A Visual Approach for Interactive Co-Training

Qi Han
Institute for Visualization and Interactive Systems
University of Stuttgart
Germany
Qi.Han@vis.uni-stuttgart.de

Weimeng Zhu
Institute for Natural Language Processing
University of Stuttgart
Germany
st113027@stud.uni-stuttgart.de

Florian Heimerl
Institute for Visualization and Interactive Systems
University of Stuttgart
Germany
Florian.Heimerl@vis.uni-stuttgart.de

Steffen Koch
Institute for Visualization and Interactive Systems
University of Stuttgart
Germany
Steffen.Koch@vis.uni-stuttgart.de

Thomas Ertl
Institute for Visualization and Interactive Systems
University of Stuttgart
Germany
Thomas.Ertl@vis.uni-stuttgart.de

ABSTRACT

Co-training is a popular semi-supervised method to build classifiers by combining labeled and unlabeled data. It trains two classifiers with a small amount of initially labeled data and iteratively retrains them after exchanging their high confidence instances. As the initial amount of labels is very small, however, the performance can suffer from the label pollution problem. We therefore propose an interactive visual approach that improves the stability of co-training through user inspection of transferred instances. It includes a visualization of classifier uncertainties and disagreement. It further helps users to quickly identify possible mistakes of the automatic approach by guiding user's attention to the instances which are labeled differently than the majority of their nearest neighbors and instances which are labeled differently by the two base classifiers. To help users examine such instances, we also include a visual explanation which shows important features of an instance along with its raw data. We show the effectiveness of our approach with a usage scenario and by comparing it with the classical co-training approach through experiments. Finally, we discuss limitations and propose several possibilities for future improvement.

Keywords

interactive machine learning, visualization, machine learning, semi-supervised learning, co-training, multi-view learning, bootstrapping

1. INTRODUCTION

Nowadays, a huge amount of text data are produced every day. The number of emails received by individuals is growing every minute. Documents produced by online communities or other organizations are also increasing quickly. This poses great challenges to human individuals as well as organizations to manage and analyze this data to obtain insights. To classify data into different categories is one of the common methods to organize it. Automatic classification can free humans from repeatedly doing the same classification tasks, as they can learn classification rules from them. However, to obtain a classifier with good performance, people need to label many data points to feed into the classification algorithm, which is very time consuming. Semi-supervised learning methods [7, 22] can reduce human effort in labeling a lot of data. They combine information from a small amount of labeled and a large amount of unlabeled data to learn classification criteria. Clustering [12] can also automatically assign documents into categories. However, the natural clustering of the data does not necessarily inconsistent with the intention of users. Recently, researchers also suggest approaches to actively integrate human intention into the clustering results [4] . In this work, we focus on improving semi-supervised learning methods through human interaction.

Co-training [5] is one of the most popular semi-supervised learning methods. It starts by training two classifiers on two different feature sets with an initial set of labeled data. It proceeds by iteratively growing the set of labeled data and retrain the classifiers on this new dataset. In each iteration, it allows each of the two classifiers to label a few unlabeled data instances, which they can classify with a high confidence. These instances are added to the set of training data for subsequent iterations. The two classifiers are retrained on this new set and co-training can start a new iteration. Zho and Li [21] use a flow-graph to explain the co-training method. We add a visual element representing users into the graph to clarify the role of users as can be seen in Figure 1.

However, as the initial amount of labeled instances is very

KDD 2016 Workshop on Interactive Data Exploration and Analytics (IDEA'16) August 14th, 2016, San Francisco, CA, USA.

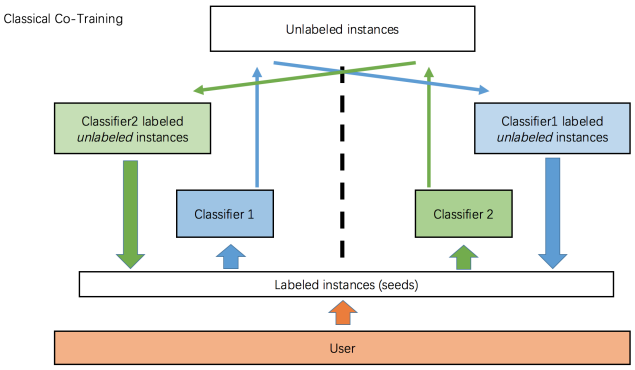

Figure 1: A graphical summarization of the classical co-training method.

