Project_Milestone

March 27, 2018

1 Title: Optimizing Transfer Learning in Sentiment Analysis

1.1 W266 Final Project by Arunima Kayath, Anamika Sinha

2 Abstract

3 Introduction

As humans we continually apply our knowledge in one area to learn things faster in another area. For example, it is reasonably easier for a soccer player to learn how to play basketball compared to someone who has not played any sport. Can we apply this knowledge transferrability to machine learning? In machine learning, a good model is one that generalizes well to unseen data based on what it learns from the training data. Creating labelled data is expensive and is not always possible. So effective utilization of existing labeled datasets can have many practical applications. In situations where there is a dearth of training data, transfer learning augments this generalization step by starting from models that have been trained on different data with similar or different tasks.

We will examine the application of transfer learning to the field of sentiment analysis specifically in the task of predicting ratings from reviews. In addition, we are picking concepts from the area of active learning to make transfer learning more effective.

4 Background

As a primary source of inspiration, we have used the PhD Thesis by Robert Remus titled Genre and Domain dependencies in Sentiment Analysis. Remus in the thesis examines the effect of genre and domain in various tasks related to sentiment analysis, one of which is sentiment polarity detection. He uses the concept of domain similarity. While various metrics for similarity are mentioned, the one we have used is JS Divergence to determine the similarity / divergence of word distributions of two domains. He also leverages JS Divergence to select a subset of instances from the source domain for creating a model for domain adaptation to the target domain. In his thesis, he uses instance selection(IS) to pick a subset from the source domain that is similar to the target domain. He shows that IS from source, and from source + target (9:1 ratio) is more effective than source only.

A paper by Hal from 2007 looks at domain adaptation in the realm of supervised learning. It proposes a simple idea of augmenting the feature space of both the source as well as the target domain and use it as the input to a classifier which aims at learning a function to minimize the loss in the target domain.The idea was to take each feature in the source domain and make two version of it namely source-specific and generic.The same is done to the target domain with target-specific and generic.By augmenting the feature space, the algorithm learns to adapt on its own.This approach has been used well for tasks like parts of speech tagging, parsing in the paper but not sentiment analysis. We found this approach very intriguing and would like to apply this to sentiment analysis task as a backup if there are issues with our primary approach.

We also looked at the transferrability of neural networks in a paper by Mou et al from 2016 titled: How Transferable are Neural Networks in NLP Applications? This paper looks at tranferrability in different tasks like sentence classification and also sentiment analysis. For sentiment analysis they looked at LSTM-RNN while CNNs were used for sentence classification. The paper also looks at how transferable different layers of NLP neural models are. The paper finds that transferrability between tasks is dependent on how semantically equivalent the two tasks are. With specific relevance to our project, the paper reports that for sentiment analysis, tranferring and finetuning the embedding layer as well as the hidden layer helps in improving performance. The output layer is more specific to the dataset and performs best when left initialized randomly. An interesting aspect of this paper is the question about when the parameters are ready to transfer. The paper reported a sharp increase in accuracy from epochs 1-5 which plateaus later. We intend to leverage this in our work.

A more recent paper that looks at the problem of domain adaptation for sentiment learning is Domain Adaptation for Large Scale Sentiment Classification : A Deep Learning Approach, by Glorot, Bordes, Bengio. The paper uses an unsupervised deep learning approach (specifically deionized auto-encoders) to extract a meaningful feature representation of each review in an unsupervised fashion. They then train a sentiment classifier (SVM) on the features extracted with this auto-encoder. They find that sentiment classifiers trained with this approach outperform state-of-the-art methods in domain adaptation for sentiment classification on a benchmark of Amazon reviews composed of 4 types of Amazon products (the benchmark was created by Blitzer in 2007). While their work shows a substantial reduction in loss in transfer from source to target domain, there is still a loss.

