Master's Essay

Subtitle

Adrian Eriksen

Thesis submitted for the degree of
Master in Language Technology
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2021

Master's Essay

Subtitle

Adrian Eriksen

Master's Essay

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Contents

List of Figures

List of Tables

Preface

Part I

Introduction

Chapter 1

Background

In this essay, I will explain the challenges of cross-domain sentiment analysis and how we might use transfer learning to solve it. Sentiment analysis is the process of trying to understand the sentiment behind a statement or document using machine learning. Sentiment analysis is, among other things, used to get information from reviews that can provide useful information. There are many different ways to formulate a sentiment. A movie review might state, "The movie is not bad at all.". If we simply look for words like "bad" and classify them as negative, we will get inaccurate results.

First, I will discuss some of the different kinds of sentiment analysis in more depth. Then, I will talk about some of the most popular datasets for sentiment analysis on sentence level and up, before I go into NoReC, which is a Norwegian dataset for sentiment analysis. Finally, I will discuss domain adaptation, which will be my thesis's primary focus.

1.1 Sentiment Analysis

Sentiment analysis (SA) is the computational treatment of opinions, sentiments, and subjectivity of texts. SA is also known as opinion mining and a few other terms and has a variety of different applications. You can use it for labeling reviews of movies or books, opinion mining from sites like Twitter.

You can use SA on different levels of text. Document-level SA is the task of classifying the sentiment of a document. The document in this context will be considered one piece of information, and the author's overall sentiment determines the score this document receives. By applying SA to different documents regarding the same topic, we get a score based on the total number of positive and negative documents.
Sentence-level SA looks at each sentence as positive, negative, or neutral, sometimes with different intensities. When looking at the sentences in a document, there are different levels of subjectivity that can be observed. Some sentences will only state a fact like "The restaurant serves Italian food", while others contain subjective opinions like "The restaurant closes too early". The subjectivity of sentences can affect the intensity, as a sentence based on an opinion or a certain belief usually indicates a stronger intensity than stating a fact.
Aspect-based SA is different than the two other approaches. Instead of classifying a word, sentence, or document as either positive or negative, aspect-based SA is tasked with identifying different aspects associated with a target [Pontiki et al., 2016]. One of the main contributions of aspect-based SA is that in addition to learning whether a review is positive or negative, you also learn which aspects of the review that made it so. If we're looking at a review for a hotel, we could retrieve information like "The breakfast was good", "They never made our bed". With information like this, we can assign the aspect, food, of the target, hotel, has a positive polarity. Likewise, the aspect, service - room, of the target, hotel, has a negative polarity. Being able to extract features like this instead of just "The hotel got a score of 3", is very valuable to most businesses since many consumers share their experiences with products online.

As of today, there are two main approaches to SA, the lexicon-based approach and the machine learning approach. The lexicon-based approach is mostly used on a document-level or sentence-level and uses a lexicon with words or multiword terms. These are usually tagged with sentiment (positive or negative) and sometimes with different intensities (very positive, slightly negative, etc.). Given the word or multiword terms, you can further calculate the sentence's value and then the entire document. One way to do this is by assigning each word a score with either positive or negative numbers while taking negation into account. For example, "The movie was not excellent" should yield a higher score than "The movie was not good", as a strongly polarized word usually reflects a somewhat mixed opinion [Taboada et al., 2011]. One of the benefits of lexicon-based SA is that you