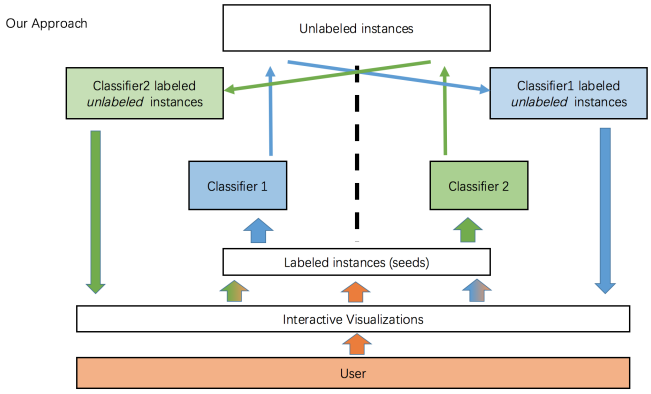

Figure 2: A graphical summarization of the interactive co-training approach proposed in this work.

small, the two base classifiers do typically not have a high classification performance at the beginning. They can thus potentially introduce labeling mistakes into the training set used to train new classifiers in the following iterations. This makes the performance of the co-training method unstable, especially when used on noisy data [9, 18]. In this paper, we propose an interactive visual approach for co-training. Our approach improves the stability of co-training through user inspection of the transferred instances. It includes a visualization based on Parallel Coordinates [11] to depict the uncertainties of the two base classifiers and the disagreement between them. Furthermore, it encodes the label distribution of the nearest neighbors of the instances. The visualization guides users' attention to the instances which are labeled differently than the majority of their nearest neighbors or the instances which are labeled differently by the two base classifiers. In doing so, it helps users to quickly identify possible mistakes of the automatic approach. Correcting those mistakes will likely boost the performance of the co-training algorithm. To help users examine these mistakes more closely, we also include a visual explanation which shows important features of an instance along with its raw data. Figure 2 depicts the main components of our approach.

2. RELATED WORK

In this section we discuss approaches that are related to ours. They can be broadly divided into two groups. The first one address machine learning in semi-supervised settings. The second group of approaches focuses on integrating interactive visualizations and machine learning to improve performance of automatic algorithms, or provide explanation for their decisions.

Blum et al. [5] suggest co-training to combine information form labeled and unlabeled data to train classifiers. They also show the effectiveness of co-training under the assumption that the two views on the data are conditionally independent. Since then, co-training has been applied in many domains, for example, to classify emails [14], or to label roles of named entities [9]. Additional research provides more insight about why and in which settings the co-training method works well [3]. In addition, limitations of the classical co-training method have been identified, such as its difficulties with noisy or unbalanced data [16, 18]. Muslea et al. [17] suggest to combine active learning and co-training to obtain a more stable and effective semi-supervised method. Our approach also tries to improve on the classical co-training method. However, we achieve this by letting users actively inspect or correct the automatic method through interactive visualizations.

Recently, there has also been many research efforts that aim to bring visual interaction and machine learning together to allow users to guide and steer machine learning methods [1, 13]. ModelTracker [2] is a visual approach for analyzing performance of machine learning models. FeatureInsight [6] and FeatureForge [10] propose visual approaches for feature engineering. Ribeiro et al. [15] propose a method for explaining the reasons behind predictions made by machine learning methods. They suggest a method to derive important features by each prediction. We propose a visual approach for classifier building in a semi-supervised way to reduce human effort and increase the trust of users in the resulting classification model.

3. APPROACH

In this section, we first describe the data processing work flow of our approach. Along with that, we also describe the reasons why we have chosen some algorithms over the others. We then highlight the tasks that we intend to support and introduce visualizations to address these challenges.

3.1 Data Processing

Our approach is based on the idea that we first build two classifiers based on two views of the data with just a few labels. These two classifiers are then utilized to label additional instances so that a high performance classifier can be obtained.

