calculated. If ${B_{low}}^i \leq {B_{up}}^i \; \forall \; i$, then the optimal solution is found and the algorithm terminates. Else the index i for which ${B_{low}}^i \leq {B_{up}}^i$ is most violated is calculated: $I = \arg\max_i \{i : {B_{low}}^i - {B_{up}}^i > 0\}$. Using (9) and (10) the maximum KKT violating pair can be calculated. Now the following cases exist: **Case 1** The most violating pair is α_l^{jl} and α_m^{jm}, **Case 2** The most violating pair is α_l^{*jl} and α_m^{*jm}, **Case 3** The most violating pair is α_l^{jl} and α_m^{*jm}, **Case 4** The most violating pair is α_l^{*jl} and α_m^{jm}, where $l \leq m$. The equality constraint $S_{r-1} = S_{r-1}^*$ must hold. So for Case 1,2 one can update the variables by adding $\Delta\alpha$ to jl^{th} variable and subtracting $\Delta\alpha$ from jm^{th} variable. In Case 3,4 both variables must be incremented by $\Delta\alpha$ ($\Delta\alpha$ can also take negative values).

Now let $G_1 \equiv W\sqrt{(\alpha^* - \alpha)^\top \mathbf{K}(\alpha^* - \alpha)} - \mathbf{d}^\top(\alpha + \alpha^*)$ denote the dual objective with current values of α, α^*. Let G_2 denote the value of dual objective after appropriately incrementing the variables α, α^* with $\Delta\alpha$. We wish to find that value of $\Delta\alpha$ for which $G_2 - G_1$ is minimized. This can be written as the following 1-d minimization problem:

$$\begin{aligned}
\min_{\Delta\alpha} \quad & \sqrt{a(\Delta\alpha)^2 + 2b(\Delta\alpha) + c} - e\Delta\alpha \\
\text{s.t.} \quad & lb \leq \Delta\alpha \leq ub
\end{aligned} \tag{11}$$

where $a = W^2(\mathbf{K}(jl, jl) - 2\mathbf{K}(jl, jm) + \mathbf{K}(jm, jm))$, $b = W^2 s_1((\mathbf{K}_{jm} - \mathbf{K}_{jl})^\top(\alpha^* - \alpha))$, $c = W^2(\alpha^* - \alpha)^\top \mathbf{K}(\alpha^* - \alpha)$ and $e = (\mathbf{d}(jl) - s_2\mathbf{d}(jm))$. The values of s_1, s_2 depend on the Case to which update belongs to. $s_1 = 1$ for Case 1,3 and $s_1 = -1$ for Case 2,4. $s_2 = 1$ for Case 1,2 and $s_2 = -1$ for Case 3,4. Here, lb, ub denote the tightest lower and upper bounds on $\Delta\alpha$ got from the inequality constraints in (6).

The optimum value of $\Delta\alpha$ that minimizes (11) is given by

$$\Delta\alpha = \begin{cases}
\left[\frac{e\sqrt{\frac{ac-b^2}{a-e^2}} - b}{a}\right]_{lb}^{ub} & \text{if } ac - b^2 > 0, a - e^2 > 0 \\
\left[\frac{-b}{a}\right]_{lb}^{ub} & \text{if } ac - b^2 = 0, a - e^2 > 0 \\
ub & \text{if } e - \sqrt{a} \geq 0 \\
lb & \text{if } e + \sqrt{a} \leq 0
\end{cases}$$

where $\Delta\alpha]_{lb}^{ub}$ denotes $\max(lb, \min(ub, \Delta\alpha))$. Once the optimum value of $\Delta\alpha$ is calculated, then the values of α and α^* are updated accordingly and the procedure is repeated in the next iteration.

3. Focused Crawling

A focused crawler usually consists of the following 3 basic components: a page fetcher, a priority queue and a scoring function. The fetcher gets the webpage pointed to by the URL at the head of the queue. Once the page is fetched, the scoring function determines the relevance of the webpage and the likelihood/probability of the links on the page leading to a webpage of interest. It then inserts these links in the priority queue based on this likelihood/probability.

