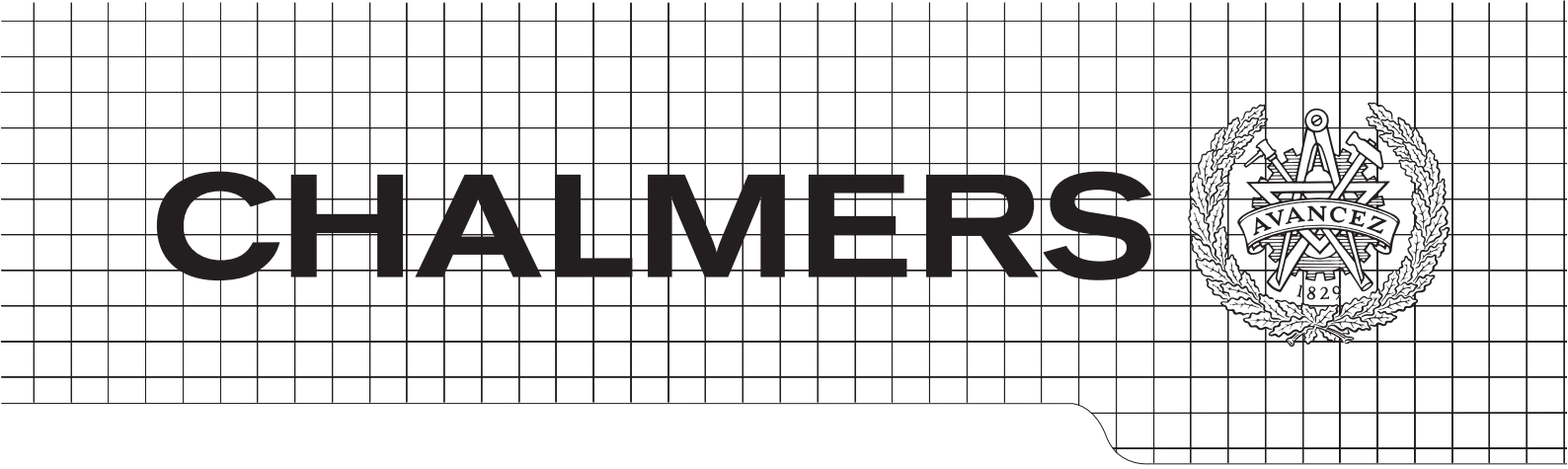

Machine Learning for On-line Advertising Using Contextual Information

Master of Science Theisis in the Programme Computer Science - Algorithms, Languages and Logic

BJÖRN BERNTSSON

Chalmers University of Technology
Department of Computer Science & Engineering
Göteborg, Sweden 2014

Machine Learning for On-line Advertising Using Contextual Information

BJÖRN BENRTSSON

Examiner: DEVDATT DUBHASHI

Chalmers University of Technology
Department of Computer Science & Engineering
SE-412 96 Göteborg Sweden
Telephone + 46 (0)31-772 1000

Abstract

This thesis considers different methods of utilising the contextual information on web-pages and ads in order to improve the fitting of a Bayesian Poisson model to historic data using L-BFGS. The data and optimization algorithm is provided by Admeta, an advertising optimization company that uses the model for click-rate predictions. The different methods tried to get added contextual information include categorization and developing different similarity measures between web-pages and ads using keywords. The similarity measures are based on WordNet, a large lexical database, and Word2Vec an open source tool that represents words as vectors. The categorization of web-pages gives good results as does some of the similarity measures. As WordNet is limited to the words found in its database Word2Vec is deemed more flexible and a superior source. For certain similarity measures it is shown that the click rate increases with the similarity. In the end using the average of the cosine distance between all keyword's vector pairs seams to give the best results among the different similarities tried for Word2Vec.