The first step in our workflow is to construct two different feature sets or so-called views from the datasets. In many cases, the dataset has a natural split of views. For example, for images with additional text descriptions we can construct one view from the image data and the other one from the textual data. For email data, we can construct one view from email meta data, like the header or sender of the emails and obtain the other view from the textual data. For datasets without a natural split, a simple procedure to acquire two views is to randomly assign features to one of the two views. Several other methods have been suggested to split a single set of features into two views, which are more suitable for

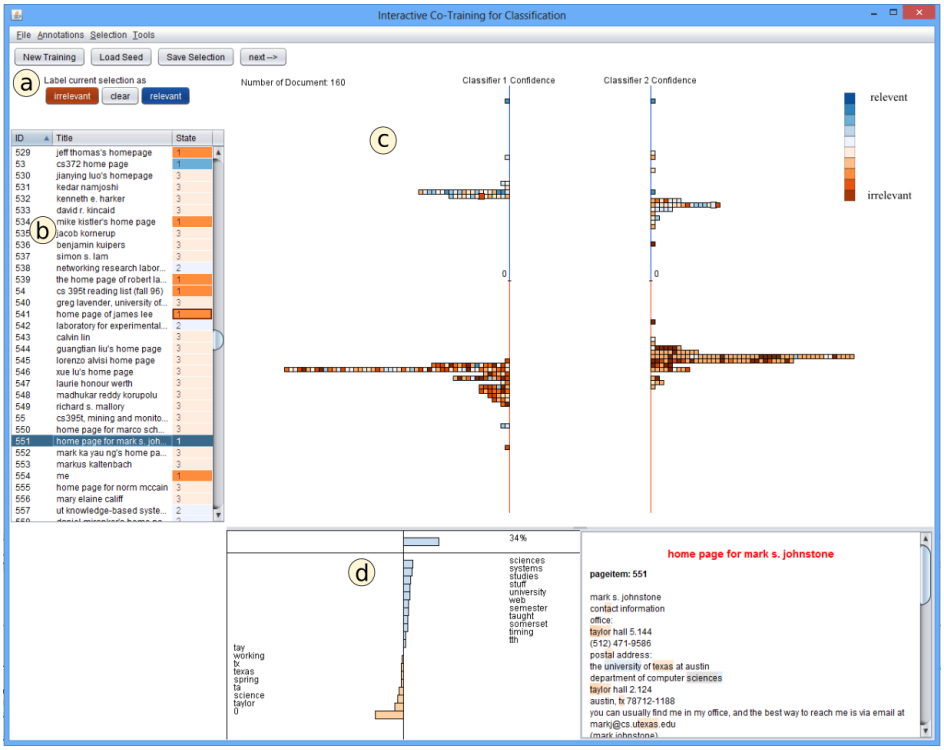

Figure 3: A screenshot of our system. It includes: a) the Control Panel comprising several control buttons b) the Instance Table view showing the instances of the dataset and classification states; c) the Overview view depicting the classification confidence of the two classifiers about the instances and d) the Explainer view showing the evidence of the classification decisions made by a classifier.

co-training algorithms than random split. Our approach can work with both types of datasets.

In the second step, we choose a classification algorithm for the base classifiers. As opposed to the original co-training paper, that proposes Naive Bayes [20], we have decided to use linear SVMs from the Liblinear package [8] in this work. SVM is a fast and robust classification algorithm that has been shown [14] to be more suitable than Naive Bayes for text classification tasks in co-training settings. In addition, we use Platt-scaling [19] to obtain a calibrated confident score of the classifications made by the SVM classifiers. This way, we can compare classification confidences of different SVM classifiers.

As mentioned in the introduction, our approach offers a view to visually explain the classification decisions. For this purpose, we need to identify the relevant features and their importance for the instances under inspection. Depending on the type of classifier, different measures of feature importance can be used for this. In this work, we use the feature weights of the linear SVM classifier.

3.2 Visual Approach

In this section we describe different views of our approach and introduce the interactions supported by them. We build interactive visualizations to help users quickly identify possibly mislabeled instances, obtain insights about the classification uncertainty of the classifiers, understand reasons behind decisions made by the classifiers, and maintain a stable mental map of all the instances.