The aim of any focused crawler is to start from a *seed* set of pages relevant to a given topic and traverse specific links to collect pages about the topic without fetching pages unrelated to the topic. The fraction of relevant pages fetched is called the harvest rate (Chakrabarti et al., 2002). Let N be the number of pages and N_R be the number of relevant pages crawled at any given time t, then the harvest rate at

time t is defined as the fraction of crawled pages that are relevant. More precisely

$$Harvest\ rate = \frac{N_R}{N} \tag{12}$$

Related work: Focused crawling was coined by (Chakrabarti et al., 1999) as an efficient resource discovery system. It has three components – crawler, classifier and distiller. The classifier and distiller are used to define the strategy of the crawl. It uses an existing document taxonomy to define the topics of interest and irrelevant topics. Classifiers are learnt at each internal node of the tree, which gives the probability of the document belonging to the node. The distiller periodically runs through the crawled pages to find the hubs and authorities (Kleinberg, 1999) to prioritize URLs that are found to be in hubs. The priority is assigned to a document and hence all URLs in the document have equal priority in the crawl.

An improved version was proposed in (Chakrabarti et al., 2002) which extends the previous baseline focused crawler to prioritize the URLs within a document by using a classifier called the apprentice. Intelligent crawling (Aggarwal et al., 2001) is a method that allows users to specify arbitrary predicates for measuring relevance and uses reinforcement based learning. Another method, which is related to the one in this paper, was proposed (Diligenti et al., 2000) which models the web as a graph and finds the relevance of any page by posing the problem as a multiclass classification formulation.

3.1. FOCUS: a new look at focused crawling

As mentioned in section 1, current methods for focused crawling determine the relevance of a page using a classifier which requires a document taxonomy and negative set of pages for training. Given a set of seed pages, the previous approaches cannot be used unless there is a topic hierarchy. Also, for any topic, the negative set is very large and diverse, which makes it difficult to construct.

To overcome these problems, the proposed crawling strategy uses the observation (Grangier & Bengio, 2005; Davison, 2000) that any document is semantically closer to documents hyperlinked with it, than to documents which are not. Thus pages which are one link away are semantically closer to seed pages than pages that are two to three links away. The idea is to rank the documents based on their link distance to the topic pages. This ranking creates an ordering among the documents of the web due to its linked nature.

Determination of link distances requires knowledge of the nature of pages which are at some link distance to the topic pages. This is done by modeling the web as a layered graph, with the pages relevant to the seed pages/topic forming the first layer. Similarly, pages which have links to topic pages form the second layer, pages having links to these second layer pages forming the third layer and so on. The first layer is called level 0, the second layer, level 1, and so on. During crawling, links on level i pages are given higher priority than those on level $i+1$. That is, links on a page are prioritized depending on how quickly these links would lead to topic page, where time is measured in terms of number of links that need to be crawled.

The layered web graph is constructed from the seed set by finding the back-links, i.e. pages that point to these seed pages. Using an existing general search engine API - GOOGLE SOAP Search API [2] the back-references for the seed pages are collected. These pages form examples of level 1. The back-links of the level 1 pages are used to construct level 2 and so on. Given this set of pages, the ranking function that ranks a webpage is determined using the CB-OR algorithm (section 2.2). Given r levels and $n_i, i = 0, \cdots, r-1$ pages in each level, the algorithm learns $\mathbf{w}$ and $b_i, i = 1, \cdots, r-1$. During crawling, the level to which a page $\mathbf{x}$ belongs is determined using the formula $\text{argmax}_{i=0,\ldots,r-1}\{I_{\{\mathbf{w}^\top \mathbf{x} - b_i > 0\}}\}$ (assuming $b_0 = -\infty, b_r = \infty$), and the links in the webpage are queued according to this rank.

