UPTEC STS 18001
Examensarbete 30 hp
Januari 2018

Sentiment analysis of Swedish reviews and transfer learning using Convolutional Neural Networks

Johan Sundström

UPPSALA UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Sentiment analysis of Swedish reviews and transfer learning using Convolutional Neural Networks

Johan Sundström

Sentiment analysis is a field within machine learning that focus on determine the contextual polarity of subjective information. It is a technique that can be used to analyze the "voice of the customer" and has been applied with success for the English language for opinionated information such as customer reviews, political opinions and social media data. A major problem regarding machine learning models is that they are domain dependent and will therefore not perform well for other domains. Transfer learning or domain adaption is a research field that study a model's ability of transferring knowledge across domains. In the extreme case a model will train on data from one domain, the source domain, and try to make accurate predictions on data from another domain, the target domain. The deep machine learning model Convolutional Neural Network (CNN) has in recent years gained much attention due to its performance in computer vision both for in-domain classification and transfer learning. It has also performed well for natural language processing problems but has not been investigated to the same extent for transfer learning within this area. The purpose of this thesis has been to investigate how well suited the CNN is for cross-domain sentiment analysis of Swedish reviews. The research has been conducted by investigating how the model perform when trained with data from different domains with varying amount of source and target data. Additionally, the impact on the model's transferability when using different text representation has also been studied.

This study has shown that a CNN without pre-trained word embedding is not that well suited for transfer learning since it performs worse than a traditional logistic regression model. Substituting 20% of source training data with target data can in many of the test cases boost the performance with 7-8% both for the logistic regression and the CNN model. Using pre-trained word embedding produced by a word2vec model increases the CNN's transferability as well as the in-domain performance and outperform the logistic regression model and the CNN model without pre-trained word embedding in the majority of test cases.

Handledare: Nils Dahlbom
Ämnesgranskare: Dave Zachariah
Examinator: Elísabet Andrésdóttir
ISSN: 1650-8319, UPTEC STS 18001

Acknowledgment

This report is the result of my Master's thesis written during the fall of 2017 on the Master's program in Sociotechincal Systems Engineering. The research has been conducted at Findwise, a company within search driven technology, machine learning and data analysis.

I would like to thank Findwise for the kind reception and the opportunity to pursue my Master's thesis at Findwise Stockholm. A special thanks to my mentor, Nils Dahlbom, for supporting me throughout the duration of this thesis project. I would also like to thank my subject reader Mr. Dave Zachariah, Associate Senior Lecturer at the division of Systems and Control and coordinator of the Machine Learning Arena, for valuable consultations, continuous support and feedback during this project.

Populärvetenskaplig sammanfattning

Sentimentanalys är ett fält inom maskininlärning och språkteknologi (eng. Natural Language Processing (NLP)) där fokus ligger på att analysera text, tal eller annan typ av subjektiv information. Ett generellt mål inom sentimentanalys är att avgöra polariteten, attityden eller känslan i text eller annan information. Det kan exempelvis vara att avgöra huruvida innehållet har positiv, negativ eller neutral ton. Sentimentanalys tillämpas i stor utsträckning inom det som kallas ”voice of the customer” där målet är att få en inblick i kunders förväntningar, preferenser och aversioner gällande produkter och tjänster. Detta kan på ett automatiskt vis göras genom att med hjälp av maskininlärning analysera kundundersökningar, kundrecenssioner eller sociala mediaflöden. Resultatet är något som sedan skulle kunna användas som beslutstöd.

Användning av sentimentanalys för att bestämma polariteten hos film-, produkt- eller andra typer av recensioner har tillämpats i stor utsträckning inom NLP-området. En svårighet vad gäller sentimentanalys och andra områden där maskininlärning ska tillämpas är att välja ut vilka kännetecken eller ”features”, som ska användas till att träna den prediktiva modellen. Features inom sentimentanalys är exempelvis ord, grupper av ord eller bokstäver. Det finns en rad maskininlärningsmodeller att välja bland för utveckling av prediktiva modeller. På senare tid har ett fält inom maskininlärning som kallas djupinlärning (eng. deep learning) fått mycket uppmärksamhet. Djupinlärningsmodeller har förmågan att lära sig underliggande strukturer i data på ett sätt som andra maskininlärningsmodeller inte kan där kännetecken eller features inte manuellt behöver bestämmas av en människa utan modellen kan själv lära sig det. Trots att sentimentanalys är utbrett inom forskningsvärlden är majoriteten av forskningen emellertid huvudsakligen fokuserad på det engelska språket och det är svårt att hitta liknande forsking för andra språk som exempelvis svenska.

Vid utveckling av maskininlärningsmodeller är det vanligt att träna en modell för en specifik kontext. Detta görs lättast genom att träna en modell på liknande typ av data som den sedan testas och tillämpas på. Ett uppenbart problem kring detta är att modellen blir domänberoende och för varje nytt område måste en ny maskininlärningsmodell utvecklas, tränas och testas vilket är både ett tidskrävande och upprepande arbete. En prediktiv modell presterar bäst om den får träna på mycket data inom samma domän. Detta är dessvärre svårt i många sammanhang då datatillgången ofta kan vara begränsad eller otillgänglig. Överföringslärning eller domänanpassning (eng. transfer learning, domain adaption) är ett aktivt forskningsområde som fokuserar på att tackla problemen gällande domänberoende och databegränsning. Huvudidén är att träna en modell med data från en distribution, källdomänen, och samtidigt få den att kunna prediktera data från en annan distribution, måldomänen. Exempelvis skulle data från en domän kunna användas för att träna en modell som kan prediktera data från en närliggande eller relaterad domän.

Djupinlärningsmodeller såsom CNN:er är ursprungligen utvecklade för bildigenkänning och har på ett framgångsrikt vis använts inom överföringslärning inom just bildigenkänning. På senare år har dessa nätverk även visat sig fungera väl för text- och sentimentanalys. Syftet med denna studie har således varit att undersöka huruvida ett CNN är lämpad att användas för överföringslärning inom sentimentanalys för svenska recensioner. Detta har undersökts genom att studera hur en modell presterar då den tränas med varierande mängd data från

käll- och måldomänen. Genom att variera mängden träningsdata från olika domäner går det att undersöka hur domänberoende modellen är för olika mängd och typ av data. Detta ger även en indikation om hur relaterade käll- och måldomänen egentligen är. För att få ett perspektiv på hur djupinlärningsmodellen presterar har den jämförts med en traditionell maskininlärningsmodell, logistisk regression, som anses vara väl lämpad för sentimentanalys.

Maskininlärningsmodeller har inte möjlighet att förstå eller hantera ren textdata utan text måste representeras på ett numeriskt vis. Det finns en rad olika sätt att representera text på numeriskt vis och vilket sätt som används kan ha stor inverkan på modellens prestation gällande sentimentklassificering samt överföringsförmågan. Olika textrepresentationer har därför undersökts i denna studie där bland annat en oövervakad (eng. unsupervised) modell kallad word2vec använts till att förträna ordvektorer.

Undersökningen har visat att CNN:en utan förtränade ordvektorer som använts i denna studie inte är vidare lämpad för överföringslärning inom sentimentanalys av svenska recensioner i extremfallen där modellen tränas med data från en domän och sedan försöker klassificera data från en annan domän. Detta då den traditionella modellen, logistisk regression, presterar lika eller bättre i majoriteten av testfallen. Däremot är det möjligt att öka CNN:ens prestation inom domänen genom att addera data från en annan domän i träningsprocessen. I testfallen där all data används från både käll- och måldomänen presterar CNN:en bättre än den logistisk regressionsmodellen i de flesta testfallen. Mängden måldomändata som används till modellträning har i många fall haft en avgörande inverkan på resultatet där denna studie visar att överföringsförmågan hos en modell ökar markant genom att ersätta 20% av källdomändata med måldomändata. Detta gäller både CNN:en och den logistiska regressionsmodellen. Denna studie har också visat att textrepresentationen har en avgörande inverkan på såväl överföringslärning som klassificering inom domänen då CNN-modellen med förtränade word2vec-ordvektorer utklassar de andra modellerna i majoriteten av fallen både gällande klassificering inom domänen men framför allt gällande överföringslärningen.

Contents

1 Introduction

We use people's opinion to make decision in our day to day life whether we are planning on buying something, vote in elections, decide what movie to watch, hotel to stay at or restaurant to visit. Similar it is crucial for businesses to get insight in what consumer think about their products or services to stay competitive. The amount of data generated on the Internet each day is huge. A lot of the data is produced by people sharing or expressing their opinion on various forums such as social media or blogs sites. Products, hotels, restaurants, movies and similar are a common subjects people frequently express their opinions about. Sentiment analysis or opinion mining is a subfield to machine learning that focus on determine the polarity or emotion of events, e.g. determine if a text has a positive, negative or neutral tone. Machine learning models often require a lot of data in order to build accurate predictive models. Therefore, with the increasing amount of freely available data generated on the Internet every day, sentiment analysis or opinion mining has gained much attention [1]. It can effectively be applied by businesses to analyse the so called "voice of the customer" to get an overall understanding of customer opinions, something that subsequently can help in decision making.

1.1 Problem

Using sentiment analysis to determine the contextual polarity of movie and product reviews or social media posts like Twitter tweets and Facebook updates is widely applied in the field of Natural Language Processing (NLP). This has lately been done with excellent result using deep learning techniques [2], [3], [4]. A crucial factor that affect traditional machine learning model performance is the feature selection process. Features in sentiment analysis can be letters, words or combination of words. Determine what features are relevant as well as irrelevant for a particular problem is a difficult and time-consuming task [5]. Deep learning models have the ability to extract higher level features all by themselves and hence do not actually need pre-defined features defined by humans [4], [6]. Even though sentiment analysis has been extensively investigated, the majority of previous research are focusing on the English language and it is hence difficult to find sentiment analysis research for other languages such as Swedish.

The traditional way of building machine learning models is to focus on a specific context [7]. What this means is that normally a model is trained and evaluated on the same sort of data, e.g. train a model based on movie reviews, test and evaluate it on other movie reviews in order to achieve the best possible predictive model for unseen movie reviews. Even though these models perform really well they have a serious drawback. That is the fact that they are domain dependent and hence will not perform well on data from other contexts or domains [1]. This is not really a problem if only a specific domain is evaluated and there exist plenty of readily available labeled data. However, this is usually not what reality looks like. Often data comes in different forms, from different domains, is limited and unlabeled [8], [9]. To build machine learning or predictive models for every new domain might include feature selection, development, training, tuning, maintaining and sometimes manually labeling data. This is a process that is difficult, repetitive and time consuming. Transfer learning and domain adaption is a highly active research field that aims to tackle the problem of domain dependence and data limitation where the distribution of source and target data differs [7]. The main idea is to utilize data from different but related domains where data is readily available to build predictive models for the target domain.

1.2 Purpose

The goal of this research is to investigate how transferable the deep machine learning model, Convolutional Neural Network (CNN), is for sentiment analysis of Swedish reviews. There are obvious needs for more generic machine learning models that can perform well on various type of domains as well as in other languages than English. A way to address the issue of domain dependent models can be to utilize data that are readily accessible from different but related domains when building and training models. The related data in combination with limited desired domain data can be used together to train, pre-train and fine-tune machine learning models such as neural networks and deep neural networks. Deep learning models such as CNNs has successfully been applied for transfer learning in the field of computer vision but has not been investigated to the same extent within the field of NLP [10]. Therefore, the aim of this research is to study how well suited CNNs are for transfer learning or domain adaption for sentiment analysis of Swedish reviews. While the choice of machine learning model most likely have a great impact on the end result the data representation, pre-processing and pre-training phase must also be considered. This leads to the following research questions.

- How well suited are Convolutional Neural Networks for cross-domain sentiment classification of Swedish reviews?
- What impact does the data representation have on the transferability across domains?

1.3 Disposition

This thesis report starts with a background section covering the fundamentals of machine learning, a thorough description of supervised learning with artificial neural networks followed by an overview of deep learning and a more in-depth explanation of what CNNs are and how they work. This section also covers sentiment analysis, how to work with textual data in machine learning as well as how to evaluate machine learning models. Lastly, a section covering problem background along with related studies is presented. The experiment section covers the type of experiments that will be conducted in this thesis to answer the research questions. In the methodology section every necessary step taken to fulfill this research are covered. This include, amongst other, data gathering, pre-processing, machine learning modeling and evaluation. Next, is the data section where the data used in this thesis is presented along with information such as amount of data and data distribution of categories and ratings.

The result section provides visual and tabular results of the conducted experiments. Thereafter, the findings in the result section are highlighted where the CNN model is compared with the baseline model and the impact of the text representation are discussed. Lastly, the gained insights are summarized and suggestions of future work is presented.

2 Background

In this section, the field of machine learning is briefly described, followed by a more in-depth explanation of supervised learning and neural networks. Next, Convolutional Neural

Networks (CNN), some difficulties in machine learning, what sentiment analysis is, how to work with text data and how to evaluate a machine learning model performance are covered. Lastly, the problem background along with related studies in the area is described.

2.1 Machine learning

Machine learning is as described in [11] a field within computer science that allows for computers to learn from previous experience without being explicitly programmed. A common goal in machine learning is to build computational models that, as accurately as possible, can predict the output of some input data. These models are constructed by combining knowledge from the fields of computer science, optimization, statistics and probability. Machine learning has successfully been applied in a variety of applications such as natural language processing, text or document classification, speech and image recognition as well as recommendation systems to name a few.

2.1.1 Different types of learning

Training machine learning models can be done in various ways, either through supervised, unsupervised, reinforcement learning or a combination of these. Supervised learning is essentially when a learner is trained on data points associated with labels to then predict or classify the labels of unseen data points. In unsupervised learning the training data do not have any labels. The learner train on that data and try to make predictions of unseen data by for instance clustering similar data points together. A common problem in unsupervised learning is that it is difficult to evaluate how well the model perform since the ground truth is unknown. Reinforcement learning is quite different and instead of training on a lot of data an agent interact with an environment with a set of rules specified by the programmer. The agent receive rewards based on its actions and the rules of the environment. Reinforcement learning is a type of trial-and-error process where the agent constantly learn from feedback and previous experience. It is in a sense closely related to how humans learn new things such as how we learn to ride a bike [11]. In this thesis both supervised and unsupervised learning will be used to train models where the goal is to classify sentiment in Swedish reviews across domains.

2.1.2 Artificial Neuron

Inspired by the biological brain the Artificial Neural Network (ANN) is widely used in the field of machine learning to solve problems such as predictive modeling, classification and function approximation. An ANN is a network of nodes connected to each other. The nodes of the network, also called Artificial Neurons (AN) or perceptrons, are the basic building blocks of ANNs. The neurons in a network are connected by weights and the structure of a simple AN can be seen in Figure 1. An AN essentially takes an input, x, associated with a weight, w, plus a bias term, b, calculate the *net* input which usually is a weighted sum, (Equation 2.1), apply an activation function that decides whether the neuron should "fire" or not. The role of the weights associated with the input is to either strengthen or weaken the input signal. The operations of an AN can be seen as a nonlinear mapping from R^N to usually either an output in the range of 0 and 1 or -1 and 1 where N defines the number of inputs. The choice of activation function determines to what interval the input signal will be mapped and if the neuron should fire. The bias term allows for shifting the activation function to either right or left along the x-axis and is also associated with a weight and an additional *biasvalue*, see Equation 2.2. [12]

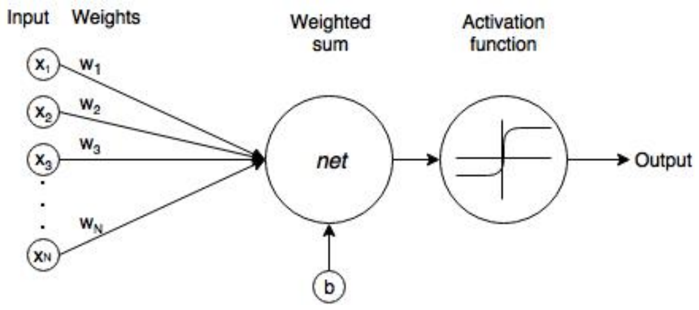


Figure 1: *Artificial Neuron*

$$net = \sum_{i=1}^{N} (x_i w_i) + b, \text{ where } b \text{ is as in Equation 2.2} \tag{2.1}$$

$$b = w_0 * biasvalue, \text{ where } biasvalue \text{ usually is equal to 1} \tag{2.2}$$

2.1.3 Activation function

There are various types of activation functions or transfer functions as they are sometimes called. Step function, sigmoid function, tanh function or rectifier (ReLU) function are all common activation functions [12]. A plot of the sigmoid activation function where λ = 1 can be seen in Figure 2 and its corresponding mathematical expression in Equation 2.3. The activation function maps the net input value to a value in the range of [0, 1]. The λ parameter controls the steepness of the function. In a binary classification problem the output value can be seen as the probability that the input belongs to a certain class. Say that the activation function outputs the value 0.8, this indicate that there is 80% chance that the input belongs to class 1 and 20% chance that it belongs to class 2. A similar activation function used for multiclassification task is the softmax function which will output a probability for every class in the range [0,1] where the sum of the probabilities are equal to 1.