don't have any need for labeled data, as the lexicon is pre-defined, and that you get some robustness when applying it on different domains if the lexicon is well-made [Taboada et al., 2011].
The machine learning approach can create a model from a labeled training dataset and then apply it to the target data through standard machine learning methods [Pang et al., 2002], vectorization [Peters et al., 2018, Mikolov et al., 2013, Pennington et al., 2014] or use a premade model and fine-tuning it on the target data [Devlin et al., 2019]. The aforementioned machine learning approaches use unsupervised, semi-supervised, and supervised learning. What they all have in common is that they don't reference any lexicon with pre-defined sentiments. You train an algorithm on graded reviews and then have it predict the grade given to unseen reviews. Much like the lexicon-based approach, machine learning is done on both document-level and sentence-level SA.
One final approach that has been used for all granularities of SA is a hybrid between the machine learning and lexicon-based approach [Zhang et al., 2011], where you first train an algorithm on labeled data before comparing the results with a lexicon to improve accuracy. This approach can also be used for aspect-based SA [Brun et al., 2016]. One of the original challenges with SA was that sentiment is rarely identifiable by keywords alone [Pang et al., 2002]. When humans are presented with the task of selecting a set of keywords to tell whether a movie review is positive or negative, our intuition often leads us towards words like "horrible", "boring", and "sucks" for negative reviews and "excellent", "thrilling" and "amazing" for positive reviews. As it turns out, selecting words like these gives us a much lower accuracy than if we train a model on labeled reviews, letting the model figure out which words are important. For my thesis, I will only focus on document-level sentiment analysis. In the next section, I will discuss some of the most popular datasets for document-level sentiment analysis.

1.2 Datasets

There are many different datasets that are commonly used for sentiment analysis, covering a variety of domains. The dataset that I will use is in Norwegian, but most of the data used in NLP is in English. This section will discuss some of the most common English ones before taking a closer look at the one I will use, which is in Norwegian. I will only discuss datasets that have been used for sentiment analysis on sentence-level and up.

1.2.1 Stanford Sentiment Treebank

The Standford Sentiment Treebank [Socher et al., 2013] is a dataset consisting of 11.855 single sentences from movie reviews and fine-grained sentiment labels for 215 thousand phrases. The intensity of the polarities is divided into five classes, from very positive to very negative. The dataset has been used as a benchmark to test new language models, as a way to

demonstrate high performance.

1.2.2 IMDb Movie Review Dataset

The IMDb Movie Review Dataset [Maas et al., 2011] has close to 50.000 movie reviews, with 25.000 being labeled as positive and 25.000 as negative. It is a dataset for binary sentiment classification, where there are no more than 30 reviews for any given movie. The dataset contains reviews with a score equal to or below 4/10, or a score equal to or above 7/10 so that there are no neutral ratings.

1.2.3 Amazon Review Data

Amazon Review Data [Ni et al., 2019] is a dataset containing 233.1 million product reviews and metadata from Amazon. It includes reviews consisting of text, ratings, and helpfulness votes, product metadata consisting of descriptions, category information, price, brand and image features, and links to "also viewed/also bought graphs. The reviews cover 29 different domains, including books, music, electronics, video games, beauty, and toys, albeit all of them are Amazon-products.

1.2.4 Twitter

Twitter has long been one of the most important and influential data-source for opinion mining. Already back in 2010, it had millions of users tweeting daily, sharing their opinion on almost everything [Pak and Paroubek, 2010]. The use of hashtags makes the data more easily separable, the word-limit per tweet makes each document concise, and the amount of data grows every day.

1.2.5 Yelp

The Yelp Review dataset [Zhang et al., 2016] consists of more than 500,000 Yelp reviews and is one of the datasets used for benchmarking SA models. There is both a binary and fine-grained version of the dataset.

1.3 NoReC

The Norwegian Review Corpus (NoReC) is a dataset containing more than 43,000 full-text reviews from Norwegian news sources [Velldal et al., 2018]. NoReC covers a range of different domains, including literature, movies, video games, restaurants, music, and theater, in addition to product reviews across a range of categories. Each review is labeled with a score ranging from 1-6, provided by the review author. NoReC was primarily created for training and evaluating models for document-level sentiment analysis, making it ideal for testing differences between domains on a document-level.

