Building a General Purpose Cross-Domain Sentiment Mining Model

Matthew Whitehead
Indiana University
School of Informatics
901 E. 10th St.
Bloomington, IN 47408
mewhiteh@indiana.edu

Larry Yaeger
Indiana University
School of Informatics
901 E. 10th St.
Bloomington, IN 47408
larryy@indiana.edu

Abstract

Building a model using machine learning that can classify the sentiment of natural language text often requires an extensive set of labeled training data from the same domain as the target text. Gathering and labeling new datasets whenever a model is needed for a new domain is time-consuming and difficult, especially if a dataset with numeric ratings is not available. In this paper we consider the problem of building models that have a high sentiment classification accuracy without the aid of a labeled dataset from the target domain. We show that an adjusted form of cosine similarity between domain lexicons can be used to predict which models will be effective in a new target domain. We also show that ensembles of existing domain models can be used to achieve a classification accuracy that approaches that of models trained on data from the target domain.

1 Introduction

We are constantly being faced with the challenge of dealing with the ever-increasing amount of data being produced. Data mining algorithms and methods are being designed to intelligently process this data and make the underlying patterns and trends more easily understandable by people.

The World Wide Web has facilitated the widespread availability of huge amounts of human-created content. Viewed as a whole, it is a rich database of nearly all human ideas and endeavors. A major challenge for the coming years is to make sense of the available content and use it to enrich our lives in ways that were never before possible. One of the major hurdles that must be addressed is that of making computers understand and intelligently process the large amount of data available in natural language format.

The problem of making a computer understand human natural language is very difficult. Instead of trying to tackle that complex problem directly, we consider the small sub-problem of determining the sentiment of a given piece of text in natural language. For example, if someone writes a review of a particular movie, then the model would be given the task of automatically determining whether or not that review is positive or negative in sentiment. In order to make this determination, a sentiment mining model would, to some degree, have to understand the language being used by the reviewer. Sentiment mining models with a high level of classification accuracy would be useful across many different domains.

Various machine learning models for determining review sentiment work quite well, achieving a better than 85% classification accuracy for standard datasets (Pang et al., 2002). The models are not perfect, but they work well enough to be useful. One of the main drawbacks of using a machine learning model is that it can be difficult and time-consuming to train effectively. Typically, researchers construct a model that is trained on labeled data from the target domain—the domain in which new classifications will be made. The problem with this approach is that a new, separate model must be built for each new target domain. This is difficult since training the new model requires the availability of a rather large amount of labeled training data.

Producing labeled datasets for domains that frequently have numerical ratings coupled with plain natural language reviews is time-consuming, but not too difficult. There are numerous websites which combine ratings and reviews for books, CDs, movies, etc. It is much more difficult to create labeled datasets for domains without such numerical ratings. Examples of these domains include political opinion (what do people think of a politician's latest speech?), personal blogs with their various themes and wandering commentary, interpersonal opinion (what do people think of a particular plumber/real estate agent/lawyer?), and others. Typically, training datasets for these domains must be manually labeled by the model creator in a subjective way since few datasets with numerical ratings exist.

If one were to build and train a different model for every new target domain, then it would take a substantial amount of time to produce the labeled training sets and actually perform the training for each separate domain.

2 Related Work

Dave et al. (2003) and Pang et al. (2002) give nice overviews of the area of sentiment mining in general.

Blitzer et al. (2007) use pivot features based on domain mutual information to relate training and target domains. They also measure domain adaptability by estimating the classification accuracy loss by adapting one domain to another without the use of a labeled target dataset.

Aue and Gamon (2005) compared results using four different training and testing domains. Using unigram, trigram, and ngram feature sets, log likelihood ratios to limit input vector sizes, and paying attention to valence shifters (words that reverse the sentiment of other words), they obtained high performance for in-domain sentiment classification using SVM classifiers, but obtained mixed results for cross-domain sentiment classification, with results ranging from barely above chance to near in-domain accuracies.