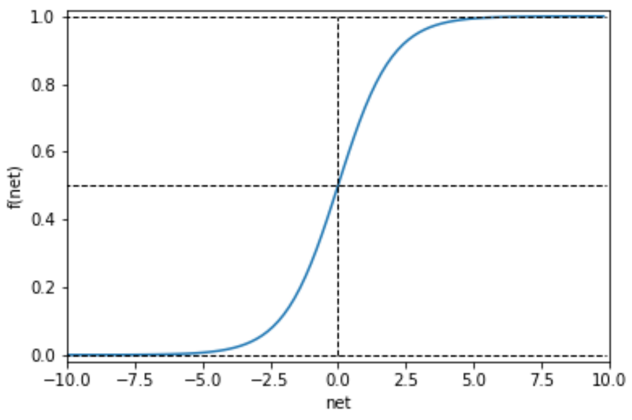


Figure 2: *Sigmoid activation function*

$$f(net) = \frac{1}{1+e^{-\lambda net}} \tag{2.3}$$

The ReLU activation function, Figure 3 and mathematically expressed in Equation 2.4, maps the net input to a value in the interval [0, ∞]. This means that all negative values are set to zero. By setting negative values to zero and hence make the neuron inactive will ease the computational load in a network with many neurons since fewer neurons are activated and hence less computations are required. Also computations are linear for ReLU compared to the exponential sigmoid function. It has also been shown in [13] that ReLU activation function is well suited for not only image recognition but also for sentiment analysis with sparse text data.

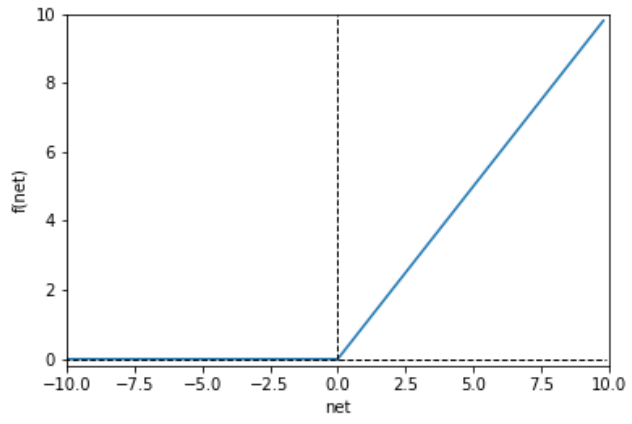


Figure 3: *ReLU activation function*

$$f(net) = max(0, net) \tag{2.4}$$

2.1.4 Neural Networks

As explained in [12] regular feed forward neural network has at least three layers; an input, hidden and output layer where every layer consist of a few or several neurons. An example of a feed forward neural network with two hidden layers, also called a Multi-Layer Perceptron (MLP), can be seen Figure 4. An MLP receives an input signal and passes that signal through the network, layer by layer to finally reach the output layer. The output of a layer is the input of the subsequent layer. In supervised learning, as briefly described in Section 2.1.1 the goal is to train a machine learning model so that it can classify or predict unseen and unlabeled data by letting the model train on known labeled data. In this thesis the polarity of reviews are to be determined hence the model should be able to predict whether a review has a positive or negative tone.

Neural networks can be used for supervised learning where the model, during training, is given both an input as well as a label or target associated with that input. The network then tries to minimize the error between the networks predicted output value and the target value by adjusting its weights. The weight adjustments of the neurons are made by utilizing optimization and a method called backpropagation, further described in Section 2.1.6. Once a network is trained it can be saved and used for later hence there is no need to retrain the network with the same data every time it shall be used. What saving a trained network actually means is that the architecture as well as the trained weights are saved then either the entire network or just desired parts, i.e. layers or specific neurons of the network can be used for other occasions.

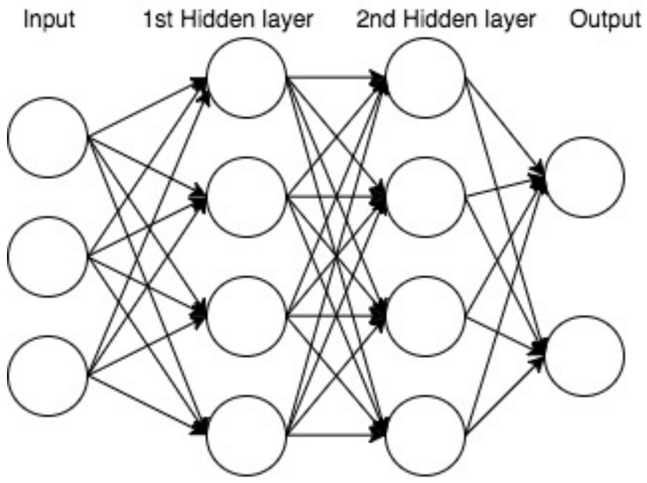


Figure 4: *Artificial Feed Forward Neural Network with two hidden layers*

2.1.5 Cost function

Measuring how well a machine learning or statistical model predict the outcome of an event can be done using a cost or objective function, sometimes also called a loss function [11]. Essentially what the cost function is measuring is the error rate between the predicted value and the correct target value. The goal of the machine learning model is to obtain the smallest possible error, i.e. to minimize the error and hence the cost function. There are several types of cost functions. A common function is the Mean-Squared Error (MSE) that is used to calculate the average squared difference between the predicted values, p, and the target values, t, see Equation 2.5. Cross Entropy (CE) (Equation 2.6), or binary cross-entropy in binary classification problems, is another common objective function that has proven to converge faster as well as yield better outcome in classification error rates compared to MSE for ANNs [14]. In [15] the author demonstrate the advantage of CE over MSE when training an ANN and demonstrates that for CE the closeness of the prediction is considered while MSE tend to give much emphasis to incorrect outputs.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(p_i - t_i)^2 \tag{2.5}$$

$$CE = \frac{1}{N}\sum_{i=1}^{N} t_i log(p_i) \tag{2.6}$$

Where p is the predicted value, t is the correct target value and N number of samples.

2.1.6 Gradient descent and back propagation

Running through an epoch include both a forward pass and a backward pass over the network [16]. The forward pass include passing the data through the network, calculates the net input or the weighted sum, apply an activation function and predict the outcome and calculate the error rate, i.e. the difference between the predicted value and the target value. An explanation of the forward pass for an AN is described in Section 2.1.2.

The goal of the backward pass is to obtain better predictions, i.e. minimize the error, for the subsequent forward pass by updating the weight and biases of the network. This is an optimization process used for training neural networks and it is called back propagation.

The process of minimizing the error or cost is built on using derivatives. Derivatives are useful when it comes to finding the local minimum of a function. Gradient descent is a popular optimization algorithm that uses derivatives to step wise follow the direction of the negative gradient of the cost function with the ultimate goal of finding a minimum. Calculating the derivatives of the cost function, E, in a neural network can be done using the chain rule. This is necessary since the cost function is a function of the activation function, $y = f(net)$. The activation function is a function of the weighted sum or net input, net, which is a function of weights, w. Hence to calculate the gradient of the cost function with respect to a certain weight i for a single neuron can be done using Equation 2.7 [12].

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial net}\frac{\partial net}{\partial w_i} \tag{2.7}$$

Assuming the MSE cost function (Equation 2.5), sigmoid activation function (Equation 2.3) and net input as the weighted sum as in Equation 2.1, then the separate derivatives of Equation 2.7 can be derived as follow.

$$\begin{aligned}\frac{\partial E}{\partial y} &= 2(\frac{1}{2})(p-t)1 \\ &= (p-t)\end{aligned} \tag{2.8}$$

$$\begin{aligned}\frac{\partial y}{\partial net} &= f'(net) \\ &= y(1-y)\end{aligned} \tag{2.9}$$

$$\frac{\partial net}{\partial w_i} = x_i \tag{2.10}$$

Then Equation 2.7 can be expressed using Equation 2.8, 2.9, 2.10 as.

$$\frac{\partial E}{\partial w_i} = (p-t)(f'(net))x_i \tag{2.11}$$

To decrease the error, the weight, w_i, is updated to w_i' using the update rule in Equation 2.12.

$$w_i' = w_i + \Delta w \tag{2.12}$$

Where Δw is.

$$\Delta w = -\eta \frac{\partial E}{\partial w_i} \tag{2.13}$$

Where η is the learning rate. Equation 2.13 can hence be written as.

$$\Delta w = \eta \delta x_i \tag{2.14}$$

Where δ is.

$$\begin{aligned}\delta &= (p-t)(f'(net)) \\ &= (p-t)(y)(1-y)\end{aligned} \tag{2.15}$$

For large neural networks containing several neurons and hidden layers every weight w is updated according to the same principle as described in this section.

2.2 Deep learning

Deep neural networks has recently gained much attention due to its state of the art performance in machine learning for task such as image, audio and video recognition as well as for tasks related to text and speech [17]. Choosing feature or feature extraction is a vital part when working with traditional machine learning models. In NLP tasks this can for instance be to choose what words to keep, how many words or decide whether to use single words or combination of words. In deep learning, feature extraction is built-in and the model can learn to extract features on its on with different levels of abstraction [6]. There are different types of deep neural networks such as CNNs and Recurrent Neural Networks (RNN). CNNs has shown particularly impressive performance for image recognition while RNNs are more suited for sequential data like speech and text [17]. However, [2] has proven that CNNs also achieves state of the art performance in NLP problems such as text and sentiment classification. CNNs, what they are and how they work are further explained in subsequent sections.

2.2.1 Convolutional Neural Networks

[18] explains that CNNs are similar to the standard MLP explained in Section 2.1.4. Both networks contain neurons with weights and biases that are trainable. Weights are updated with gradient descent and backpropagation and they both use non-linear activation function.

Since CNNs were initially developed for image classification and computer vision the intuition of CNNs are somewhat easier to grasp from a computer vision point of view. Therefor in the following sections CNNs for NLP will be described with illustrations both from the NLP as well as the computer vision perspective.

The main components in a CNN are the convolutional layer, pooling layer and fully connected layer. Other important components are activation function, optimization algorithm, dropout and a range of other hyperparameters that can be tuned. The fully connected layer can be seen as a standard feed forward neural network described in Section 2.1.4, where the final classification takes place. An example of a CNN architecture for sentiment analysis can be seen in Figure 5.

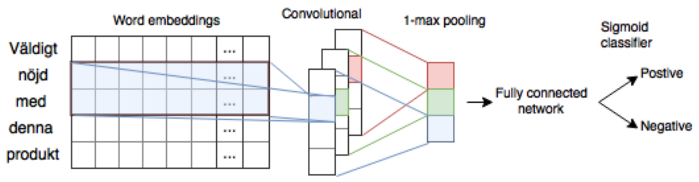


Figure 5: *Example CNN architecture for sentiment analysis*

Convolutional layer

The input to a CNN is a numerical matrix that represent an image of pixels, a sentence of words or characters. The primary task of the convolutional layer is to extract features from the input matrix. By applying the convolutional operation spatial relations between the pixels or words in the input matrix can be preserved and a new matrix with convolved features can be obtained. The convolutional step is essentially a filter with weights that slides across the input matrix and performs a dot product between the weights in the filter and the input matrix [19]. An illustration of the convolutional step can be seen in Figure 6. The filter slides from left to right starting in the top left corner and finishing in the bottom right corner jumping down one row at a time. In the illustration in Figure 6 the mapping from the input matrix through the filter to the convolved feature can be seen. After every training session the weights in the filter are updated using backpropagation [20].

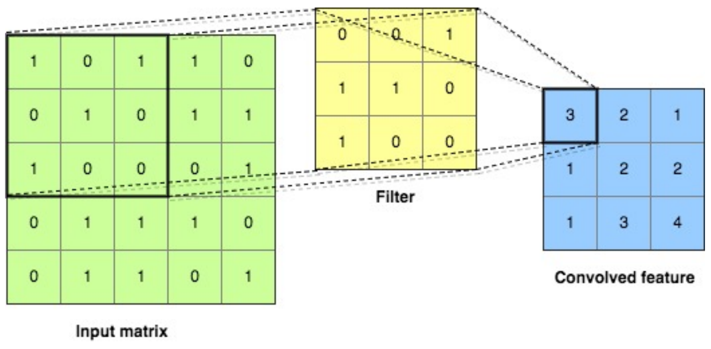


Figure 6: *Convolutional operation*

When dealing with images the input matrix is a set of pixels that represents the image. For text applications each row in the input matrix can be seen as a word. However, the rows can also represent one or more characters instead of entire words though it is more common to use words over characters. Since each row corresponds to a word, every word is represented by a vector of numbers. These vectors are usually either one-hot vectors or word embedding, concepts that are further explained in Section 2.5.

For images the filters are often quadratic and slides over different parts of the image region, see Figure 6. When dealing with text it is more common to slide over full rows, i.e. words, using a filter width equal to the width of the matrix. However filters with different heights can be used. The height of the filter indicate how many words the window should capture or slide over and is sometimes called kernel, window or filter size. Applying the filter to the corners or edges where no neighboring pixel values exists can be solved with something called zero padding. What this means is that all the values that are not in the matrix but captured by the filter is set to zero. This allow for applying the filter to every element in the input matrix. The movement of the filter can be adjusted with a stride parameter. In the sentence-word analogy a stride value of one implies that the filter moves one word at a time capturing every word while a stride parameter of two capture every other word in the sentence. Different number of filters with different kernel sizes can be used in the same model. [20]

Pooling layer
In [21] the author explain that after the convolved feature matrices has been obtained typically a pooling layer is applied. One major purpose of the pooling operation, also called sub-sampling, is to reduce the size of the convolved feature matrix while keeping the most implicit information. This help to cut down the computation time and control overfitting since the number of parameters is reduced [20]. The pooling layer extract specific and hopefully important features from the input. As an example, if a sentence contains something like "horrible movie", this region may yield a higher value for some filter compared to surrounding regions and hence be detected by the pooling layer. There are different types of pooling such as maximum, average or summation pooling. The most commonly used in previous research for sentiment analysis is maximum pooling [2], [6], [19]. The way max pooling work is basically that it extracts the largest value in the pooling window. When dealing with text data the pooling window often spans over the entire filtered matrix extracting the global maximum and hence a single number (although smaller windows can also be applied). Figure 7 illustrate both global max pooling as well as max pooling with a window size of 2x2 and stride value 2.

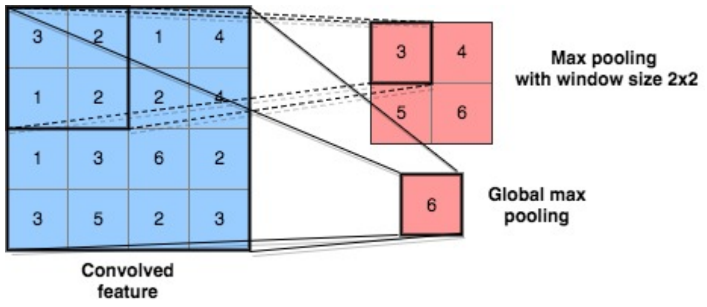


Figure 7: *Pooling operation*

Additional components
Convolutional layer and pooling layer may be the major building blocks of CNNs but there are some additional features or layers that can be of importance as well. Dropout is one such layer whose purpose is to prevent overfitting, it is further described in Section 2.3. Another technique similar to dropout is to add noise to the data [22]. In CNN architectures this can be applied by adding a noise layer just after the input layer that will make small changes or corrupt the input data slightly. One type of noise is the Gaussian noise which is a statistical noise with normal distribution.

Optimization
There are several types of optimization algorithms. Gradient descent is one such optimization algorithm briefly explained in Section 2.1.6. Other optimization algorithms are Adaptive Gradient Algorithms (AdaGrad) and Root Mean Square Propagation (RMSProp) which are both extensions of gradient descent. A further extension of gradient descent as described in [23] is an algorithm called Adam which it is a combination of AdaGrad and RMSProp. It has been shown that Adam is well suited for problems with large datasets and/or many parameters. It is also an efficient algorithm for deep learning that works well for computer vision as well as NLP problems.

2.3 Difficulties in machine learning

Although statistical modeling and machine learning can be useful in many situations such as function approximation and predictive learning there are also some common difficulties related to these techniques. Overfitting and underfitting are examples of two such difficulties. Overfitting in machine learning as described in [24] refers to a model that models the training data too well. This means that the model learns the characteristics of the training data in such extent that it has trouble to generalize and hence model new and unseen data. As for overfitted models underfitted models also has trouble generalizing new unseen data, however underfitted model cannot even model the training data.

For a neural network that tries to approximate a function the main causes for overfitting is letting the model train for too long with too many nodes, i.e. parameters describing the desired function. Deep neural network are powerful machine learning models but due to the large number of nodes they are prone to overfitting [22]. There are however techniques to address the problem of overfitting in deep neural networks. Dropout is a technique applied during training where randomly selected nodes are dropped along with their connections [22]. Other approaches is to use regularization, add noise to the input data and utilize a validation dataset to determine when to stop training [25]. Adding noise or small changes to the training data will challenge the neural networks robustness to handle a dataset that is slightly modified.

