Robust Cross-Domain Sentiment Analysis for Low-Resource Languages

Jakob Elming **Dirk Hovy** **Barbara Plank**
Centre for Language Technology
University of Copenhagen
zmk867@hum.ku.dk,{dirk,bplank}@cst.dk

Abstract

While various approaches to domain adaptation exist, the majority of them requires knowledge of the target domain, and additional data, preferably labeled. For a language like English, it is often feasible to match most of those conditions, but in low-resource languages, it presents a problem. We explore the situation when neither data nor other information about the target domain is available. We use two samples of Danish, a low-resource language, from the consumer review domain (film vs. company reviews) in a sentiment analysis task. We observe dramatic performance drops when moving from one domain to the other. We then introduce a simple offline method that makes models more robust towards unseen domains, and observe relative improvements of more than 50%.

1 Introduction

Sentiment analysis, the task of determining the polarity of a text, is a valuable tool for gathering information from the vast amount of opinionated text produced today. It is actively used in reputation management and consumer assessment (Amigó et al., 2012; Amigó et al., 2013). While supervised approaches achieve reasonable performance (Mohammad et al., 2013), they are typically highly domain-dependent. In fact, moving from one (source) domain to a different (target) domain will often lead to severe performance drops (Blitzer et al., 2007; Daumé et al., 2010). This is mainly due to the models overfitting the source (training) data, both in terms of its label and word distribution. The task of overcoming this tendency is known as domain adaptation (DA) (Blitzer et al., 2007; Daumé et al., 2010).

There are three different approaches to DA: in *Supervised DA*, labeled training data for the target domain exists, in *Unsupervised DA*, data for the target domain exists, but it is unlabeled. A third, less investigated scenario is *Blind DA*: the target domain is not known at all in advance. Supervised DA effectively counteracts domain-bias by including labeled data from the target domain during training, thus preventing overfitting to both the label and the word distribution of the source. Unsupervised methods usually rely either on external data, in the form of gazetteers, dictionaries, or on unlabeled data from the target domain. While they do not prevent overfitting to the source domain's label distribution, the additional data acts as a regularizer by introducing a larger vocabulary.

However, both cases presuppose that we already know the target domain and have data from it. In many real-world settings, these conditions are not met, especially when dealing with low-resource languages. We thus need to regularize our models independent of the possible target domains. Effectively, this means that we need to prevent our models from memorizing the observed label distribution, and from putting too much weight on features that are predictive in the source domain, but might not even be present in the target domain.

In this paper, we investigate sentiment analysis for Danish, a low-resource language, and therefore approach it as a *Blind DA* problem. We perform experiments on two types of domains, namely reviews for movies and companies. The challenge lies in the fact that the label distribution (positive, negative, neutral) changes dramatically when moving from one domain to the other, and many highly predictive words in the company domain (e.g., "reliable") are unlikely to carry over to the movie domain, and vice versa. To the best of our knowledge, this is the first study to perform sentiment analysis for Danish, a low-resource language where relevant resources like polarity dictionaries

Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 2–7,
Baltimore, Maryland, USA. June 27, 2014.

are hard to come by.

We present a simple offline-learning version inspired by previous work on corruptions (Søgaard, 2013), which also addresses the sparsity of available training data. Our method introduces a relative improvement on out-of-domain performance by up to 54%.

2 Robust Learning

The main idea behind robust learning is to steer the model away from overfitting the source domain. Overfitting can occur either by

1. putting too much weight on certain features (which might not be present in the target domain), or

2. over-using certain labels (since the label distribution on the target domain might differ).

One approach that has been proven to reduce overfitting is data corruption, also known as dropout training (Søgaard and Johannsen, 2012; Søgaard, 2013), which is a way of regularizing the model by randomly leaving out features. Intuitively, this approach can be viewed as coercing the learning algorithm to rely on more general, but less consistent features. Rather than learning to mainly trust the features that are highly predictive for the given training data, the algorithm is encouraged to use the less predictive features, since the highly predictive features might be deleted by the corruption. Most prior work on dropout regularization (Søgaard and Johannsen, 2012; Wang and Manning, 2012; Søgaard, 2013) has used online corruptions, i.e., the specific dropout function is integrated into the learning objective and thus tied to the specific learner. Here, we propose a simple approximation, i.e., a wrapper function that corrupts instances in an off-line fashion based on the weights learned from a base model. The advantage is that it can be used for any learning function, thereby abstracting away from the underlying learner.

