Joint Supervised Learning of Ratings and Rankings

Rafael Moreno Ferrer, Avinayan Senthi Velayutham, and Ying Wang
Advisor: Ramesh Nallapati

***Abstract*— Supervised recommender systems can be broadly classified into (a) regression models that predict ratings of unlabeled items and (b) ranking models that rank items according to an order of interest. Although both models assess the value of an item, regression may be less precise than ranking because the regressor does not learn a distinction between two different training items with identical ratings. Ranking systems do not have this drawback because they assume perfect ordering between all pairs of training items. However, ranking requires the annotator to make judgments between all pairs of items, which can be more time consuming than simply rating each item independently.**

In this work, we investigate the possibility of combining rating and ranking systems. Our goal is to assess whether ratings prediction can be improved by supplementing a regression model with ranking information. We present two Support Vector Machine models that learn jointly from ratings and rankings.

I. Introduction

Rating and ranking are well known problems in machine learning with widespread applications. Rating is common in services like the Netflix Movie Recommender, while ranking is essential in web search. In ratings prediction, a regression model trains on a set of user-annotated ratings, typically on an integer scale of 1 to 5, and learns to predict absolute ratings for unlabeled items. In ranking, the model trains on a set of pairwise rankings annotated by the user and assigns relative values to unlabeled items in order to place them in an ordered list.

Although these problems are related, they have complementary trade-offs in terms of acquiring human annotation for supervised learning. To acquire labeled data for ranking, the human annotator needs to consider all pairs of items, which is significantly more expensive than assigning a rating to each individual item. Conversely, ranking annotations often provide fine-grained information that ratings cannot, because in rating systems with a small set of rating values, many items will have identical ratings.

In this work, we go beyond the individual models and formulate joint supervised models based on training data for both rating and ranking. In doing so, we aim to improve upon the coarseness of a ratings-only model as well as investigate the inherent similarities between regression and ranking.

We use a dataset of text documents scored for their readability, described in Section II. In Section III, we define the existing SVM models for rating and ranking. Then in Section IV, we present two models for learning jointly from ranking and rating, and we discuss their results in Section V. We conclude the paper in Section VI by listing the next steps in this project.

II. Dataset

Our data comes from the DARPA Machine Readability dataset, a set of 540 text documents whose readability levels have been assessed by eight human expert annotators with professional experience in linguistic analysis. Each document is labeled with a readability rating on a scale of 1 to 5, and the documents are further divided into subsets of 10 or fewer, over each of which an annotator ranked the member documents. Every annotator rates and ranks each of the 540 documents exactly once. Since the rating values assigned to a given document vary according to an annotator's tastes, we considered every annotator's data separately. For each annotator, we split the 540 documents into a 390-document training set and a 150-document test set. The training set was further split into a 300-document development training set and a 90-document development test set. The features in our models were a set of 62 preselected NLP features aimed at assessing readability (provided courtesy of Ramesh Nallapati).

III. Definitions and Notation

We now define two existing SVM models for regression and ranking. These are the models we build upon in our formulation of the joint models.

A. Ratings model

Our ratings model is an SVM Regression model with m training examples[1].

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*)$$

$$\text{s.t.} \quad \begin{cases} y_i - (\langle w, \Phi(x_i)\rangle + b) & \leq \epsilon + \xi_i, \quad \forall i \\ (\langle w, \Phi(x_i)\rangle + b) - y_i & \leq \epsilon + \xi_i^*, \quad \forall i \\ \xi_i, \xi_i^* & \geq 0, \quad \forall i \end{cases}$$

Where y_i denotes the rating label for document x_i, and ϵ is a tolerance parameter. In our tests, we set ϵ to zero to penalize any deviation from the true label on a training example. This choice is makes the model more compatible with our later error measurements using average absolute error.

B. Ranking model

For ranking, we use Ranking SVM[2].

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 + C\sum_{p_{i,j}\in P}\xi_{i,j}$$

$$\text{s.t.} \begin{cases} \langle w, \Phi(x_i)\rangle - \langle w, \Phi(x_j)\rangle & \geq 1 - \xi_{i,j}, \quad \forall p_{i,j} \in P \\ \xi_{i,j} & \geq 0, \quad \forall p_{i,j} \in P \end{cases}$$

In this model, P is the set of all pairwise orderings from the training set. For example if $p_{i,j} \in P$, then a human has ranked document x_i higher than x_j. Hence, $\sum_{p_{i,j} \in P}$ denotes "sum over all orderings in P".