2.4 Sentiment analysis

Sentiment analysis is as described in [26] a subfield to machine learning and natural language processing (NLP) that focus on analyzing text, speech or other type of subjective information. A general goal within sentiment analysis is to determine the polarity or emotion of events, i.e. determine if a text has a positive, negative or neutral tone. More sophisticated emotions like anger, happiness, irony or sadness can also be investigated with the same technique.

Sentiment analysis is sometimes called opinion mining and is widely adopted in the field of NLP and has been applied to many different areas where people's opinions can be found. One such area is social media on the web. The increasing amount of freely available opinionated data generated on the Internet and especially on social media and other social networks today has made it easier to do research on large scale sentiment analysis. A variety of fields have been investigated using sentiment analysis or opinion mining such as movie, product, restaurant or hotel reviews, political opinions and recommendation systems [8]. Sentiment analysis or opinion mining can be used for decision making for both businesses as well as consumers. It is crucial for businesses to know what consumers thinks about their products or services. Simultaneously consumers want to know what other people think about a product before they buy it themselves. Taking advantage of the vast amount of data on the Internet using sentiment analysis can increase the understanding of ”the voice of the customer” and hence help in the decision making process. [26]

Recently sentiment analysis has been studied using deep learning techniques. State of the art performance for NLP tasks such as binary as well as multi sentiment classification and opinion detection have been obtained using CNNs [2]. In [2] the author is using one convolutional layer with kernel size of 3, 4, 5 and 100 filters. Other settings used are dropout with a dropout rate of 0.5, ReLU activation function and global max pooling. Similarly in [19] where the authors predicts the sentiment of Twitter tweets they use a single convolutional layer, ReLU activation function and max pooling. They do however not specify all hyperparameters such as number of filters, kernel size and dropout rate etcetera. Other research have used two convolutional layers [27].

2.5 Working with text data

Machine learning models cannot handle raw text as input therefore the text must be converted or represented in a numerical way. There are a few common approaches to represent text data when dealing with sentiment analysis and document or sentence classification problems. A few of these are described in the following sections.

2.5.1 Bag of Words

One common and simple approach is called Bag of Words (BoW) where the frequency of words is used while the order of words and grammar is not considered [28]. It is easiest illustrated with an example. Consider the following two sentences:

```
1. Justin likes to drink coffee with milk and sugar.
2. Katy likes sugar, she also likes to drink coffee with milk.
```

The vocabulary based on these sentences are:

```
['Justin', 'likes', 'to', 'drink', 'coffee', 'with', 'milk',
'and', 'sugar', 'Katy', 'she', 'also']
```

Which are the unique words from the sentences grouped together. The two sentences can then be numerically represented as follows:

```
1. [1,1,1,1,1,1,1,1,1,0,0,0]
2. [0,2,1,1,1,1,1,0,1,1,1,1]
```

Where the numbers indicates how many times each word appears in that sentence or document. Here only single words or so called 1-grams or unigrams are used in the BoW model. Another possible representation to capture a broader context would be to used n-grams. An example of a 2-grams or bigram vocabulary based on the two sentences above is as follows:

```
['Justin likes', 'likes to', 'to drink', 'drink coffee', 'coffee with',
'with milk', 'milk and', 'and sugar', 'Katy likes', 'likes sugar',
'sugar she', 'she also', 'also likes']
```

Instead of using every word in a corpus it is common to reduce the vocabulary size. Choosing the number of words and which words to use can be done in different ways. Two common ways to do this is to either choose the most frequent words in the corpus or choose the words based on something called ”Term frequency-inverse document frequency” (Tf-idf). As explained in [29] calculating Tf-idf is a two step procedure. First Tf, which essentially is the number of times a word appears in a document normalized by the total number of words in that document, is calculated. Idf which indicate the importance of a word is calculated by taking the logarithm of number of documents divided by the number of documents that has the word in it. This can help filtering out stop words like ”the”, ”a”, ”and” etcetera which are words that probably occur frequently in a corpus but are not that important for the context.

2.5.2 Ont-hot encoding

One-hot encoding is another approach for numerically representing words in text data. This approach is also easiest illustrated with a simple example. Consider a vocabulary of three words in a specific order, ”coffee”, ”milk” and ”tea”. Considering the order of the vocabulary the individual words can be numerically represented using one-hot encoded vectors. The first word, i.e. ”coffee” is represented as [1,0,0] where the 1 at position one indicate that ”coffee” appears at position one in the vocabulary. Similarly ”milk” is represented as [0,1,0] and ”tea” as [0,0,1] following the same principles [30].

2.5.3 Word embedding

Other more sophisticated word representations can be obtained by using models such as Tomas Mikolov’s word2vec or Facebook’s fastText. They are so called word embedding and constructed by training large corpus of text data in an unsupervised fashioned. The aim of these models are to obtain numerical vectors representing each word in a vocabulary where words that are similar should be close to each other in the vector space. These models are probabilistic and can achieve tasks like $vector(king) - vector(man) + vector(woman) \approx vector(queen)$ [31]. A graphical example of this vector equation in 2D can be seen in Figure 8.

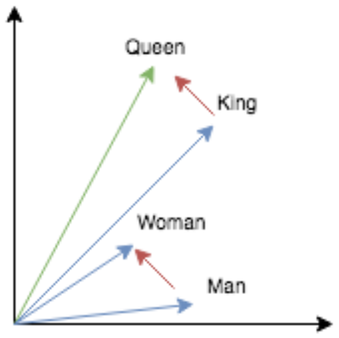


Figure 8: *A classic word2vec example*

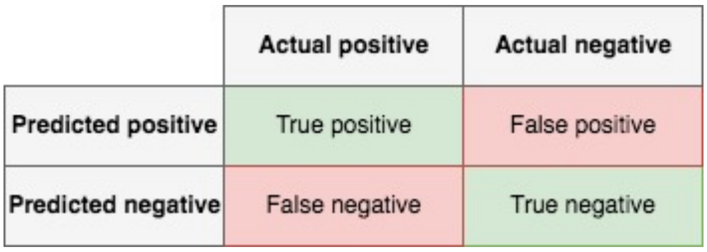
The word2vec model is essentially a neural network with one input layer, one hidden layer with linear neurons and no activation function and one output layer with a softmax function used for multiple classification [31]. There are two approaches when building a word2vec model, Continuous Bag Of Word (CBOW) and Skip-gram model. The CBOW model will be used in this thesis and hence further explained. In the CBOW approach the neural network tries to predict a word given a context where a context can be a word, a group of words or a sentence. In the case where the context is a single word the input and target to the neural network are one-hot encoded vectors. Initially, weight values are randomly assigned to the hidden layers. The input is passed through the hidden layer till the output layer which in turn predicts the output. The error between the output and the target is then calculated and back propagation, as explain in Section 2.1.6, is applied to update the weights of the hidden layer. After the network is trained, the output layer is removed and the trained weights of the hidden layer represents the actual word vectors. Instead of the input or context being a single word it can be several. The only difference is that the output of the hidden layer is averaged element wise over the number of input words. The Skip-gram model does basically the opposite, i.e. it will try to predict a context given a word.

Another type of embedding is to map positive integer values in an array to float values in a lower dimension. When dealing with sparse vectors this can be an efficient way to represent data in a more dense form. In deep learning this can be achieved by adding an embedding layer as the first layer of the model. The embedding layer will take arrays of integer values and map these to float values. An example is given in [32], the Keras documentation, where the embedding layer maps the input array $[[4],[20]]$ to the output $[[0.5, 0.1],[0.6, -0.2]]$. In this example the integer value 4 respectively 20 with dimension 1 x 1 has been transformed to a vector of size 1 x n where n equals 2.

2.6 Evaluate performance

Measure the performance of a machine learning model can be done in a several ways. For classification task a straightforward approach is to calculate the percentage of correctly classified and or incorrectly classified data points. This measure is called accuracy (see Equation 2.16) and as the name indicates measure how accurately the machine learning model can classify unseen data points [12]. This is a reliable metric of performance when the dataset used is balanced, i.e. the number of data points in each class are evenly distributed. Say that a dataset contains 50% positive and 50% negative data points. A machine learning model with accuracy 80% performs way better compared to randomly guessing if the next data points is either positive or negative which in the long run should give an accuracy of 50%. If the dataset is instead imbalanced e.g. there are 80% positive and 20% negative data points then just guessing that all unseen data points are positive will yield an accuracy of 80%. This is somewhat misleading since the accuracy of classifying positive data points seem to be high, but at the same time the accuracy of classifying negative data points is 0% which is really low.

There are several metrics to use when dealing with imbalanced datasets such as Precision, Recall and Reception Operating Characteristic (ROC) curve. To describe the performance of a model a confusion matrix can be used [33], see Figure 9. The table displays the actual condition in the columns and the predicted values in the rows. The green cells indicates correct predictions, i.e. for all conditions that actually are positive, predict positive and for all conditions that actually are negative, predict negative. The red cells indicates error predictions, e.g. if a condition is true, the model predicts it to be false or vice versa.



	Actual positive	**Actual negative**
Predicted positive	True positive	False positive
Predicted negative	False negative	True negative

Figure 9: *Confusion matrix*

Precision (Equation 2.17) and Recall (Equation 2.18) are two frequently used metrics that can be derived from the confusion matrix (Figure 9). Precision is a measure that when the model predicts true can tell how often the models is correct. Recall or True Positive Rate (TPR) is a measure that when the condition actually is true can tell how often the model predicts true. Another measure is F_1 score that calculates the weighted mean of Precision and Recall, see Equation 2.19.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.16}$$

$$Precision = \frac{TP}{TP+FP} \tag{2.17}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.18}$$

$$F_1 score = 2\frac{PrecisionRecall}{Precision + recall} \tag{2.19}$$

Where TP is the number of true positive, TN is the number of true negative, FP is the number of false positive and FN is the number of false negative.

The ROC-curve metric displays the relation between the TPR and the False Positive Rate (FPR) and is a common way to display the performance of a classifier [33]. An example of an ROC-curve can be seen in Figure 10 where TPR and FPR are calculated for every possible threshold. A good classifier will yield a ROC-curve that stretches up to the upper left corner while a poor classifier will be positioned close to the diagonal dotted line. The diagonal line essentially illustrates the result of random guesses. The Area Under the (ROC) Curve (AUC) is a metric that represent the ROC-curve as a single value. The AUC value can be in the range of 0 to 1 where 1 is the desired value and 0.5 represent the diagonal line, i.e. random guessing. In Figure 10 the AUC value in percent can be seen in the box in the bottom right corner of the plot.

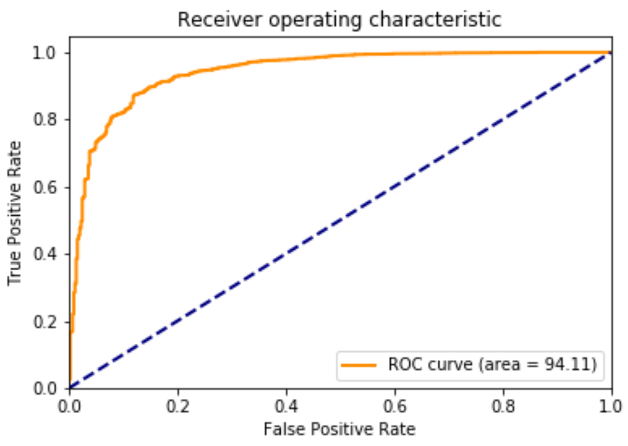


Figure 10: *Example of an ROC-curve with corresponding AUC value*

For sentiment classification of reviews where there are well-defined characteristics for positive and negative reviews the ROC-curve and hence also AUC are reliable measures of performance [34]. Also according to [35] for imbalanced dataset where the positive class is overrepresented ROC-curve and AUC is preferred over F_1 score which is derived from Precision and Recall. An example that illustrate that Precision, Recall and hence F_1 is not preferred is presented in [35] where a dataset with 10 samples are used and 1 sample is negative and 9 are positive. Say that a model predict every sample to be positive. This will yield the metrics where TP = 9, FP = 1, TN = 0, FN = 0 and by using Equation 2.17 and 2.18 the Precision = 0.9 and Recall = 1.0. Both Precision and Recall are one or close to one which by a first glance looks promising. However, by calculating TPR = TP/(TP+FN) = 1.0 and FPR = FP/(FP+TN) = 1.0 indicates that this is a poor classifier due to the high value of FPR. FPR is also known as the "false alarm rate" since it measure the ratio between the number of negative points falsely predicted as positive and the total number of negative points.

2.6.1 Baseline

When evaluating the performance of a machine learning model or essentially any kind of system or scientific experiment a baseline for comparison is fundamental [36]. Model accuracy, AUC or F_1 score do not actually say anything about the performance if not compared to a baseline. A common baseline for classification tasks is random guessing. For a balanced binary dataset random guessing corresponds to a 50% accuracy. The same principle can be applied to other investigations such as comparing a new model to a well-known model where the latter act as a baseline. By feeding the two models with the same input and calculate different metrics of how they perform the models can easily be compared.

The aim in this thesis is to investigate a CNN's transferability between different domains. In order to get a notion of the models performance it is interesting to not only investigate the metrics described above for different CNN architectures but rather compare these metrics to another well known statistical or machine learning model. The baseline model used in this thesis is a traditional machine learning model called *Maximum Entropy* or *Logistic Regression* for binary classification known to perform well on sentiment analysis tasks and commonly used as baseline for sentiment analysis research along with other machine learning models such as *Naive Bayes* and *Support Vector Machines* [37], [38]. In logistic regression the logistic function or sigmoid function explained in Section 2.1.3 is used to predict the probability that a data point, x, belongs to a certain class y [39]. This is further done by minimizing a cost function, see Equation 2.20, and use gradient decent, as briefly mentioned in Section 2.1.6, to update the weights, w.

$$Cost = -\frac{1}{m}\sum_{i=1}^{m}[y^i log(h_w(x^i)) + (1-y^i)log(1-h_w(x^i))] + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2 \tag{2.20}$$

Where

$$h_w(x) = h(w^T x) = \frac{1}{1+e^{-w^T x}} \tag{2.21}$$

is the hypothesis with the logistic function and

$$\frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2 \tag{2.22}$$

is the L2 regularization term that reduces the magnitude of w_j and is used to prevent overfitting, something that is discussed in Section 2.3. λ is the regularization parameter that control the magnitude of the fitting parameters, w_j.

2.7 Problem background and related studies

Domain adaption or transfer learning as it is also called is a highly active research field within the machine learning paradigm. The most common way to study transfer learning is to use data from two separate domains, although more than two domains can be used, and consider one of the domains as the source domain and the other domain as the target domain [7]. This means that data from source and target are sampled from different distributions [8]. Often the source domain data is labeled while all or most of the target domain data is unlabeled. The aim is to make use of the source data in the training processes to build a model for the target domain.

As explained in [40], the lack of training data with high quality is one of the biggest challenges when applying machine learning models to new domains. This is especially true for deep learning models where a lot of data often is needed to achieve good performance. These are typical use cases for transfer learning, i.e. to utilize non-target domain data which might be of higher quality, labeled and readily available to help train models for the target domain.

Transfer learning is widely known in the field of computer vision where deep neural networks seldom are trained from scratch but instead leverage networks that already have been trained on huge datasets [41]. These already trained neural networks can be reused and fine-tuned with new data to fit other domains. The main idea is that in a neural network with several layers the first layers, i.e. the layers further away from the output seem to capture more elementary features from the input signal compared to layers closer to the output [42]. A transfer learning approach is to freeze initial layers and fine-tune layers closer to the output. Using pre-trained networks and transfer learning will both reduce the training phase, require less data and hopefully still achieve similar performance as in-domain trained models.

Even though transfer learning has been successfully applied in the field of computer vision it has not been successful to the same extent for NLP and sentiment analysis. Although not as good results have been obtained for NLP compared to computer vision this does not mean that good results have not been obtained. It is however difficult to compare performance between separate fields such as computer vision and NLP. Transfer learning or domain adaption has been an active research field within NLP for over a decade where a lot of different approaches to tackle the difficulties have been studied. However, similar to sentiment analysis research the majority of work has focused on the English language.

Training a model on the source domain and directly apply it on the target domain often yields poor results [40]. Studies have shown that the target data, even if it is limited and/or unlabeled should be utilized and somehow be part of the pre-training and/or training process [8], [40]. Utilizing both source and target data can be achieved in several ways. In [1] they try to align domain-specific words from different domains into unified clusters using domain-independent words as a bridge, something they call Spectral Feature Alignment. This is done in an unsupervised fashioned where they cluster data from different domains to find domain specific and independent words. In [8] the authors are using a deep learning approach for domain adaption for sentiment analysis. They are using something called Stacked Denoising Autoencoders on data from both source and target to extract higher level features. Autoencoders can be used for dimensionality reduction, i.e. they try to find a way to reproduce the input signal but in a compressed way. The denoising part, i.e. applying noise to the input signal which slightly modifies the input signal can simulate that data might come from another distribution. This is an unsupervised approach and once the higher level features are extracted a supervised classifier known to perform well on sentiment analysis tasks called Support Vector Machines are trained and used for classification. The authors of [43] extends this research and instead suggest something they call marginalized Stacked Denoising Autoencoders which yields roughly the same performance as in [8] but in a more time effective way.