3.2.1 *Overview visualization*

The overview visualization is updated after each round of co-training. All instances labeled in the last round of co-training are visualized in this view. Small squared glyphs are used to represent the instances. As each instance is given two different confidence scores by the two base classifiers, we depict each one of the instances as two glyphs. They are placed along two axes according to their confidence scores by the two classifiers. As the amount of instances labeled in one round of co-training can be large, we divide the whole range of the axes into bins. We then assign the instances to these bins. When more than one instance is placed into one bin, we stack them on top of each other. This way, users can identify individual instances easily and gain an impression of the distribution of classifier confidence scores (see Fig. 4). ModelTracker [2] uses a similar design to depict the test instances of one classifier. We aim to display disagreement between two classifiers and the mislabel possibility of the instances.

We assign colors to the instances to encode the label distributions of their neighboring instances. The similarity is measured according to the Euclidean distance in the feature space of the classifiers. For each instance in the visualization, we count how many of its neighbors are labeled as positive and how many of them are labeled as negative. The ratio of these two counts is mapped to color between positive color and negative color with linear interpolation. Thus, the more neighbors of an instance are labeled as positive, the more similar its color is to the positive color. As

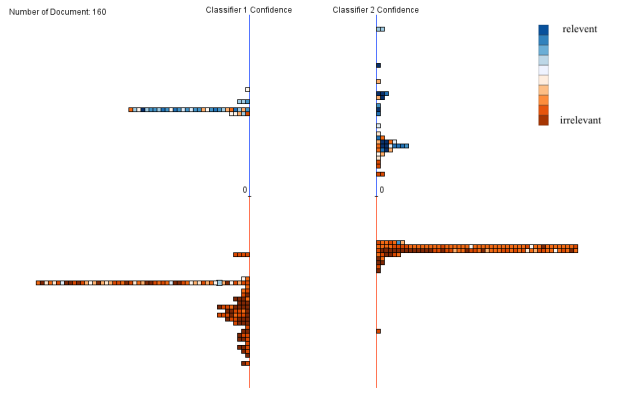

Figure 4: In the overview visualization, the instances are placed according to the confidence score of the classifiers. The colors of the instances encode the label distributions of their neighboring instances in the feature space of the classifiers.

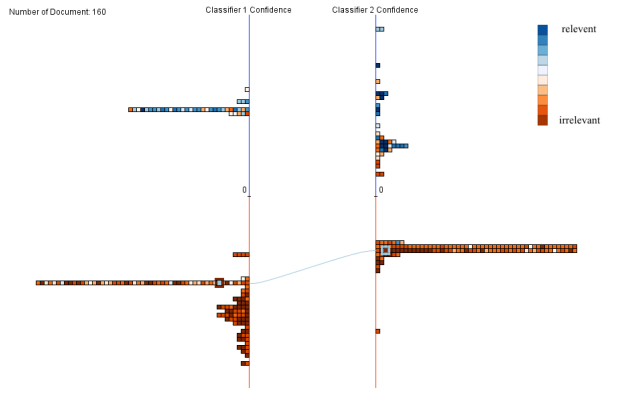

Figure 5: This figure shows the highlighting effect. The instance was positioned at the lower part of the overview view which means the instance was labeled as irrelevant. However, on the left side, the color of this instance is blue. It shows that the neighbors of the instance were labeled mostly as relevant. This inconsistence suggests that the instance might be labeled incorrectly.

distinct colors can be perceived in a pre-attentive way, users can quickly see instances that are classified as positive but have more negative neighbors or vice versa.

When users hover over one of the squares, it is highlighted by a halo around it (see Fig. 5). The color of the halo is identical to that of the square representing the same instances in the second classifier. We also show a spline connecting those two squares representing the same instance. By comparing the information from two views of the same instance, users can more easily decide whether it is worth to inspect the instance more closely.

3.2.2 *Interactive explainer view*

The explainer view is located directly below the overview visualization. It consists of one view depicting the important features along with their importance rating, and another view showing the raw data of an instance (see Fig. 6). When users hover over an instance, the explainer shows more detailed information about the instance and the classification.