Kobayashi et al. (2007) apply what they call opinion extraction, that captures semantic content and structure in weblog data. Their "intra-sentential" model exhibits somewhat modest accuracy in the training domain, but seems to work nearly as well out-of-domain as it does in-domain. Daumé and Marcu (2006) investigate what they term domain adaptation to infer key word locations and types (and make capitalization decisions) in one domain based on data predominantly (but not exclusively) from another. In analyzing their system's performance they characterize corpus similarity using a learned degree of relatedness from their model and Kulback-Liebler divergence between unigram language models of their pairs of domains, but do not use this similarity in a predictive fashion.

Though less directly related, Swarup and Ray (2006) mine recurring sub-graphs in artificial neural networks to discover network motifs that may speed up learning in new domains. Dai et al. (2007) have used co-clustering to perform cross-domain document classification. Blitzer et al. (2008) have established theoretical error bounds for cross-domain classification when target domain data is present but limited compared to source domain data and have demonstrated agreement with the form of error curves on real cross-domain training, although not with quantitative error values. Jiang and Zhai (2006) consider the problem of cross-domain models for the named entity recognition task. Part of their focus was to train a model based on generalizable features with the hope that performance in new domains would improve.

3 General Setup

We conducted a series of experiments to attempt to determine the viability of models trained across domains and general purpose models with high performance in many different domains. For all of our experiments involving machine learning we used the libsvm library (Chang and Lin, 2001). For all SVM models we used the following parameters: linear kernel, C = 2.0, γ = 2.0.

Datasets

In order to determine how well cross-trained models perform, we investigate performance over a variety of review-plus-rating datasets. These datasets represent a wide variety of domains that could all benefit from the creation of accurate sentiment mining models. Two of the datasets were

prepared and used for previous sentiment mining work (see descriptions below), but the majority of the datasets we collected and compiled ourselves. The datasets are publicly available online: http://www.cs.indiana.edu/~mewhiteh/html/opinion_mining.html.

Following are brief descriptions of the datasets used in this study:

camera - Digital camera reviews from Amazon.com. These reviews were taken from cameras that had a large number of ratings. This dataset and the laptop review set both fall under the broader domain of consumer electronics.

camp - Summer camp reviews from CampRatingz.com. A significant number of these reviews were written by the young people who attended the summer camps.

doctor - Reviews of physicians from RateMDs.com. This dataset and the lawyer review set could both be considered part of the larger "ratings of people" domain.

drug - Reviews of pharmaceutical drugs from DrugRatingz.com.

laptop - Laptop reviews from Amazon.com. Various laptops are reviewed from different manufacturers.

lawyer - Reviews of lawyers from LawyerRatingz.com.

movie - Movie reviews of various movies from (Pang and Lee, 2004). Because of the nature of the domain and dataset source, these reviews are typically longer than reviews from other domains.

music - Musical CD reviews from Amazon.com. The albums being reviewed were recently released popular music from a variety of musical genres.

radio - Reviews of radio shows from RadioRatingz.com. This dataset and the tv dataset had the shortest reviews on average.

restaurant - Restaurant reviews from We8There.com as compiled by (Snyder and Barzilay, 2007). We used a subset of this dataset balanced between positive and negative reviews. These reviews are from various restaurants in a number of different cities.

tv - Television show reviews from TVRatingz.com. These reviews were typically very short and not very detailed.

To create these datasets, we ran a web spider to download and organize ratings and reviews from a number of different websites. The resulting datasets are individual text files that alternate reviews and corresponding ratings on consecutive lines. Ratings have values of either 1.0 for a positive sentiment or -1.0 for a negative sentiment. All the websites we gathered reviews and ratings from used a numerical rating scale of 1-5. 5 is the highest score and 1 is the lowest. After a brief preliminary investigation we decided that the threshold for a positive rating should be 3.5 out of 5. So all ratings 3.5 and above were labeled as positive and all ratings less than 3.5 were labeled as negative.