1.3.1 Distribution of categories and ratings

The dataset has a good spread between scores, with 3, 4, and 5 being the most frequent. The spread makes sense, as it usually takes something particularly bad to give a score of 1 or 2 or something extraordinary to give a score of 6. Figure 1.1 shows a distribution of the ratings. Looking

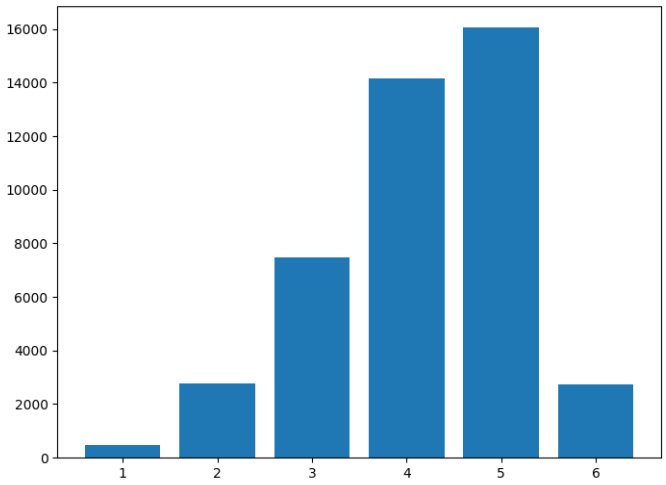

Figure 1.1: Distribution of ratings in the NoReC dataset

at the distributions of the categories, we can see from table 1.2 that the categories 'screen' and 'music' is the most represented by far. One of the reasons for this is that the different sources label their categories differently. Upon further inspection, we can see from figure 1.3 and figure 1.4 that these cover a variety of different categories from the different sources (I have removed the categories 'utenriks', 'kjendis' and 'nyheter' from figure 1.3 because these represented one, four and one reviews respectively). The dataset mainly cover the timespan 2003-2019, but it does contain a handful of reviews dating as far back as 1998. To make up for how language changes over time, the reviews were first sorted by date from old to new. Next, they were divided into 80% for training data, 10% for development data and 10% for test data. In a real world scenario, a model trained on this data is more likely to be applied on newer reviews, so by training on the oldest data and finetuning on newer data, it takes effects like shifts in vocabulary over time into account. As we can see from table 1.4 this also affects the average length of reviews. From this table I would namely like to mention the sports category, as this has a significant decrease of average length in the development and test set. As we will see later, this also drastically affects the accuracy of the model when testing on the sports domain.

Further, I wanted to see whether there was any significant difference in the tokens within each domain. To do so, I ran a Proxy A-Distance approximation [Ganin et al., 2016] by modifying some of the code found

Bokmål	Nynorsk
43,062	552

Table 1.1: Distribution of languages

	Train	Dev	Test	Total
Screen	11,439	1,429	1,429	14,297
Music	10,565	1,320	1,319	13,204
Misc	3,696	462	461	4,619
Literature	3,451	432	430	4,313
Products	2,778	347	345	3,470
Games	1,411	179	179	1,799
Restaurants	733	91	91	915
Stage	613	76	75	764
Sports	187	24	22	233

Table 1.2: Distribution of reviews across train, test and development sets.

here [Link Github?]. Proxy A-Distance (PAD) aims to approximate the similarity between domains, by running a learning algorithm looking at the divergence between a source and a target. To avoid memory issues, I took the 55.000 most frequent tokens from each domain, excluding Games, Restaurants, Stage and Sports. Figure 1.2 indicates that the distance between the domains are rather small. While this alone gives us very little additional information, there might be some merit to investigate whether there is some corellation between the PAD and the accuracy for the domains. However, since the accuracy on domains is largely dependent on the size of the domain, this might not tell us anything. [???]

		Avg. Rating				Avg. Tokens			
	Reviews	Train	Dev	Test	All	Train	Dev	Test	All
Screen	14,296	3.86	3.90	3.98	3.91	422.2	500.2	546.2	489.5
Music	13,203	4.14	4.28	4.20	4.20	325.9	406.5	385.7	372.7
Misc	4,618	4.41	4.37	4.43	4.40	516.3	540.7	538.4	531.8
Literature	4,312	4.38	4.45	4.50	4.44	445.5	584.4	579.9	533.9
Products	3,469	4.59	4.60	4.63	4.61	981.0	1082.5	1008	1023.8
Games	1,798	4.25	4.27	4.39	4.30	569.4	628.8	740.6	646.3
Restaurants	914	4.14	4.26	4.07	4.16	789.9	832.8	894.7	839.1
Stage	763	4.49	4.51	4.58	4.53	567.0	606.2	648.6	607.3
Sports	232	3.68	3.54	3.63	3.62	503.9	165.4	266.9	312.1