IV. Two models for joint learning

We now present two models for learning jointly from ratings and rankings.

A. Perturbation model

In the perturbation model, we work on breaking the tie that exists between many documents that have the same rating. For each group of n documents that have the same rating r for a given annotator within a ranking subset, we apply a "perturbation" in each document's rating based on the rank ordering. The highest-ranked document receives a rating of $r+\delta$, the lowest receives a rating of $r-\delta$, and the rest of the $n-2$ documents receive ratings evenly spaced between the $r+\delta$ and $r-\delta$ based on their rank position. These adjusted ratings then become the inputs into SVM Regression.

B. Joint SVM model

In the joint SVM model, we combine the constraints of SVM Regression and Ranking SVM.

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 + C_{Reg}\sum_{i=1}^{m}(\xi_{Reg,i} + \xi^*_{Reg,i}) + C_{Rank}\sum_{p_{i,j}\in P}^{m} \xi_{Rank,i,j}$$

$$\text{s.t.} \begin{cases} y_i - (\langle w, \Phi(x_i)\rangle + b) & \leq \xi_{Reg,i}, \quad \forall i \\ (\langle w, \Phi(x_i)\rangle + b) - y_i & \leq \xi^*_{Reg,i}, \quad \forall i \\ \xi_{Reg,i}, \xi^*_{Reg,i} & \geq 0, \quad \forall i \\ \langle w, \Phi(x_i)\rangle - \langle w, \Phi(x_j)\rangle & \geq 1 - \xi_{Rank,i,j}, \forall p_{i,j} \in P \\ \xi_{Rank,i,j} & \geq 0, \quad \forall p_{i,j} \in P \end{cases}$$

V. Results and Discussion

A. Perturbation model

Table I shows the results for the perturbation experiment. The first row shows the baseline performance of an SVM regression algorithm for each of the eight annotators. By baseline performance, we mean the test error obtained in each annotator without any perturbation on the labels. The second row contains the performance obtained by perturbing the ratings labels based on ranking information and feeding them into an SVM regression algorithm. Annotators marked with an asterisk (*) are those whose results pass the statistical significance test with a p-threshold of 0.05.

The results were obtained by running SVM regression on the development set to obtain the cost parameter C with minimum absolute error for each annotator, which gives us the baseline for our experiment. Next, we perform a grid search along a combination of values for the cost parameter C and the perturbation parameter δ. We report the minimum absolute error obtained from the grid search for each annotator. We use the C and δ corresponding to this minimum absolute error while applying the SVM regression model on the official test set (with 150 documents) and report its absolute error. The SVM Regression model was implemented using MATLAB and CVX software[3].

TABLE I
Perturbation Experiment Performance Improvements

	Ann. 1*	Ann. 2*	Ann. 3*	Ann. 4*
Baseline: Unperturbed SVM absolute error	0.6049	0.6599	0.5971	0.5077
Perturbed SVM model absolute error	0.6035	0.6559	0.5974	0.5034
Improvement	0.23%	0.6%	-0.05%	3.1%

	Ann. 5*	Ann. 6*	Ann. 7*	Ann. 8*
Baseline: Unperturbed SVM absolute error	0.6500	0.6549	0.5365	0.6100
Perturbed SVM model absolute error	0.6512	0.6560	0.5337	0.6080
Improvement	-0.18%	-0.09%	0.52%	0.33%

As evidenced by the results in Table I the improvements are modest and not entirely consistent across annotators. However, these results reveal that incorporating additional information from rankings to the regression algorithm – done by means of label perturbation in this experiment – can improve the predictive ability of the ratings only model. Hence, our next step is to implement the Joint SVM Model which incorporates the ranking information as part of the optimization constraints.

B. Joint SVM model

The Table II below shows the results of the Joint SVM model for each of the eight annotators. The first row contains the baseline performance on SVM Regression alone, and the second row contains the performance on the joint model, which includes 1252 to 1283 pairwise ranking constraints per annotator. Annotators marked with an asterisk (*) are those whose results pass the statistical significance test with a p-threshold of 0.05. The baseline performance in Table II differs from that in Table I because of the different sets of C_{reg} values used during the grid search.

