IJCSC Volume 5 • Number 1 March-Sep 2014 pp. 121-125 ISSN-0973-7391

Evaluation of Usefulness of Unlabeled Data in Learning a Recommender

Vikas Parmar[1],Priyank Thakkar[2], K Kotecha[3]
Nirma University, Ahmedabad, 382481, Gujarat, India
Email: [1]12mcec32@nirmauni.ac.in, [2]priyankbthakkar@gmail.com, [3]drketankotecha@gmail.com

Abstract: Supervised learning algorithms require labeled training examples from every class to engender a classification function. One of the shortcomings of this classical paradigm is that in order to learn the function accurately, a large number of labeled examples are needed. There are many situations (e.g. a new user in an online recommender system) where for every class,only a small set of labeled examples is available. Situations such as these encourage to investigate about the usefulness of unlabeled examples in learning a recommender. The main objective of this paper is to examine the influence on the accuracy of the recommender when it is built using unlabeled examples in addition to the labeled examples. Co-Training algorithm which allows to incorporate unlabeled examples while learning a classifier/recommender. Usefulness of this algorithm is investigated by means of experimental study using hetrec2011-movielens-2k data set.Accuracy and f-measure are used as the evaluation measures.
Index Terms— Classification, Recommender, Labeled and Unlabeled Examples, Co-Training.

I. Introduction

Recommender system is one of the applications to predict rating or preference for the items that have not been seen by a user. This system typically produces a list of recommendations. Recommending books, CDs, and other products at amazon.com, movies by MovieLens, and news at VERSIFI Technologies (formerly adaptiveInfo.com) are examples of such applications to name a few [8].
In a recommender system, the items which are liked or disliked by the user are the labeled examples about the users' preference. However, there are many other items which are not rated by the user. These items form the set of unlabeled examples. One of the major problem with recommender system is that it does not have adequate number of labeled examples for the new user.
In supervised learning, labeled training examples from every class are used by the learning algorithms to generate a classification function. The biggest problem of this common prototype is that in order to learn accurately, large number of labeled examples are needed. Labeling can be very labor intensive and time consuming, since it is often done manually. In many applications, such as movie recommender system, for many user only small number of ratings (i.e. labeled examples) are available. A recommender learnt through these small number of ratings may not beadequately accurate and useful.The number of ratings of a user increases only gradually.A partially supervised learning algorithm as its name suggest, do not need full supervision, and thus are able to lessen the labeling effort. This type of learning exhibits a small set of labeled examples of every class, and a large set of unlabeled examples. The idea is to make use of the unlabeled examples to increase the prediction accuracy of a learner. Co-training algorithm belongs to the family of partially supervised learning algorithms and able to incorporate unlabeled data in addition to the labeled data in learning a classifier/recommender. Efficacy of co-training algorithm in learning a recommender by exploiting unlabeled examples in the presence of small number of labeled examples is critical.

II. Related Work

Co-Training algorithm is special in a way that it allows learning from labeled and unlabeled examples. Co-training was formalized in [9]. They provided a theoretical guarantee for accurate learning subject to certain assumption. In [10], it was shown that co-training produces more accurate classifiers than the EM algorithm, even for data sets which did not satisfy the strict underlying assumptions. Co-training algorithm basically allows a classifier/recommender to exploit unlabeled data in addition to the labeled data during the learning process. A new user of the collaborative recommender system exhibits small number of labeled examples about his/her interest. It is difficult to learn an accurate recommender based on this few labeled examples. Therefore, it may be worth investigating the use of the unlabeled examples to improve the performance of a recommender through co-training algorithm.
Application of co-training was investigated for word sense disambiguation in [1]. Author investigated optimal and empirical parameter setting methods. Author proposed a new method that combined co-training with majority voting. A PAC analysis on co-training style algorithm was presented in [2]. They showed that co-training process could succeed even without two views, given that two learners had large differences. They also proved theoretically that why co-training process could not improve the performance further after a number of rounds. A spectral clustering algorithm having a flavor of co-training was proposed in [3]. The major advantage

of their algorithm was that there were no hyperparameters to set. They empirically compared their proposal with number of baseline methods on synthetic and real-world data set.Cross-lingual sentiment classification is addressed with the help of co-training in [4]. Experimental results showed that their approach performed better than that of standard inductive and transductive classifiers.

