Kernel Regression with Order Preferences*

Xiaojin Zhu and **Andrew B. Goldberg**
Department of Computer Sciences
University of Wisconsin, Madison, WI 53706, USA

Abstract

We propose a novel kernel regression algorithm which takes into account order preferences on unlabeled data. Such preferences have the form that point x_1 has a larger target value than that of x_2, although the target values for x_1, x_2 are unknown. The order preferences can be viewed as side information or a form of weak labels, and our algorithm can be related to semi-supervised learning. Learning consists of formulating the order preferences as additional regularization in a risk minimization framework. We define a linear program to effectively solve the optimization problem. Experiments on benchmark datasets, sentiment analysis, and housing price problems show that the proposed algorithm outperforms standard regression, even when the order preferences are noisy.

Introduction

We propose a novel algorithm for kernel regression. The proposed regression algorithm is able to incorporate domain knowledge about the relative order of target values on unlabeled examples. As a motivating example, consider the task of predicting real estate prices. The price of a house varies significantly depending on its location and many other factors. However, a domain expert may determine that everything being "roughly equal," the feature *number of bedrooms* determines the order of house prices. For instance, a 4-bedroom house is more expensive than a 3-bedroom one.

At first glance, it may appear that such knowledge can be enforced by a positive correlation between the feature and the target. However, modeling such knowledge as positive correlation can be difficult in *non-linear* kernel regression, because of the non-linear feature mapping. Besides, in general a correlation may only hold for part of the range of the feature value, and it would be inappropriate to force the correlation across the range. We would like a more general approach to capture such knowledge.

*We thank Grace Wahba for discussions on the representer theorem; Olvi Mangasarian, Edward Wild and Michael Ferris on optimization; and Wei Chu on the benchmark datasets. Research supported in part by Wisconsin Alumni Research Foundation.

We propose to encode such domain knowledge with *order preferences* on unlabeled examples. That is, for all pairs of unlabeled examples x_i, x_j satisfying the "roughly equal" condition, such domain knowledge specifies the *order* between their target values $f(x_i)$ and $f(x_j)$, even though their actual target values are unknown. Respecting the domain knowledge then amounts to incorporating the order preferences into a kernel regression framework. When labeled data is scarce, these order preferences should improve our regression model.

Another practical application of our approach is in predicting Internet file transfer rates based on network properties like round trip time, available bandwidth, queuing delay, package loss rate, and so on (Mizra *et al.* 2007). The features have intuitive impact on transfer rate, but the exact relation is highly non-linear and unknown. We can, however, easily create order preferences on unlabeled data using domain knowledge. In general, order preferences can encode certain complex domain knowledge.

Regression with Order Preferences

Let us formally define our regression problem. In addition to a labeled training set $\{(x_1, y_1), \ldots, (x_l, y_l)\}$, we assume that we are given p *order preferences* between pairs of unlabeled examples. An order preference is defined by a tuple (i, j, d, w), with the interpretation that we would like $f(x_i) - f(x_j) \geq d$. As discussed below, we encode it as a soft preference rather than a hard constraint. The scalar $w \geq 0$ is the weight (confidence) for the preference. Obviously knowing the order preferences is much weaker than knowing the labels of the unlabeled examples. In this sense the preferences are a form of weakly labeled data or side information. We would like to use them to improve regression.

It is possible to represent the order preferences as directed edges in a graph (Dekel, Manning, & Singer 2004), where the edges represent asymmetric order information. However, it is worth noting that order preferences can also encode similarity. For example, the two preferences $(i, j, 0, w), (j, i, 0, w)$ encode $f(x_i) = f(x_j)$. More generally, the two preferences $(i, j, -\epsilon, w), (j, i, -\epsilon, w)$ encode closeness: $|f(x_i) - f(x_j)| \leq \epsilon$. It is also easy to encode $a \leq f(x_i) - f(x_j) \leq b$. As special cases of order preference, one can also encode unary preferences $f(x_i) \leq g(x_i)$, $f(x_i) = g(x_i)$, or $f(x_i) \geq g(x_i)$, where g is some given function. The unary preferences are closely related to the work of Mangasarian et al. (2004), which adds them to kernel machines.