Table 1.3: The distribution of categories in the NoReC dataset

	Unique Tokens			
	Train	Dev	Test	All
Screen	209,978	55,354	57,741	235,715
Music	161,948	47,267	44,599	183,473
Misc	110,423	30,660	30,692	124,762
Literature	93,467	28,888	29,055	107,465
Products	92,910	27,682	25,339	106,781
Games	47,448	14,147	15,660	55,605
Restaurants	37,235	10,268	10,325	40,266
Stage	35,109	8,803	9,375	40,602
Sports	8,076	763	1,362	8,741

Table 1.4: Number of unique tokens in the domains and splits

	Screen	Music	Misc	Literature	Products	Games	Rest.	Stage	Sports
Screen	-								
Music	15.40	-							
Misc	18.82	18.69	-						
Literature	16.98	15.02	22.04	-					
Products	9.55	10.04	11.84	11.61	-				
Games	5.75	6.15	8.30	8.51	7.85	-			
Restaurants	7.49	8.47	11.38	11.72	10.45	8.74	-		
Stage	10.37	11.20	16.14	16.15	10.44	10.25	14.28	-	
Sports	2.51	3.04	4.25	4.69	4.35	7.90	8.38	8.75	-

Table 1.5: Percentage of tokens overlapping between domains

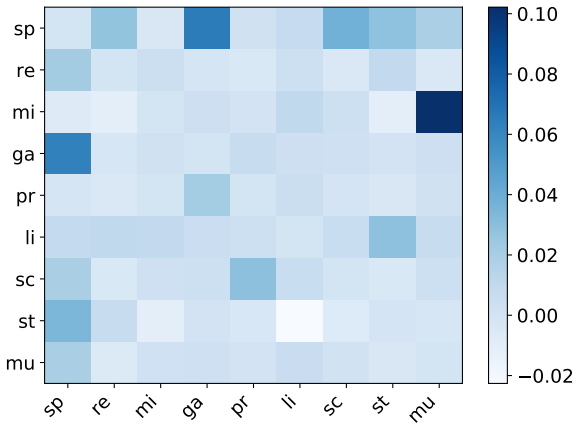

Figure 1.2: Heatmap showing Proxy-A Distance of domains in NoReC

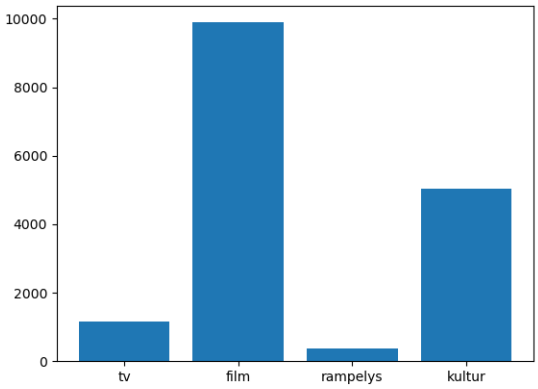

Figure 1.3: The different categories found under 'screen'

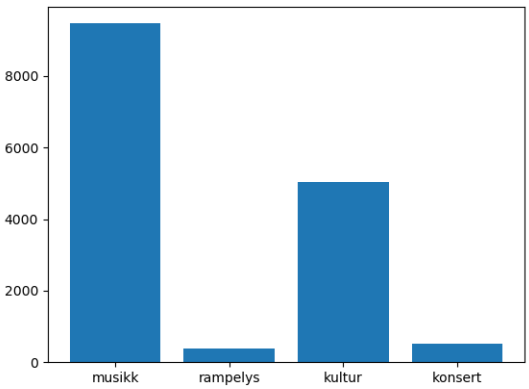

Figure 1.4: The different categories found under 'music'