III. Co-training Algorithm

Co-training assumes that the set of features can be partitioned into two subsets (x_1 and x_2). For learning the target classification function, each of them is sufficient. This division of features is not considered by the traditional learning algorithms. This feature division is exploited by co-training to learn separate classifiers over each of the feature sets. The co-training algorithm employed by us is shown in algorithm 1 [5], [9]. In the implementation reported here, x_1 portion refers to rating profile while x_2 refers to genre profile of a movie. Co-training returns two classifiers. At classification time, the two classifiers are applied separately for each test example and their scores are combined to decide the final class. For naive Bayesian classifiers, two probability scores are multiplied, i.e.

$$\Pr(c_j|x) = \Pr(c_j|x_1)\Pr(c_j|x_2) \qquad (1)$$

Algorithm 1: Co-Training Algorithm

```
Input: labeled set L,unlabeled set U
  Output: Two Classifier $f_1$ and $f_2$
  1)  Create a poolU' of examples by selecting u examples at random from U
  2)  Learn a classifier fusing L based on $x_1 + x_2$(i.e. all features) of the examplesx.
  3)  repeat
  4)  Learn a classifier $f_1$using L based on only $x_1$ portion of the examples x.
  5)  Learn a classifier $f_2$ using L based on only $x_2$ portion of the examples x.
  6)  Apply $f_1$to classify the examples in$U'$, for each class $c_i$,pick$n_i$examples that $f_1$hasmost confidently
      classified as class $c_i$,and add them to L.
  7)  Apply $f_2$ to classify the examples in $U'$, for each class $c_i$, pick $n_i$examples that $f_2$hasmost confidently
      classified as class $c_i$,and add them to L.
  8)  Randomly select $2n_1 + 2n_2$ examples from U to replenish $U'$.
  9)  until k iterations.
  10) return $f_1$ and $f_2$
```

The probability calculated in equation 1 represents results of the recommender based on labelled and unlabeled data. Let this recommender be denoted as f^*.

IV. Experimental Evaluation

Dataset

In all the experiments carried out, dataset hetrec2011-movielens-2k [6], (http://www.rottentomatoes.com/), (http://www.imdb.com/), (http://www.grouplens.com/) dated May 2011 is used. It was made publically available by [6]. It is built on the original MovieLens10M data set, distributed by the GroupLens research group. In this data set, movies also refer to their analogous web pages at the IMDB website. The data set comprises 2,113 users, 10,197 movies and a total of 13,222 unique tags. It also comprehends 855,598 user ratings ranging from 0.5 to 5.0, in additions of 0.5, leading to a total of 10 distinct rating values. There is an average of 405 ratings per user, and 85 per movie. There are 20 genre types, 20,809 movie-genre assignments, 4060 directors and 95321 actors. There are average 22 actors per movie.

Evaluation Measures

Accuracy and f-measure are used to evaluate the performance of recommenders. Accuracy is defined as the ratio of the number of correctly classified instances in the test set to the total number of instances in the test set. In this paper, user liking a movie is considered as positive class while user disliking a movie as negative class. In this sense, true positive (TP), false negative (FN), false positive (FP) and true negative (TN) are defined as under [5].

TP: the number of correct classifications of the positive instances
FN: the number of incorrect classifications of the positive instances
FP: the number of incorrect classifications of the negative instances
TN: the number of correct classifications of the negative instances

Based on the above interpretations precision (p) and recall (r) are defined in equations (2) and (3) respectively.

$$p = \frac{TP}{TP + FP} \qquad (2)$$

$$r = \frac{TP}{TP + FN} \qquad (3)$$

F-measure (F) is used to compare classifier on a single measure and it is represented by the equation (4)

$$F = \frac{2pr}{p + r} \qquad (4)$$

Experimental Methodology, Results and Discussions

For experimentation purpose, 24 users who have rated at least 100 items are selected as the test user. For each of the test user, 80% of the movies rated by him/her are considered as a part of training set. Remaining movies form the testing set. It is to be noted that when x_1 portion of the features is considered, each of the observation in training set and testing setis a real-valued vector describing a movie in terms of ratings of users other than the test user under focus.When x_2 portion of the features is considered, each of the observation in training set and testing setis a Boolean vector describing a movie in terms of its associated genres. This implies that each vector based on x_1 portion of the features is a real-valued vector but with large number of missing values. A missing value at a particular position in this vector indicates that corresponding user has not rated the movie represented by the vector. Experiments are carried out for all the test users and reported results are aggregated over the results of each of the test user. Co-training algorithm is executed with parameter values ofu = 200, $2n_1 + 2n_2 = 10$, k = 50. All three classifiers f, f_1, &$f_2$are learnt through naïve-Bayesian algorithm.