Our approach to add order preferences to kernel regression is to treat them as regularization. Recall the standard risk minimization framework for kernel regression is

$$\min_{\mathbf{f}\in\mathcal{H}} \quad \textstyle\sum_{i=1}^{l} c(x_i, y_i, f(x_i)) + \lambda\Omega(\|\mathbf{f}\|_{\mathcal{H}}), \tag{1}$$

where $\mathcal{H}$ is the Reproducing Kernel Hilbert Space (RKHS) induced by some kernel, $c()$ is a loss function for regression, λ is a weight parameter on the regularizer, and $\Omega()$ is a monotonic increasing function.

With the order preferences we now define an additional regularization term $r(x, f)$. Intuitively if the function $\mathbf{f}$ satisfies all order preferences, r should be zero; if $\mathbf{f}$ violates some, r increases. A natural choice is to use a shifted hinge function: for order preference (i, j, d, w), the regularization term for this single preference is $w \max(d - (f(x_i) - f(x_j)), 0)$. That is, it is zero if the preference is satisfied; otherwise it is the amount the preference is violated, weighted by w. As a side note, we point out that if we have two preferences $(i, j, -\epsilon, w), (j, i, -\epsilon, w)$, this would form the ϵ-insensitive loss (Smola & Schölkopf 2004).

We define the regularization term $r(x, f)$ as the sum of shifted hinge function on all order preferences:

$$r(x, f) = \sum_{q=1}^{p} w_q \max(d_q - (f(x_{iq}) - f(x_{jq})), 0). \tag{2}$$

We note that order preferences have been used in ranking problems (Herbrich, Obermayer, & Graepel 2000; Burges *et al.* 2005; Yu *et al.* 2006; Chu & Ghahramani 2005); in particular (Joachims 2002) employed a similar shifted hinge function for ranking. However they have not been used in regression before. Our problem is

$$\min_{\mathbf{f}\in\mathcal{H}} \quad \textstyle\sum_{i=1}^{l} c(x_i, y_i, f(x_i)) + \lambda_1\Omega(\|\mathbf{f}\|_{\mathcal{H}}) + \lambda_2 r(x, f). \tag{3}$$

A Linear Program Formulation

To fully specify the above problem, we choose to use the ϵ-insensitive loss $c(x, y, f) = |y - f|_\epsilon$ in support vector regression:

$$|y - f|_\epsilon = \begin{cases} 0 & \text{if } |y - f| \le \epsilon \\ |y - f| - \epsilon & \text{otherwise.} \end{cases} \tag{4}$$

We further choose $\Omega(\|f\|_{\mathcal{H}})$ to be a linear function, in this case the 1-norm of the dual parameters discussed below, resulting in 1-norm support vector machines (Bradley & Mangasarian 1998; Bi *et al.* 2003; Zhu *et al.* 2004). The formulation originates from generalized support vector machines (Mangasarian 2000). Such 1-norm support vector machines are comparable in performance to the standard 2-norm support vector machines, but with the advantage that they can be solved as *linear programs*, which tends to be more efficient.

The solution can be characterized by a representer theorem (Kimeldorf & Wahba 1971; Schölkopf, Herbrich, & Smola 2001): The minimizer $\mathbf{f}^* \in \mathcal{H}$ admits the form $f^*(x) = \sum_{i=1}^{l+2p} \alpha_i K(x_i, x)$, where x_i ranges from the labeled examples to the unlabeled examples involved in the p order preferences. The proof uses the standard orthogonality argument, and is omitted for space consideration.

Let $K(x, \mathbf{x}_{1:l})$ denote the row vector of kernel values between a point x and the labeled data $\mathbf{x}_{1:l}$. We represent our function $\mathbf{f}$ in dual form by

$$f(x) = K(x, \mathbf{x}_{l:l})\alpha + \alpha_0 \tag{5}$$

where α is a column vector of dual parameters, one for each labeled point; α_0 is a bias scalar. This amounts to approximating the representer theorem by setting dual parameters not on the labeled data to zero for a sparse representation. Our linear-program regression problem is

$$\begin{aligned} \min_{\alpha,\alpha_0} \quad & \tfrac{1}{l}\textstyle\sum_{i=1}^{l} |y_i - f(x_i)|_\epsilon + \lambda_1\|\alpha\|_1 + \\ & \lambda_2 \tfrac{1}{p}\textstyle\sum_{q=1}^{p} w_q \max(d_q - (f(x_{iq}) - f(x_{jq})), 0), \end{aligned} \tag{6}$$

where $\|\alpha\|_1 = \sum_{i=1}^{l} |\alpha_i|$ is the 1-norm of α. The bias α_0 is not regularized.