We pared down each of the datasets to include exactly 50% positive reviews and 50% negative reviews. Some of the testing domains are heavily skewed toward one or the other class. This rebalancing eliminates that bias, normalizing the data distributions across domains. This simplifies the comparison of classification accuracies across domains and yields a consistent 50% chance (random guessing) baseline for all domains.

Lexicon Reduction via Odds Ratio

To reduce the computational complexity of the test setups and increase overall classification accuracy, we reduced the size of the domain lexicons. First, we eliminated stopwords (words appearing in over half the reviews) and unique words (words appearing in only a single review).

Second, we chose to use the odds ratio method of feature selection to further reduce the lexicon size. This method eliminates terms from the lexicon which are not useful in distinguishing between positive and negative reviews. Using odds ratio feature selection was particularly important for the general models that combined all the separate datasets since the resulting full lexicons were very large. We found that keeping around 15-25% of the original lexicon was optimal. Yang and Pedersen (1997) offer a comparison of several other dimensionality reduction methods for text categorization.

4 Experiments

4.1 Single Model, Cross Domain Test

Can models trained in one sentiment mining domain be effective classifiers in other domains?

If a sentiment model is trained using restaurant

reviews, what level of performance does that model have classifying movie reviews? Does it have a similar performance classifying reviews of physicians? If we can show that a single model is accurate over multiple domains, then a significant amount of time and effort could be saved.

Setup

The first test measured the performance of classification models trained in one domain and then tested in a different domain. To do this, we trained a new SVM classification model for each domain dataset and then tested each model on all the other datasets. The models were constructed using a bag-of-words representation using normalized word counts. Instead of using normalized word counts, it also would have been possible to use TF/IDF weightings. Final lexicon size was reduced using the odds ratio metric as described above.

We also ran K-fold tests (K=25) for each model being trained and tested on the same dataset. Each reported classification accuracy is the average of the K-fold tests. The results from these models show the performance level that could be achieved given labeled data from the target domain. If the models trained in other domains could approach these levels, then the collection and labeling of data in the target domain would not be necessary.

Results

Table 1 shows the results from the cross domain classification tests. The first observation is that models trained and tested on different domains always had lower classification accuracy than models trained and tested in a single domain. This is an unsurprising result, and simply shows that the way to achieve the highest possible accuracy is to use a model trained in the target domain.

Another result of note is that the accuracies of cross-domain models vary widely. For example, the model trained in the drug domain and tested with the tv domain model only achieved a 53% classification accuracy. This is negligibly better than completely random guessing, which would have produced a 50% accuracy. Compare that result to the model trained in the doctor domain and tested in the restaurant domain. This model had a cross-domain classification accuracy of 76%. That result is much better than guessing and is approaching the accuracy of the model trained in the restaurant domain itself (85%).

By comparing columns of results we can see the classification accuracies across all other domains given each model trained on a single domain. The model trained on the movie domain had poor results in all the other domains tested, with the highest accuracy at only 57%. Training in the doctor or lawyer domains yielded better results, with a majority of the accuracies being 60% or higher.

Examining isolated rows in the table shows how all the different models performed on a single testing domain. Testing in the drug domain resulted in the overall lowest scores, with only a single model achieving a classification accuracy above 60%. This was most likely due to the domain-specific language used in reviews in that domain (brand names of drugs, specific side effects, etc.). The restaurant and camp domains were the easiest for the various models to classify with high scores, with most models yielding 60% accuracy or higher.

4.2 General Model Test

Can a single model be trained using a variety of different sentiment mining domains that has a high classification accuracy throughout all those domains?

The tests described in 4.1 only considered models trained in a single domain and then tested in some other domain. We would also like to consider the performance of a single, general model that is trained on a dataset consisting of a mix of reviews from all the available datasets. If a single model such as this were able to achieve an accuracy that is competitive with (or better than) the models trained on labeled data from the target domain, then the result would be significant, as it would be simpler to build and maintain a single model that would work well in all domains instead of keeping separate models for each different domain.