On top of the explainer view, we use a bar chart starting at the center line of the view to depict classification confidence. When an instance is classified as positive, the bar grows to the right, when it is classified as negative, the bar grows to the left. Colors of the bars are consistent with the overview visualization. Features are sorted according to their importance and the bar charts are placed accordingly from top to bottom. This visualization allows users to see the most important features for the classification of an instance. This gives users cues about how the classification decision about that instance is made. To get additional information about the features influencing the classification decision, users can turn to the raw data view. The raw data view highlights highly weighted features by coloring the background of the features in the raw data. The highlights guide the attention of users during inspection of the raw data.

Additional interactions are implemented in the view to further improve the usefulness of the explainer view. Users can click on the bars depicting the features, which causes the raw data view to scroll to the position where the feature appears for the first time. If they click the bar once more, the raw data view scrolls to the next appearance of that feature. This interaction further helps users to efficiently pick important information from the raw data. If they press the shift-key while clicking on the feature bar, all the instances containing this feature will be highlighted in the overview visualization and the instance table view. When users find a feature which is responsible for a erroneous label, they can use this interaction to find out how the feature is distributed over the labels. This gives users hints on explanations of feature importance.

3.2.3 *Control panel and instances table*

The instance table (Fig. 7) works as a notebook for the users. It maintains a table of all the instances in the dataset and uses the third column of the table to depict if that instance is labeled as positive, negative, or not labeled at all. Users can sort the instances according to their ids or the status of their labels. The control panel consists a group of controls for users to interact with the co-training process. For example, they can change the labels of the instances manually using the controls in this panel. We highlight the instances which have been manually labeled by the users during the current round in the instance table.

4. EVALUATION

In this section, we first demonstrate how users can interact with the system through a detailed description of a usage scenario. We then go on to show the effectiveness of our system by comparing our approach with the classical co-training algorithms.

4.1 Usage Scenario

At the beginning, all instances are displayed within the instance list view 7. Users can label a few instances by clicking on them and reading through the raw data within the explanation panel. Once they have decided on the label of the instance, they can use the buttons in the control panel to label the instance as positive or negative.

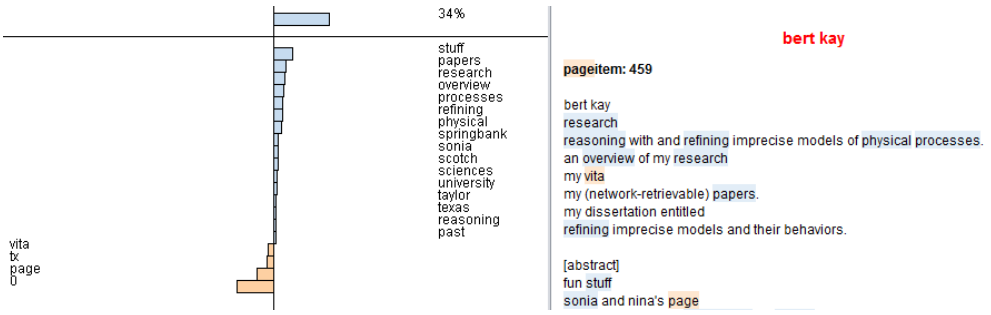

Figure 6: The explainer view shows most important features for the classification of an instance and the raw data of it.

ID	Title	State
529	jeff thomas's homepage	1
53	cs372 home page	1
530	jianying luo's homepage	3
531	kedar namjoshi	3
532	kenneth e. harker	3
533	david r. kincaid	3
534	mike kistler's home page	1
535	jacob kornerup	3
536	benjamin kuipers	3
537	simon s. lam	3
538	networking research labor...	2
539	the home page of robert la...	1
54	cs 395t reading list (fall 96)	1
540	greg lavender, university of...	3
541	home page of james lee	1
542	laboratory for experimental...	2

Figure 7: The list view shows the instances of the dataset and their current labeling status. Dark red indicates that the instance is labeled as irrelevant. Light red means that the instance has not been labeled yet but it is predicted by the classifiers as irrelevant. Similarly, dark blue and light blue means the instance is labeled or classified as relevant. User labeled instances are highlighted.