We transform (6) into a standard linear program by introducing auxiliary variables for the three terms respectively. Let $\mathbf{1}$ be the all-one vector, ξ an l-vector of slack variables, η an l-vector, ν a p-vector, $\mathbf{d}$ the difference vector, $\mathbf{w}$ the weight vector, $K(\mathbf{x}^i_{1:p}, \mathbf{x}_{1:l})$ the $p \times l$ kernel matrix between the first points in the order constraints and the labeled data, and $K(\mathbf{x}^j_{1:p}, \mathbf{x}_{1:l})$ the same sized kernel matrix between the second points in the order constraints and the labeled data. Vector inequalities are element-wise. With standard transform techniques, our linear program for kernel regression with order preferences can be written as:

$$\begin{aligned} & \min_{\alpha,\alpha_0,\xi,\eta,\nu} \quad \tfrac{1}{l}\mathbf{1}^\top\xi + \lambda_1\mathbf{1}^\top\eta + \tfrac{\lambda_2}{p}\mathbf{w}^\top\nu \\ & \text{s.t.} -\xi - \epsilon\mathbf{1} \le \mathbf{y}_{1:l} - K(\mathbf{x}_{1:l}, \mathbf{x}_{1:l})\alpha - \alpha_0\mathbf{1} \le \xi + \epsilon\mathbf{1} \\ & \quad \xi \ge 0 \\ & \quad -\eta \le \alpha \le \eta \\ & \quad (K(\mathbf{x}^i_{1:p}, \mathbf{x}_{1:l}) - K(\mathbf{x}^j_{1:p}, \mathbf{x}_{1:l}))\alpha \ge \mathbf{d} - \nu \\ & \quad \nu \ge 0. \end{aligned} \tag{7}$$

This is a linear program with $3l + p + 1$ variables and $5l + 2p$ constraints. The global optimal solution can be found efficiently.

Connections to Semi-Supervised Learning

It is instructive to note that several semi-supervised learning approaches can be expressed in a similar form as (3). In particular, manifold regularization (Belkin, Niyogi, & Sindhwani 2004) uses

$$r(x, f) = \sum_{i,j\in U} w_{ij}(f(x_i) - f(x_j))^2,$$

where U is the unlabeled data, and w_{ij} represents the similarity between x_i, x_j based on domain knowledge. S3VMs (Collobert *et al.* 2006; Joachims 1999b) uses

$$r(x, f) = \sum_{i\in U} \max(1 - |f(x_i)|, 0)$$

to attempt to push unlabeled examples out of the margin. Co-train style multiview learning (Brefeld *et al.* 2006; Sindhwani, Niyogi, & Belkin 2005) uses

$$r(x, f) = \sum_{u,v=1}^{M} \sum_{i \in U} (f_u(x_i) - f_v(x_i))^2$$

for the M different views, to encourage them to make the same prediction on the same unlabeled example. These methods and our order preferences all encode some domain knowledge other than labels. One might establish many order preferences on unlabeled data. For example, higher bandwidth, shorter delay and fewer package loss leads to higher file transfer rates. They can all be viewed as unlabeled-data-dependent regularizers $r(x, f)$. Our order preferences may contain slightly stronger information, and we view them as filling in the continuum between supervised learning and semi-supervised learning. It is possible to combine order preferences with existing semi-supervised learning methods by adding the respective $r(x, f)$ terms together (with appropriate weights) to form a new regularizer.

Experiments

We demonstrate the benefit of order preferences with four groups of experiments. We implemented our linear program (7) using CPLEX. All experiments ran quickly. Solving the LP for each trial takes 0.2 to 0.5 seconds depending on the number of order preferences and unlabeled data size. In all experiments, ϵ in the ϵ-insensitive loss (4) was set to 0, and preference weights $\mathbf{w}$ were set to 1. We use the acronym SSL for (7), and SVR for the corresponding standard 1-norm support vector regression (i.e., $\lambda_2 = 0$). We also experimented with standard 2-norm support vector regression using SVM$^{\text{light}}$ (Joachims 1999a), and the results were comparable to SVR and not reported here.