Setup

To test the viability of a general model, we combined all the distinct datasets together to form a new dataset called “total”. We then used this dataset to perform K-fold tests (K=25) to see how well the general model would perform across all domains in

Table 1: Cross Domain Classification

Testing Dataset \ Training Dataset	camera	camp	doctor	drug	laptop	lawyer	movie	music	radio	rest	tv	Ave. Test
camera	**90**	64	67	57	64	63	51	55	60	61	62	63
camp	71	**85**	69	57	53	68	52	63	62	66	71	65
doctor	59	65	**84**	58	53	72	50	57	63	72	65	63
drug	57	59	59	**72**	53	62	50	54	57	59	56	58
laptop	74	56	56	53	**96**	63	57	50	55	61	52	61
lawyer	57	61	71	55	59	**83**	51	60	60	66	65	63
movie	57	63	56	52	59	59	**81**	55	53	58	59	59
music	58	58	65	61	50	53	50	**88**	61	63	56	60
radio	55	64	62	53	52	62	50	58	**73**	59	58	59
rest	64	67	76	64	53	62	56	64	64	**85**	65	65
tv	61	70	63	53	51	71	52	58	62	63	**82**	62
Ave. Train	64	65	66	58	58	65	55	60	61	65	63	

Table 2: General Model Classification Accuracy

	Classification Accuracy
General Model	80
Ave. of Same-Domain Models	83

which it had previously been trained.

Results

The general model trained and tested on the total dataset had an overall classification accuracy of 80% (see Table 2). This is close to the 83% accuracy that is the average of all the models that had the same domain for training and testing (average of the main diagonal in Table 1). Since the general model performs at a level that approaches the average of the single-domain models, it is a useful alternative when the overhead of training, storing, and choosing from a set of domain-specific models is considered too high.

4.3 Leave-One-Out Test

Given a new target domain for which there is no labeled data available, can a single model trained on a variety of other domains make accurate classifications in the target domain?

The general model test from 4.2 showed that a single model could be competitive with a set of domain-specific models for domains for which there exists labeled data. To evaluate the case where a completely new domain is being explored and no labeled dataset is available, we can test the difference in performance between a general model trained on all other available datasets together versus models trained on individual datasets.

Table 3: Leave-One-Out vs. Single-Domain

Test Set	LOO	Ave. Other	Best Other	Same
camera	62	63	67	90
camp	66	65	71	85
doctor	61	64	72	84
drug	53	58	62	72
laptop	65	61	74	96
lawyer	58	63	71	83
movie	62	59	63	81
music	61	60	65	88
radio	56	59	64	73
restaurant	67	65	76	85
tv	60	62	71	82
Average	61	62	69	83

Setup

To carry out this comparison, we performed leave-one-out tests for each domain. That is, a separate model was trained using all the datasets combined except for the target dataset upon which it was tested.

Results

Table 3 shows how the leave-one-out models compared to the average of the single cross-domain models, the best single cross-domain model, and the single same-domain model. The leave-one-out models performed similarly to the average of the single cross-domain models. The leave-one-out model accuracy was always within about 4-5% of the average of the cross-domain models. In five of the domains, the leave-one-out model was superior, while the average cross-domain model was more accurate in the remaining six domains.

Comparing those results to the best single cross-domain model per testing dataset (the "Best Other" column of Table 3) suggests that there is room for improvement. The best single cross-domain models always had the highest of the cross-domain accuracies (though, as the final column of Table 3 shows, none of the cross-domain formulations approached the accuracy of the models trained and tested on the same domain). Still, the modest difference in accuracy between the best cross-domain models and the average or leave-one-out cross-domain models suggested to us that there may be a way to exploit some form of domain selection to improve our cross-domain performance.

4.4 Lexicon Similarity for Predicting Classification Accuracy Test

Can we predict which single-domain models will do well in a new domain based on the similarities between the training domain's lexicon and the target domain's lexicon?

The results in Table 1 show that models trained in certain domains had much higher performance than models trained in other domains. Table 3 shows that the best single-domain models always outperformed the average of all single-domain models and the leave-one-out general model.