In [40] they study the impact of using pre-trained word vectors from the target domain before training a model with data from another domain. They study different deep learning models such as CNNs, RNNs and Long-Short-Term-Memory Networks. They conclude that using pre-trained word vectors from the target domain significantly boost the performance of transfer learning. [44] use CNNs to train a model based on job descriptions and evaluate it on resume data. Since job description data is easier to collect and somewhat related to resume data they use domain adaption to classify the more scarce resume data. The authors consider CNN to be useful in a domain adaption scenario where the goal is to classify short texts. They also suggest that the model most likely will perform better if the model is trained with both source and target data compared to merely source data.

3 Experimental setup

This section covers how the investigation of the Convolutional Neural Network's (CNN) transferability for sentiment analysis of Swedish reviews across domains will be conducted. Two types of experimental setups are presented along with how to study and interpret the results. The idea behind which data domains to investigate is also covered in this section.

3.1 Experiment

There are two experimental setups used in this thesis, the first is based on training a model on data from a source domain called S along with a fraction, α, of data from a target domain, T, and test the model on unseen target domain data, T'. The experimental setup can be seen in Figure 11 where the number next to the arrow indicate the order of the action. The experimental idea is to start with α equal to 1 and gradually decrease its value by a certain step length, h, down to 0. The model can hence be rigorously trained and tested for different amount of target data T. This approach yields different training dataset sizes for every α and when the source and target data size differs this might affect the transferability investigation and making it difficult to determine whether the model perform in a certain way due to the amount of data used for training. Therefore another approach where the training dataset size is fixed is also investigated.

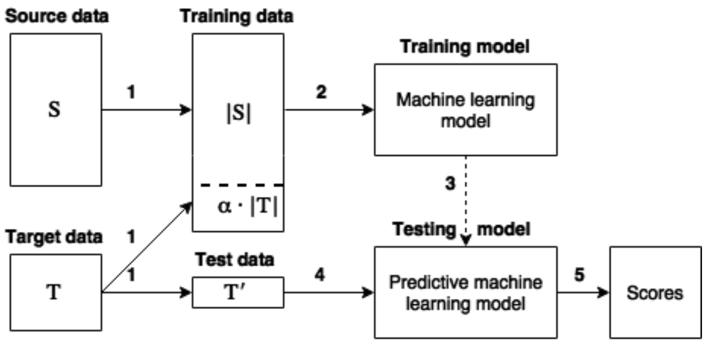


Figure 11: *Experimental framework I*

Instead of using the whole dataset of S, while $|T|$ is multiplied by α, $|S|$ is multiplied by $(1-\alpha)$ keeping the training dataset size constant throughout the experiment. $|S|$ and $|T|$ represent the dataset sizes for the source respectively the target datasets. Since the source and target dataset differs initially in size and to get a constant size, N, the larger of the two dataset will be truncated to the size of the smaller. The dataset size, N, is hence chosen by $N = min\{|S|, |T|\}$. The experimental setup in Figure 11 is hence modify to the experimental setup in Figure 12. Using the whole dataset of S as in Experiment I would probably be recommended in a real system since more data often yields better results for deep learning models. However, by keeping the data size fixed as in Experiment II there will be no bias towards larger datasets for individual tests and more focus can be directed towards the models' transferability.

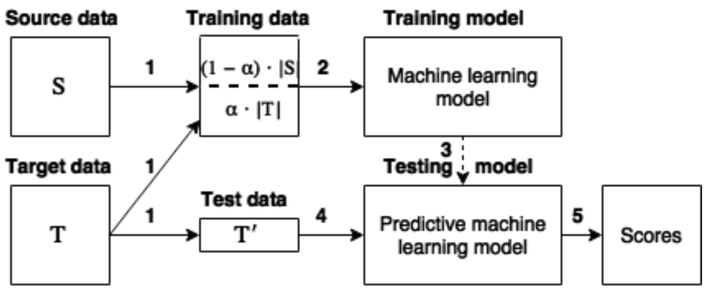


Figure 12: *Experimental framework II with fixed dataset size*

3.1.1 Experiment evaluation

By starting with α set to 1 and gradually decrease its value down to 0 by a certain step length, h, should indicate how target data dependent the model is between certain domains. Plotting the models AUC score against α will yield a visual indication of how transferable the model is between the chosen domains or how related the domains are. A transferable model should have a somewhat constant AUC score and hence a constant line in the plot. Also, inspecting the AUC score for different α will indicate whether there is a critical point in the relation between the amount of target respectively source data. An AUC score that drops significantly when α decreases probably indicate that the model is not that transferable or there are no or little relation between the domains. The goal is hence not necessarily to achieve the highest possible accuracy or AUC score, instead a model that can keep or not loose too much performance when α is decreased is desired. Comparing Experiment I and Experiment II will give an indication of the dataset size impact on the transferability, especially for the extreme case where $\alpha = 0$ and the datasets differs notably in size.

To get an overview of how transferable the models are in its extreme cases the AUC loss will be studied. The AUC loss is in this thesis calculated by taking the difference in AUC score for $\alpha = 1$ and $\alpha = 0$ for the second experiment with fixed dataset size. When α is equal to 1 the model is trained and evaluated on the target domain (in-domain classification) while when α is equal to 0 the model is trained on merely the source domain and evaluated on the target domain (out-domain classification). Furthermore, some data domain characteristics such as frequently used words and intersection of words across domains will be studied. This will indicate whether different domains are using the same type of words and how domain specific the most frequent words are. Also, misclassification statistics such as proportion of misclassified positive and negative reviews and average review text length will also be investigated. Studying misclassification can reveal if there are any particular type of reviews the model having trouble to classify.

3.1.2 Data domains

The choice of which source and target domain data to use for different tests is based on the idea of starting with domains that intuitively seem to be more related to then test the transferability between more distant domains. An example hypothesis is that the domain category ”Datorer & Tillbehör” is closer related to ”Ljud, Bild & Musik” and ”Elektronik” than to ”Hem & Trädgård” in dataset *A* or to ”Restauranger” in dataset *C*, see Section 5 for more information about the datasets. This hypothesis is based on word similarities across domains where domains related to say technology may intuitively use more similar words. Reviews for technical products such as computers or monitors may include words such as ”portable”, ”battery”, ”sharp”, ”resolution” or ”memory”. Although, some words may be even more domain specific and only suitable for a specific product or service. However, these words are probably not that common to use when reviewing restaurants where words like ”service”, ”tasty”, ”disgusting” and ”fresh” are more frequently used. Some words that are more generic and may be found across almost all domains are words such as ”expensive”, ”cheap”, ”great”, ”good” and ”bad”.

3.1.3 Models

The models that are going to be investigated are foremost a CNN designed for sentiment analysis, (for detailed information about the CNN model see Section 4.7). To compare the results of the CNN model a traditional statistical or machine learning model called logistic regression will undergo the same experiments. In order to evaluate the impact of different text representations the CNN model will be tested with both standard word embedding and pre-trained word embedding using a word2vec model. The baseline, i.e. the logistic regression model will be tested with both a unigram as well as a unigram and bigram text representation.

4 Methodology

In this section, the methodological approach of the research is explained. It include a detailed explanation of the research workflow, i.e. every necessary step taken from data gathering and data preparation to modeling and model evaluation. Furthermore, tools, libraries and frameworks used throughout this thesis are also described.

4.1 Delimitation

The CNN architecture used for the main experiments in this thesis were chosen based on initial testing of various different CNN architectures with different hyperparameter settings on different domains along with inspiration from previous work. There is however no guarantee that the particular CNN architecture used in this thesis is the best performing CNN architecture for transfer learning in this context. The aim and scope of this thesis is however not to evaluate every possible CNN architecture and to find the optimal CNN architecture for transfer learning, it rather focuses on investigating a CNN’s transferability across different domains and how the model behave when varying the amount of source and target data used for training. Therefore, through initial testing of several CNN architectures for different domains a fairly simple model that performed similar as more complex models were used for the main experiments.

4.2 Workflow

The investigation of how well suited CNNs are for transfer learning for sentiment analysis of Swedish reviews included several steps. An overview of the workflow or the necessary steps taken in this thesis can be seen in the list, Workflow, below. The first step was to gather data. Data is of course a vital part when dealing with machine learning models and without data this research would not be possible. Next, inspection of the collected data gave an over all understanding of its characteristics. The next step was to pre-process and prepare the data for machine learning followed by building desired machine learning models. To keep track of and for easy access during the subsequent analysis phase all tests and test results were stored in a database. The individual steps in the Workflow list below will be further explained in the following sections but first a brief explanation of the software tools that were used is presented.

Workflow

1. Gather data
2. Inspect data
3. Prepare data for machine learning
4. Build machine learning models
5. Train and test machine learning models
6. Store tests, results and metadata
7. Analysis of test results together with data inspection in step 2
8. Repeat from step 4

4.3 Tools

To fulfill every step in the workflow presented in the previous section a set of software tools were necessary to use. A brief overview of the most vital and frequently used tools and libraries in this thesis is presented next.

4.3.1 Python

Python is a general-purpose programming language extensively used throughout this thesis. It is as described in [45] an object-oriented, interpreted, high-level programming language that can be applied to many different problems and scenarios. Python has a standard library for handling areas such as string processing, Internet protocols and operating system interfaces among others. However, Python supports a wide range of third-party extensions as well [46]. The major Python libraries or extensions used in this thesis are Scrapy, IPython, Jupyter Notebook, Pandas, Matplotlib, Scikit-learn, Tensorflow, Keras and Gensim. These libraries and what they were used for are briefly described next.

Scrapy
Scrapy is an open source library or framework for extracting or scraping data from websites

[47]. This tool was used in this thesis to develop scripts that could crawl three different websites and extract review data along with metadata.

IPython and Jupyter Notebook
IPython is an interactive shell for python with a Jupyter Notebook kernel allowing for interactive development of code in the browser [48], [49]. It is widely used in the field of data science and similar for data processing, statistical modeling, machine learning, numerical simulations and visualization. Jupyter Notebook was used throughout this thesis for data processing, machine learning, analysis and visualization.

Pandas
Pandas is an open source library for data manipulation and data analysis [50]. It provide easy handling of data structures along with high performance. Pandas has been used throughout this thesis for processing and handling data along with data analysis.

Matplotlib
Matplotlib is a Python library dedicated for plotting figures, graphs and diagrams [51]. It was used in combination with Jupyter Notebook for visualization.

Scikit-learn
Scikit-learn is an open source machine learning, data mining and data analysis library for Python [52]. The library provide, among others, the logistic regression model used as baseline in this thesis.

Tensorflow
Tensorflow is an open source library for machine intelligence and numerical computations originally developed by Google [53]. The library is especially well suited for deep learning where the flexible architecture allows computations to be deployed to one or more CPUs or GPUs. Tensorflow was used as the backend software to allow deep learning in this thesis.

Keras
Keras is a high-level API that can be used together with Tensorflow and other deep learning libraries [54]. Its main focus is to allow for fast prototyping and experiment. The Keras API was used to develop the CNNs with different architectures examined in this thesis.

Gensim
Gensim is a library for unsupervised semantic modeling from plain text. The library is specialized in processing raw, unstructured digital text in an efficient way [55]. Building and training the word2vec model based on review texts was achieved using the Gensim library.

4.3.2 PostgreSQL

PostgreSQL is an open-source relational database system [56]. In this thesis PostgreSQL were used as a database to store data in a structured way. Examples of what was stored is information related to the datasets, various tests, test results and metrics. The data can easily be accessed by querying the database using SQL queries.

4.4 Data gathering

The data used for this research was scraped from three different websites that provide reviews. These websites contain everything from product and restaurant reviews to reviews of companies, hair designer, dentists, plumbers and many more. Along with a written opinion the review is associated with a rating, either in the range of 1 to 5 or 1 to 10 depending on the website, with 5 respectively 10 as the highest grade and 1 the lowest. Review text, rating as well as meta information such as brand, superior category, category and sub category, company name and the corresponding url was gathered. An example of a gathered review along with metadata can be seen in Section 5.2.

Scraping the websites for information included several steps. First the underlying structure of the website was inspected. This included inspection of the HTML structure, searching for urls as well as tags and their identifiers to determine where the relevant information was located. After that it was possible to develop scripts that found these tags and extracted the desired information. The gathered information was then stored on disk in JSON format.

The underlying structure of websites usually differs. This means that the scraping script had to be tailor made for every new website. Although several sites underlying structure had similarities and hence much of the scraping code could be reused.

4.5 Data inspection

After the data was collected it was inspected to get an overall understanding of the data, its characteristics, its shape and its distribution. This phase included looking into and reading samples of reviews to grasp what makes a high as well as poor rated review. Another important inspection step was to get an overall understanding of the whole dataset such as number of reviews, how the reviews are distributed over different categories, the overall distribution of ratings and the average length of review texts in terms of words. A more in-depth inspection of the datasets used in this thesis can be seen in Section 5.

4.6 Preparing data

Pre-processing and cleaning data before feeding it to a predictive model is a major and an important part when working with machine learning. ”Garbage in garbage out” is a well know saying in machine learning, automatic control or essentially anywhere where an input signal is fed to a system to produce an output signal. There are of course many aspects of data quality, sometimes no matter how much the data is pre-processed it is still considered garbage. However pre-processing can somewhat increase the data quality and some pre-processing is almost always needed to transform the data into a format that the system or model can interpret.

The pre-processing of the text data included removal of HTML-tags, non alphanumeric tokens, blank lines, whitespace in beginning and ending of texts. A few of the collected reviews only contained rating and no review text, other review texts only contained a single character. These reviews were of no use and hence directly removed. The rating value was converted from strings to integers. After the major pre-processing was done the data was saved and stored on disk in a Comma Separated Value (CSV) format for easy access later on.

Further data preparations included removal of reviews with review text shorter than a certain threshold. The threshold used in this thesis was to remove all reviews that contained less than two words. Since reviews with a text length shorter than two words were directly removed a lower threshold could never be used, no restrictions were however set for an upper threshold at this point.

In this thesis only positive and negative reviews were considered. Since the review rating values were used as sentiment indicators, i.e. high ratings were associated with positive review text while low ratings were associated with negative review text, the neutral reviews were removed. For reviews where the ratings ranged from 1 to 5, ratings of 1 and 2 were considered negative while ratings of 4 and 5 were consider positive. Ratings with value 3 were considered neutral. In the case where it was possible to set a rating in the range from 1 to 10, reviews with rating 1, 2 and 3 were treated as negative, 4, 5, 6, 7 as neutral and 8, 9 and 10 as positive. For both rating metrics the neutral reviews were removed. The transformation from rating to sentiment was inspired by previous research where [1], [8], [43] and [57] used a dataset collected from *Amazon* that contain reviews of *DVD*, *Books*, *Electronics* and *Kitchen appliance* with minimum rating 0 and maximum rating 5. The rating were converted to sentiment where reviews with rating > 3 were considered positive and reviews with rating < 3 were considered negative and the remaining reviews were neglected.

4.6.1 Text representation

Several types of numerical representations of the review text were used for different models. These included BoW and Tf-Idf with unigram and bigram described in Section 2.5. Different types of word embedding were tested for the deep neural networks. Both Keras embedding layer, which later on will be called standard word embedding, as well as pre-trained word embedding produced by a word2vec model were investigated.

In order to feed a CNN with text data the first step was to map every word in the dataset to an integer or index value representing the word in a vocabulary. Each review is then converted to a sequence of integers where each integer represent a word. Since a CNN requires that the input is of the shape $n\ x\ m$ and reviews are of different length the sequences are padded to equal length. Longer reviews are truncated while shorter reviews are padded with 0 values. The threshold for review text length were set to 300 tokens.

Over a million raw reviews from different categories and websites were gathered in this thesis. Out of these reviews 927250 contained at least two words in their review text. These review texts were used to train a word2vec model using the CBOW approach with word dimension 100 described in Section 2.5.3. The minimum count was set to 5 meaning words that appeared less than 5 times in the whole corpus were removed. The reason was that words that appear less than a certain threshold can be considered typos or garbage. In the case where the less frequent words are not garbage the limited quantity is still not enough for training. The threshold of 5 was chosen arbitrarily and based on the fact that it was the default threshold in the Python library Gensim used to train the word2vec model.

4.7 Modeling

A lot of different CNN architectures were tested but the basic structure used can be seen in Figure 13. Different number of convolutional filters, kernel, filter or window sizes and pooling sizes were tried out. The use of denoising layer and dropout layer were also tested. Other settings that was tuned were settings like number of epochs to train the model, batch size and whether certain weights should be trainable or not. Different text representations as described in previous section such as a standard embedding layer and an embedding layer were the weights are pre-trained using an unsupervised word2vec approach were thoroughly investigated. Although a lot of different settings were tried a few of the settings were early on kept constant throughout the major part of the experiments. These settings were based on both inspiration from previous work and through testing. Among these settings were ReLU activation function, sigmoid function for classification in the output layer, binary cross-entropy as cost function and Adam as optimization function.