Yet another interesting work which may lend itself to transfer learning in sentiment analysis has been that of deep contextualized word representation in a paper published this year by Clark el al. This paper explores word vectors each token is assigned a representation that is a function of the entire input sentence. The vectors are derived from a bidirectional LSTM that is trained with a coupled language model(LM) on a large text corpus. The word vectors are learned functions of the internal states of a deep bidirectional language model (biLM). These representations are called ELMo(Embeddings from a language model). This paper investigates the performance of ELMo on several tasks including sentiment analysis where it reports an improvement of 1% in accuracy over state of the art. Adding ELMo to a model increases the sample efficiency considerably, both in terms of number of parameter updates to reach state-of-the-art performance and the overall training set size. Although, this paper does not allude to transfer learning, the efficiency gained here can be helpful when we want to minimize training time on source domain in order to be able to transfer sooner.

Our work in also about leveraging active learning to make transfer learning more effective. Active Learning Literature Survey by Burr Settles (Ref 15) is an interesting summary of many possible approaches to picking the instances we label in the target domain using active learning. There's a few that seemed particularly interesting. a) Labeling instances with maximum uncertainty in labeling using classification b) Using instances that are most likely to change the model. c) Using instances that have some combination of uncertainty and representativeness of the domain we are trying to model. The paper also concludes that active learning in general has been observed to reduce the number of labeled instances needed when laveling is expensive.

We would like to focus on ways to get to the same accuracy as on the target domain model only, by enhancing the source domain with a few examples from the target domain. We want to a) detemine how many examples are needed from the target domain when they are selected randomly b) whether we can reduce the number of examples needed by applying heuristics to pick the examples from the target domain that are least represented in the source domain.

5 Methods

We are using Amazon review data, which is available at http://jmcauley.ucsd.edu/data/amazon/links.html

Data Description: Several million reviews with ratings from Amazon are available, in 24 categories. (Usage of this data requires citation of Ref 4,5.).

We picked 4 categories for our initial analysis:
Toys and Games : (2,252,771 reviews)
Video Games : (1,324,753 reviews)
Automotive : (1,373,768 reviews)
Home & Kitchen : (4,253,926 reviews)

We want to divide the project into 3 parts, the 3rd is a stretch goal. depending on time and challenge of the tasks.

Part 1 : How well do different machine learning models transfer across domains ? We will evaluate 2-3 machine learning techniques (Naïve Bayes, SVM, and Neural Networks) and determine which transfers the best. For neural networks, we will start with pretrained GLOVE embeddings and explore the performance with or without training it further. (Our objective is not to find the best model on the source / target domain by itself, but rather the model that lends best to transferability from source to target).

Measure: Comparative Accuracy of predicting rating from reviews when using one domain versus another. This will define the baseline for transfer learning for part 2.

Part 2: Main part of the project. We will examine and develop techniques to improve transfer learning from one domain to another.

Examples include:

a. Using domain similarity (source, target domain) to pick the source domain for transfer learning. (Ref: 2) We will use JS Divergence or cosine similarity as the metric for domain similarity / difference.

b. Using similarity to pick instances from source domain for transfer learning to target domain. (Ref: 2)

c. Variation of b above, with multiple source domains to pick instances for developing the source model to transfer. (Ref: 2)

d. Evaluate whether adding a small number of labeled target instances can help us get to desired accuracy in target domain.

1st Metric: Nts/Nt where

Nts = # labeled instances from target domain needed when starting from a model built on a source domain, to get to similar accuracy as a model built entirely on the target domain.

Nt = ~ minimum number of labeled instances from target domain to reach maximum possible accuracy when building a model directly on the target domain only.

2nd Metric: - Tts/Tt where

Tts = Epochs needed to get to similar accuracy when starting with a model trained on source domain

Tt = Minimum Epochs needed to reach maximum accuracy with a model trained on the target domain only.

e. Try to determine which examples from target domain would be most helpful in improving accuracy of the model (eg those most different from the source domain) Sample methods to pick examples from target domain :

- Instances in the target domain with the most new vocabulary words vs the source domain.
- Target domain instances with the highest JS Divergence, or lowest cosine similarity vs the source domain. Metric: Same as in 2d, and vs the result in 2d.

f. Stretch goal: If time permits, and we develop sufficient expertise to write the algorithms, we will attempt using Attention based neural networks (Ref: 11), or Auto-Encoders (Ref : 8) to see if they help improve accuracy of transfer learning, and how much fewer training instances are needed with those models.