Then they can set the number of instances they wish the co-training algorithm to label in the next round and press the start button. The system iteratively labels more instances by choosing those instances rated most confidently by the classifiers.

Those newly labeled instances are depicted in the overview visualization. Users can identify the label given by the classifiers at the transfer time by observing the position of the instance squares. If the instance is positioned in the top half of the visualization, it is labeled as positive. The other way around, it is labeled as negative. They can also get an impression of how the labels of the nearest neighbors are distributed for each instance, by observing the color of squares representing the instances. Instances with more positive nearest neighbors are colored blue. Instances with more negative nearest neighbors are colored red. Seeing this can help them to quickly pick those instances, which have been labeled as positive but have a neighborhood which is labeled mostly as negative, or vice versa. Because disagreement to the majority of the labels of the nearest neighbors indicates a possible labeling error. If at least one of the classifiers has given high confidence to this label, users should be more motivated to check it, because it could have been mislabeled with a high probability. Correcting this will likely give a boost to the performance of the learned classifier.

Once they have identified one possible mistake of the classifier, they probably want to check if it is really mislabeled or not. By clicking on the instance, the explainer view shows the decisions of the classifiers and the important features on which the decision is based. Clicking on one of the shown features, the text panel will scroll to the position where that feature shows up in the text for the first time. This way users can quickly skim through long documents and concentrate only on information which is important for the classification. By clicking on one of the features, the instances which contain this feature will be also highlighted in the overview view.

Once users find a mislabeled instance, they can relabel it, and continue to examine other instances until they are satisfied with these set of newly labeled instances. They can then continue the co-training by clicking on the next button in the control panel. The system will use all the instances that were newly labeled in the subsequent co-training step, by letting the two classifiers retrain themselves based on the updated labeled set and further iteratively label more instances.

4.2 Comparison to Classical Co-Training

In this subsection, we describe the experiments to compare our approach and the classical co-training approach.

4.2.1 Experimental setup

The WebKB-Course dataset is composed by Blum et al. [5] to demonstrate the effectiveness of the co-training approach and has been subsequently used in several other works. We conduct our experiments on this dataset to compare our approach with the classical co-training algorithm.

This dataset contains data of 1051 web pages. For each web page we can construct two views/feature sets. One of them is based on the text on the web page. The other one is based on the anchor text of the links pointing to the page. The whole dataset can be divided into two parts: web pages of courses from a university or web pages of researchers in the university.

In the first step, we do experiments to obtain a estimation of error rate that we can obtain from the specific combination of the two views and the base classifier we use, which is a Support Vector Machine (SVM). For each experiment, we randomly choose 263 (25%) pages as test documents. From the rest of the 788 pages, we further randomly choose

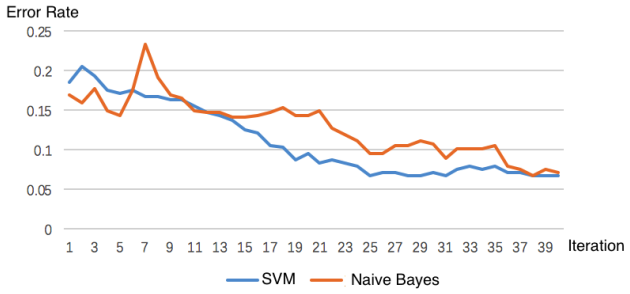

Figure 8: This figure shows the error curve of SVM compared to Naive Bayes.

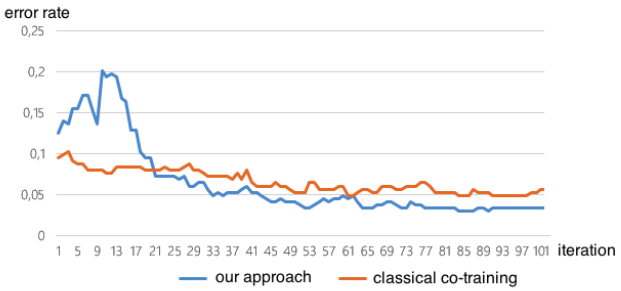

Figure 9: This figure shows the error curve of our approach compared with classical co-training approach with additional initial labels.