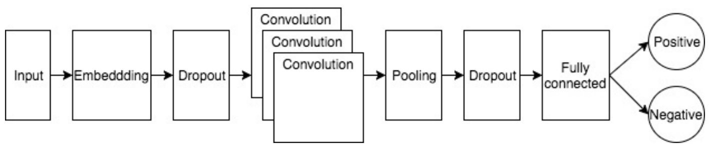


Figure 13: *Basic CNN architecture for sentiment classification*

An example of setting up a CNN with one convolution layer, several filters, a pooling layer and a fully connected layer using Python and Keras with Tensorflow backend is shown in the code below along with additional comments. Note that this is merely an example of how to set up a CNN in Keras, no dropout layer is for instance present in this example.

```
from Keras import ... # assume all necessary imports

# Specify parameters
EMBEDDING_DIM = 100
MAX_SEQUENCE_LENGTH = 300
TRAINABLE = True
..
# Instantiate a sequential model
model = Sequential()
# Embedding layer as the first layer
model.add(Embedding(input_dim=len(word_index)+1,
            output_dim=EMBEDDING_DIM,
            weights=[embedding_matrix], # initialize embedding weights
            input_length=MAX_SEQUENCE_LENGTH,
            trainable=TRAINABLE))
# One dimensional convolutional layer with specified number of filters
# filter size, padding and activation function
model.add(Conv1D(filters=100, kernel_size=4,
                 padding='same', activation='relu'))
# One dimensional max pooling layer with window size 2
model.add(MaxPooling1D(pool_size=2))
# Flatten the output from the pooling layer
# so it fits the fully connected layer
model.add(Flatten())
# Fully connected layer with ReLU activation function
model.add(Dense(units=250, activation='relu'))
# Output layer predicting the output using sigmoid activation function
model.add(Dense(units=1, activation='sigmoid'))
# Other settings such as loss function,
# optimization algorithm and desired metrics
model.compile(loss='binary_crossentropy', optimizer='adadelta',
            metrics=['accuracy', 'precision', 'recall', 'f1'])
# Train the model with training data and validation data
# for a given number of epochs and batch size
model.fit(X_train, y_train,
          validation_data=(X_val, y_val),
          epochs=3,
          batch_size=32)
# Evaluate model on new data...
scores = model.evaluate(X_test, y_test)
# ...or generate predictions based on new data
predictions = model.predict(X_test)
```

Through testing it turned out that a fairly simple model performed similar as a more complex model and since the training time increases significantly when the number of parameters increases the simpler model was used for the extensive experiments in this thesis. The most important features of this model are summarized in Table 1. Dropout is used both directly after the embedding layer and prior to the fully connected layer with dropout probability of 0.2 and 0.5 respectively. The use of dropout in the initial part of the network was inspired by previous research where denoising or corruption were added to the data in order to train a robust classifier more suited for transfer learning. The early dropout is also a way of simulating that the data comes from a slightly modified distribution. A dropout layer close to the output layer can also increase the robustness of the network and is mainly used to prevent overfitting. Choosing number of convolutional filters and their kernel sizes as well as other hyperparameters were determined by trial-and-error along with inspiration of previous work in the area (and the fact that complex models are more computationally intensive to train). As mentioned earlier, since no major improvement was gained using a more complex network the simplest possible but still well performing network was used. The weights of the embedding layer were both initialized randomly using standard embedding as well as with the pre-trained word2vec representation of the review text data. All the weights of the network including the embedding layer weights were set to be trainable, i.e. no weights were locked or frozen during training. Freezing the weights of the embedding layer were also tested but keeping them trainable yielded better performance for initial tests. One epoch means a full training cycle on the training set. Batch size is the number of training samples that are fed to the model at a time.

Table 1: CNN model overview.

Conv.	**Embedding**	**Filters**	**Kernel size**	**Pooling**	**Dropout**	**Epochs**	**Batch**
1	random, w2v	32	3	2	0.2, 0.5	5	128

For the baseline an L2 regularized logistic regression model with BoW data representation with unigram and unigram and bigram were used. The regularization parameters, λ, was set to 1 and the solver used was the default efficient large scale linear classification library, liblinear [58]. Using techniques such as Tf-idf might improve the performance but for fair comparison both models, the CNN and logistic regression, used a vocabulary that consisted of the 5000 most frequent words.

4.7.1 Training and testing

The datasets for the CNN model were split into three sets; a training set, a validation set and a test set with distribution 60%, 20% and 20% respectively. The validation set was mainly used to prevent overfitting and to know when to stop training and can be seen as an extended training set. Therefore the validation set included, like the training set, both source and target data with the same proportion as for the training set. For the baseline the datasets were instead split into two sets, a training set and a test set with distribution 80% and 20%. Both the CNN and baseline were trained and tested in a K-fold cross-validation fashioned which means that the model trained on the training set and was tested on the test set K times with randomly selected train and test data for every K. An average of the scores of the K runs were then calculated. K was set to 5 for the CNN and 10 for the baseline. The reason for that the two models differ in the way they were trained and how the datasets were split is because of how the models work. The more traditional baseline model cannot take a validation set as input therefor it got a training set of size 80% while the CNN got a training size of 60% plus a validation set of 20%. For all experiments the test set contained only target data and was not used in training in any way.

Several different domains were investigated. Table 2 displays the investigated domains and an indication of whether the domain acted as source or target along with the type of reivew data. The *Datasize* column indicate whether the data size was fixed (Experimental II), unfixed (Experiment I) or both. Both fixed and unfixed datasets were used only when the source dataset was larger than the target dataset. For experiment II, when the source dataset was smaller than the target dataset the larger dataset was truncated to the same size as the smaller of the two datasets. All experiment were executed by starting with $\alpha = 1$ and gradually decrease its value by a step length, $h = 0.2$, down to $\alpha = 0$.

Table 2: Domains investigated.

Source	Target	Type	Data size
Elektronik	Restauranger	Company→Company	Fixed, unfixed
Elektronik	Datorer & Tillbehör	Company→Product	Fixed, unfixed
Elektronik	Ljud, Bild & Musik	Company→Product	Fixed, unfixed
Elektronik	Hem & Trädgård	Company→Product	Fixed, unfixed
Restauranger	Elektronik	Company→Company	Fixed
Restauranger	Datorer & Tillbehör	Company→Product	Fixed, unfixed
Restauranger	Ljud, Bild & Musik	Company→Product	Fixed, unfixed
Restauranger	Hem & Trädgård	Company→Product	Fixed, unfixed
Datorer & Tillbehör	Elektronik	Product→Company	Fixed
Datorer & Tillbehör	Restauranger	Product→Company	Fixed
Datorer & Tillbehör	Ljud, Bild & Musik	Product→Product	Fixed, unfixed
Datorer & Tillbehör	Hem & Trädgård	Product→Product	Fixed, unfixed
Ljud, Bild & Musik	Elektronik	Product→Company	Fixed
Ljud, Bild & Musik	Restauranger	Product→Company	Fixed
Ljud, Bild & Musik	Datorer & Tillbehör	Product→Product	Fixed
Ljud, Bild & Musik	Hem & Trädgård	Product→Product	Fixed, unfixed
Hem & Trädgård	Elektronik	Product→Company	Fixed
Hem & Trädgård	Restauranger	Product→Compnay	Fixed
Hem & Trädgård	Datorer & Tillbehör	Product→Product	Fixed
Hem & Trädgård	Ljud, Bild & Musik	Product→Product	Fixed

4.8 Evaluation

The experimental results from the two experimental for all models were visualized by plotting the AUC score against α. To compare the different models, CNN and baseline, for two given domains, the resulting AUC score for different α were plotted in the same figure. This gave a clear visual representation of how transferable the models were for different domains and for different amount of data. The AUC loss explained in Section 3.1.1, were summarized in two bar plots giving an overview of the models transferability across domains. Furthermore, some information regarding the best performing models misclassification statistics were summarized in a table. Statistics in this sense mean proportion of positive and negative reviews misclassified (FP, FN) along with average review text length in words for $\alpha = 1$ and $\alpha = 0$ for Experiment II, i.e. in-domain classification versus out-domain classification. This gave an indication of what kind of reviews the model struggled to predict. Also, some characteristics regarding the different data domains such as most frequent words and intersection of words across domains were also conducted. This gave a hint of the data domains vocabulary richness along with common words used across domains.

5 Data

In this section, information regarding the datasets used in this research are presented. Information such as size of the datasets i.e. number of reviews, average review text length in words, distribution of ratings and a few example ratings are displayed. A more detailed inspection of the datasets and domains used for the main experiments is shown. Lastly, some results regarding the pre-trained word2vec model are also presented.

5.1 Datasets

Three datasets gathered from different websites were used in this thesis. Table 3 displays an overview of the number of reviews gathered for each dataset prior to and after pre-processing. The number of reviews after the neutral reviews were deleted is also displayed along with the average review text length in words for each dataset. Dataset *A* is more focused on individual product reviews while dataset *B* and dataset *C* rather contain company reviews. However, all datasets are divided into categories on their respective website.

Table 3: Overview of datasets.

Source	**Raw**	**Pre-processed**	**Positive and negative**	**Avg. review length**
A	345765	144541	112626	57
B	737131	733555	694990	19
C	49405	49133	44531	24

5.1.1 Dataset details

The individual reviews in dataset *A* belongs to a superior category. The distribution of reviews over their superior categories can be seen in the Table 4. The table displays the number of reviews after pre-processing is done and neutral reviews are removed. Categories containing less than 1000 reviews are not included in the table.

Table 4: Overview of dataset *A*.

Category	**Number of reviews**
Spel och Film	38996
Datorer och Tillbehör	24336
Ljud, Bild och Musik	21507
Mobil och GPS	12049
Hem och Trädgård	6872
Skönhet och Hälsa	3624
Foto och Video	2595
Skor, Kläder och Accessorarer	1139

The overall distribution of reviews for dataset *A* can be seen in Figure 14 and the distribution of the dataset where the neutral reviews are removed and the positive and negative reviews are respectively grouped together can be seen in Figure 15.

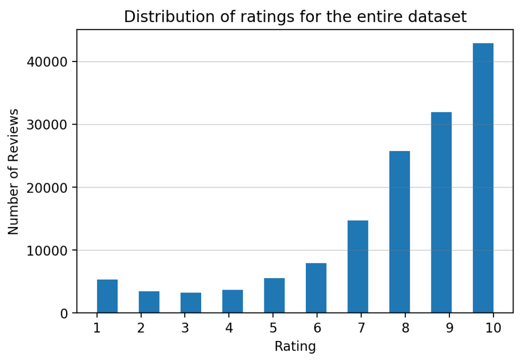


Figure 14: Distribution of rating

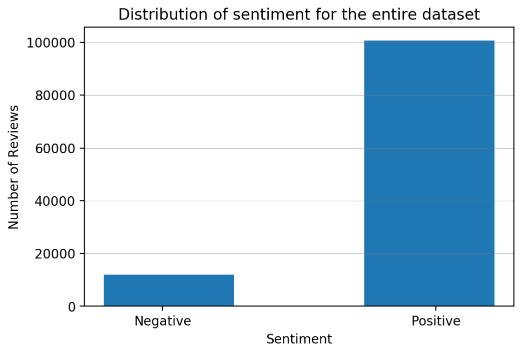


Figure 15: Distribution of sentiment

The number of reviews for each category in dataset *B* can be seen in Table 5. The table displays the number of reviews after pre-processing is done and neutral reviews are removed. As for dataset *A*, categories containing less than 1000 reviews are not included in the table.

Table 5: Overview of dataset *B*.

Category	**Number of reviews**
Hem och Trädgård	167844
Kläder och Mode	109588
Transport	90232
Fritid	70065
Elektronik	54429
Hälsa och Välbefinnande	43848
Resor och semester	31045
Pengar	29103
Sport	22496
För Företag	20966
Mat och Dryck	10372
Hantverkare	10022
Barn	9890
Konst	7915
Datorer och Tillbehör	7325
Telefon- och Internettjänster	3442
Sällskapsdjur och husdjur	2564
Underhållning	2279
Tobaksprodukter	1203

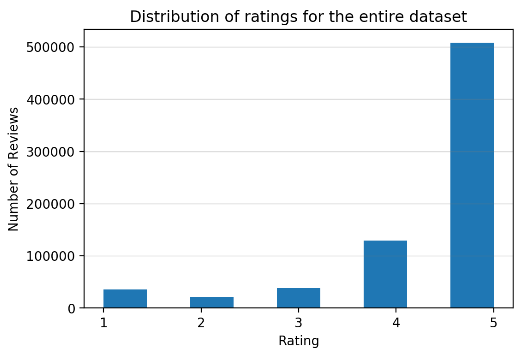


Figure 16: Distribution of rating

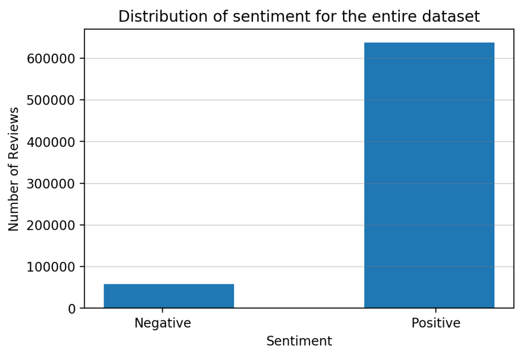


Figure 17: Distribution of sentiment

The number of reviews for each category in dataset C can be seen in Table 6. The table displays the number of reviews after pre-processing is done and neutral reviews are removed.

Table 6: Overview of dataset C.

Category	**Number of reviews**
Restauranger	26089
Bilverkstäder	11509
Frisörer	6934

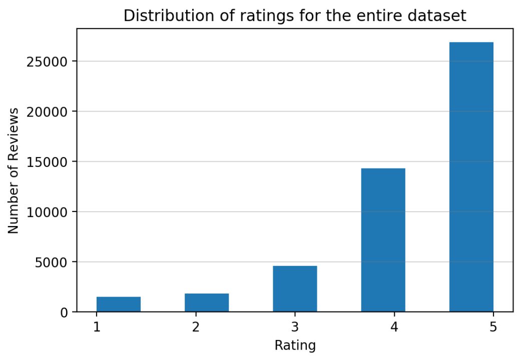


Figure 18: Distribution of rating

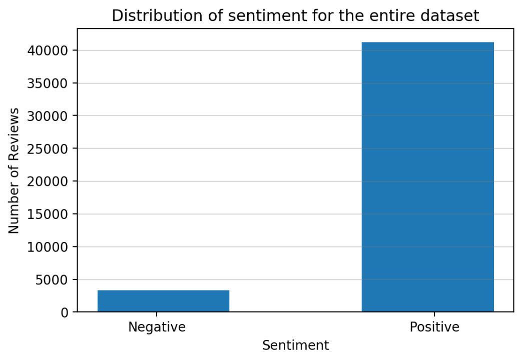


Figure 19: Distribution of sentiment

5.1.2 Investigated domains

Figure 20 displays the distribution of sentiment for the domains used in the main experiments. The data domain names are hereafter often abbreviated, Datorer & Tillbehör (D&T), Ljud, Bild & Musik (LB&M), Hem & Trädgård (H&T), Elektronik (E), Restauranger (R). See Table 13 in Appendix A for more details about what subcategories and the amount of data the investigated domains contain.

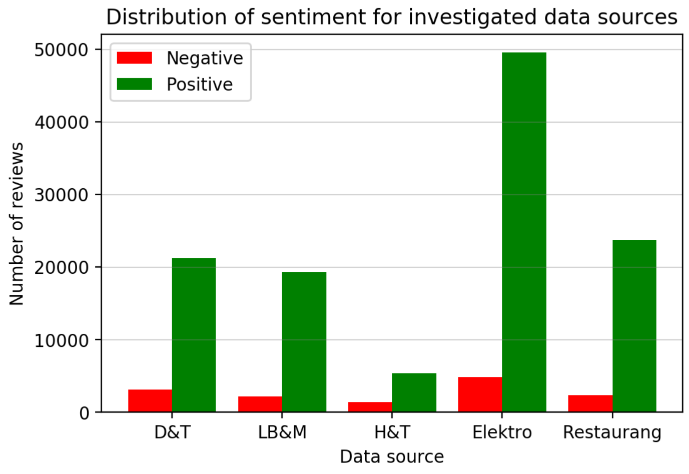


Figure 20: *Distribution of sentiment for investigated domains*

5.2 Example reviews

Every review was gathered along with some metadata. A more detailed example of what a review object would look like is shown below.

```
Brand: Samsung
Broad (Superior) category: Ljud, Bild & Musik
Category: Hembiosystem & stereopaket
Specific category: Hembiosystem
Name: Samsung MS660/MS661
Review text: Väldigt nöjd med musik- och filmupplevelsen. Mer djup i filmen och
    discomusiken flödade. Helt klart värt pengarna. Barnen har använt den flitigt och
    vi vuxna med. Rummet fylls med en känsla som är så verklig så vi lever oss
    verkligen in i filmen och musiken.
Rating: 10
Url: http://wwww.link.to.review/review_id
```

However not all meta attributes were available for all websites and reviews but were gathered where possible.