(Diligenti et al., 2000) proposed a related work, where a context graph is created using layer information and the problem is posed as a multiclass classification problem. During a crawl, the webpages assigned to a particular class are preferred over the webpages assigned to other classes. Thus they inherently assume a ranking over the classes. This ranking information is not considered during training of the multiclass classifier. The present method thus attempts to pose the problem as a ranking problem, and OR is employed; it being more suited for ranking than multiclass classification. The OR formulation has different penalties for misclassification among classes; greater the distance between levels, greater the penalty. Whereas, a multiclass formulation penalizes all miclassifications equally.

4. Experiments and Discussion

The section consists of 2 parts: the first part looks at the performance and scalability of the CB-OR algorithm. The second part discusses the implementation of the crawler and the quality of the pages fetched.

[2] `http://code.google.com/apis/soapsearch/`

Table 1. Comparison of training times (in sec) with **CB-OR**, **SMO-OR** and **SeDuMi** on benchmark datasets. The test set error rate is given in brackets. (CH-California Housing, CS-Census datasets).

	S-Size	CB-OR	SMO-OR	SeDuMi
		sec (err)	sec (err)	sec
CH	10,320	.5 (.623)	551.9 (.619)	112
	13,762	1.5 (.634)	1033.2 (.616)	768.8
	15,482	8.4 (.618)	1142 (.617)	×
	17,202	14.3 (.621)	1410 (.617)	×
	20,230	10.4 (.62)	1838.5 (.62)	×
CS	5,690	.3 (.109)	893 (.128)	20.4
	11,393	.7 (.112)	5281.6 (.107)	108.8
	15,191	1 (.108)	9997.5 (.107)	271.1
	22,331	1.5 (.119)	×	435.7

4.1. Scalability of CB-OR

In this section we present results of scaling experiments on two large benchmark and synthetic datasets and show that the CB-OR algorithm (denoted by **CB-OR**) scales well to large datasets. The two benchmark datasets used are California Housing dataset [3] and Census dataset [4]. California Housing dataset has $20,640$ data points in 8 dimensions and Census dataset has $22,784$ data points in 16 dimensions. The training time and generalization of **CB-OR** is compared with the SMO algorithm for OR formulation (1) (Chu & Keerthi, 2005) (denoted by **SMO-OR**). The training time of solving the dual (6) with SeDuMi (denoted by **SeDuMi**) is also compared in order to show the advantage of using the **CB-OR** algorithm to solve the dual (6) instead of a generic solver like SeDuMi. The benchmark datasets were randomly partitioned into training and test datasets of different sizes to show the scalability of various algorithms. In all cases the results shown are for the parameters that gave best accuracy on the test set. Table 1 summarizes the scaling experiments on California housing dataset and Census dataset. Note that the training time for **CB-OR** is $T_{CB-OR} = T_{clust} + T_{SMO}$ where T_{clust} is the time required to cluster the data using BIRCH and T_{SMO} if the time required to solve the dual (6) using the proposed **CB-OR** algorithm. A '×' in the table implies that the corresponding algorithm failed to complete training, due to lack of memory. The results clearly show that the training time with **CB-OR** is very less when compared to **SMO-OR** and **SeDuMi**. Also the average error rate on the test sets with **CB-OR** on California Housing dataset is 0.6221 and on Census dataset is 0.1122. These are comparable to the average error rates 0.6184 and 0.1172 given by **SMO-OR** on the two benchmark datasets. This shows that the **CB-OR** algorithm achieves similar generalization as **SMO-OR**, but requires very less training time. Note that the reason why **SeDuMi** failed to complete training is presence of large number of clusters in the dataset.

[3] `http://lib.stat.cmu.edu/datasets/`

[4] `http://www.liacc.up.pt/~ltorgo/Regression/DataSets.html`

Table 2. Comparison of training times in sec with **CB-OR** and **SMO-OR** on synthetic dataset.