Keywords: Machine Learning, Poisson model, L-BFGS, Word2Vec, WordNet, Keyword similarity, Advertising

Acknowledgements

First I would like to thank the people at Admeta for the opportunity to work at their offices with their code and data as well as for a couple of fun go-cart races. I would also thank my supervisor at Chalmers, Devdatt Dubhashi, for many useful tips and inputs as well as Mikael Kågebäck for some much appreciated input on the use of Word2Vec. I should also mention the people at AlchemyAPI who gave me an academic access key to their API as well as the people behind WordNet, Word2Vec and its python implementation in Gensim. Last but not least I thank Knut Nordin my supervisor at Admeta who has given me a lot of his time and expertise.

Björn Berntsson, Borås 04/06 - 14

Contents

1

Introduction

Today statistical models are being used in many areas of science and business. Our increasing computing power and the vast availability of data has allowed these models to be applied in new contexts and become more and more complex.

1.1 Admeta

Admeta ABs product Private Ad Exchange is targeted towards on-line publishers (e.g. GP, Blocket, etc.). Advertisers provide ads they want displayed on the publishers websites. When a visitor to a web-page is shown an ad it is known as an impression. The visitor shown the advertisement then may click, or not click, on the ad and subsequently may, or may not, perform some type of action, such as a purchase, on the advertiser's web site. The publisher is paid by the advertiser per impression (CPM), per click on the ad (CPC) and/or per action on the advertisers website after clicking the ad (CPA).

For each impression, a real-time auction is performed between the available ads where the expected revenue for each participating ad is evaluated, given a number of covariates at hand. In order to calculate the expected revenue, a statistical model is utilized to predict the number of subsequent clicks and the number of each type of action. These predicted numbers are then combined with the CPM, CPC and CPA bids given by the advertiser to form the expected revenue.

The core of the Private Ad Exchange product is the ability to select the best ad for each impression, thereby maximizing revenue for the publisher. To achieve that, a statistical model with dozens of covariates and millions of parameters is used. The parameters are estimated from tens of billions of observations.

1.2 Contextual information in on-line advertising

Intuition would have it that displaying ads that are closely related to the content of websites should increase the probability of the ads being clicked as well as improve the user experience. Studies have shown that this is indeed the case [1, 2]. Ribeiro-Neto et al[3] examines a number of different ways of utilizing keywords to match web-pages and ads.

Researchers at Yahoo have done a lot of work in the area of contextual advertising. Broder et al [4] use classification as well as uni-grams and phrases to get a relevance measure between ads and web-pages. Chakrabarti et al[5] does similar work but also include user-click data to improve the relevance measure. Other big players in the area, such as Google, have no doubt done similar work.

1.3 Purpose

The aims and goals of this thesis are:

- To improve the accuracy of the predictions by using covariates based on contextual information extracted from web-pages and ads.
- To evaluate if covariates based on contextual information can be effective for new ads when the contextual parameters have been trained on other data.

1.4 Scope

While Admeta uses a large statistical model this work has been carried out with a simplified version containing only a few of the original covariates and a lot less parameters. Furthermore all tests are limited to data in Spanish from a single publisher. Spanish was chosen since Admeta's largest customers are Spanish and because some third party programs used in the thesis can handle Spanish. As stated before publishers are paid in CPM, CPC and/or CPA. This thesis will only look at optimizing CPC, or rather at improving click-rate predictions, since clicks on ads, while not frequent, are much more so than actions. The experiments performed in this thesis should be easy to apply for the CPA case as well, all that is needed is larger quantities of data.

1.5 Thesis outline

Section 2 introduces the problem to be studied in this thesis and explains various terms that will be used throughout. In section 3 the simplified statistical model used for training and testing is explained briefly. Following this section 4 introduces some techniques and tools that will be used to extract contextual information from ads and web-pages. Section 5 explains what work was carried out, what data was used and how the different techniques and tools were utilized. Section 6 presents the results of the tests described

in section 5 and section 7 discusses the result as well as possible work to be done in the future. Finally section 8 summarizes the thesis.