Below in Table 7 are a few examples of positive and negative reviews randomly selected from the investigated data domains along with their original corresponding rating.

Table 7: Example reviews with corresponding rating.

Review text	**Rating**	**Category**
Mycket nöjd. Mycket ljud till ett rimligt pris. Lätt att installera, lätt att hantera ur kontroll, iPad eller mobiltelefon. Litet fotavtryck och ger mycket ljud.	10/10	LB&M
Gör det den ska och jag upplever ingen fördröjning alls. Väldigt simpel att installera på min mac, använder den i garageband och funkar kanon. Man kan även koppla in förutom en mic, ett till instrument. Den är dock väldigt simpel och inget extra så tycker 900kr är ett ganska kraftigt pris för den, dock köpte jag den hos X för ca 600 vilket är ett mer logiskt pris.	8/10	D&T
Läcker vatten emellanåt. Vid lägre temperaturval stänger den av sig själv. Man får ”jobba” ganska ordentligt för att få det riktigt slätt vilket jag inte har behövt med andra strykjärn. Köpte den billigt på erbjudande men skulle inte rekommendera produkten.	3/10	H&T
Lämnade tillbaka min på garantin när den började lukta brännt. X hade ingen ny på lager och vägrade att ge mig en V2. Kass produkt och kasst av X att inte ge mig en likvärdig produkt. Fick istället ett tillgodokvitto.	1/10	H&T
En mysig restaurang, jättegod mat och mycket trevlig personal! Vi rekommenderar gärna denna restaurang till andra och kommer säkerligen att besöka den igen vi också!	5/5	R
Bra grejor. Vad jag behöver till mobilen. Bra priser och enkel leverans. Tyvärr gillade min iPhone 7 plus bara Apple orginaldelar, X lightning till 3 mm piratdelen gillas inte efter nån minut. Alla övriga delar funkar bra.	4/5	E
Jag var där och käkade igår den xx-xx-xx. Jag har aldrig blivit så magsjuk som nu. Är typ grön i ansiktet och spyr som en räv. Gör mig en tjänst passa dig för detta stället.	1/5	R

5.3 Word2vec

Since the word2vec model trained on the review text produced numerical vectors that represents each word in the training corpus it is possible to see how similar different words are, i.e. how close they are in the vector space. A few examples of word similarities for the word2vec model used in this research can be seen in Table 8. Obtaining the word similarities are done by utilizing the built-in, *most_similar*, feature in the Python library Gensim.

Table 8: Top 5 similar words

Word	Similar word	Similarity
jättebra	superbra	0.894
	kanon	0.853
	kanonbra	0.852
	toppen	0.850
	utmärkt	0.820
dåligt	uselt	0.760
	kasst	0.743
	illa	0.706
	konstigt	0.694
	märkligt	0.692
extremt	oerhört	0.853
	otroligt	0.810
	väldigt	0.809
	fruktansvärt	0.797
	sjukt	0.749
skarp	ljusstark	0.876
	knivskarp	0.798
	färgåtergivning	0.714
	skarpa	0.713
	bakgrundsoskärpa	0.702

6 Result

In this section the results from the experiments are presented. First an overview in terms of AUC loss across domains is presented followed by more detailed plots showing the AUC score for different amount of source and target data. Next, some data domains and word characteristics across domains are presented followed by some misclassification statistics. The representation of the data domains in this section are often abbreviated, as in Section 5. Datorer & Tillbehör (D&T), Ljud, Bild & Musik (LB&M), Hem & Trädgård (H&T), Elektronik (E), Restauranger (R). To display which bar or curve that belongs to a certain model in the plots, all plots are labeled. The labels are $LR-uni$ which stand for logistic regression with unigram text representation, $LR-unibi$ stand for logistic regression with unigram and bigram text representation, $CNN-emb$ represents a Convolutional Neural Network (CNN) with standard word embedding and $CNN-w2v$ represents a CNN with pre-trained word2vec text representation. All the plots are annotated with a domain and an arrow pointing to another domain. This representation means that the domain to the left of the arrow is the source (S) domain and the other is the target (T) domain, e.g. Elektronik (S) $\rightarrow$ Datorer & Tillbehör (T).

6.1 Overview

Figure 21 and 22 shows an overview of the AUC loss between domains. The AUC loss is the difference in AUC score for $\alpha = 1$ and $\alpha = 0$, i.e. in-domain versus out-domain classification. Higher AUC loss value indicate worse performance in terms of transferability between two domains. A few rare cases displays negative AUC loss, this indicates that the AUC score is actually higher for the out-domain case.

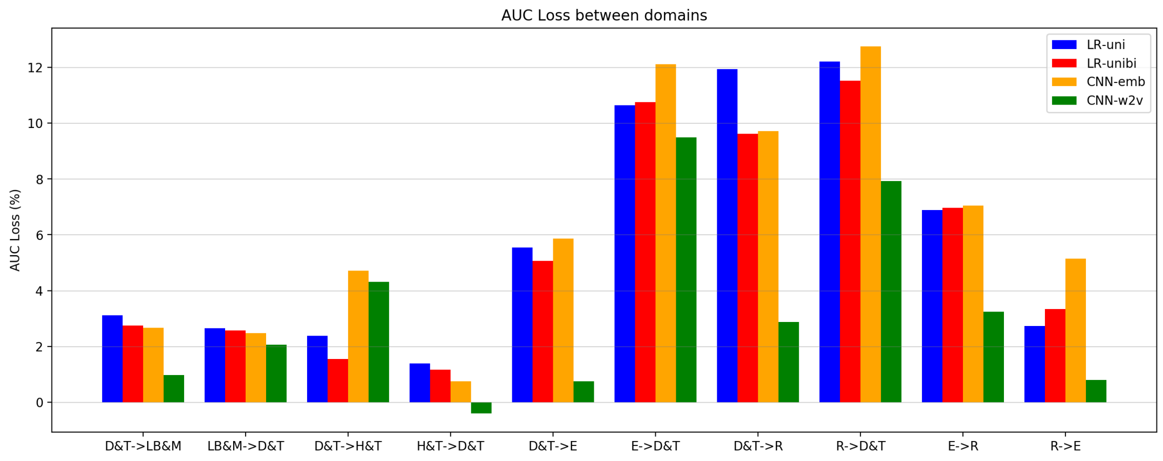


Figure 21: *AUC loss between domains for in-domain vs. out-domain classification.*

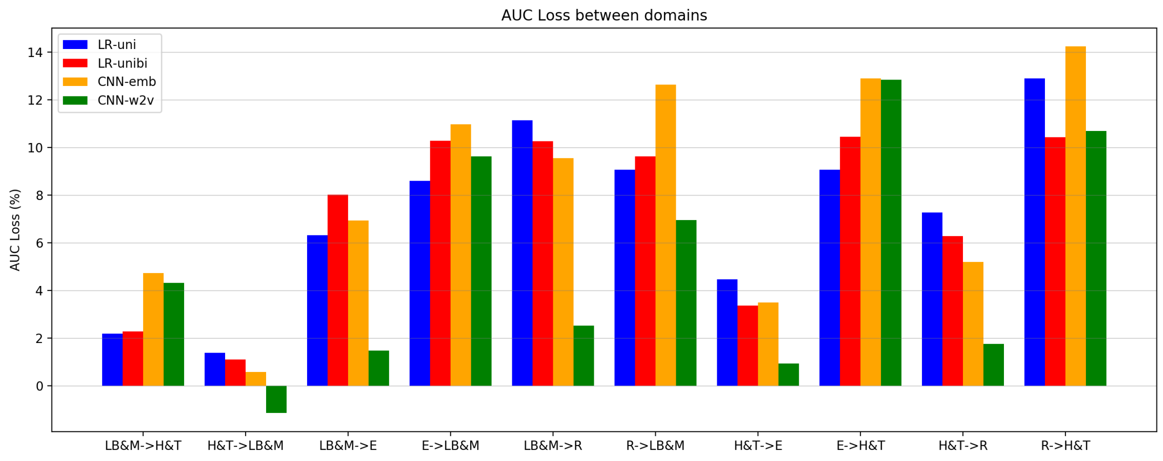


Figure 22: *AUC loss between domains for in-domain vs. out-domain classification.*

6.2 Area Under the Curve score

Several plots displaying the investigated machine learning models' transferability between domains are presented next. The following ten figures are structured in the same fashioned, i.e. the figures contain three plots where the plot to the:

- Left - represent experiment I
- Middle - represent experiment II
- Right - represent experiment II where the source and target are switched

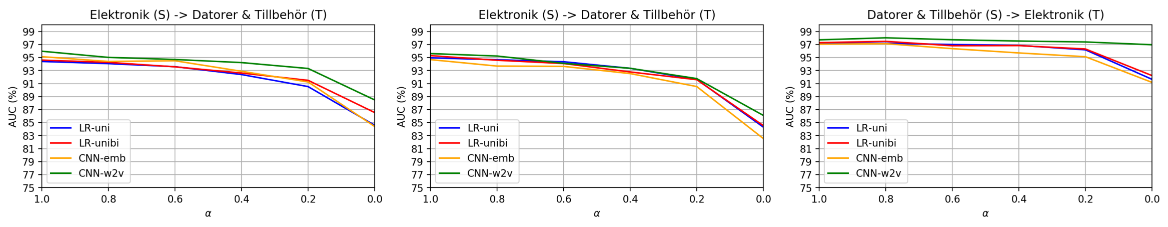


Figure 23: *AUC score for different amount of source respectively target data for Elektronik → Datorer & Tillbehör and Datorer & Tillbehör → Elektronik.*

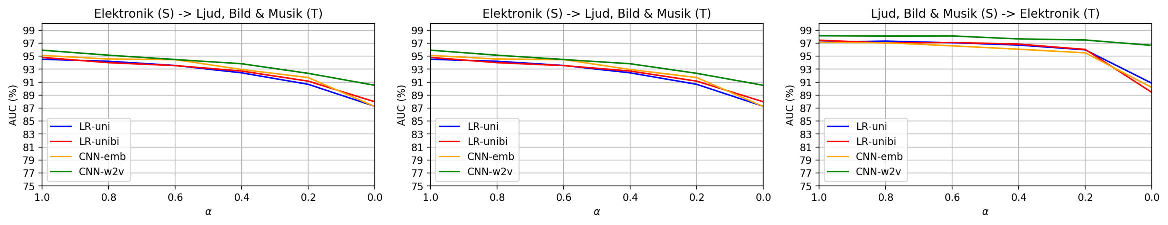


Figure 24: *AUC score for different amount of source respectively target data for Elektronik → Ljud, Bild & Musik and Ljud, Bild & Musik → Elektronik.*

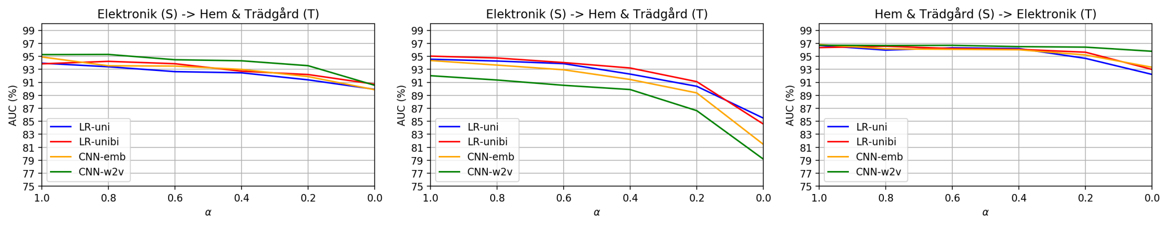


Figure 25: *AUC score for different amount of source respectively target data for Elektronik → Hem & Trädgård and Hem & Trädgård → Elektronik.*

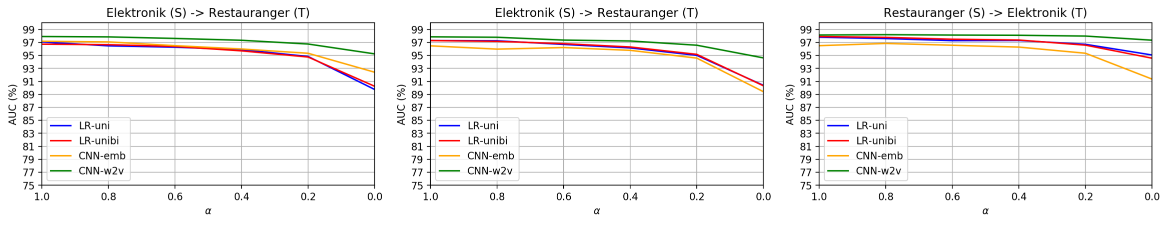


Figure 26: *AUC score for different amount of source respectively target data for Elektronik → Restauranger and Restauranger → Elektronik.*

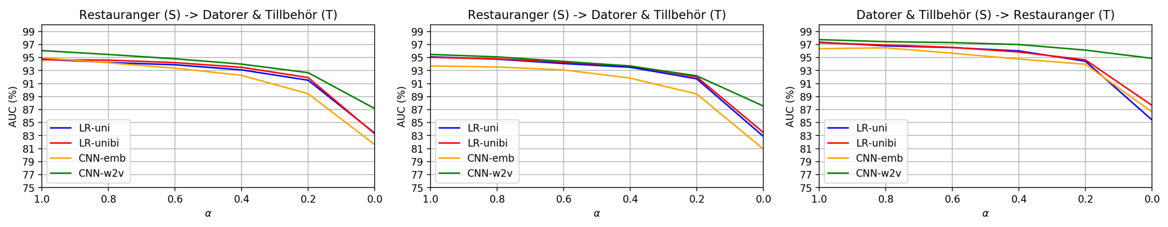


Figure 27: *AUC score for different amount of source respectively target data for Restauranger → Datorer & Tillbehör and Datorer & Tillbehör → Restauranger.*

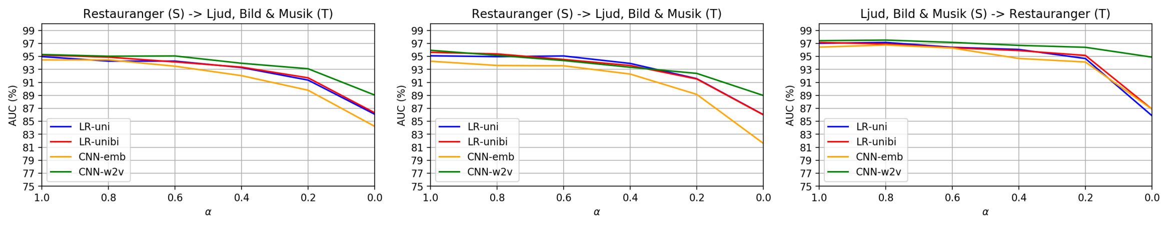


Figure 28: *AUC score for different amount of source respectively target data for Restauranger → Ljud, Bild & Musik and Ljud, Bild & Musik → Restauranger.*

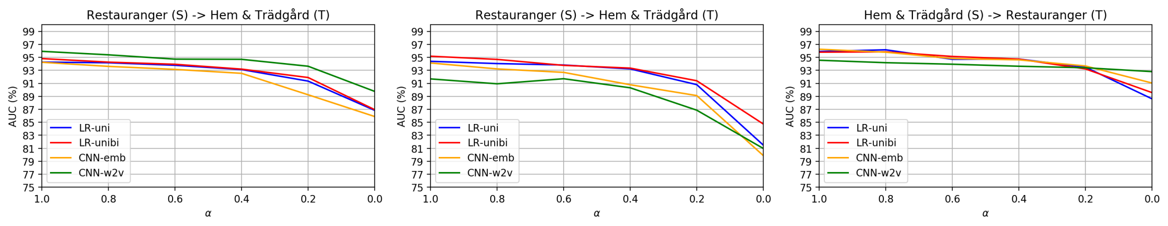


Figure 29: *AUC score for different amount of source respectively target data for Restauranger → Hem & Trädgård and Hem & Trädgård → Restauranger.*

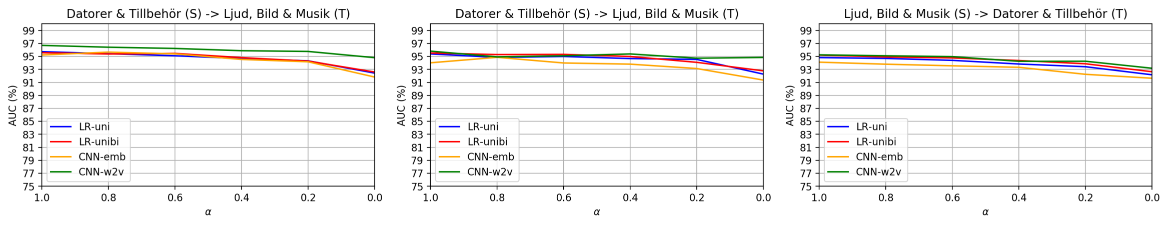


Figure 30: *AUC score for different amount of source respectively target data for Datorer & Tillbehör → Ljud, Bild & Musik and Ljud, Bild & Musik → Datorer & Tillbehör.*

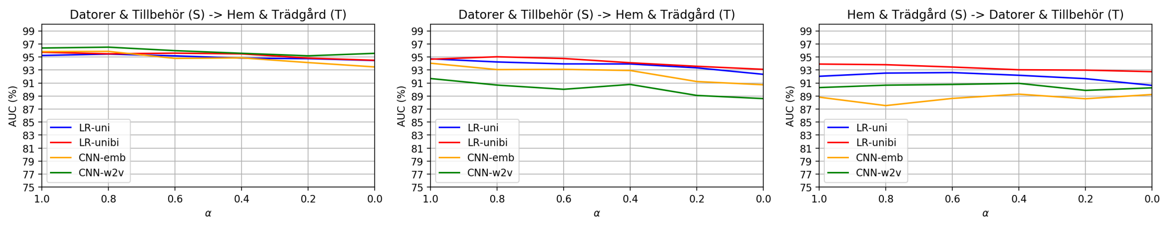


Figure 31: *AUC score for different amount of source respectively target data for Datorer & Tillbehör → Hem & Trädgård and Hem & Trädgård → Datorer & Tillbehör.*

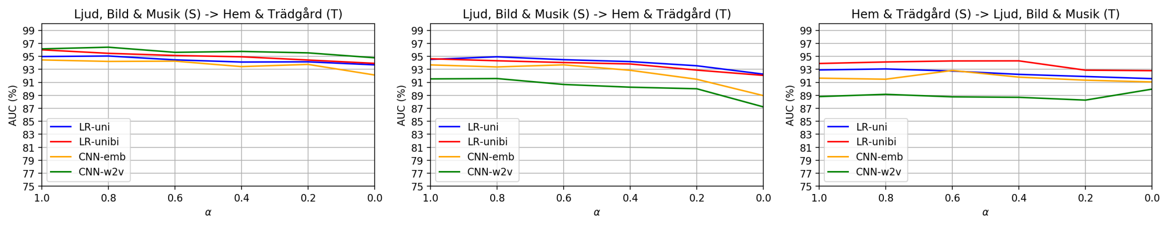


Figure 32: *AUC score for different amount of source respectively target data for Ljud, Bild & Musik → Hem & Trädgård and Hem & Trädgård → Ljud, Bild & Musik.*

6.3 Domain word characteristics

Table 9 displays the amount of unique words a given data domain contain along with the intersection of words across domains. As an example, Datorer & Tillbehör contain 54941 unique words and Hem & Trädgård contain 31328 words and the two domains have 14141 unique words in common.