5.0.1 Note:

In the interest of time, we may skip parts b and c, and focus on d and e. In our opinion, since b and c are about picking a subset of instances from the source domain(s), they are likely to be less effective than d and e.

We would also like to expand e to include additional approaches from active learning literature(Ref 15). ie we would also like to try using uncertainty in label prediction to pick instances to label in the target domain, as well as using a combination of uncertainty and one of the methods in e to pick instances to enhance the source domain with instances from the target domain.

We would love to get feedback on whether that is fine, and if it makes sense to modify our approach in this way.

5.0.2 Details on JS Divergence :

JS Divergence is similar to KL Divergence, but better suited to comparing different domains.

KL Divergence =

$$D_{KL}(P \mid\mid Q) = \sum_x P(x) \log_2 (P(x)/Q(x))$$

JS Divergence =

$$D_{JS}(P \mid\mid Q) = 1/2 * (D_{KL}(P \mid\mid M) + D_{KL}(Q \mid\mid M))$$

where M = 1/2(Q+R)

JS Divergence has the benefit that it is defined even when Q(x) is 0 for a given word, whereas KL Divergence is not.

5.0.3 Details on RNN :

Motivated by the paper on Transferability of Neural networks, which uses an LSTM based RNN for sentiment analysis task, we will use a recurrent neural network with long short term memory(LSTM)units.

Model trained on Source domain = M_S

Model trained on Source domain + Target domain = M_{S+T}

We will start with pretrained GLOVE embeddings with 100 dimensions and finetune it in training of M_S. Then we intend to transfer it to train M_{S+T} allowing it to fine tune further.

We will also initialize the hidden layer of M_{S+T} from the M_S and let it train further.

A softmax layer is added to the last word's hidden state for classification. This layer will be randomly initialized.

Part 3: Stretch goal If time permits, we would like to also examine the effectiveness of transfer learning with different tasks within sentiment analysis. Specifically, we would like to see how transfer learning works in Aspect Based Sentiment Analysis. For this, we would be using the data from SemEval 2016, and a CNN / RNN to predict the aspects and associated sentiments.

Data: Labeled data for fine grained sentiment analysis in two categories in English : restaurants, and laptops. Paper describing the data(Ref 3). Sample analysis paper (Ref 6)

6 Results and discussion

Code is uploaded on Github at : https://github.com/ArunimaKay/Sentiment_Analysis_Transfer_Learning/blob

So far, we have used 4 domains from the Amazon review dataset, and tried two Baseline models (SVM, Naive Bayes), and an initial calculation of similarity metrics across domains. The details of our finding are included in this section.

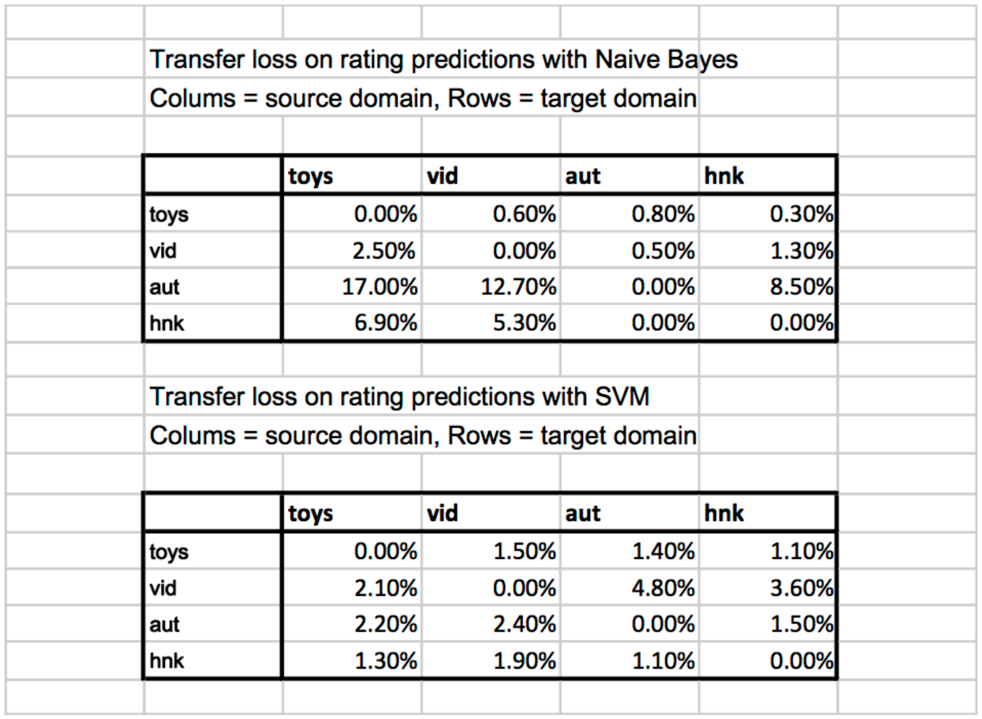
Data Used : The 4 datasets we have used are : Toys and Games : (2,252,771 reviews)
Video Games : (1,324,753 reviews)
Automotive : (1,373,768 reviews)
Home & Kitchen : (4,253,926 reviews)