S-Rate	S-Size	CB-OR	SMO-OR
0.002	10,000	1	182
0.0025	12,500	1	260
0.003	15,000	1	340
0.3	1,500,000	9	×
1	5,000,000	36	×

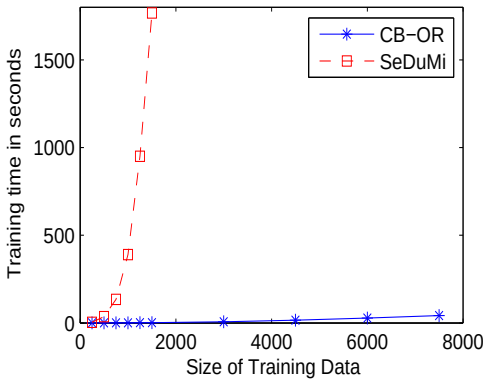

Figure 1. Dashed line represents training time with **SeDuMi** and continuous line that with **CB-OR** on a synthetic dataset.

In order to show that the **CB-OR** algorithm can scale up to very large datasets containing millions of data points, we present scaling results on a large synthetic OR dataset in 2 dimensions having 5 classes. The data points of each class were generated using a GMM with 5 components. Thus the size of problem for **CB-OR** is 25, whereas that for **SMO-OR** it is the size of the whole training set. Table 2 shows the summary of the scaling experiment. In order to show the scalability of the **CB-OR** when compared to **SeDuMi**, scaling experiments on a synthetic ordinal regression dataset with 5 class was done assuming each data point is a cluster. Figure 1 shows the summary of the results. As the results show, the proposed **CB-OR** algorithm requires run time of under 1 minute for few thousands of clusters; whereas **SeDuMi** does not scale well if the number of clusters are large.

Table 3. Datasets: Categories and training set sizes

Category	Seed	1	2	3	4
NASCAR	1705	1944	1747	1464	1177
Soccer	119	750	1109	1542	3149
Cancer	138	760	895	858	660
Mutual Funds	371	395	540	813	1059

Table 4. #R(#I) is the number of relevant (irrelevant) web pages crawled by baseline crawler. It indicates the difficulty in crawling these categories. The next two columns show the harvest rates for Baseline and OR.

Dataset	#R/#I	BL	OR
NASCAR	11530/19646	.3698	.6977
Soccer	10167/9131	.34	.4952
Cancer	6616/12397	.4714	.58
Mutual Fund	9960/10992	.526	.5969

4.2. FOCUS

The purpose of these experiments was to study the performance of the crawler on topics which are difficult to crawl, where difficulty is measured in terms of the harvest rate of the *baseline crawler* (Chakrabarti et al., 1999; Chakrabarti et al., 2002). Nalanda iVia focused crawler [5] was used as the baseline crawler; it implements the focused crawler in (Chakrabarti et al., 1999) with a logistic regression based binary classifier. The *apprentice* (Chakrabarti et al., 2002) was not employed in the experiments. But this does not affect the conclusions, as the apprentice is orthogonal to any crawler and will only lead to performance improvement.

The topics chosen were Mutual Funds, NASCAR, Soccer, and Cancer. Mutual Funds pages are heavily interlinked with pages on investment. This was observed (Chakrabarti et al., 1999) to cause difficulties when crawling. NASCAR pages seem very similar to pages on other motorsports organizations/races (e.g. INDY 500). This is due to the fact that a lot of the pages have similar content including terminology, names, racing circuits etc. **Preparation of the training set:** A set of seed pages was first collected for each topic, from sources like Wikipedia and links returned by queries to general purpose search engines. Then the layered graph was created as mentioned in the previous section. This graph consisted of 5 layers. The number of pages that were collected through backlinks varied for different levels and categories. This was done to test the robustness of the regressor. For each document, a vector of 4000 features is used. The details of the size of the training sets is given in the Table 3.

[5] `http://ivia.ucr.edu/`

Crawling: During the training stage, the input data was split into training and validation data. CB-OR was then used to determine the regressor. The parameters η and W were tuned on the validation set by grid search. During the crawling phase, each newly crawled document was fed to the regressor which returned a label indicating the predicted level. If a page was marked as level 0, 1, 2, or 3, the links from the pages were added to the priority queue; any page marked as level 4 was discarded. But only pages belonging to levels 0, 1 and 2 were considered as relevant pages when measuring the performance of the crawler.