Table 9: Unique and overlapping words across domains.

Data	**D&T**	**LB&M**	**H&T**	**E**	**R**
D&T	54941	21079	14141	13742	10562
LB&M	-	51663	13525	13621	10879
H&T	-	-	31328	10887	9298
E	-	-	-	43619	9670
R	-	-	-	-	31310

Table 10 displays the top 20 frequent words in each data domain where stop words are not considered. The table is ordered in a descending order where the most frequent words are at the top of the table.

Table 10: Top 20 frequent words in investigated domains.

D&T	**LB&M**	**H&T**	**E**	**R**
fungerar	ljud	väldigt	snabb	mat
väldigt	ljudet	riktigt	leverans	trevlig
problem	riktigt	kaffe	snabbt	service
riktigt	väldigt	köpte	nöjd	personal
köpte	lurar	maskinen	priser	maten
nöjd	nöjd	fungerar	service	trevligt
funkar	bas	nöjd	pris	bemötande
skärmen	låter	maskin	beställde	väldigt
tyst	par	problem	beställa	miljö
perfekt	högtalare	ca	snabba	personlaen
känns	musik	känns	varan	fantastisk
skärm	sitter	använda	smidigt	mysig
priset	fungerar	snygg	handla	ställe
datorn	köpte	tyst	väldigt	bord
använder	tv	tycker	beställning	riktigt
snabb	priset	köpa	produkter	restaurang
köp	lurarna	perfekt	produkt	atmosfär
ca	funkar	köp	problem	åt
enda	problem	enda	leveranser	fantastiskt
spel	känns	vatten	hitta	tillbaka

6.4 Misclassification statistics

Table 11 displays some statistics for the CNN model with pre-trained word embedding using a word2vec model. The statistics include the proportion of False Negative (FN) and False Positive (FP) misclassification along with the average review text length, *Rev. len.*, for $\alpha = 1$ and $\alpha = 0$. Column, *Test length*, to the far right displays the average review text length for the test set. The review text length are length in terms of number of words. All the data in Table 11 are averages over $K = 5$ runs and with a threshold equal to 0.5.

Table 11: Statistics over misclassification

Dataset	$\alpha = 1$			$\alpha = 0$			Test
	FN	**FP**	**Rev. len.**	**FN**	**FP**	**Rev. len.**	**length**
E→R	34%	66%	46	12%	88%	47	24
E→D&T	29%	71%	85	67%	33%	123	59
E→LB&M	29%	71%	102	73.5%	26.5%	171	67
E→H&T	30%	60%	90	65%	35%	121	73
R→E	43%	57%	36	67.6%	32.5%	31	17
R→D&T	37.5%	62.5%	84	89.5%	10.5%	101	59
R→LB&M	24%	76%	98	90%	10%	137	66
R→H&T	42%	58%	93	82%	18%	123	75
D&T→E	40%	60%	34	48%	52%	35	17
D&T→R	33%	67%	45	16%	84%	47	24
D&T→LB&M	24%	76%	96	21.5%	78.5%	93	65
D&T→H&T	40%	60%	98	21.5%	78.5%	96	74
LB&M→E	50%	50%	34	52%	48%	36	17
LB&M→R	33%	67%	45	9%	91%	51	25
LB&M→D&T	30%	70%	86	37%	63%	84	57
LB&M→H&T	37%	63%	89	23%	77%	89	74
H&T→E	34.5%	65.5%	35	73.5%	26.5%	33	16
H&T→R	31.5%	68.5%	43	27%	73%	40	24
H&T→D&T	28.5%	71.5%	88	70%	30%	84	61
H&T→LB&M	21.5%	78.5%	92	64.5%	35.5%	99	66

Table 12 displays a few example reviews where the CNN-model with pre-trained word2vec embedding strongly misclassified. A value close to 1 in the *Pred*. column indicate that the model believes the review to be positive while a value closer to 0 indicate that the model predict the review to be negative. The *Rating* column displays the original review rating. The *Category* column indicates whether it was an in-domain or out-domain classification. A single category in the column meaning in-domain while $S \rightarrow T$ meaning out-domain.

Table 12: Misclassification examples.

Review text	**Rating**	**Pred.**	**Category**
Fantastiskt lätt att installera. Bra med möjlighet att koppla in usbdisk. Snygg och liten och fungerar väldigt bra. Update; Tappar anslutning med ca 30 minuters intervaller. Samma uppsättning som tidigare och problemet har börjat för några månader sen.	2/10	0.977	D&T

Har älskat den här plattan, mycket bra kvalitet och allt fungerade perfekt så jag skaffade en till. En månad senare var båda sönder, en med trasig usb port och en med skärmen spräckt pga lekande barn. Allt frid och fröjd in med dom på reparation tänkte jag. Där har jag nu efter flera månaders väntan fått veta att det inte finns några reservdelar till dessa. Hur Google och ASUS kan sätta sina namn på en produkt som kännetecknas av sådant slöseri och slit och släng är för mig en gåta. MYCKET Besviken och kommer inte att köpa fler tablets från dessa tillverkare. Om du köper se till att packa in den i ordentligt med skydd för du har inga extra chanser att laga den senare.	1/10	0.974	D&T
Makalös prestanda, datorn startar på ett klick, alla program som ska ladda är klarladdade efter två sekunder. Min solid statare gick tyvärr sönder. Efter en vecka ungefär så började konstiga frysningar uppträda lite då och då. När den användes kunde allting bara avstanna i fem till tio sekunder för att sedan hosta igång som inget har hänt. Synnerligen irriterande vid spelande. Har du liknande problem, skicka tillbaks den. Läste en del på nätet och visade sig att rätt många har problem med sina SSD. Så jag väntar nog ett halvår till innan jag köper en ny!	8/10	0.026	D&T
Bra och relativt enkla inställnings funktioner. Stödjer flertalet ddns. Fick tyvärr problem med tappad anslutning sporadiskt. Ping mot ntp1.sp.se varierar 33-55-150 ms och tappade paket. Detta slår även på ip-tele, trots att den är inkopplad före routern. Bytte tillfälligt tillbaka till den gamla Linksys WRT-54GL med Tomato. Det kan vara ngn. inställning som jag missat, som påverkar detta. Köpte denna Asus p.g.a grabbarna i familjen klagade på dåliga svarstider i spel, och bra recensioner här. Men ngt i Asus försämrar ännu mer. Det märks även vid vanlig surf, då webbsidor tar lång tid att ladda ibland. Fw uppgraderad till 3.0.0.4.376.1071 efter köp. Det kanske finns en senare fw som är bättre. Tänkte kolla efter alternativ FW, tomato/merlin/dd-wrt. Update: Problemet med droppade paket är inte routern. Verkar vara modemet som bråkar och att Asus routern är känsligare än Linksysen för detta. I morse var linan väldigt dålig. Kollade direkt efter modemet, vilket gav 12% packet loss under 3-4 minuter. Höjer betyget 5 till 8.	8/10	0.0045	E→D&T

Efter ombyggnaden har menyn blivit katastrofalt dålig och dyr, ny personal och troligen nya ägare som gjort det sämre, det enda som blivit bättre är att lokalen är grymt fräsch men priserna blivit mkt dyrare och menyn väldigt liten,.. 100kr för 1 pizza är för mkt, visserligen har de mängdrabatt om du köper flera 2st = 150kr men kvaliten är dålig tyvärr... Före ombyggnaden sommaren 2010: Grymt goda pizzor och trevlig personal, lite högre pris men det är det verkligen värt, ibland ses kändisar äta här och restaurangen är så mysig att det är värt att ta med dejten om ni är hungriga, helt klart söders bästa pizzeria!	2/5	0.991	LB&M$\rightarrow$R
Mycket god mat men snorkig och otrevlig service vi kände oss förnedrade.	1/5	0.997	R

7 Discussion

7.1 AUC loss and transferability

The AUC loss plots, Figure 21 and 22, displays an overview of the transferability across domains. Since the AUC loss is calculated as the difference of AUC for in-domain compared to out-domain classification it gives a clear overview of how the models perform in the extreme cases where all target data is used compared to when no target data is used for training. By inspecting the AUC loss plots and comparing the baseline with both unigram and unigram and bigram text representation to the CNN with standard text embedding one can see that the CNN perform better in only 4 out of 20 cases. Judging from these results one can conclude that the CNN used in this thesis is not that well suited for transfer learning in the extreme cases since the more traditional logistic regression model performs equal or even better in most of the cases. Also, by inspecting the more detailed plots in Figure 23 to Figure 32 both the baseline and standard CNN are somewhat close in terms of absolute AUC score for in-domain classification. The CNN with standard word embedding is however performing better than the baselines in 6 out of 10 cases where $\alpha = 1$ for experiment I. This is the settings that probably would be used in a real life system where all data is utilized, i.e. all source and target data is used for training. Comparing the two experiment for $\alpha = 1$ for the CNN models reveals that adding data from another domain in the training process can in many cases increase the overall performance by around 1-3% instead of just training on in-domain data.

One can see in Figure 23 to Figure 32 that the AUC score decreases gradually as α decreases for the majority of the tests. This is somewhat intuitive and expected. What is interesting is however the fact that in many cases such as E→D&T, D&T→E, LB&M→E, E→R, R→E, R→D&T, D&T→R, R→LB&M, LB&M→R the AUC score drops significantly when α drops from 0.2 to 0. This is true for both CNN with standard text embedding and the baselines. This means that by just substituting a small amount of source data with target data in the training processes one can significantly improve the transferability performance. In some cases, such as R→H&T for the CNN with standard text embedding and unfixed data size (leftmost plot in Figure 29) the critical point where the performance drops is rather at $\alpha = 0.4$ compared to $\alpha = 0.2$ as in many other cases. Other, not that obvious, examples where the threshold is at $\alpha = 0.4$ or maybe $\alpha = 0.6$ are R→D&T for the unfixed data size, leftmost plot in Figure 27. Exactly where the threshold is differs slightly for the individual tests and having a smaller step size, h, could help pinpoint more precisely where that critical point is. What is clear is that utilizing target data in the training process increases the performance in almost every case.

By studying the model performance between intuitively related domains it is possible to see that both the CNN and the baseline model are more transferable in those cases compared to more distant domains, e.g. D&T→LB&M performs better than D&T→R, at least in the extreme cases as can be seen in Figure 21. However, this is not always the case. E and D&T can be seen as somewhat related but the transferability from E to D&T is rather poor, especially in the extreme cases where α is equal to 0. However, the other way around, D&T→E shows better performance. Similar results can be found for E→LB&M and LB&M→E. A possible cause for these results is most likely that the domain E is of the type company reviews while D&T and LB&M are product reviews. One can imagine that company reviews are more limited in their vocabulary, where words such as “snabb leverans”, “bra service”, “billigt och fraktfritt” can be applied to almost any kind of company regardless of their type of business. While product reviews have a richer vocabulary where every product can have more specific positives and negative attributes. The richness of the vocabulary can somewhat be confirmed when inspecting Table 9, both D&T and LB&M are much smaller datasets than E but contain several thousand more unique words than E.

Another similar example is that the transferability between D&T and R is better than the other way around, i.e. R→D&T. From Table 9 it can be seen that R reviews have a smaller vocabulary than D&T even though R is a slightly larger dataset. The R dataset also just contain restaurant reviews while D&T covers a wide range of subcategories as can be seen in Table 13 in Appendix A. Other categories that intuitively seem more distant is H&T and LB&M but the AUC score for LB&M→H&T do not drop that much as α decreases, which at first seem strange. H&T→D&T and H&T→LB&M also reveals high transferability, in fact, the AUC score actually increases as α decreases. By inspecting the subcategories of H&T, Table 13 in Appendix A, it reveals that it is a quite diverse superior category containing review about everything from $kitchen\ appliance$, $cleaning$, *smart homes* to $tools$, $food$ and $interior\ design$. Also, by comparing the top 20 frequent words in Table 10 for D&T and H&T it is possible to see that they share 12 out of 20 words. This is the highest number of words any two domains share when considering their top 20 frequent words. Also, LB&M and H&T share 7 words and D&T and LB&M share 9 words while D&T and E share only 4. So, H&T might be closer related to D&T and LB&M than one might intuitively think when just considering their domain names.

Another reason that the Experiments with H&T produces rather different results compared to the other tests might be that it is by far the smallest of the investigated datasets. The most notable results is as mentioned in the previous section H&T→D&T and H&T→LB&M where the AUC increases as α decreases. That behavior cannot be seen for H&T→E and H&T→R. A possible reason for that is when using H&T as source in Experiment II the larger dataset is truncated. The D&T and LB&M datasets might lose vital information in that step while the E and R datasets might be more robust and undiversified. This should however be addressed since the experiments are performed in a cross-validation fashioned. Another notable result is that the baselines actually outperform both CNN models for H&T→D&T and H&T→LB&M in terms of absolute AUC score and in-domain classification. Once again, this might have to do with the dataset size since a deep neural network often require a lot of data to perform well. The size of the H&T dataset might be more suitable for the traditional logistic regression model for the more diverse product review datasets. Although the AUC loss in the extreme cases when using H&T as source is lower than for many other domains, by inspecting the actual AUC score in the rightmost plots of Figure 31, 32 one can see that the in-domain AUC score is worse compared to when using one of the other datasets as source. This is clearly a data size issue. Comparing the rightmost plot of Figure 31 with the middle plot of Figure 23 one can see the obvious difference in AUC score for $\alpha = 1$, i.e. in-domain classification. In both these cases the models are only using D&T data but for the latter case the whole D&T set is used while for the first case only D&T data in the size of H&T is used for training.

Some difference in transferability can be seen between experiment I and II, i.e. keeping just the source domain constant for all α versus keeping the total training size constant. The difference is quite subtle, at least for datasets that are in the range of 20 - 45 thousand reviews. The dataset size seem however to be of greater importance when comparing the two experiment for a smaller target dataset as mentioned in the previous discussion. Comparing the experiments with fixed and unfixed data size when H&T is the target domain, it is possible to see that the overall performance is higher when the source data size is kept constant and is in this case much larger than the target dataset. This is mainly the case for the CNN models when inspecting the case where $\alpha = 1$, i.e. the full source and target datasets are used for training. This further strengthen the fact that deep neural networks are more data size dependent compared to more traditional machine learning models. Comparing the two experiments AUC score for $\alpha = 0$, i.e. only source data is used for training but the data size differs between the experiments one can see a significant difference where the AUC score for experiment I is higher for all models. Concrete examples can be seen in the leftmost and middle plots of Figure 25, 29, 31 and 32 where H&T acts as target, i.e E→H&T, R→H&T, D&T→H&T, LB&M→H&T. To fully understand if this is the case it would be necessary to do more test with a larger data size of H&T, that was however not possible since at the time of this research no more H&T data were available.