If it were possible to predict which existing model would be most effective for making classifications in the target domain, then that model could be used whenever labeled training data were not available for building a domain-specific model for the target domain. This could also help avoid those models which provide no improvement in accuracy over guessing and potentially achieve a greater classification accuracy in a general model.

Setup

In order to try to make this type of prediction, we decided to look first at a simple form of lexical similarity between domains. The intuition is that if lexicons from two different domains are highly similar, then cross-domain classification will be more accurate than if the lexicons are dissimilar. We calculated an adjusted form of cosine similarity between domain lexicons using the odds ratio values found in the training domain lexicons. Note that in an unlabeled target domain dataset, odds ratio values are not available, so only the values in the training domain dataset can be used. For the purposes of computing the adjusted cosine similarity between lexicons, each unknown odds ratio is given the value of its known counterpart from the training domain. This makes the assumption that if a word is useful distinguishing sentiment in the training domain then it will also be useful in the target domain (as long as it appears in the target domain's lexicon). Using this method, we only need to look for occurrences of words that were useful in other domains, instead of having the requirement of fully labeled target domain datasets.

To investigate whether it might be possible to predict which model would be most accurate for a new target domain, we first computed the cosine similarities between all pairs of domain lexicons and then plotted the previously measured cross-domain model accuracies versus this measure of lexicon similarity. The result showed a positive relationship between increasing lexicon similarity and improved model classification accuracy.

With the positive relationship between lexicon similarity and classification accuracy in mind, we ran another test that used the existing cross-domain model with the lexicon most similar to the target domain lexicon for classifications. If lexicon similarity is a good predictor, then we would expect the classification accuracy of this approach to approach the best single-domain model.

Results

Table 4 shows that choosing the model with a lexicon most similar to the target domain lexicon produced results that were nearly equal to the best of all the single-domain models. (Note that as the test is designed it is not possible for this method to do better than the best single cross-domain classifier; the best it can do is to serve as an oracle and select that best single cross-domain classifier.) Using lexicon similarity to pick a cross-domain model recaptured over half the difference between average and best cross-domain model accuracy, and also improved substantially over the leave-one-out general model.

Being able to predict (approximately) the most accurate existing model trained in a different domain from the target domain is encouraging, but many of

Table 4: Model Choice by Lexicon Similarity

Test Dataset	Ave. Other	Similar Lexicon	Best Other
camera	63	64	67
camp	65	66	71
doctor	64	72	72
drug	58	59	62
laptop	61	74	74
lawyer	63	71	71
movie	59	55	63
music	60	63	65
radio	59	58	64
restaurant	65	76	76
tv	62	62	71
Average	62	66	69

the classification scores are still rather low. Almost all of the scores are in the 60-75% range. Trying to construct a usable end-user application with these accuracies may be problematic.

4.5 Ensemble Models Test

Can we combine groups of models to form ensembles that can outperform leave-one-out and models chosen based on lexicon similarity described in the other tests?

The accuracies we saw using lexicon similarity model prediction were an improvement over the earlier tests, but still lagged behind the performance of the models trained in the target domain. Given the demonstrated benefit of applying lexical similarity, we wondered if domain similarity might be leveraged further.

Our final test attempts to get closer to the trained-in-domain models by leveraging the power of ensemble classification. This method works by creating groups of models that can vote on each new classification. A simple or weighted majority vote can be used to produce the ensemble's final classification for each problem.

Setup

We tried building two different kinds of ensembles. In each case we only used data from datasets outside of the target domain.

The first ensemble used a simple majority vote of its component models for each new classification. Note that if a majority of the models make correct classifications more often than any single cross-domain model does, then this setup would show an improvement over the classification accuracy of even the most accurate single cross-domain model.