2.1 Our approach

Our off-line feature corruption algorithm works as follows:

1. train an uncorrupted (base) model,
2. create k copies of the training data instances,
3. corrupt copies based on the feature weights of the base model and an exponential function (described below), and
4. train a new model on the corrupted training data.

The advantages of this algorithm compared to online corruption are

1. it is a wrapper method, so it becomes very easy to move to a different learning algorithm, and

2. corruption is done based on knowledge from a full, uncorrupted model, which provides a better picture of the overfitting.

This comes, however, at the cost of longer training times, but in a low-resource language training time is less of an issue.

Specifically, multiple copies of the training data are used in the corrupted training stage. This results in each data point appearing in different, corrupted versions, as visualized in Figure 1. The copying process retains more of the information in the training data, since it is unlikely that the same feature is deleted in each copy. In our experiments, we used k=5. Larger values of k resulted in longer training times without improving performance.

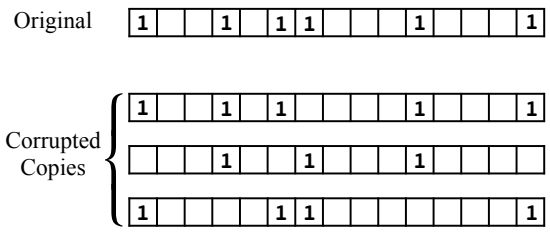

Figure 1: Example of an original feature vector and its multiple corrupted copies.

We experiment with a $random$ and a $biased$ corruption approach. The first approach (Søgaard and Johannsen, 2012) does not utilize the feature weight information from the base model, but randomly deletes 10% of the features. We use this approach to test whether an effect is merely the result of deleting features.

The biased approach, on the other hand, targets the most predictive features in the base model for deletion. We use a function that increases the probability of deleting a feature exponentially

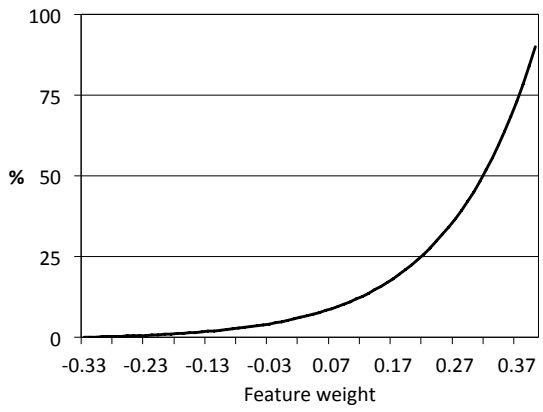

Figure 2: The corruption function conditioning the probability of deleting a feature in a positive instance on its weight in the Scope baseline model.

with its model weight. That is, a highly predictive feature (with a high weight in the model) will be more likely to be deleted. A feature with a low weight, on the other hand, has a much lower chance of being deleted. Figure 2 visualizes the exponential corruption function used. The function assigns the lowest weighted feature of the model zero likelihood of deletion, and the highest weighted feature a 0.9 likelihood of deletion. In order to mainly corrupt the highly predictive features, the exponential function is shifted to an area with a steeper gradient. That is, instead of scaling to the exponential function between 0 and 1, it is scaled to the area between -3 and 2 (parameters set experimentally on the development set). The corruption probability p_{cor} of deleting a feature f given a category c is defined as

$$p_{cor}(f|c) = \frac{exp(\frac{w(f|c)-w_{min}(c)}{w_{max}(c)-w_{min}(c)} * 5 - 3) - exp(-3)}{exp(2) - exp(-3)} * 0.9 \quad (1)$$

with $w(f|c)$ being the weight of f given the instance category c in the model, and $w_{min}(c)$ and $w_{max}(c)$ being the lowest and highest weights of the model respectively for category c.

3 Experiments

Our experiments use Danish reviews from two domains: movies and companies. The specifications of the data sets are listed in Table 1 and Figure 3. The two data sets differ considerably in data size and label distribution.