For each of the test users, two kinds of experiments are carried out. In the first kind of experiment, recommender (f in the algorithm) is learnt only using the labeled data while in the second type of experiment, recommender (f_1 and f_2) is learnt through labeled and some number of unlabeled data. Experiments are carried out by changing the number of labeled data to see the impact of unlabeled data in the presence of different number of labeled data. This results are summarized in Table 1. It can be seen from the results that using unlabeled data in addition to the labeled data while learning the recommender improves the performance of the recommender. It is also evident that the performance of the recommenders improve when number of labeled data increases.

Table 1: Aggregated Results

Number of Labeled Examples	**Aggregated Results of Recommender (f) (Only Labeled Data)**		**Aggregated Results of Recommender (f^*) (Labeled + Unlabeled Data) using Co-training Algorithm**	
	Accuracy (%)	**F-measure (%)**	**Accuracy (%)**	**F-measure (%)**
10	56.44	59.05	68.84	69.57
20	57.18	59.86	69.17	69.86
30	60.32	62.36	67.73	68.49
50	62.13	64.28	69.81	70.56
100	62.45	64.18	70.56	71.20
All	65.39	66.01	71.36	71.31

Impact of feature selection on recommender while using only labeled data and labeled plus unlabeled data is also studied. In this experiment, for each of the test user, all the ratings which are available are used as the labeled data. Feature selection is performed only on the rating profile (e.g. x_1 portion of the features). Results are depicted in Figure 1 and 2. Graphs clearly suggest that learning a recommender using appropriate number of features definitely affects the performance of the recommender. This is not only true for the conventional recommender that uses only labeled data but also true for the recommender based on co-training algorithm which incorporates unlabeled data in learning process.