7.2 Text representation

The model that outperform all the other models in the absolute majority of the cases in terms of transferability is the CNN with unsupervised pre-trained word embedding using word2vec. This can be seen in the extreme cases in the AUC loss plots, Figure 21 and Figure 22. The CNN model with word2vec embedding is in most of the cases also performing best for in-domain classification. Using pre-trained word vectors can hence improve both in-domain as well as the transferability of a model significantly. In several cases such as D&T→E, D&T→R, LB&M→E, LB&M→R, the AUC loss are over 5% less compared to the CNN with standard word embedding and the baselines. The use of pre-trained word vector also increase the overall in-domain performance in most of the cases. Comparing to standard word embedding it is possible to see an in-domain AUC difference of 1-3% to the CNN with word2vec embedding's favor. However, in a few cases, where the smaller dataset H&T is used for the fixed data size experiments this model actually performs worst in terms of absolute in-domain AUC score and transferability.

The drawback of using pre-trained word vectors is the fact that training them require a lot of textual data. On the other hand, there exists already pre-trained word vectors free of use such as a word2vec model trained on millions of Google news articles in English. Since this thesis investigate Swedish reviews, such a pre-trained word vector would be of no use. The word2vec model trained in this thesis were trained with the same sort of data as used for the experiments, i.e. reviews. It is however possible to pre-train word vectors using other text corpus's such as a Wikipedia dump, news articles or Twitter tweets. Although, one can imagine that using the same type of data for the word2vec model as for the predictive model will yield better performance compared to training a word2vec model on arbitrary text data. Mainly because different words are used in different contexts and in different ways. This is however only speculations and tests would have to be conducted to confirm this claim.

Varying between unigram and bigram for the baseline seem not to have that great impact on the transferability. In the extreme cases unigram seem to be better suited for LB&M→E

and E→H&T while unigram and bigram seem to be a better fit for D&T→R and LB&M→R. When using the smaller dataset, H&T, it seems like unigram and bigram perform slightly better in most of the cases. It is difficult to pinpoint why that is and this might hence be easiest determined through testing and evaluation to see what works best for a particular domain or domains. Inspecting the AUC plots for varying α the two baseline curves follow each other closely more or less in every test. Using both unigram and unigram and bigram in this thesis was mainly to cover how logistic regression is commonly used for sentiment analysis.

7.3 Review level

The overall performance of the CNN with the pre-trained word embedding is rather high across domains, at least compared to the other models investigated in this thesis. It is however difficult to grasp what is really happening inside a deep neural network designed for sentiment analysis compared to a CNN dedicated for image recognition. In the latter case it is possible to extract what happens in each layer of the network, i.e. it is possible to step by step see how an image is processed and what features are identified by the network, layer by layer. When working with textual data it seem to be more complex.

Since it is difficult to know exactly how the CNN work the misclassified review were studied. From Table 11 it is possible to determine that in all in-domain cases the most difficult reviews to classify seem to be negative reviews, i.e. the model predicted positive when the actual sentiment is negative. This might have to do with the skewness of the datasets, i.e. the number of positive reviews are a lot more than the number of negative reviews. When instead inspecting the out-domain misclassification distribution the proportion of FP and FN is almost 50/50. So for transfer learning it seems like the model having equally hard to predict positive as negative reviews seen over all tests. There are however major differences between specific datasets and domains.

An overall observation valid for all the tests for the CNN with word2vec embedding is that longer reviews in terms of words are more difficult to classify, this is true both for in-domain as well as out-domain classification. In Table 11 it is possible to see that the average review text length for misclassified reviews for both $\alpha = 1$ and $\alpha = 0$ is longer compared to the average review text length of the actual test set (Table 11 rightmost column). A reason might be that longer reviews tend to contain both positive and negative aspects and it is hence difficult for the classifier to determine if the positives outweigh the negatives and vice versa.

A few examples of misclassified reviews where the model strongly believed a review was positive but turned out to be negative and vice versa can be seen in Table 12. These reviews are quite difficult even for a human to classify as positive or negative since they contain both positive and negative aspects. A few of the reviews were also updated later on making it even more difficult. These are just a few examples and no statistical conclusion can be drawn from this but it is interesting to see examples of what kind of reviews the model struggle to classify.

7.4 Rating as sentiment

The validity of using ratings strictly as an indicator of positive and negative reviews can be discussed. It is commonly used in previous research to use the review rating as a sentiment indicator [1], [8], [43], [57]. One can understand why this is so broadly used. In machine learning it is often difficult to get hold of big labeled datasets, so using the review rating as sentiment indicator is an easy way to get a lot of freely available and labeled data. However, a lot of reviews are not only positive or negative, instead many contain both positive and negative aspects. A more valid approach might be to restrict negative and positive reviews to the minimum respectively maximum rating, i.e. 1/5 and 5/5 or 1/10 and 10/10. By doing that a lot of training data is however lost.

Intuitively or maybe ideally, a review with rating of 1/5 should contain merely negative aspects while a review with rating of 2/5 should contain some positive segments. The same reasoning can be done for rating 5/5 and 4/5 as well. The machine learning model should hence be able to determine if the positive outweigh the negatives and vice versa. So, the problem of restricting reviews to either positive or negative might just be the fact that reviews are labeled positive and negative and should rather be labeled as mostly positive and mostly negative. Another approach is instead of seeing it as a binary problem every rating can be associated with a label, something like negative, weakly negative, neutral, weakly positive, and positive. This will however most likely require more data so that each rating is well represented over all datasets. The amount of reviews with rating 1 and 2 (and 3) were often few for the datasets used in this research, hence this approach was not an option.

Another problem regarding review data can be that the actual ratings do not correspond to the review text that well. In [59] they compare reviewer rating and reader rating, i.e. they study the difference between how a reviewer rate its own review text with how a reader would rate that same review text. They mean that even though a reviewer consider the same product or service while writing the review text and giving the rating there might be a mismatch between the text and rating since the reviewer can leave out perceptrons or thoughts of the product or service in the review text but include it in the rating. Therefore, if a reader, that has not experienced the same thing as the reviewer, rate the review text solely on the review text, the text and rating might correlate more accurately. By comparing these two approaches and train a Naive Bayes classifier they find that reader ratings is a better measure compared to reviewer rating. This can be an important factor to keep in mind when dealing with review data and sentiment analysis. However, the advantage of easily access big corpus of annotated data is lost if reader rating should be considered over reviewer rating.

8 Conclusion

This study has shown that the CNN model with standard word embedding used in this thesis perform worse in terms of transferability in the extreme cases for a fixed data size compared to the baselines, i.e. a logistic regression model with unigram and unigram and bigram text representation, when trying to predict the sentiment of Swedish reviews across domains. The CNN model perform better in terms of transferability in the extreme case in only 4 out of 20 tests. Judging from these results one can conclude that the CNN model used in this thesis by itself is not that well suited for transfer learning in the extreme cases since it does not beat the baseline model. The CNN does however perform better than the baseline in 6 out of 10 cases when the full source and target data are used for training. This study has shown that instead of merely using one domain for training it is possible to boost the performance of a CNN model by adding data from another domain in the training phase.

Substituting a smaller amount of source data with target data for training can significantly increase the transferability performance across domains. In this thesis, substitution of 20% of the total source domain training set with target data has proven to increase the AUC score up to 7-8%. This is true for the majority of the tests done in this thesis both for the standard CNN model and the baselines. Another important factor is the training data size where a larger dataset size can increase both the overall as well as the transferability performance of especially the CNN models.

A fairly simple CNN with pre-trained word embedding like word2vec outperform the other models in almost every test case. This is true both for in-domain classification but foremost for the transferability aspect. The text representation has a great impact on the transferability of the model and using pre-trained word embedding is therefore recommended both for transfer learning as well as for in-domain sentiment analysis. The pre-training is done in an unsupervised fashioned and labeled data is hence not required for this step.

It is difficult to determine the relatedness of two domains by just looking at what categories they belong to. In this thesis it seems like product reviews from different categories are more related compared to say product and company reviews within the same area. Other aspects that might better describe two domains relatedness seem to be the intersection of total words and frequently used words across domains. It is hence important to study the characteristics of the datasets when dealing with sentiment analysis and transfer learning.

9 Future work

In this thesis the best overall performing model in terms of transferability and in-domain sentiment classification is a CNN with pre-trained word embedding using word2vec. The word2vec model is trained on review type data and it would be interesting to see how the performance differs if the word2vec model is instead trained with arbitrarily text data such as a Wikipedia corpus.

A lot of focus in this research was to investigate transfer learning by combining different amount of source and target data for training. Another focus would be to investigate the maximum transferability achieved by training a deep learning network with a large amount of data from different source domains and fine tune some parts such as the last fully connected layer of the network with a smaller amount of target data. A further extension to this research would be to add more levels of sentiment such as neutral, weakly and strongly positive and negative sentiment instead of just grouping reviews into either positive or negative sentiment. Although, more data would most likely be required. This might however help to capture the more ambiguous reviews that a binary classifier struggle to classify.

References

[1] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World wide web*, pages 751–760. ACM, 2010.

[2] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[3] Cícero Nogueira Dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78, 2014.

[4] Houshmand Shirani-Mehr. Applications of deep learning to sentiment analysis of movie reviews. Technical report, Technical report, Stanford University, 2014.

[5] Avrim L Blum and Pat Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997.

[6] Dario Stojanovski, Gjorgji Strezoski, Gjorgji Madjarov, and Ivica Dimitrovski. Twitter sentiment analysis using deep convolutional neural network. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 726–737. Springer, 2015.

[7] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 10(3):1–145, 2016.

[8] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

[9] Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham Taylor, and David W Aha. Unsupervised and transfer learning challenge. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 793–800. IEEE, 2011.

[10] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*, 2016.

[11] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[12] Andries P Engelbrecht. *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[13] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[14] Pavel Golik, Patrick Doetsch, and Hermann Ney. Cross-entropy vs. squared error training: a theoretical and experimental comparison. In *Interspeech*, volume 13, pages 1756–1760, 2013.

[15] James D. McCaffrey. Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training, 2013. Available: `https://jamesmccaffrey.wordpress.com/2013/11/05/why-you-should-use-cross-entropy-error-instead-of-classification-error-or-mean-squared-error-for-neural-network-classifier-training/`, [Online; accessed 18-October-2017].

[16] Matt Mazur. A step by step backpropagation example, 2015. Available: `https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/`, [Online; accessed 16-September-2017].

[17] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[18] Stanford University. Cs231n — convolutional neural networks for visual recognition. Available: `http://cs231n.github.io/convolutional-networks/`, [Online; accessed 17-September-2017].

[19] Aliaksei Severyn and Alessandro Moschitti. Twitter sentiment analysis with deep convolutional neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 959–962. ACM, 2015.

[20] Denny Britz. Understanding convolutional neural networks for nlp. Available: `http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/`, [Online; accessed 17-September-2017].

[21] Michael Nielsen. Chapter 6, deep learning — neural networks and deep learning, 2017. Available: `http://neuralnetworksanddeeplearning.com/chap6.html`, [Online; accessed 17-September-2017].

[22] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958, 2014.

[23] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Michael A Babyak. What you see may not be what you get: a brief, nontechnical introduction to overfitting in regression-type models. *Psychosomatic medicine*, 66(3):411–421, 2004.

[25] L Graning, Yaochu Jin, and Bernhard Sendhoff. Generalization improvement in multi-objective learning. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 4839–4846. IEEE, 2006.

[26] Bing Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

[27] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[28] Wikipedia. Bag-of-words model — wikipedia, the free encyclopedia, 2017. Available: `https://en.wikipedia.org/w/index.php?title=Bag-of-words_model&oldid=783273254`, [Online; accessed 5-September-2017].

[29] Thorsten Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Technical report, Carnegie-mellon univ pittsburgh pa dept of computer science, 1996.

[30] Jason Brownlee. Why one-hot encode data in machine learning?, 2017. Available: `https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/`, [Online; accessed 24-October-2017].

[31] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[32] Keras. Keras: The python deep learning library - embedding. Available: `https://keras.io/layers/embeddings//`, [Online; accessed 30-October-2017].

[33] Andrew P Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern recognition*, 30(7):1145–1159, 1997.

[34] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437, 2009.

[35] Shir Meir Lador. What metrics should be used for evaluating a model on an imbalanced data set? (precision + recall or roc=tpr+fpr). Available: `https://towardsdatascience.com/what-metrics-should-we-use-on-imbalanced-data-set-precision-recall-roc-e2e79252aeba`, [Online; accessed 11-October-2017].

[36] Debra A Gusnard and Marcus E Raichle. Searching for a baseline: functional imaging and the resting human brain. *Nature Reviews Neuroscience*, 2(10):685–694, 2001.

[37] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(2009):12, 2009.

[38] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[39] Andrew Ng. Cs229 lecture notes - logistic regression. *Stanford University*, 2012. Available: `http://cs229.stanford.edu/notes/cs229-notes1.pdf`, [Online; accessed 27-November-2017].

[40] Nikhil Garg and Arjun Seshadri. Transfer learning: The impact of test set word vectors, with applications to political tweets.

[41] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[42] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. *Shape, contour and grouping in computer vision*, pages 823–823, 1999.

[43] Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. *arXiv preprint arXiv:1206.4683*, 2012.

[44] Luiza Sayfullina, Eric Malmi, Yiping Liao, and Alex Jung. Domain adaptation for resume classification using convolutional neural networks. *arXiv preprint arXiv:1707.05576*, 2017.

[45] Python.org. What is python? executive summary. Available: `https://www.python.org/doc/essays/blurb/`, [Online; accessed 26-October-2017].

[46] Python FAQ. General python faq. Available: `https://docs.python.org/2/faq/general.html`, [Online; accessed 26-October-2017].

[47] Scrapy. A fast and powerful scraping and web crawling framework. Available: `https://scrapy.org/`, [Online; accessed 26-October-2017].

[48] IPython. Interactive computing. Available: `https://ipython.org/`, [Online; accessed 26-October-2017].

[49] Jupyter Notebook. The jupyter notebook. Available: `https://scrapy.org/`, [Online; accessed 26-October-2017].

[50] Pandas. Python data analysis library. Available: `http://pandas.pydata.org/`, [Online; accessed 26-October-2017].

[51] Matplotlib. Python plotting. Available: `https://matplotlib.org/`, [Online; accessed 26-October-2017].

[52] Scikit-learn. Machine learning in python, 2017. Available: `http://scikit-learn.org/`, [Online; accessed 26-October-2017].

[53] Tensorflow. An open-source software library for machine intelligence. Available: `https://www.tensorflow.org/`, [Online; accessed 26-October-2017].

[54] Keras. Keras: The python deep learning library. Available: `https://keras.io/`, [Online; accessed 26-October-2017].

[55] Gensim. Topic modelling for humans. Available: `https://radimrehurek.com/gensim/about.html`, [Online; accessed 26-October-2017].

[56] PostgreSQL. The world's most advanced open source database. Available: `https://www.postgresql.org/about/`, [Online; accessed 30-October-2017].

[57] John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL*, volume 7, pages 440–447, 2007.

[58] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. Liblinear: A library for large linear classification. *Journal of machine learning research*, 9(Aug):1871–1874, 2008.

[59] Isa Maks and Piek Vossen. Sentiment analysis of reviews: Should we analyze writer intentions or reader perceptions? In *RANLP*, pages 415–419, 2013.

Appendix A

Table 13: Investigated domains and their subcategories

Category	Subcategories	Amount
Datorer & Tillbehör	Datorkomponenter	9597
	Möss & tangentbord	3592
	Nätverk	3485
	Datorer	2073
	Kablar	1971
	Bildskärmar	1437
	Datortillbehör	839
	Lagrinsmedia	675
	Skrivare & skannrar	482
	Mjukvaror	114
	VR-glasögon	71
Ljud, Bild & Musik	Högtalare & hörlurar	11831
	Hifi- & hembiotillbehör	1578
	Förstärkare	1551
	Projektorer & dukar	1407
	Mediaspelare	1213
	TV	1044
	Stereodelar	854
	Hembiosystem & stereopaket	568
	PA-utrustning	482
	DVD-spelare & Blu-ray-spelare	323
	Musikinstrument	252
	Billjud	202
	Bärbart ljud	202
Hem & Trädgård	Kök	2450
	Städ & klädvård	850
	Vitvaror	567
	Inredning	554
	Smarta hem	486
	Trädgårdsredskap	479
	Verktyg	461
	Klimat & värme	393
	Grillar	236
	Livsmedel	166
	Till bilen	93
	Badrum	69
	Kontorsutrustning	35
	Djurfoder	33
Elektronik	Mobiltillbehör	17562
	Mobiltelefoner	8114
	Batterier & strömförsörjning	7836
	Ljud	2985
	TV & bildskärmar	2359
	Hushållsapparater	2337
	Elektroniska reserverlar & komponenter	1705

	Elartiklar	1050
	Foto & video	294
	Elektronikreparation	158
Restauranger	Restauranger	26089