A Toy Example

First we use a toy example to illustrate order preferences. We constructed a polynomial function of degree 3 as our target (the dotted line in Figure 1(a)). We randomly sampled three points (the open circles) from the target function as training data and gave them to SVR. For this experiment we set $\lambda_1 = 0$. Since there were not enough training data points, SVR produced a fit (the dashed line) through the training points but very different from the target.

We then randomly selected a pair of unlabeled points $-0.15, 0.30$. Note they did not coincide with the training points. Without revealing the actual target values at these points, we constructed an order preference using their true order: $(0.30, -0.15, 0, 1)$, or equivalently $f(0.30) - f(-0.15) \geq 0$. Note we set $d = 0$ so that the order preference specified their order but not the true difference; hence it was weaker. We set $w = 1$. In Figure 1(a) the order preference is shown at the lower left as a line linking the two unlabeled points (black dots). The point with the larger value has a larger dot. SVR happened to violate the order preference. With the three training points and this order preference, SSL produced a better fit (the solid line).

In Figure 1(b) we added more order preferences, generated similarly from random unlabeled point pairs and their true order. Note some preferences were already satisfied by SVR. The SSL function was further improved. We consistently observed such behavior in repeated random trials.

Benchmark Datasets

We experimented with five regression benchmark datasets (Boston, Abalone, Computer, California, Census; Available at `http://www.niaad.liacc.up.pt/~ltorgo/Regression/DataSets.html`), and report results on all of them. One difficulty in working with such standard datasets is creating sensible order preferences on unlabeled data. Ideally the order preferences would be prepared by experts with domain knowledge on the tasks. Lacking such experts, we had to create simulated order preferences from the relation of true values on unlabeled points (more details later; Note, however, we never give out the true values themselves). Therefore our results on benchmark datasets should be viewed as "oracle experiments." Nonetheless they are useful indications of how well our regression would perform given such domain knowledge.

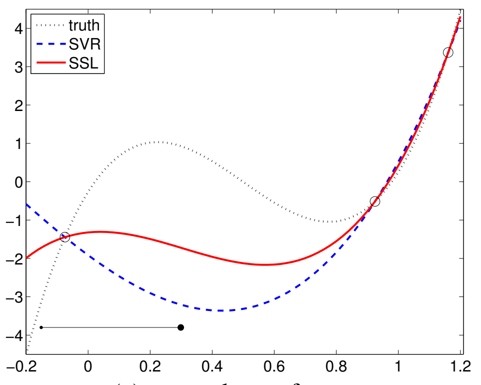
For each benchmark dataset, we normalized its input features to zero mean, unit variance. For categorical features with k distinct values, we mapped them into indicator vectors of length k. We used Radial Basis Function (RBF) kernels $k(x, x') = \exp(-\sigma \|x - x'\|^2)$ for all datasets. We used 5-fold cross validation to find the optimal RBF bandwidth σ, and SVR 1-norm weight λ_1. The parameters were tuned for SVR on a 9×9 logarithmic grid in $10^{-4} \leq \sigma \leq 10^4$ and $10^{-4} \leq \lambda_1 \leq 10^4$. We simply fixed λ_2 at 1. This is partly justified by the fact that in (6), the 'shifted hinge function' is on a similar scale to the ϵ-insensitive loss; both incur a linear penalty when violated. Tuning λ_2 might produce better results than reported here, but with limited labeled data (which has been used to tune λ_1 and σ for SVR already) it is hard to do.