Table 5: Accuracy of Ensemble Models

Test Dataset	Best Other	Simple Ensemble	Weighted Ensemble	Same Dom.
camera	67	77	77	90
camp	71	79	78	85
doctor	72	77	82	84
drug	62	65	64	72
laptop	74	79	80	96
lawyer	71	80	80	83
movie	63	71	70	81
music	65	72	74	88
radio	64	70	70	73
restaurant	76	85	85	85
tv	71	80	82	82
Average	69	76	77	83

The second ensemble used weighted majority votes, where each model had a weight proportional to the cosine similarity between its lexicon and the target domain's lexicon. The intuition behind the weighting comes from the previous test's results, where we saw that applying lexicon cosine similarity could help predict which models would be most accurate. This ensemble used squared (and then re-normalized) lexicon similarities as weights to increase the influence of the more similar lexicon models and decrease the influence of the less similar lexicon models.

For each testing dataset we built each of these ensembles using the single-domain models from all datasets except the testing set. We then tested each ensemble model using all testing dataset patterns.

Results

Though still short of the best same-domain performance, results for the ensemble models are promising. Table 5 shows that both the ensemble setups significantly surpassed the best single-domain models from outside of the target domain, recapturing over half the accuracy difference between the best cross-domain model and the same-domain model, on average (and over 70% of the accuracy difference between averaged cross-domain models and the same-domain model). Indeed, over half of the ensemble classification accuracies approached the perfor-

mance of the models trained in their respective target domains. Though the spread is high, with same-vs-ensemble accuracy differences ranging from 0% to 16%, this is still fairly encouraging since it shows it is possible to build a general ensemble model that exhibits an accuracy within about 6%, on average, of the accuracy possible with domain-specific models. For a modest performance decrease, it may therefore be possible to use an ensemble model trained in domains for which labeled data already exists, instead of spending time preparing a labeled dataset for each new target domain.

The weighted ensemble model was only a marginal improvement over the simple majority vote ensemble. Future investigation of different weighting schemes might yield further improvement.

5 Conclusion

In keeping with previous work, our experiments show that using a model trained in one domain and then tested in another produces mixed results. However, we have also shown how a substantial improvement in accuracy can be had by selecting cross-domain models with lexicons that are most similar to the target domain's lexicon. We also demonstrated that a single, combined model can do almost as well across multiple domains as a collection of domain-specific models.

Finally, we have demonstrated a novel method for building ensemble models, using lexicon similarity, that yield a high classification accuracy for domains in which no training was performed. This suggests it may be possible to build a single ensemble model for use in a potentially wide variety of new domains for which labeled training data does not exist.

References

Anthony Aue and Michael Gamon. 2005. Customizing sentiment classifiers to new domains: a case study. In *Submitted to RANLP-05, the International Conference on Recent Advances in Natural Language Processing*, Borovets, BG.

John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jenn Wortman. 2007. Biographies, bollywood, boom-boxes, and blenders: Domain adaptation for sentiment classification. Association for Computational Linguistics (ACL).

John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jenn Wortman. 2008. Learning bounds for domain adaptation. In *Advances in Neural Information Processing Systems 21*, Cambridge, MA. MIT Press.

C. C. Chang and C. J. Lin. 2001. *LIBSVM: a library for support vector machines.* http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. 2007. Co-clustering based classification for out-of-domain documents. In Pavel Berkhin, Rich Caruana, and Xindong Wu, editors, *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*, pages 210–219. ACM.

Kushal Dave, Steve Lawrence, and David M. Pennock. 2003. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *WWW*, pages 519–528.

Hal Daumé III and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *J. Artif. Intell. Res. (JAIR)*, 26:101–126.

Jing Jiang and ChengXiang Zhai. 2006. Exploiting domain structure for named entity recognition. In *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'06)*, pages 74–81.

Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, June 2007*, pages 1065–1074.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, pages 271–278.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. *CoRR*, cs.CL/0205070.

Benjamin Snyder and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *Proceedings of NAACL HLT*, pages 300–307.

Samarth Swarup and Sylvian R. Ray. 2006. Cross-domain knowledge transfer using structured representations. In *AAAI*. AAAI Press.

Y. Yang and J. Pedersen. 1997. A comparative study on feature selection in text categorization. In *ICML*.