Data Exploration : The most striking observation from data exploration is that the ratings are strongly skewed towards positive, with 80-85% ratings being 4-5 out of 5. We have not made an attempt to balance the ratings for analysis. We do see better accuracy for positive ratings (~90%) vs for negative ratings (~65-80%). So far we have not chosen to balance the sample since that would mean discarding some positive sample.

Baseline models and transfer learning Below are the results of transfer loss after running Baseline models(Naive Bayes and SVM) on four different domains, namely toys, video games(vid), aut(Automotives) & Home and kitchen(hnk). Transfer loss is defined as (Accuracy of binary rating prediction for target domain wiht a model built on `target` domain) - (Accuracy of binary rating prediction for target domain wiht a model built on `source` domain)

We see that the SVM model does better in total accuracy of prediction in each domain as well as in transfer from one domain to another.

We intend to do a deeper error analysis with respect to precision and recall for each class during transfer to see if transfer is more effective for one class vs another.



Transfer loss on rating predictions with Naive Bayes

Colums = source domain, Rows = target domain

	toys	vid	aut	hnk
toys	0.00%	0.60%	0.80%	0.30%
vid	2.50%	0.00%	0.50%	1.30%
aut	17.00%	12.70%	0.00%	8.50%
hnk	6.90%	5.30%	0.00%	0.00%

Transfer loss on rating predictions with SVM

Colums = source domain, Rows = target domain

	toys	vid	aut	hnk
toys	0.00%	1.50%	1.40%	1.10%
vid	2.10%	0.00%	4.80%	3.60%
aut	2.20%	2.40%	0.00%	1.50%
hnk	1.30%	1.90%	1.10%	0.00%

Note that the transfer loss with SVM is quite low - in the range or 1-5% for overall ratings. This makes us wonder if this is sufficient for us to attempt adding target domain samples to improve accuracy. We may need to pick a task where there is higher transfer loss for part 2. eg we could look at transfer loss for negative class, or at tranfer loss with predicting more granular ratings - 1,2,3,4,5 instead of binary values.

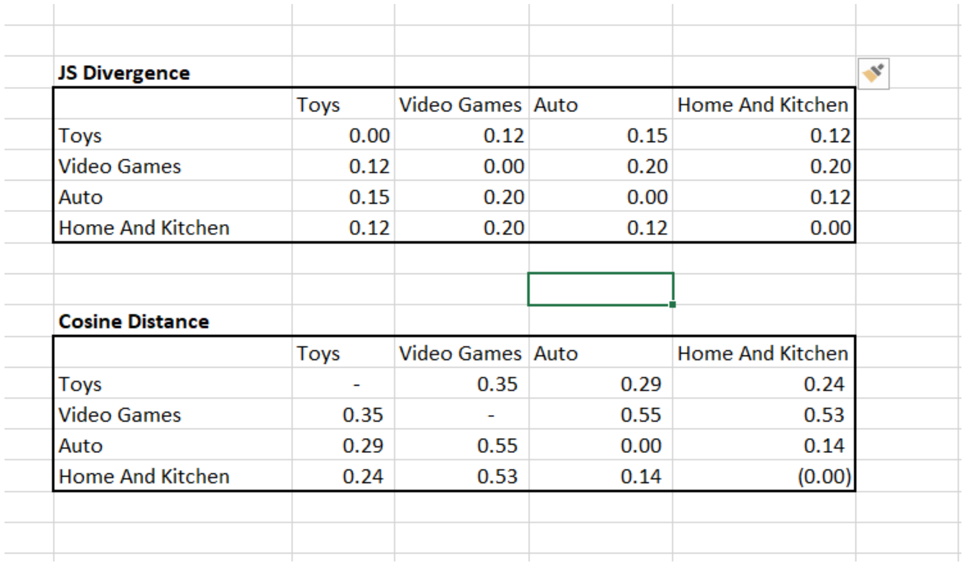
Similarity metrics calculations. We also calculated two similarity / distance metrics for the 4 domains with respect to each other. The metrics we calculated are JS Divergence, and Cosine Similarity. Picking a good metric is critical for moving forward with Part 2, the main part of this project, as we intend to compare the effectiveness of examples picked from transfer domain at random, vs examples picked from transfer domain that are least similar to source domain.