All experiments were repeated for 20 random trials. Different algorithms shared the same random trials so we could perform paired statistical tests. In each trial we split the data into three parts: l labeled points, u unlabeled points that were used to generate order preferences, and test points that were the rest of the dataset (see Table 1 Partition). Test points were unseen by either algorithm during training. All results we report are test-set mean-absolute-error over the 20 trials. Let t be the test set size. Test-set mean-absolute-error is defined as $\sum_{i \in \text{test}} |y_i - f(x_i)|/t$. We address the following questions:

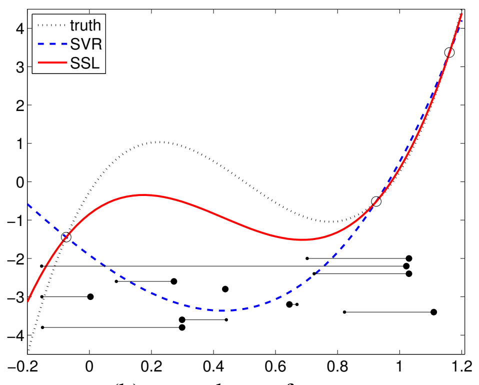
Can order preferences improve regression? We randomly sampled with replacement $p = 1000$ pairs (x_i, x_j) from the u unlabeled points. For each sampled pair, we generated an order preference from the true target values y_i, y_j. Without loss of generality let $y_i \geq y_j$. Our simulated order preference was

$$f(x_i) - f(x_j) \geq 0.5(y_i - y_j). \qquad (8)$$

Let us explain our order preferences. We could have created the 'perfect' order preferences with the pair: $f(x_i) - f(x_j) \geq y_i - y_j$ and $f(x_i) - f(x_j) \leq y_i - y_j$. They together



(a) one order preference



(b) ten order preferences

Figure 1: A toy example comparing SVR and SSL, showing the benefit of order preferences.

encode $f(x_i) - f(x_j) = y_i - y_j$. But in real tasks it might be difficult to know the exact difference $y_i - y_j$, so we did not do that. On the other hand, with inequality preferences we could have set $f(x_i) - f(x_j) \geq 0$. It would only encode order, without any information on the actual difference. But in real tasks one might have some rough estimate of the difference, and (8) was meant to simulate this estimate. Table 1 compares the test-set mean-absolute-error of SVR and SSL. The differences on all datasets are significant with a paired t-test at the 0.01 level. We conclude that, with the order preferences (8), SSL significantly improves regression performance over SVR.

What if we change the number of order preferences p? One expects a larger gain with more order preferences p. We systematically varied p from 10 to 5000, keeping everything else the same as in Table 1. Figure 2(a) shows that it was indeed the case. A very small p sometimes hurts SSL, making it worse than SVR. But as p grows larger SSL rapidly improves, and levels off at around $p = 100$. This indicates that one needs only a moderate amount of order preferences to enjoy the benefit.

What if we change the labeled data size l? The benefit of order preferences is expected to diminish with more labeled data. We fixed the number of order preferences $p = 1000$, and systematically varied l. As expected, Figure 2(b) shows that SSL is most useful when l is small, and the benefit reduces as l grows.

How precise do the order preferences need to be? Extending (8), one can define order preferences as $f(x_i) - f(x_j) \geq \beta(y_i - y_j)$ where β controls how precise they are. As mentioned earlier, $\beta = 0$ only supplies order information, and a larger β estimates the differences. We varied β from 0 to 2 (over-estimate) for the experiments in Table 1. Figure 2(c) shows that with only the order ($\beta = 0$) SSL already outperformed SVR. With a conservative estimate of the differences ($0 < \beta < 1$) SSL was even better. However larger β seems to be inferior. This might be advantageous in practice, since one does not need to know the precise differences, and can err on the safe side.