Results: Table 4 gives a comparison of the performance of FOCUS with the baseline crawler. The number presented are the harvest rates as defined in Equation (12). As seen, FOCUS performs better than the baseline crawler on categories considered very difficult.

5. Conclusion

This paper presented 2 key ideas: A fast, easy to implement, algorithm to solve the dual of the novel clustering based OR formulation, which is an SOCP with one SOC constraint. An ordinal regression formulation for the focused crawling problem which removes the need for a negative class definition and is independent of a topic taxonomy. As seen from the experimental evaluation, the new OR formulation can be used to crawl for topics which are difficult to crawl using the baseline crawler. The formulation involves solving a large scale OR problem during training, which can be very efficiently done using the new CB-OR algorithm.

The current system assigns equal priority to all URLs in each page. A strategy that uses the page contents and URL tokens to assign different scores has already been studied in (Chakrabarti et al., 2002). Presently the crawler does not use the DOM structure to prioritize the URLs. Its usage would increase the accuracy of the simple model of the web being used currently.

Appendix

Derivation of Dual: In this section we derive the dual of the primal formulation (4). Using the dual norm $\|\mathbf{w}\|_2 = \sup_{\|\mathbf{u}\|_2 \leq 1} \mathbf{w}^\top \mathbf{u}$, the Lagrangian function can be written as:

$$\mathcal{L} = \sum_{i=1}^{r} \sum_{j=1}^{n_i} \Big\{ \xi_i^j + \xi_i^{*j} + \alpha_i^j C_i^j$$

$$+ \quad \alpha_i^{*j} C_i^{*j} - \beta_i^j \xi_i^j - \beta_i^{*j} \xi_i^{*j} \Big\} - \sum_{i=1}^{r} \gamma_i (b_i - b_{i-1}) + \rho(\mathbf{w}^\top \mathbf{u} - W) \quad (13)$$

where the Lagrange multipliers satisfy $\alpha_i^j \geq 0, \alpha_i^{*j} \geq 0$, $\beta_i^j \geq 0, \beta_i^{*j} \geq 0, \gamma_i \geq 0, \rho \geq 0, \|\mathbf{u}\|_2 \leq 1$ and $C_i^j = \mathbf{w}^\top \phi(\mu_i^j) - b_i + 1 - \xi_i^j + r_i^j W, C_i^{*j} = 1 - \xi_i^{*j} + r_i^j W + b_{i-1} - \mathbf{w}^\top \phi(\mu_i^j)$. The KKT conditions for optimality can be summarized as follows:

$$\begin{aligned}
\nabla_{\mathbf{w}} \mathcal{L} = 0 \quad &\Rightarrow \quad \rho \mathbf{u} = \sum_{i=1}^{r} \sum_{j=1}^{n_i} \left(\alpha_i^{*j} - \alpha_i^j \right) \phi(\mu_i^j) \\
\frac{\partial \mathcal{L}}{\partial b_{i:1 \leq i \leq r-1}} = 0 \quad &\Rightarrow \quad \sum_{j=1}^{n_{i+1}} \alpha_{i+1}^{*j} + \gamma_{i+1} = \sum_{j=1}^{n_i} \alpha_i^j + \gamma_i \\
\frac{\partial \mathcal{L}}{\partial \xi_i^j} = 0, \frac{\partial \mathcal{L}}{\partial \xi_i^{*j}} = 0 \quad &\Rightarrow \quad \alpha_i^j + \beta_i^j = 1, \alpha_i^{*j} + \beta_i^{*j} = 1 \\
\text{Compl.Slack.} \quad &\Rightarrow \quad \alpha_i^j C_i^j = 0, \ \alpha_i^{*j} C_i^{*j} = 0 \\
&\Rightarrow \quad \beta_i^j \xi_i^j = 0, \ \beta_i^{*j} \xi_i^{*j} = 0 \\
&\Rightarrow \quad \gamma_i (b_i - b_{i-1}), \rho(\mathbf{w}^\top \mathbf{u} - W) \quad (14)
\end{aligned}$$