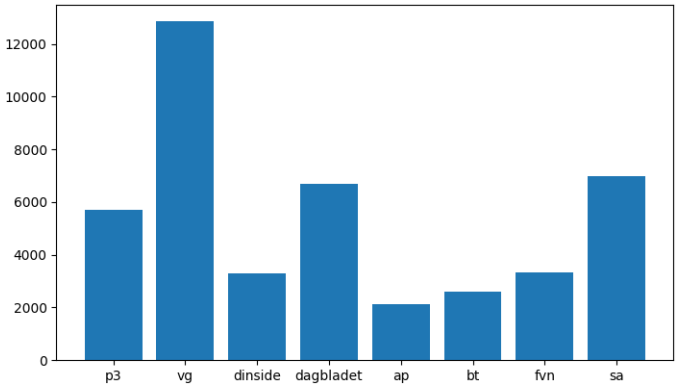

Figure 1.5: The different sources in the NoReC dataset

1.4 Domain Adaptation

Domain adaptation is the task of developing learning algorithms that take knowledge from labeled data in a source domain and adapt the knowledge to different domains. While domains can be anything from medical journals to Wikipedia, the most common domains for sentiment analysis consists of different kinds of reviews, opinionated sites like Twitter, and generally documents where feelings or opinions are expressed. Domain adaptation is especially interesting in NLP, as we often have a large amount of labeled data in a source domain (e.g IMDb Movie Review Dataset), and we want to apply the knowledge from an algorithm trained on this domain, to a domain where we have little to none labeled data. [Daumé III, 2007] This raises a challenge, however. If we train a model on a specific domain (e.g movie reviews), it transfers poorly to other domains like restaurant reviews. In the movie review domain, some of the words that carry negative weight are words like "2", "series" and "tv", which makes sense in that specific domain (people tend to disfavor movies based on tv series and sequels). However, for the restaurant domain, the opposite might be true. If a restaurant has been on tv, or is part of a series, it might be positive. This creates a challenge when we train a model on a specific domain and try to apply it to another. Domain adaptation is an instance of transfer learning, which I will go into next.

1.4.1 Transfer Learning

Transfer learning is a means to extract knowledge from a source setting and apply it to a different target setting. If you have a source domain D_S and learning task T_S, a target domain D_T and learning task T_T, transfer learning aims to improve the performance on D_T using knowledge from D_S and T_S [Pan and Yang, 2010]. According to Pan and Yang, there are three main research issues in transfer learning: 1) what to transfer, 2) how to transfer, and 3) when to transfer. What to transfer, is the task of finding

the information that is relevant as well as irrelevant to transfer between the domains. Secondly, we must develop an algorithm that can transfer the information in a satisfactory manner, which is how to transfer. When to transfer is the task of knowing when transfer learning is helpful and when it's disruptive. Using transfer learning on two completely separate domains may hurt the model's performance[Pan and Yang, 2010].
There are a variety of different types within transfer learning. A taxonomy that shows the variations can be seen in 1.6. In NLP, this can be especially

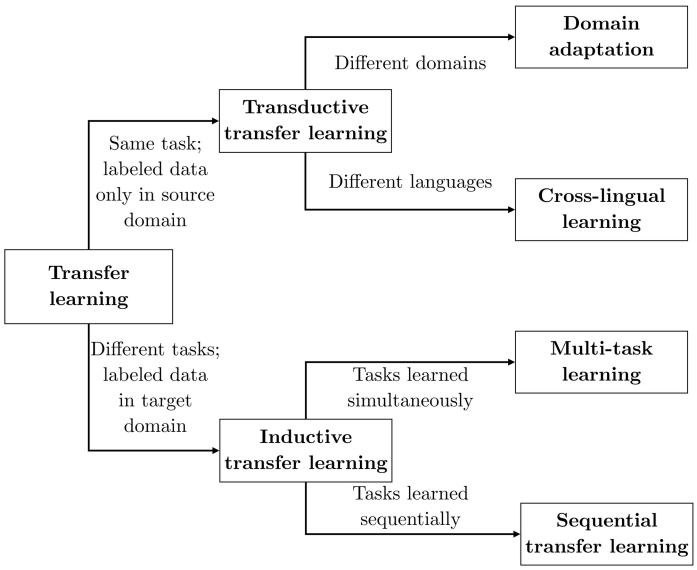

Figure 1.6: A taxonomy for transfer learning in NLP Ruder, 2019

useful because words often mean the same in a given context. There are, however, a few different types of transfer learning. One is when you have labeled data in the source domain and adapt the knowledge to different domains, also known as domain adaptation. A different, more common approach is training on a large amount of unlabeled data before adapting the representations using self-supervised learning.