Sentiment Analysis in Movie Reviews

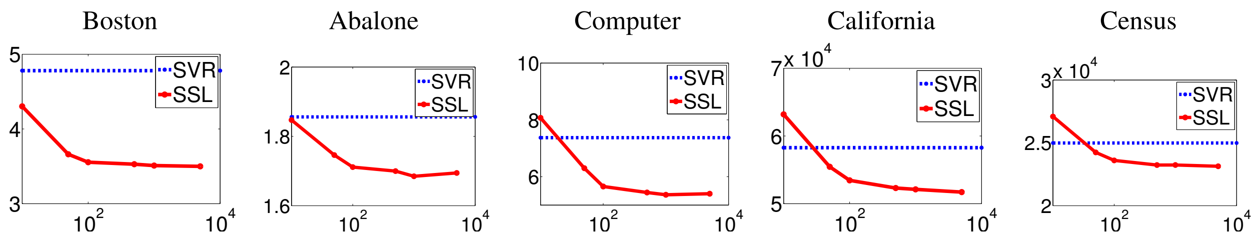
We next experimented with the real-world problem of sentiment analysis in movie reviews. Given a movie review text document x, we would like to predict $f(x)$, the rating (e.g., '4 stars') given to the movie by the reviewer. We assume that by looking at the wording of unlabeled reviews, one can determine that some movies will likely be rated higher than others (even though we do not know their actual ratings). These are incorporated as order preferences. We worked on the "scale dataset v1.0" with continuous ratings, available at `http://www.cs.cornell.edu/people/pabo/movie-review-data/` and first used in (Pang & Lee 2005). It contains four authors with 1770, 902, 1307, 1027 reviews respectively. For each author, we varied $l \in \{30, 60, 120\}$, and let $u = 500, p = 500$. The remaining reviews were test examples. Each experiment was repeated for 20 random trials. All reported results are test-set mean-absolute-error. Each review document was represented as a word-presence vector, normalized to sum to 1. We used a linear kernel, set $\lambda_1 = 10^{-7}$ and $\lambda_2 = 1$.

As a proxy for expert knowledge, we used a completely separate "snippet dataset" also located at the above URL. The snippet dataset is very different from the scale dataset: it contains single punch line sentences (snippets) instead of full reviews; the snippets have binary positive/negative labels instead of continuous ratings; it comes from different authors on different movies. We trained a standard binary, linear-kernel SVM classifier g on the *snippet* data using SVM$^{\text{light}}$. We then applied g on random pairs of unlabeled movie reviews x_i, x_j in the *scale* dataset. The order of the continuous margin output $g(x_i), g(x_j)$ serves as our proxy for expert knowledge[1]. Since this is a very crude and noisy estimate, we created an order preference $(i, j, 0, 1)$ only if $g(x_i) - g(x_j) > 0.25$, where 0.25 is an arbitrary threshold. Note we set $d = 0$ since we do not know the

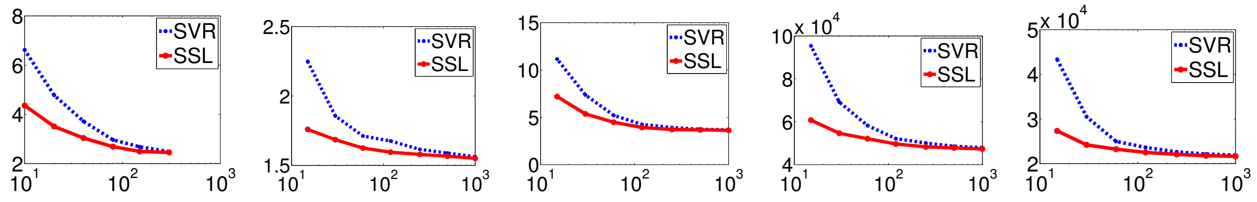
[1]Our use of g simulates a layman (not an expert) reading two reviews and saying "the author liked this one more than that one." This layman does not have enough experience to predict the actual star ratings, but is able to tell that one sounds more positive than the other.

Table 1: Benchmark data. All improvements are statistically significant.

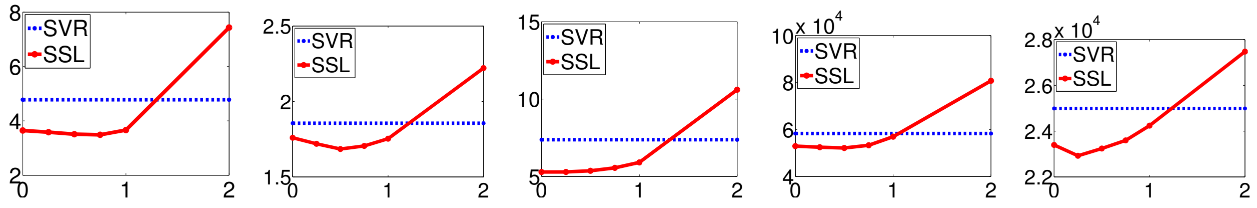
Dataset	Partition		Mean absolute error		Improvement
	dim	$l/u/$test	SVR	SSL	
Boston	13	20/200/286	4.780 ± 1.351	3.511 ± 0.376	27%
Abalone	8	30/1000/3147	1.856 ± 0.180	1.685 ± 0.102	9%
Computer	21	30/1000/7162	7.373 ± 3.445	5.364 ± 0.998	27%
California	8	60/1000/19580	58268 ± 4435	52120 ± 1843	11%
Census	16	60/1000/21724	24992 ± 1377	23241 ± 901	7%



(a) The effect of the number of order preferences p (x-axis).