3 positive and 9 negative documents as seeds, as suggested by Blum et.al in their work. The rest of the 776 pages are treated as unlabeled data. We then train the SVM classifier with default parameter setting with the classical co-training algorithm and calculate the error rate of the obtained model on test data. We repeat the experiments 10 times and report the average of the results.

Fig. 8 shows that SVM achieves lower or comparable error rates as Naive Bayes Classifier in this setting. The results is in consistent with the suggestion of [14] and confirms that our choice of base classifier is reasonable.

In the second step, we conduct experiments to compare our approach with the classical co-training approach. In these experiments, we follow the same process to divide the dataset into test data, seeds and unlabeled data.

We repeat the experiment 10 times with two master students of computer science and set a time limitation of 10 minutes in total to finish each experiment. We set the number of newly labeled instances in each round to be 70, which means users have to manually inspect the instances labeled by the classifiers for about 6 times during each experiment. In average, users only read the raw data for examining possible errors about 20 times in one experiment session, most of which are mislabeled.

To enable a fair comparison with our approach, we randomly choose 20 more documents as extra seeds, to feed into the classical co-training algorithm, which does not receive further interaction during co-training. We also run ten times of the experiment on the classical co-training algorithm, and report the average performance.

4.2.2 *Experiment results*

The average error rate of the iterations during the experiment session is shown in Fig. 9.

It is obvious that error rate of both approaches decreases along with the iterations, showing that co-training is effective for such classification task. It is also reasonable that our approach makes more errors comparing to the classical co-training at the beginning, because our approach only uses 12 seeds, and the classical co-training uses 32 seeds. But the error rate decreases faster than the classical co-training, with the help of user's reviewing labels and is more stable in the later part of the co-training process. But in fact, our approach beats the classical co-training quite fast, far before the end, and keeps the advantage all the time.

During the experiments, we also notice that with the help of our visualization users tend to put less effort in manually reviewing the labels by reading the raw document text. Usually, users only checked suspicious instances obvious to them within the overview visualization, then they quickly view some information provided by the explainer view. If they keep the doubt regarding the label, they can go on to read the raw text and manually label them as a fall-back strategy. Even when they have to read the raw text, they say, the highlights made it easier for them to grasp important information from the text.

5. DISCUSSIONS AND FUTURE WORK

The result of our experiment indicates that users' interaction is effective in improving the co-training, as well as that we require less user effort in labeling the documents to achieve better performance. The latter may be due to the benefits of our visualization that helps users to identify mislabeled documents, with much less effort. Even through, our approach achieves a relatively lower error rate (5%) on WebKB-Course dataset, the improvement over the classical co-training algorithm is not very significant. The reason might be that the dataset with which we conduct our experiments is highly suitable for the co-training algorithm, so that both our approach and the classical co-training algorithm can obtain a low error rate with relatively few initial labeled data. More experiments on other datasets with different levels of difficulty for co-training algorithm will bring more insight about this aspect.

One limitation of our experiments is that we repeat the experiment with our approach 5 times for each of the two users. Although we did not observe significant learning effect between different runs of the experiments, due to the random assignment of the initial set of labeled data, an experiment with more users would be necessary to increase the validity of the results.

As we have mentioned in the data processing section, our approach needs to identify the relevant features and their importance for the instances under inspection. For some type of classifiers, it is straight forward to rate the importance of their features. With linear classifiers, such as linear SVMs, we can use the feature weights of the trained model as feature importance. For Naive Bayes classifier, we could use the probabilities of the features conditioned on the classes as a measure of their importance. For other types of classifiers, like kernel SVMs, there is research that proposes ways to

derive feature importance for individual test instances [15] for general type of classifiers. In this regard, our approach is quite general.