2

Problem definition

Few people today have been exempt from on-line advertisement. These can take the more traditional form of a picture with some text, but often they are short flash movies. Every such ad in Admeta's system is called a **material** and is given a unique id called the material id. On a single web-page there are usually many ads displayed. Every location where a material may be shown is called a **placement** and given an id called the placement id that is unique among placements on all websites utilizing Admeta's system. When an ad is displayed on a website this is called an **impression**. The outcome of an impression is either that the user clicks or ignores the ad. The **covariates** (material, placement, country, time, e.t.c.) of an impression together with the outcome of that same impression is called an **observation**.

Admeta's statistical model is used to find the likelihood that displaying a certain material will lead to the user clicking it. This is done by linking the covariates, via the model's parameters, to the expected number of clicks. The model have many so called **effects** which makes up the parameters of the model. Each effect is an attempt to model a part of the advertisements behaviour. In the simplified model there are **simple effects**, which map a single covariate to a parameter, and there are **interaction effects**. The interaction effects takes two covariates and map them to a single value showing if the two covariates work well together, have little or no interaction or counteract each other. These interaction effects would create a huge matrix with one parameter for every possible combination of the two covariates. As some covariates can take hundreds of thousands of different values that would be too large. Therefore this large matrix is instead approximated using a matrix factorization approach described in section 3.4.

The problem then is to extract information from web-pages and ads, somehow use this information as covariates and find effects that improve how well the model fits to test data after training it on other data. This should mean improved click-rate predictions which in turn means increased revenue.

3

Statistical Background

This section introduces the theory behind the statistical model used at Admeta. It is not meant to give a complete understanding of the theories used, but rather to give direction for further reading on the subjects. The methods used was implemented, evaluated and recommended by a masters student at Chalmers in 2012. Most of what is said in this section can be read about in greater detail in his thesis "Application of L-BFGS to Large-Scale Poisson MAP Estimate" [6].

3.1 Bayesian statistics

In Bayesian statistics a *prior* belief is assigned to the parameters before seeing any data and a *posterior* belief afterwards. These beliefs are represented using probability densities, the prior and posterior density.

Theorem 3.1: Bayes' formula. Let A, B be two random variables, and $f_{A,B}$ be their joint distribution

$$f_A(a|B=b) = \frac{f_B(b|A=a)f_A(a)}{f_B(b)} \tag{3.1}$$

Here $f_A(a|B=b)$ is the posterior belief of the parameters, $f_B(b|A=a)$ is the likelihood function, that is the probability of an outcome b given parameters a, $f_A(a)$ is the prior belief and $f_B(b)$ is simply the probability of the outcome b.

3.2 L-BFGS

To train the model to new data the limited memory BFGS update, L-BFGS, is used. It is an approximation of the Quasi-Newton method BFGS that uses less memory. L-BFGS was first introduced in [7] and it is considered to be one of the most successful

algorithms for large scale optimization[8]. More information on Quasi-Newton methods and L-BFGS in particular can be found in [6].

3.3 The statistical model

The simplified model used in this thesis is a GAM, or more precisely, a Bayesian additive Poisson model with log link. A GAM is a generalized additive model. The model attempts to relate a response variable Y to some predictive variables X, called covariates. The expectation of Y_i; μ_i, where i is a single observation, is linked to the covariates by a *link function*. This function is often the logarithm, $log(\mu_i) = X_i\theta$, the inverse, $\frac{1}{\mu_i} = X_i\theta$ or the logit $log(\frac{\mu_i}{1-\mu_i}) = X_i\theta$, where θ is the model parameters. Every effect in the model has a set of parameters. The choice of link function usually depends on the distributional assumption on Y_i. A common choice for similar applications is the Poisson distribution and it is used at Admeta.

$$Y \ f(Y_i|\theta, X_i) = \frac{\mu_i^{Y_i} e^{-\mu_i}}{Y_i!} \tag{3.2}$$

Therefore the logarithm was chosen as link function since it comes with some nice properties, including simplifying calculations.

A GAM can capture rather complex relations between Y and X but with this flexibility comes an increased risk of over-fitting. Over-fitting the model to old data can lead to poor predictions for new data. To counter this a Bayesian model is used. This means that the prior distributions assigned to the parameters act as regulators, reducing over-fitting.