(b) The effect of labeled data size l (x-axis).



(c) The effect of the difference scaling factor β (x-axis).

Figure 2: The effect of various parameters on SSL on the Benchmark data. y-axis is test-set mean-absolute-error.

difference in rating. Table 2 presents the results of our sentiment analysis experiments. As expected, SSL is most useful when l is small, and the gain over SVR gradually diminishes with larger l. SSL leads to improvements in all cases, and the differences are significant (*) with paired t-tests at the 0.05 level in about half of the cases[2]. We expect better order preferences from advanced natural language processing (e.g., parsing) to bring larger improvements.

Predicting Housing Prices Using Heuristic Order Preferences

As a final real-world experiment, we played the role of real estate experts to carry out the scenario introduced in the beginning of the paper. We used the same California dataset in Table 1, but this time with order preferences derived from domain knowledge instead of oracles. The task is to predict the median house value for 20640 groups of houses throughout the state. With other factors being roughly equal, we believe the value is largely determined by the number of bedrooms. We decided that two groups are "roughly equal" if they are located within 25 miles of each other (i.e., they are in the same community), their median house ages differ by at most 10 years, and they are inhabited by residents whose median income level differs by at most \$1000. We repeated the experimental setup in the benchmark section, and for each random trial, we created approximately 1200 order preferences. Specifically, for all pairs of housing groups in the labeled and unlabeled data that satisfy the "roughly equal" criteria, we created a preference that the group with more bedrooms has a higher target value. We omitted preferences between two labeled groups, since they are either redundant or incorrect. We set $w = 1$ and $d = 0$, and used the same λ parameters as in the benchmark section. Note that the order preferences are created without any knowledge of the actual target values, and that the relations we constructed

[2]As a sanity check, we also experimented with *wrong* order preferences by intentionally flipping all preferences $(i, j, 0, 1)$ into $(j, i, 0, 1)$. As expected, SSL with wrong orders became *worse* than SVR by 1% – 13% for different authors at $l = 120$.

Table 2: Movie review sentiment analysis mean-absolute-error for each author.

Dataset	$l/u/$test	SVR	SSL	Improvement
Author (a)	30/500/1240	0.1383 ± 0.0072	0.1362 ± 0.0028	1.5%
	60/500/1210	0.1323 ± 0.0042	0.1311 ± 0.0025	0.9%
	120/500/1150	0.1224 ± 0.0042	0.1219 ± 0.0024	0.4%
Author (b)	30/500/372	0.1645 ± 0.0146	0.1540 ± 0.0046	* 6.4%
	60/500/342	0.1514 ± 0.0063	0.1496 ± 0.0046	* 1.2%
	120/500/282	0.1431 ± 0.0063	0.1416 ± 0.0062	* 1.0%
Author (c)	30/500/777	0.1405 ± 0.0163	0.1357 ± 0.0070	3.4%
	60/500/747	0.1268 ± 0.0072	0.1258 ± 0.0038	0.8%
	120/500/687	0.1150 ± 0.0048	0.1138 ± 0.0047	1.0%
Author (d)	30/500/497	0.1433 ± 0.0151	0.1350 ± 0.0052	* 5.8%
	60/500/467	0.1366 ± 0.0104	0.1293 ± 0.0037	* 5.3%
	120/500/407	0.1256 ± 0.0092	0.1226 ± 0.0038	2.4%

are highly non-linear. We found that the heuristic preferences led to a 6% reduction in test-set mean-absolute-error in SSL (54664 ± 2521) compared to SVR (58268 ± 4435). The difference is statistically significant with a paired t-test at the 0.01 level.