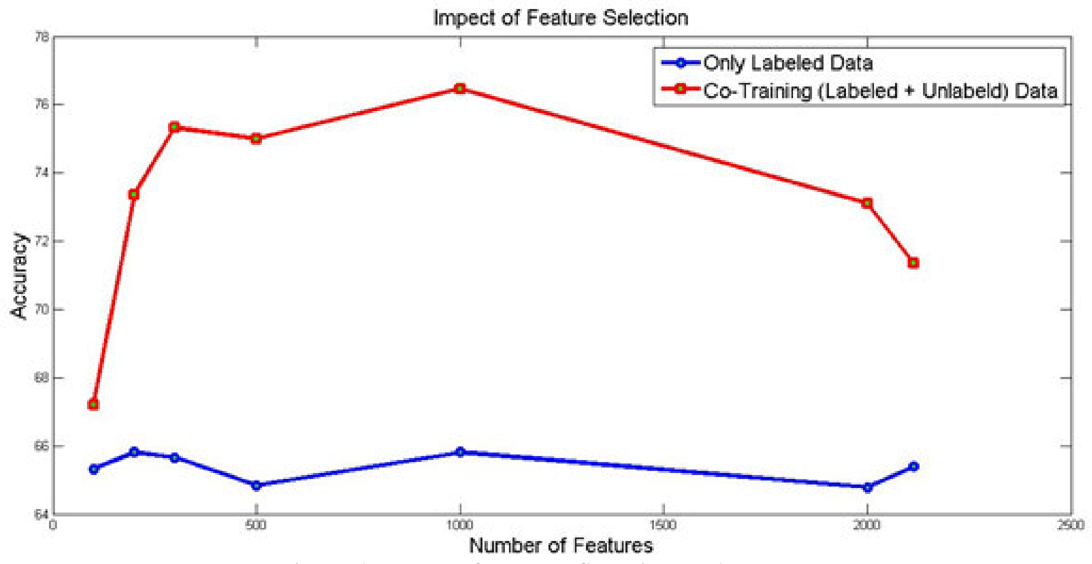

Figure 1:Impact of Feature Selection on Accuracy

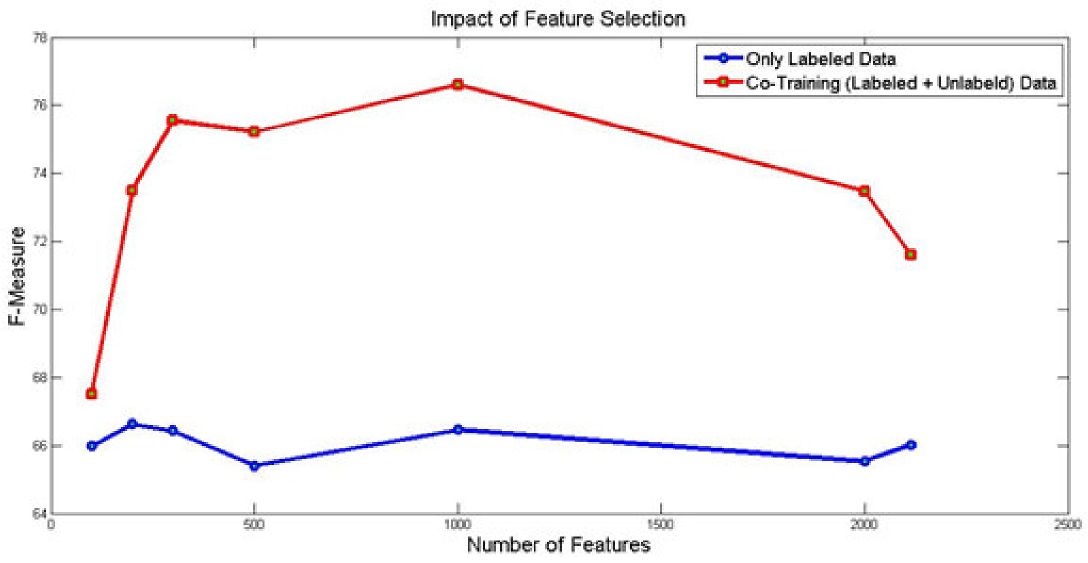

Figure 2:Impact of Feature Selection on F-measure

V. Conclusions & Future Work

The usefulness of unlabeled examples in learning the recommender in the presence of varying number of labeled examples is evaluated in this paper. From the experimental results, it is evident that accuracy of the recommender is improved when it is learnt using unlabeled data in addition to the labeled data.The improvement in the performance is almost 12% when the labeled data is as few as 10 to 20. Results also demonstrate that selecting right number of features while learning a recommender further improves the performance. One potential direction for the future work can be the experimentation on other data sets.

References

[1] Rada Mihalcea, “Co-training and self-training for word sense disambiguation”, Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2004), 2004.

[2]Wei Wang, Zhi-Hua Zhou, “Analyzing co-training style algorithms”, Machine Learning: ECML 2007, 454—465, 2007, Springer.

[3] Abhishek Kumar, Hal Daum ́e, “A co-training approach for multi-view spectral clustering”, Proceedings of the 28th International Conference on Machine Learning (ICML-11), 393—400, 2011.

[4] Xiaojun Wan, “Co-Training for Cross-Lingual Sentiment Classification”, Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1, 235—243, 2009, Association for Computational Linguistics.

[5] Bing Liu, “Web Data Mining”, Department of Computer Science University of Illinois at Chicago.

[6]Cantador, Peter Brusilovsky, Tsvi Kuflik, “2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)”, Proceedings of the 5th ACM conference on Recommender systems, RecSys 2011, 2011, Chicago, IL, USA, ACM, New York, NY, USA, information heterogeneity, information integration, recommender systems.

[7]Bradley N. Miller, Istvan Albert, Shyong K. Lam, Joseph A. Konstan, John Riedl, “MovieLens unplugged: experiences with an occasionally connected recommender system”, Proceedings of the 8th international conference on Intelligent user interfaces, 263—266, 2003, ACM.

[8]Gediminas Adomavicius, Alexander Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions”, Knowledge and Data Engineering, IEEE Transactions on, 17, 6,734-749, 2005, IEEE.

[9]Avrim Blum, Tom Mitchell, “Combining labeled and unlabeled data with co-training”, Proceedings of the eleventh annual conference on Computational learning theory, 92-100,1998, ACM.

[10]Kamal Nigam, Rayid Ghani, “Analyzing the effectiveness and applicability of co-training”, Proceedings of the ninth international conference on Information and knowledge management, 86-93, 2000, ACM.