DOMAIN	SPLIT	REVIEWS	WORDS
Scope	Train	8,718	749,952
	Dev	1,198	107,351
	Test	2,454	210,367
	Total	12,370	1,067,670
Trustpilot	Train	170,137	7,180,160
	Dev	23,958	1,000,443
	Test	48,252	2,040,956
	Total	242,347	10,221,559

Table 1: Overview of data set and split sizes in number of reviews and number of words.

3.1 Data preparation

The movie reviews are downloaded from a Danish movie website, `www.scope.dk`. They contain reviews of 829 movies, each rated on a scale from 1 to 6 stars. The company reviews are downloaded from a Danish consumer review website, `www.trustpilot.dk`. They consist of reviews of 19k companies, each rated between 1 and 5 stars.

Similar to prior work on sentiment analysis (Blitzer et al., 2007), the star ratings are binned into the three standard categories; positive, neutral, and negative. For the Scope data, a 6 star rating is considered positive, a 3 or 4 rating neutral, and a 1 star rating negative. 2 and 5 star ratings are excluded to retain more distinct categories. For the Trustpilot data, 5 star reviews are categorized as positive, 3 stars as neutral, and 1 star as negative. Similar to Scope data, 2 and 4 stars are excluded.

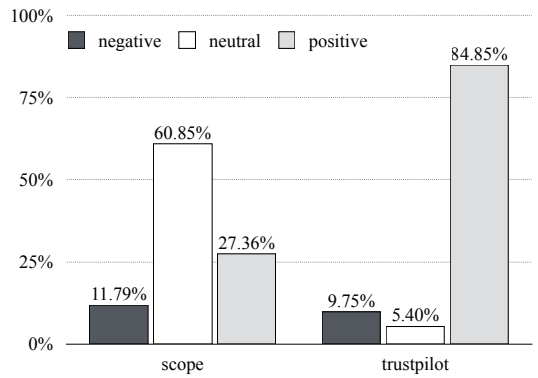

Figure 3: Label distribution in the two data sets.

Apart from the difference in size, the two data sets also differ in the distribution of categories (see Figure 3). This means that a majority label baseline estimated from one would perform horribly on

- N-gram presence for token lengths 1 to 4
- Skip-grams (n-gram with one middle word replaced by *) presence for token lengths 3 to 4
- Character n-gram presence for entire document string for token lengths 1 to 5
- Brown clusters (Brown et al., 1992; Liang, 2005) estimated on the source training data
- Number of words with only upper case characters
- Number of contiguous sequences of question marks, exclamation marks, or both
- Presence of question mark or exclamation mark in last word
- Number of words with characters repeated more than two times e.g. 'sooooo'
- Number of negated contexts using algorithm described in the text
- Most positive, most negative, or same amount of polar words according to a sentiment lexicon

Table 2: Feature set description.

the other domain. For instance, the majority baseline on Scope (assigning $neutral$ to all instances) achieves a 5% accuracy on Trustpilot data. Similarly, the Trustpilot majority baseline obtains an accuracy of 27% on Scope data by always assigning $positive$.

We choose not to balance the data sets, in keeping with the blind DA setup. Knowing the target label distribution can help greatly, but we can assume no prior knowledge about that. In fact, the difference in label distribution is one of the major challenges when predicting on out-of-domain data.

3.2 Features

The features we use (described in Table 2) are inspired by the top performing system from the SemEval-2013 task on Twitter sentiment analysis (Mohammad et al., 2013).

One main difference is that Mohammad et al. (2013) had several high-quality sentiment lexicons at their disposal, shown to be effective. Working with a low-resource language, we only have access to a single lexicon created by an MA student (containing 2248 positive and 4736 negative word forms). Our lexicon features are therefore simpler, i.e., based on whether words are considered positive or negative in the lexicon, as opposed to the score-based features in Mohammad et al. (2013).

We adopted the simple negation scope resolution algorithm directly from Mohammad et al. (2013). Anything appearing in-between a negation token[1] and the first following punctuation mark is considered a negated context. This works well for English, but Danish has different sentence adverbial placement, so the negation may also appear after the negated constituent. This simple algorithm is therefore less likely to be beneficial in a Danish system. We plan to extend the system for better negation handling in future work.