1.4.2 Pre-Training

Pre-training in NLP is the task of modeling complex characteristics of word use and apply this model on several different tasks. The intuition behind pre-training is that if a model has general knowledge about a domain, it can more easily use existing knowledge to derive information from unseen domains. Pre-training is usually done on large amounts of data and has been used to produce word embeddings in recent years. The most commonly used word embeddings can be divided into static and contextualized embeddings. I will first do a short explanation of static embeddings, before discussing contextualized embeddings and finetuning in more detail.

1.4.3 Static Embeddings

Static word embeddings had a breakthrough when Google published their Word2Vec algorithm in 2013 [Mikolov et al., 2013]. Word2Vec was a breakthrough in NLP, as we now had access to over 1.4 million vectors trained on more than 100 billion words. Word2Vec could be implemented using two different approaches. The first is the continuous bag of words (CBOW), where we try to predict which word is most likely given its context. The second is skip-gram, where we try to predict the context from a word. About a year later, Stanford published their version of static word embeddings called GloVe [Pennington et al., 2014]. In short, what GloVe did differently was that it focused on co-occurrences of words, looking at the probabilities that two words appear together.

1.4.4 Contextualized Embeddings

Contextualized embeddings were the next step in pre-training language models. In 2018, Embeddings from Language Models (ELMo) was published [Peters et al., 2018]. Where we previously assigned a vector to each word, ELMo looks at the context the word appears in. If we take the word "fall", this could have multiple meanings. One being the verb "to fall", another being the time of year as in "autumn". With traditional embeddings, we would learn the vectors based on a dataset and assign only one vector to "fall". One of the revolutionary things that ELMo did is that each token is assigned a representation that is a function of the entire input sentence. In other words, the embedding assigned to "fall" is calculated from the sentence it appears in. The way ELMo does this is by using a bidirectional long short-term memory (BiLSTM) RNN to calculate the probability of both previous and future words in the sentence before returning the contextualized embedding.
Not long after the release of ELMO, Bidirectional Encoder Representations from Transformers (BERT) was published [Devlin et al., 2019]. Upon its release in 2018, it obtained state-of-the-art results on eleven NLP tasks in various fields, using Google's transformer architecture [Vaswani et al., 2017]. Whereas previous language representation models using transformers had been unidirectional [Radford et al., 2018], BERT uses attention mechanisms to learn the contextual relations between all words in a sentence. The way BERT does this is by using a "masked language model" (MLM) pre-training objective. First, the model replaces some of the words in the dataset with the [MASK] token, and then the model attempts to predict the actual value of the token, based on the context provided by the unmasked words in the sentence. Next, the model does "next sentence prediction" (NSP). By pairing 50% of the sentences in the dataset, BERT is tasked with predicting whether the next sentence in a document is the next sentence, with a 50% chance it will be. This has proven very useful for tasks like question answering, where models are required to produce fine-grained output at the token level. Upon the release of the paper, Google also released the models used in the paper, BERT_{BASE} and BERT_{LARGE}. These are both incredibly

large models with 110M and 340M parameters, respectively. Training a model of this size requires an enormous amount of computational power, energy, and time. By making both the code and pre-trained models from the paper publicly available, it became possible for small research groups with limited computational power and funding to fine-tune BERT and apply it as they saw fit.