The results of JS Divergence and Cosine Distance Calculations are shown below. Cosine Similarity shows a wider range than JS Divergence.

We also plotted the JS Divergence vs the transfer loss, and cosine similarity vs transfer loss. Note that transfer loss for domain 1 + domain 2 is calculated as the average of ((domain 1 source & domain 2 target), (domain 2 source, domain 1 target)).

We see that the SVM transfer loss is well correlated with both metrics, though cosine distance gives a wider spread. Based on the graphs, we don't see as much of a correlation between Naive Bayes based transfer loss, and distance / divergence metrics. This seems to be influenced by the fact that the source model built on the auto domain in particular does not transfer well to the other domains with Naive Bayes. Naive Bayes in particular may not handle out of vocabulary words in target domain well - likely setting the probability to 0 in those cases. A deeper error analysis may help our understanding here.

Based on this graph, we can say that picking a source domain that has the least cosine distance from the target domain is likely to lead to the least transfer loss. ideally the conclusion could be



JS Divergence

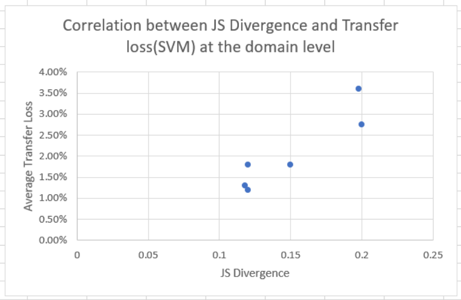
	Toys	Video Games	Auto	Home And Kitchen
Toys	0.00	0.12	0.15	0.12
Video Games	0.12	0.00	0.20	0.20
Auto	0.15	0.20	0.00	0.12
Home And Kitchen	0.12	0.20	0.12	0.00

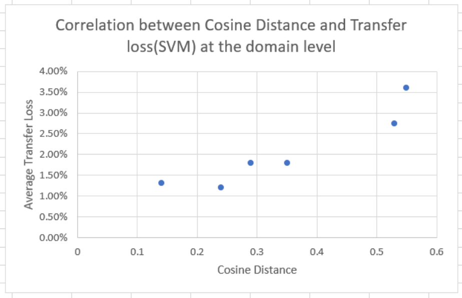
Cosine Distance

	Toys	Video Games	Auto	Home And Kitchen
Toys	-	0.35	0.29	0.24
Video Games	0.35	-	0.55	0.53
Auto	0.29	0.55	0.00	0.14
Home And Kitchen	0.24	0.53	0.14	(0.00)

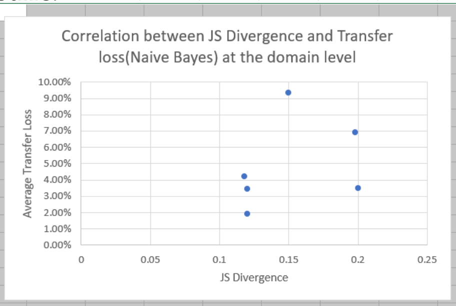
grounded with more domains as data points, but we will focus on doing parts 2 d and e first.