The goal of the model is to predict the outcome of future impressions. To do this the model's parameters need to be fit to historic observations. As a Bayesian approach is used this means finding the posterior distribution of the parameters. Using Bayes formula we have,

$$f(\theta|Y,X) = \frac{f(Y|\theta,X)f(\theta|X)}{f(Y|X)} \propto f(Y|\theta,X)f(\theta) \tag{3.3}$$

where $f(Y|X)$ is constant with regard to θ and $f(\theta|X)$ is assumed to be equal to $f(\theta)$. The likelihood $f(Y|\theta,X)$ as said before is assumed to follow the Poisson distribution and $f(\theta)$ is assumed to follow the Normal distribution with mean μ and variance σ.

$$N(\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(\theta-\mu)^2}{2\sigma}} \tag{3.4}$$

The mean and variance of all the different parameters is set to 0 and 0.25 respectively in the simplified model, but they could also be trained to take the most likely values. The term $f(\theta)$ then is close to one for parameter values close to zero and it approaches zero as parameter values diverges from zero. When training the model the maximum a-posteriori estimate (MAP) of θ is found by applying L-BFGS (section 3.2) to the problem of maximizing $f(Y|\theta,X)f(\theta)$. This then is the θ that maximizes the posterior

distribution (equation 3.3) and, as it consists of the most likely parameters given past observations, it is believed to be good for future predictions.

3.4 Matrix approximation

In some cases matrices are to big to use for certain applications and then one needs to approximate them. There are many methods of doing this. Some well known ones are SVD (singular value decomposition) and Eigendecomposition. Another is the probabilistic matrix factorization (PMF) introduced by Salakhutdinov and Minh[9]. What is done at Admeta may not necessarily have a name but it is somewhat related to all three of these methods. Suppose we have a large matrix R with dimensions XxY. We represent this matrix by using two matrices, one for the X rows and one for the Y columns. The first matrix gets dimensions XxF, the other YxF. F is the number of factors we decide to use, the more we use the better the approximation gets but the longer it takes to construct the matrices. A value in the original matrix is retrieved by taking the inner product of the two rows corresponding to the X and Y values wanted.

The two matrices are constructed during training. We optimize one feature f at a time moving to the next feature once the first can not be improved. When we see an observation with covariates $x \in X$ and $y \in Y$ we calculate the error in prediction using the inner product of the rows and the outcome of the observation. Then feature f of row x is updated by the value of feature f in row y times the error and the learning-rate and wise versa for row y.

4

Contextual Background

This section introduces different methods of extracting contextual information from web-pages and texts used in this thesis.

4.1 Alchemy API

In order to use the textual information on a web-page all relevant text must be extracted from a html document. There are many services that does that but for this thesis the natural language processing service AlchemyAPI was chosen. In addition to extracting the text AlchemyAPI also uses machine learning techniques to find keywords, categorize the text and much more.

4.2 WordNet

WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms called synsets, each expressing a distinct concept. An example of a synset could be the concept described by the three words ”audacity”, ”audaciousness”, and ”temerity”. As a word can have many meanings it can also appear in many synsets, for instance the word ”dog” could refer to the animal dog, but it could also be used as an informal term for a man; ”You lucky dog”. Synsets are interlinked by means of conceptual-semantic and lexical relations. For instance this means that the animals ”cat” and ”dog” are closely related because they are both linked to the concept ”pet”. One good thing about this structure is that it allows for good translations because the concepts are translated rather than just individual words. While there is no translation of WordNet that covers all the synsetes and relations in the English original, there is a Spanish translation within the Multilingual Central Repository (MCR) that includes 38.500 words and covers about 21% of the around 117.000 English sysnsets. This is as of WordNet and MCR versions 3.0 which were used in this paper and can be retrieved from

[10]. For further information on WordNet the reader is referred to [11] and for more on MCR see [12].

The semantic relations between concepts in WordNet makes it well suited to infer similarities between words. Many different similarity measures have been tried, the simplest of which is the inverse path length, $1/pathlength$. Where the $pathlength$ is the number of synsets on the shortest path between the two words in question. Another measure was proposed by Leacock and Chodorow; $-log(pathlength/(2 * D))$, where D is the maximum depth of the taxonomy. Other measures take into account the depth of the two words, their information content and/or their least common subsumer. For an extensive list see [13].