To get these results, we first ran SVM regression for each annotator on the development set to select the regression cost parameter C with the lowest absoluted errors. Then, for each annotator, we ran our SVM joint model on the development set with C_{reg} set equal to the C parameter we obtained from the SVM Regression model. We tested values of C_{rank} from the set $\{C_{reg}, \frac{3}{4}C_{reg}, \frac{1}{2}C_{reg}, \frac{1}{10}C_{reg}\}$, and chose whichever value of C_{rank} minimized test error. Using these chosen parameters, we ran SVM regression on the official set to get

TABLE II

JOINT MODEL PERFORMANCE IMPROVEMENTS

	Ann. 1*	Ann. 2*	Ann. 3*	Ann. 4*
Baseline: SVM Regression absolute error	0.6115	0.6672	0.5852	0.4942
Joint SVM model absolute error	0.5989	0.6541	0.5839	0.4888
Improvement	2.1%	2.0%	0.2%	1.1%

	Ann. 5	Ann. 6*	Ann. 7*	Ann. 8*
Baseline: SVM Regression absolute error	0.6500	0.6617	0.5232	0.6255
Joint SVM model absolute error	0.6444	0.6560	0.5211	0.6113
Improvement	0.1%	0.9%	0.4%	2.3%

our baseline values and on the SVM joint model to assess the new model's performance.

As Table II shows, adding ranking constraints to a regression model can indeed improve regression performance, though the improvements are only by modest margins. In our testing, we also tried to gauge the effect of the volume of ranking constraints by using smaller subsets of the constraints available for each annotator. In one test, we removed ranking constraints randomly, reducing the number of constraints from the complete set of about 1250, to 800, and then 400 and 200. However, this decrease revealed no clear pattern in the algorithm's performance, most likely due to the random nature of the selections. Then, to determine whether the quality of the constraints mattered, we tested the joint model using only pairwise ranking constraints for items with tied ratings. This reduced the number of ranking constraints to about 450 per annotator. While this setting achieved improvements over the baseline errors, the improvements were only on the order of 0.1 percent, and half of the annotators did not pass the statistical significance test. These results suggest that it is better to use the entire set of ranking constraints rather than a smaller, targeted subset.

While it is encouraging to find that regression performance can improve with rankings data, we would still like to see greater improvements and to investigate the relationship between regression and ranking more thoroughly. There are several other tests we'd like to try, which we will discuss in Section VI on Future Work.

VI. Future Work

The objective of our work was to determine how valuable rankings data can be in situations where the rating labels are coarse. It may be possible to gain more insight into the value of rankings by training with less ratings data. That is, we would withhold some rating labels from our training set while keeping all the ranking constraints in place. While in a regression-only model the performance might be expected to decline with less ratings data, we hope to see that the added rankings information in a joint model can counteract that decline. We would especially expect to see a stronger counteracting effect in the case where the data trained on had more rating ties than the original data.

Additionally, we may try adjusting the loss functions in the optimization objective of the SVM joint model or combining pairwise constraints with another regression model altogether. In previous informal experiments, we combined pairwise restrictions with regularized least squares regression but did not achieve any favorable results, but perhaps with more careful parameter selection we will have greater success. Finally, a richer appreciation of the value of a joint model could be provided by training and testing on different datasets.

VII. Acknowledgements

We would like to thank Ramesh Nallapati, Research Associate in the Stanford Natural Language Processing group, for advising us and providing us with the Machine Readability Dataset. We would also like to acknowledge the Stanford NLP group for allowing us to use its JavaNLP software.

References

[1] Smola, A. J., and Scholkopf, B., (1998), "A Tutorial on Support Vector Regression," NeuroCOLT2 Technical Report Series.

[2] Joachims, T., (2002), "Optimizing Search Engines using Clickthrough Data," ACM Conference on Knowledge Discovery and Data Mining (KDD).

[3] Boyd, S., and Grant, M. (2010),CVX: Matlab Software for Disciplined Convex Programming, Software Version 1.21. http://cvxr.com/cvx.

[4] Ng, A. (2010), Lecture notes from CS229 Machine Learning course (Fall 2010), Department of Computer Science, Stanford University, CA.