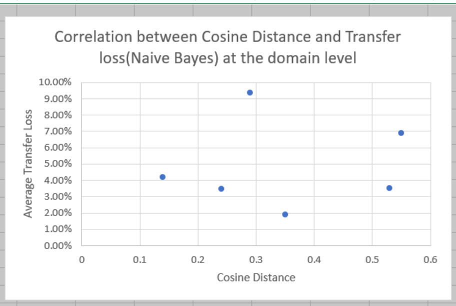
Based on this, it seems to make sense to use SVM for our baseline (pending analysis with a neural net), and use cosine distance for the similarity / distance





metric.





7 Next Steps section for work you plan to do before submitting the final version

1. Check in-domain accuracy with different sample sizes with SVM to determine the minimum sample required to achieve close to max accuracy within domain. This would be the baseline sample size for part 2, when we look at what amount of sample we need to add from the target domain to improve accuracy of a model built on source domain.

2. Get the RNN model working, and determine baseline in-domain sample size, and transfer accuracy without adding any more sample from the target domain.

3. Deeper Error analysis on transfer at the domain and class level

4. Check a few more domains to see if we can find some with greater cosine distance, worse transfer accuracy for more effective analysis.

5. Implement part 2, section d and e of the Methods section above to see how many of the target domain instances we need to add to source domain to get to same accuracy as source domain, when they are

   - selected at random,
   - selected based on cosine distance from source domain
   - selected based on occurence of out of source vocabulary words
   - selected based on uncertainty in prediction using the source model.
   - selected using a combination of uncertainty, and cosine distance or out of vocabulary words.

Within part 2 of the methods section, we are planning to focus on parts d and e, and skip parts b and c in the interest of time. Instead we would like to enhance section e as outlined above.

8 Requesting feedback on the following from instructors

1. What is a good Similarity metric. Does our choice of cosine distance make sense given analysis to date ? is there a better metric, different way of calculating the metric that we should be using.

2. Which is better to use CNN or RNN for transfer learning? Our referenced paper on transferrability of Neural networks suggests RNNs for sentiment analysis but other research suggests CNNs since sequence does not matter for sentiment analysis. Is either OK ? Is one clearly better for predicting at review/document level since each review in our dataset is comprised of multiple sentences?

3. Please refer Methods section. Is it okay to skip parts b and c, and focus on d and e in the interest of time.

4. When using source domain model for target domain, should we build word id vectors on a vocabulary created only on the source domain, or a vocabulary built on the source and target domain. The latter is feasible since it only uses the unlabeled data to build the vocabulary.

9 References

1. Book : Sentiment Analysis and Opinion Mining, Bing Liu.
2. Genre and Domain Dependencies in Sentiment Analysis (PhD Thesis) http://www.qucosa.de/fileadmin/data/qucosa/documents/16543/dissertation_rremus_angenommen_2
3. SemEval 2016, Task 5, paper describing the data: https://www.researchgate.net/publication/305334494_Se 2016_Task_5_Aspect_Based_Sentiment_Analysis)
4. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering R. He, J. McAuley WWW, 2016 Pdf
5. Image-based recommendations on styles and substitutes J. McAuley, C. Targett, J. Shi, A. van den Hengel SIGIR, 2015 pdf
6. INSIGHT-1 at SemEval-2016 Task 5: Deep Learning for Multilingual Aspect-based Sentiment Analysis. https://arxiv.org/abs/1609.02748v2
7. How Transferable are Neural Networks in NLP Applications? https://arxiv.org/pdf/1603.06111.pdf
8. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach http://www.icml-2011.org/papers/342_icmlpaper.pdf
9. Frustratingly Easy Domain Adaptation: Hal Daumt'e III. www.umiacs.umd.edu/~hal/docs/daume07easyadapt.pdf
10. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification http://john.blitzer.com/papers/sentiment_domain.pdf
11. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs http://www.aclweb.org/anthology/Q16-1019
12. Learning Attitudes and Attributes from Multi-Aspect Reviews http://i.stanford.edu/~julian/pdfs/icdm2012.pdf
13. Deep contextualized word representations https://arxiv.org/pdf/1802.05365.pdf
14. Domain Adaptation with Structural Correspondence Learning http://john.blitzer.com/papers/emnlp06.pdf
15. http://burrsettles.com/pub/settles.activelearning.pdf