1.4.5 Fine-tuning

While contextualized embeddings can bre trained from scratch, it's more common to use the published models because of the aforementioned time, power and energy it requires. Using pre-trained embeddings in this way, is actually an instance of transfer learning. When applying the pre-trained embeddings, you can also fine-tune them to a specific task. Fine-tuning is the task of training a pre-trained model on a new dataset, which makes it an instance of domain adaptation. When we fine-tune a model, we have a few different options. First of all, we want to make sure that the model remembers the important information it was originally trained on. One way to avoid this, is to freeze different combinations of layers as we train our model. Freezing the layers means that we only allow some of the parameters to change during fine-tuning, which in turn forces the model to remember some of the original information. One approach to this is gradual unfreezing, where all layers are frozen except for the last one, which is then trained for one epoch. Next we unfreeze the the second to last layer and train the model for one epoch on the last two layers. This is repeated until all layers are unfrozen and then the model is trained until convergence. This has successfully been applied in ULMFiT [Howard and Ruder, 2018]. Another concept in fine-tuning which can also be seen in the ULMFiT paper is discriminative fine-tuning. The intuition here is that you want to start out with a low learning rate for the last layer because this holds the most general information, then rapidly increase the learning rate, before gradually reducing it until convergance. This will adjust how much the different layers learn, as they all hold different information.

Chapter 2

Temp

2.1 Preprocessing of NoReC

The NoReC dataset consists of over 43.000 full-text reviews as raw text. It also contains a metadata-file in json-format. Each review's filename is six digits, corresponding to the review's metadata in the json-file. The original dataset has all reviews divided into training set, development set and test set. To more easily conduct domain adaptation experiments, it would be sensible to have the reviews sorted by domains while keeping the original split in the data. Further, as the reviews are in a raw text format, we need to tokenize them. To achieve this, I used NLTK's pre-trained PunktSentenceTokenizer for Norwegian, as a basis for NLTK's word_tokenize(). I iterated through all reviews in their respective split, found their rating and category in the metadata, and created CSV-files with "rating,text" as headers, dividing them into folders based on category. This way we get to keep the train, test and development split, while we're able to easily use the rating for each review as a label. Further, we would not want to read every review each time we ran an experiment for a given category, so I appended all reviews in each category to a pandas dataframe, and stored them as a pickle-file.

2.2 Initial Experiments with Bag of Words and TF-IDF

To create a baseline for domain adaptation on the NoReC dataset, I started conducting experiments with Scikit Learn's TfidfVectorizer and C-Support Vector Classification. By default, the TfidfVectorizer converts all characters to lowercase, which we might want to try without at a later point, ngram range to 1,1 and L2 normalization. For the C-Support Vector Classification, we want to change the kernel to linear, rather than the radial basis function. We then proceed to train on one domain before testing it on the other domains and repeat this for all nine domains.
The second experiment I wanted to conduct, was to train the model on all of the training data before testing on separate domains, and to trian the model on separate domains before testing it on all the test data as one test set. As we can see from table 2.1 we get fairly decent results from a baseline

Train/Test	Screen	Music	Misc	Literature	Products	Games	Rest.	Stage	Sports	All	Avg.
Screen	**57.45**	48.06	50.75	49.76	38.84	48.60	54.94	50.66	22.72	50.95	**45.54**
Music	51.50	**56.02**	51.84	46.74	42.60	47.48	46.15	49.33	27.27	51.29	44,73
Misc	44.22	49.65	54.01	52.09	49.56	44.13	27.47	**56.00**	22.72	47.85	43.23
Literature	40.65	46.47	50.75	**54.41**	39.42	43.01	39.56	48.00	22.72	44.86	41.32
Products	32.82	41.16	48.37	45.81	**50.43**	48.04	23.07	49.33	9.09	40.26	37.21
Games	36.31	43.36	**50.75**	44.65	49.85	48.04	23.07	48.00	9.09	42.15	38.13
Rest.	38.34	31.84	31.01	33.25	25.79	27.37	**45.05**	32.00	18.18	33.50	27.72
Stage	34.07	42.83	48.80	45.81	49.85	46.36	26.37	**53.33**	27.27	41.34	40.17
Sports	**20.22**	15.92	13.44	11.86	9.56	12.84	21.97	6.66	13.63	15.99	14.05
All	57.80	56.55	**58.35**	57.90	53.33	50.27	57.14	57.33	4.54	56.53	-
Avg.	37.26	39.91	**43.21**	41.24	38.18	39.72	32.82	42.49	19.26	37.12	-