3.3 Corruption

The corruption happens at the feature-instance level. When we refer to the deletion of a feature in the following, it does not mean the deletion of this feature throughout the training data, but the deletion of a single instance in a feature vector (cf. Figure 1).

Corrupting the Scope data deleted 9.24% of all feature instances in the training data. Most features are deleted from $positive$ instances (16.7% of all features) and least from the majority $neutral$ instances (6.5% of all features). Only 9.4% of the minority class $negative$ are deleted.

For Trustpilot, the corruption deleted 11.73% of the feature instances. The pattern is the same here, though more extreme. The majority $positive$ class has the fewest features removed (2.2%), the minority class $neutral$ has 22.8% of its features deleted, and the $negative$ class has an overwhelming 35.6% of its features deleted.

The fact that the corruption function does not take the weight distribution of the individual labels into account, and therefore corrupts the data of some labels much more than others, does prove to be a problem. We will get back to this in the results section.

4 Results

Table 3 shows the results of the experiments. We report both accuracy and the average f-score for positive and negative instances (AF).

AF is the official SemEval-2013 metric (Nakov et al., 2013). It offers a more detailed insight into

[1] We use the following negation markers: *ikke, ingen, intet, ingenting, aldrig, hverken. næppe*.

System	In-domain				Out-of-domain			
	Dev set		Test set		Dev set		Test set	
	Acc.	AF	Acc.	AF	Acc.	AF	Acc.	AF
Scope baseline	**84.2**	**75.6**	82.4	72.1	35.5	43.3	36.0	44.3
Scope random corrupt	83.1	72.9	**82.7**	**72.8**	35.7	43.9	36.2	44.5
Scope biased corrupt	82.7	72.6	81.5	70.6	**55.5**	**48.6**	**55.5**	**44.9**
Trustpilot baseline	**94.8**	**91.8**	94.3	91.2	39.9	45.0	39.9	**46.2**
Trustpilot random corrupt	**94.8**	91.7	**94.4**	**91.4**	39.8	45.6	40.0	46.0
Trustpilot biased corrupt	93.7	89.0	93.4	89.5	**43.6**	**45.7**	**43.4**	44.7

Table 3: Evaluation on development and test sets measured in accuracy (Acc.) and the average f-score for positive and negative instances (AF).

the model's performance on the two "extreme" classes, but it is highly skewed, since it ignores the $neutral$ label. As we have seen in our data, this can make up the majority of the instances. Accuracy has the advantage that it provides a clear picture of how often the system makes a correct prediction, but can be harder to interpret when the data sets are highly skewed in favor of one class.

The results show that randomly corrupting the data (cf. Søgaard and Johannsen (2012), Sec. 5) does not have much influence on the model. Performance on in- and out-of-domain data is similar to the baseline system. This indicates that we can not just delete *any* features to help domain adaptation.

The biased corruption model, on the other hand, makes informed choices about deleting features. As expected, this leads to a drop on in-domain data, since we are underfitting the model. Considering that the algorithm is targeting the most important features for this particular domain, the drop is relatively small, though. The percentage of features deleted is roughly the same as the 10% for the random system (see section 3.3).

With the exception of AF on Trustpilot test, our biased corruption approach always increases out-of-domain performance. The increase is especially notable when the model is trained on the small domain, Scope. On both test and development, the corruption approach increases accuracy more than 50%. On the AF measure, the increase is smaller, which indicates that most of the increase stems from the $neutral$ category. On the test set, the f-score for $positive$ labels increases from 49.1% to 71.2%, $neutral$ increases from 13.5% to 18.4%, but $negative$ decreases from 39.4% to 27.5%. The fact that f-score decreases on $negative$ indicates that the corruption algorithm is too aggressive for this category. We previously saw that this was the category where 35% of the features are deleted.

The lower degree of overfitting in the corrupted model is also reflected in the overall label distribution. For the Scope system, the training data has a negative/neutral/positive distribution (in percentages) of 27/61/12. The baseline predictions on the Trustpilot data has a very similar distribution of 30/63/7, while the corrupted system results in a very different distribution of 52/35/13, which is more similar to the Trustpilot gold distribution of 85/5/10. The KL divergence between the baseline system and the Trustpilot data is 1.26, while for the corrupted system it is 0.46.