4.3 Vector representation of words

Representations of words as vectors has a long history [14]. The idea is to give similar words similar vector representations and thus be able to infer more than is possible with simple frequency counts. Many different models have been used to train the vectors but one problem has been that these methods require very much data to be accurate and/or take a long time to train. To overcome these limitations researchers at Goggle have developed two new models that are simple but can be trained on a lot of data very fast. The two models are called the *Continuous Bag-of-Words Model* (CBOW) and the *Continuous Skip-gram Model* (Skip-gram)[15]. In the CBOW model we move through all words and learn to predict the current word based on the surrounding words. The surrounding words' vectors are updated based on the error in the prediction. The Skip gram model is very similar to CBOW, the difference is that it predicts the surrounding words based on the current word instead. Figure 4.1 is taken from [15] and gives a graphical representation of the CBOW and Skip gram models.

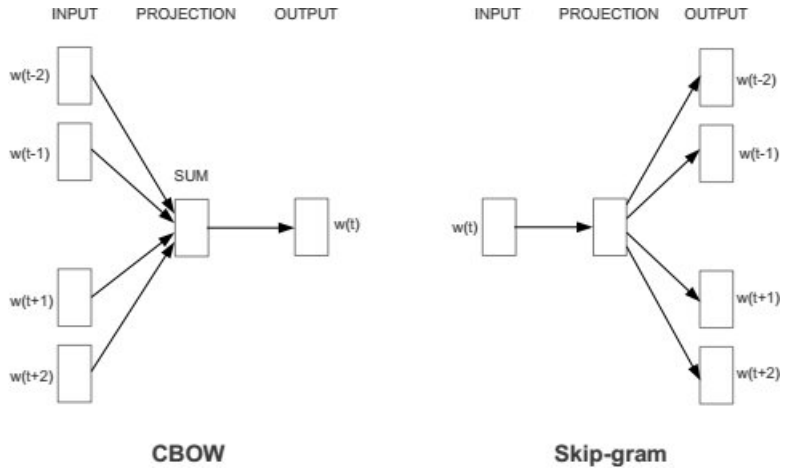

Figure 4.1: Two vector representation of words models. CBOW predicts the current word based on the context, and the Skip-gram predicts the context given the current word.

The Skip-gram model is trained by maximizing the objective function

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c<=i<=c\ i!=0} log(p(w_{t+i}|w_t)) \tag{4.1}$$

where T is the number of words in the training set, and c is the number of words before and after the current word t that is used in the training. The probability $p(w_{t+i}|w_t)$ is approximated using the hierarchical softmax first introduced by Bengio et al[16] and later evaluated in a paper by Morin and Bengio[17].

It has been shown that simple algebraic operations on the vectors can yield surprising results [15, 18, 19]. For instance taking the vectors of the words and performing the addition and subtraction: London - England + Sweden generates a vector very close to Stockholm. Doing the same for: aunt - girl + boy generates a vector close to uncle. Clearly the word-vectors can capture some rather subtle connections between words.

CBOW and Skip-gram have been used for many applications. In their paper Tomas Mikolov et al[20] use both for machine translation with good results and Mikael Kågebäck et al[21] use Skip-gram for document summarization. Both CBOW and Skip-gram have been released in an open source project called Word2Vec[22]. This code is implemented in C but there is a port in the python package Gensim[23] that have been used in this paper.

4.3.1 Cosine similarity

Similar words should appear with roughly the same words around them and as the vector of a word is defined by the vectors of the words around it the vectors can be used to infer a similarity between two words. To do this Word2Vec uses the cosine of the angle between the vectors of the words. This value for two vectors v_1 and v_2 is obtained by taking their dot product and dividing by their magnitudes as seen in equation 4.2.

$$Cos(v_1, v_2) = \frac{v_1 \cdot v_2}{||v_1||\ ||v_2||} \tag{4.2}$$

The resulting similarities range from 1, exactly the same, to -1, being very dissimilar. A cosine similarity of 0 should just mean that the words are not similar.