Table 2.1: Bag of Words classifier with SVM and TF-IDF, with averaged out of domain predictions.

classifier, where six out of our nine domains has the highest accuracy when tested on their respective test set.
The table is sorted by the amount of data, with Screen beeing the highest and Sports the lowest, with the reported accuracy for training/testing on all domains as one dataset is reported last. As for the outliers in the table, Misc yields a better score when tested on Stage. This might be because Misc contains reviews belonging to one of the other categories, being the thrid largest dataset. This suspicion is further backed up by the fact that Games yields a higher test score when tested on Misc than on Games. This is something that should be looked further into. Sports is as of now preforming a lot worse than any other domain, most likely due to the low amounts of data we have for Sports. Another reason for the low accuracy on the Sports domain, is the average length of the reviews in the test data. As mentioned in subsection 1.3 the average length of the reviews in the Sports domain is almost half of the length in its training data (266.9 versus 503.9), and significantly shorter than the other domains. Further, if we look at the Music domain, we can see that the average length of reviews in the test data is only 385.7, which is not that much longer than the average sports review. However, this is closer to the average length of its training data, and Music is also the second largest domain. This suggests that Sports having a short average review length in its test data alone, is not enough to justify its low accuracy.

2.3 NorBERT

NorBERT [reference for NorBERT?] is a BERT language model [Devlin et al., 2019], trained for Norwegian. NorBERT features a custom 30.000 WordPiece vocabulary, and outperforms Google's multilingual BERT on most, if not all, Norwegian tasks.

2.3.1 NorBERT Baseline

Replicate initial experiments with NorBERT

2.3.2 Further experiments with NorBERT

BERT only accepts 512 tokens, which means that we rarely get the entire review. TODO:
Majority class classifier

Bibliography

[Brun et al., 2016] Brun, C., Perez, J., and Roux, C. (2016). Xrce at semeval-2016 task 5: Feedbacked ensemble modeling on syntactico-semantic knowledge for aspect based sentiment analysis. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 277–281.

[Daumé III, 2007] Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

[Ganin et al., 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks.

[Howard and Ruder, 2018] Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

[Maas et al., 2011] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space.

[Ni et al., 2019] Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 188–197.

[Pak and Paroubek, 2010] Pak, A. and Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

[Pang et al., 2002] Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proc. of NAACL*.

[Pontiki et al., 2016] Pontiki, M., Galanis, D., Papageorgiou, H., Androutsopoulos, I., Manandhar, S., AL-Smadi, M., Al-Ayyoub, M., Zhao, Y., Qin, B., De Clercq, O., Hoste, V., Apidianaki, M., Tannier, X., Loukachevitch, N., Kotelnikov, E., Bel, N., Jiménez-Zafra, S. M., and Eryiğit, G. (2016). SemEval-2016 task 5: Aspect based sentiment analysis. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 19–30, San Diego, California. Association for Computational Linguistics.

[Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training.

[Socher et al., 2013] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

[Taboada et al., 2011] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., and Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

[Velldal et al., 2018] Velldal, E., Øvrelid, L., Bergem, E. A., Stadsnes, C., Touileb, S., and Jørgensen, F. (2018). NoReC: The Norwegian Review Corpus. In *Proceedings of the 11th edition of the Language Resources and Evaluation Conference*, pages 4186–4191, Miyazaki, Japan.

[Zhang et al., 2011] Zhang, L., Ghosh, R., Dekhil, M., Hsu, M., and Liu, B. (2011). Combining lexicon-based and learning-based methods for twitter sentiment analysis. *HP Laboratories, Technical Report HPL-2011*, 89.

[Zhang et al., 2016] Zhang, X., Zhao, J., and LeCun, Y. (2016). Character-level convolutional networks for text classification.