5 Related Work

There is a large body of prior work on sentiment analysis (Pang and Lee, 2008), ranging from work on well-edited newswire data using the MPQA corpus (Wilson et al., 2005), to Amazon reviews (Blitzer et al., 2007), blogs (Kessler et al., 2010) and user-generated content such as tweets (Mohammad et al., 2013). All of these studies worked with English, while this study – to the best of our knowledge – is the first to present results for Danish.

As far as we are aware of, the only related work on Danish is Hardt and Wulff (2012). In their exploratory paper, they investigate whether user populations differ systematically in the way they express sentiment, finding that positive ratings are far more common in U.S. reviews than in Danish ones. However, their paper focuses on a quantitative analysis and a single domain (movie reviews), while we build an actual sentiment classification system that performs well *across* domains.

Data corruption has been used for other NLP

tasks (Søgaard and Johannsen, 2012; Søgaard, 2013). Our random removal setup is basically an offline version of the approach presented in (Søgaard and Johannsen, 2012). Their online algorithm removes a random subset of the features in each iteration and was successfully applied to cross-domain experiments on part-of-speech tagging and document classification. Søgaard (2013) presents a follow-up online approach that takes the weights of the current model into consideration, regularizing the most predictive features. Our $biased$ approach is inspired by this, but has the advantage that it abstracts away from the underlying learner.

6 Discussion and Future Work

We investigate cross-domain sentiment analysis for a low-resource language, Danish. We observe that performance drops precipitously when training on one domain and evaluating on the other. We presented a robust offline-learning approach that deletes features proportionate to their predictiveness. Applied to blind domain adaptation, this corruption method prevents overfitting to the source domain, and results in relative improvements of more than 50%.

In the future, we plan to experiment with integrating the weight distribution of a label into the corruption function in order to prevent over-corrupting of certain labels.

Acknowledgments

We would like to thank Daniel Hardt for hosting the Copenhagen Sentiment Analysis Workshop and making the data sets available. The last two authors are supported by the ERC Starting Grant LOWLANDS No. 313695.

References

Enrique Amigó, Adolfo Corujo, Julio Gonzalo, Edgar Meij, and Maarten de Rijke. 2012. Overview of RepLab 2012: Evaluating Online Reputation Management Systems. In *CLEF*.

Enrique Amigó, Jorge Carrillo de Albornoz, Irina Chugur, Adolfo Corujo, Julio Gonzalo, Tamara Martín, Edgar Meij, Maarten de Rijke, and Damiano Spina. 2013. Overview of RepLab 2013: Evaluating Online Reputation Monitoring Systems. In *CLEF*.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*.

P.F. Brown, P.V. Desouza, R.L. Mercer, V.J. DellaPietra, and J.C. Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Hal Daumé, Abhishek Kumar, and Avishek Saha. 2010. Frustratingly easy semi-supervised domain adaptation. In *ACL Workshop on Domain Adaptation for NLP*.

Daniel Hardt and Julie Wulff. 2012. What is the meaning of 5 *'s? An investigation of the expression and rating of sentiment. In *Proceedings of KONVENS 2012*.

Jason S. Kessler, Miriam Eckert, Lyndsie Clark, and Nicolas Nicolov. 2010. The 2010 ICWSM JDPA sentiment corpus for the automotive domain. In *ICWSM-DWC*.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *SemEval-2013*.

Preslav Nakov, Sara Rosenthal, Zornitsa Kozareva, Veselin Stoyanov, Alan Ritter, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Second Joint Conference on Lexical and Computational Semantics (*SEM)*.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Anders Søgaard and Anders Johannsen. 2012. Robust learning in random subspaces: Equipping nlp for oov effects. In *COLING*.

Anders Søgaard. 2013. Part-of-speech tagging with antagonistic adversaries. In *ACL*.

Sida Wang and Christopher D Manning. 2012. Fast dropout training for logistic regression. In *NIPS workshop on log-linear models*.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *EMNLP*.