In this work we focus on using interactive visualizations to improve the performance of the co-training method. In the future we also want to investigate how to depict the changes of the classifiers during the co-training process. This could further increase users' trust in the resulting classifiers. In addition, we only handle binary classification problems, so far. In the future, we aim to extend the approach to multi-class classification problems. One way to achieve that is to divide multi-class classification problems into several binary problems using a 1 vs n strategy or n vs n strategies. Further, in this work, we only deal with one specific multi-view learning algorithm: the co-training algorithm. We could also extend our approach to work with general type of multi-view algorithms.

6. CONCLUSION

In this work we proposed a visual approach for co-training. It includes several visualizations to help users interact with the co-training process. The overview visualization depicts classification uncertainty and disagreement between two classifiers, and enables users to spot possible mislabels. The explainer view shows important features for an instance along with its raw data, and allows users to examine the classification of an instance more closely. The instance list view shows all the instances and their labeling status. The control panel lets user correct the mislabeled instance manually and start a next round of co-training iteration. Together they present an effective way for building classifiers in a human steered semi-supervised way. We showed the effectiveness of our approach through detailed description of a usage scenario and by comparing it to the classical co-training approach.

7. ACKNOWLEDGMENTS

We would like to thank the two master students for testing our approach, the reviewers for their valuable feedback, and our colleagues for the insightful discussions.

8. REFERENCES

[1] J. Allen, C. I. Guinn, and E. Horvtz. Mixed-initiative interaction. *IEEE Intelligent Systems and their Applications*, 14(5):14–23, 1999.

[2] S. Amershi, M. Chickering, S. M. Drucker, B. Lee, P. Simard, and J. Suh. Modeltracker: Redesigning performance analysis tools for machine learning. In *Proc. ACM CHI*, CHI '15, pages 337–346. ACM, 2015.

[3] M.-F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. In *Advances in neural information processing systems*, pages 89–96, 2004.

[4] A. Biswas and D. Jacobs. Active image clustering with pairwise constraints from humans. *Int. J. Comput. Vision*, 108(1-2):133–147, May 2014.

[5] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. ACM COLT*, COLT' 98, pages 92–100. ACM, 1998.

[6] M. Brooks, S. Amershi, B. Lee, S. M. Drucker, A. Kapoor, and P. Simard. FeatureInsight: Visual support for error-driven feature ideation in text classification. In *Proc. IEEE VAST*, pages 105–112. IEEE, 2015.

[7] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[9] S. He and D. Gildea. Self-training and co-training for semantic role labeling: Primary report. Technical report, Technical Report 891, University of Rochester, 2006.

[10] F. Heimerl, C. Jochim, S. Koch, and T. Ertl. FeatureForge: A novel tool for visually supported feature engineering and corpus revision. In *COLING (Posters)*, pages 461–470. Citeseer, 2012.

[11] A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91, 1985.

[12] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[13] A. Kapoor, B. Lee, D. Tan, and E. Horvitz. Interactive optimization for steering machine classification. In *Proc. ACM CHI*, pages 1343–1352. ACM, 2010.

[14] S. Kiritchenko and S. Matwin. Email classification with co-training. In *Proc. CASCON*, CASCON '01, pages 8–. IBM Press, 2001.

[15] C. G. Marco Tulio Ribeiro, Sameer Singh. "why should i trust you?": Explaining the predictions of any classifier. *Proc. ACM SIGKDD*, 2016.

[16] E. T. Matsubara, M. C. Monard, and R. C. Prati. On the class distribution labelling step sensitivity of co-training. In *Artificial Intelligence in Theory and Practice*, pages 199–208. Springer, 2006.

[17] I. Muslea, S. Minton, and C. Knoblock. Selective sampling with naive cotesting: preliminary results. In *The ECAI 2000 workshop on Machine Learning for information extraction*, 2000.

[18] D. Pierce and C. Cardie. Limitations of co-training for natural language learning from large datasets. In *Proc. EMNLP*, pages 1–9, 2001.

[19] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[20] I. Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM New York, 2001.

[21] Z.-H. Zhou and M. Li. Semi-supervised learning by disagreement. *Knowl. Inf. Syst.*, 24(3):415–439, Sept. 2010.

[22] X. Zhu. Semi-supervised learning literature survey. Technical report, Department of Computer Sciences, University of Wisconsin, Madison, 2005.