Since $b_0 = -\infty, b_r = \infty$, the complimentary slackness conditions immediately show that at optimality $\alpha_1^{*j} = \alpha_r^j = \gamma_1 = \gamma_r = 0 \ \forall \ j$. With these boundary conditions, one can easily eliminate the γ_i multipliers from the KKT conditions using $\frac{\partial \mathcal{L}}{\partial b_i} = 0$, giving the following conditions: $S_i^* \leq S_i, \ \forall \ i = 1, \ldots, r-2, \ S_{r-1}^* = S_{r-1}$ where $S_i = \sum_{k=1}^{i} \sum_{j=1}^{n_k} \alpha_i^k$ and $S_i^* = \sum_{k=2}^{i+1} \sum_{j=1}^{n_k} \alpha_i^{*k}$. Now let us denote the column vector containing the α_i^j by α and that containing α_i^{*j} by α^*. We once again note that the entries corresponding to $i = 1$ are zero in α^* and those corresponding to $i = r$ are zero in α. Also let us denote the vector containing $1 + r_i^j W$ with $\mathbf{d}$ and the matrix containing the dot products of centers $\phi(\mu_i^j)$ with each other as $\mathbf{K}$. Using this notation, one can write the dual of the CB-OR formulation as given in (5).

Acknowledgements: This project is partially supported by AOL India Pvt Ltd and DST, Government Of India(DST/ECA/CB/660).

References

Aggarwal, C., Al-Garawi, F., & Yu, P. (2001). Intelligent crawling on the World Wide Web with arbitrary predicates. *Proc. of 10th Intl. Conf. on WWW*.

Chakrabarti, S., Punera, K., & Subramanyam, M. (2002). Accelerated focused crawling through online relevance feedback. *Proc. of 11th Intl. Conf. on World Wide Web*, 148–159.

Chakrabarti, S., van den Berg, M., & Dom, B. (1999). Focused Crawling: A New Approach for Topic-Specific Resource Discovery. *WWW Conference*.

Chu, W., & Keerthi, S. (2005). New approaches to support vector ordinal regression. *Proc. of 22nd Intl. Conf. on Machine learning*, 145–152.

Crammer, K., & Singer, Y. (2002). Pranking with ranking. *NIPS*, *14*.

Davison, B. (2000). Topical locality in the Web. *Proc. of 23rd Intl. Conf. on Research and development in Information Retrieval*, 272–279.

Diligenti, M., Coetzee, F., Lawrence, S., Giles, C., & Gori, M. (2000). Focused crawling using context graphs. *Proc. of 26th Intl. Conf. on VLDB*.

Erdougan, E., & Iyengar, G. (2006). An active set method for single-cone second-order cone programs. *SIAM J. on Optimization*, *17*, 459–484.

Grangier, D., & Bengio, S. (2005). Exploiting Hyperlinks to Learn a Retrieval Model. *Proc. of NIPS Workshop*.

Har-Peled, S., Roth, D., & Zimak, D. Constraint classification: A new approach to multiclass classification and ranking. *NIPS*.

Herbrich, R., Graepel, T., & Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. *Advances in Large Margin Classifiers*, 115–132.

Kleinberg, J. (1999). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, *46*, 604–632.

Nath, J. S., Bhattacharyya, C., & Murty, M. N. (2006). Clustering based large margin classification: a scalable approach using socp formulation. *Proc. of 12th Intl. Conf. on KDD* (pp. 674–679).

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods—Support Vector Learning* (pp. 185–208). Cambridge, MA: MIT Press.

Shashua, A., & Levin, A. (2003). Ranking with large margin principle: Two approaches. *NIPS*, *15*.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. *Proc. of Intl. Conf. on Management of data*, 103–114.