This experiment demonstrates that order preferences with some noise can still be beneficial. In fact, a post-experimental analysis of the created order preferences revealed that only 70% were actually accurate (i.e., 30% of "roughly equal" housing group pairs do *not* have the predicted relation based on bedrooms). We expect our method to extend well to new tasks (e.g., predicting Internet file transfer rates) where large numbers of reasonably accurate order preferences can be generated automatically.

Conclusions

We presented a novel kernel regression algorithm with order preferences, formulated as a linear program. We showed that even with noisy, heuristic order preferences, the regression performance is improved. Our algorithm can be easily extended beyond regression. For example, one future direction is to apply order preferences to ordinal classification (Chu & Keerthi 2005).

References

Belkin, M.; Niyogi, P.; and Sindhwani, V. 2004. Manifold regularization: A geometric framework for learning from examples. Technical Report TR-2004-06, University of Chicago.

Bi, J.; Bennett, K.; Embrechts, M.; Breneman, C.; and Song, M. 2003. Dimensionality reduction via sparse support vector machines. *JMLR* 3:1229–1243.

Bradley, P., and Mangasarian, O. 1998. Feature selection via concave minimization and support vector machines. In *ICML98*.

Brefeld, U.; Gaertner, T.; Scheffer, T.; and Wrobel, S. 2006. Efficient co-regularized least squares regression. In *ICML06*.

Burges, C.; Shaked, T.; Renshaw, E.; Lazier, A.; Deeds, M.; Hamilton, N.; and Hullender, G. 2005. Learning to rank using gradient descent. In *ICML05*.

Chu, W., and Ghahramani, Z. 2005. Gaussian processes for ordinal regression. *JMLR* 6(July):1019–1041.

Chu, W., and Keerthi, S. S. 2005. New approaches to support vector ordinal regression. In *ICML05*, 145–152.

Collobert, R.; Sinz, F.; Weston, J.; and Bottou, L. 2006. Large scale transductive SVMs. *JMLR* 7(Aug):1687–1712.

Dekel, O.; Manning, C.; and Singer, Y. 2004. Loglinear models for label-ranking. In *NIPS 16*.

Herbrich, R.; Obermayer, K.; and Graepel, T. 2000. Large margin rank boundaries for ordinal regression. In Smola, A. J.; Bartlett, P.; Schölkopf, B.; and Schuurmans, D., eds., *Advances in Large Margin Classifiers*. MIT Press. 115–132.

Joachims, T. 1999a. Making large-scale svm learning practical. In Schölkopf, B.; Burges, C.; and Smola, A., eds., *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

Joachims, T. 1999b. Transductive inference for text classification using support vector machines. In *ICML99*, 200–209. Morgan Kaufmann, San Francisco, CA.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *KDD02*. ACM Press.

Kimeldorf, G., and Wahba, G. 1971. Some results on Tchebychean spline functions. *Journal of Mathematics Analysis and Applications* 33:82–95.

Mangasarian, O. L.; Shavlik, J. W.; and Wild, E. W. 2004. Knowledge-based kernel approximation. *JMLR* 5:1127–1141.

Mangasarian, O. 2000. Generalized support vector machines. In Smola, A. J.; Bartlett, P.; Schölkopf, B.; and Schuurmans, D., eds., *Advances in Large Margin Classifiers*. MIT Press. 135–146.

Mizra, M.; Sommers, J.; Barford, P.; and Zhu, X. 2007. A machine learning approach to TCP throughput prediction. In *ACM SIGMETRICS*.

Pang, B., and Lee, L. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Association for Computational Linguistics*.

Schölkopf, B.; Herbrich, R.; and Smola, A. J. 2001. A generalized representer theorem. In *COLT*.

Sindhwani, V.; Niyogi, P.; and Belkin, M. 2005. A co-regularized approach to semi-supervised learning with multiple views. In *Proc. of the 22nd ICML Workshop on Learning with Multiple Views*.

Smola, A., and Schölkopf, B. 2004. A tutorial on support vector regression. *Statistics and Computing* 14:199–222.

Yu, S.; Yu, K.; Tresp, V.; and Kriegel, H.-P. 2006. Collaborative ordinal regression. In *ICML06*.

Zhu, J.; Rosset, S.; Hastie, T.; and Tibshirani, R. 2004. 1-norm support vector machines. In *NIPS 16*.