5

Method

In this section it will be described how the data to be used in training was acquired and how the techniques described in section 4 were used to add more contextual information to this data.

5.1 The training data

The model was trained and tested on historic data from Admetas database where they record every impression and click on Admetas customers ads. All the data was taken from a single day and a single publisher, only the placement with the most clicks on a web-page were kept. Clicks and impressions on materials with less than ten clicks was removed as was any impressions where the placement had less than ten clicks. Before this filtration there was around 18 700 clicks after there were around 8 100 clicks. Only the impressions relevant to the clicks was gathered from the database. Of the impressions that meet the criteria a random set of 10% was used. This was done in order to speed up calculations and should not affect the results except by always overestimating the expected number of clicks, but what is interesting is just whether or not the new covariates could improve predictions and these results should not be affected. The final dataset consist of a total of around 500 000 impressions, 8 100 of witch lead to clicks.

The data was then divided into two different training-testing pairs. The first pair was intended to be used for testing the first purpose: whether or not contextual data is useful for the predictions. To this end the data was divided randomly with 90% in the training data and 10% in the testing data. This dataset will be called the *90/10 dataset*. The intent of the second training-testing pair was to see if any of the contextual covariates have a large positive effect on learning when it comes to new materials not seen before. Instead of splitting the data randomly all observations of a few materials was put in the testing data. Thus the training is done on one set of materials and the testing on another. This second dataset will be referred to as the *newAds dataset*.

5.2 Model covariates

For the simplified model used as a baseline for this thesis only a few of the covariates used in Admeta's model are included. These are the material id and the placement id. In addition to these new covariates derived from contextual data are added.

To get the context from the web-pages AlchemyAPI described in section 4.1 was used. It categorized the pages into one of the following categories:

- arts and entertainment
- business
- computer and internet
- culture and politics
- gaming
- health
- law and crime
- religion
- recreation
- science and technology
- sports
- weather

The categories were given an id and used as a covariate. AlchemyAPI was also used to generate a number of keywords for every web-page. The materials was categorized by hand into the same categories as used by AlchemyAPI. Keywords describing the materials were also added by hand.

To utilize the keywords WordNet, described in section 4.2, was used to establish a similarity measure between materials and web-pages. For every keyword from a web-page its similarity values to all of a material's keywords was found by using breadth first search in the WordNet structure and using taking $1/pathlength$. These values was then used to form a few different similarity values between a keyword and a material. One similarity was achieved by taking the average of the different values, one by taking the maximum and a last one by taking the sum. These similarity values between keywords and materials was then used to find similarity values between web-pages and materials. For every keyword of a website we take it's three similarity values to the material in question. Then we again take the average, the maximum and the sum. Finally the values are scaled and rounded into an integer value. This process gives us 9 different similarity values between web-pages and materials, they will be called:

- WNavgAvg
- WNavgSum
- WNavgMax
- WNsumAvg
- WNsumSum
- WNsumMax
- WNmaxAvg
- WNmaxSum
- WNmaxMax

It should be noted that only 20% of the website keywords and 65% of the material keywords existed in WordNet, any other words were ignored. Some of the materials' keywords also had to be edited so as to be found in WordNet, mostly it was a matter of using the most basic versions of words.

In part because of WordNet's rather low word coverage and in part in order to have something else to compare with, it was decided to try another way of measuring similarity as well. Vector representations of words lends themselves well to expressing similarity between words so the tool Word2Vec described in section 4.3 was used. In order to use Word2Vec it needs to have a trained model. There are several already trained models as well as datasets to train on to be found on Word2Vec's google code site, but the problem is that all of them use English words only. Instead a Spanish dataset with 220 MB of news articles was found. After converting it to lower-case and removing all ".", ",", "?" and similar characters the dataset was used for training. The Skip-gram model was employed in the training because it has generally had better results than the CBOW model[15]. The training took between six and twelve hours depending on the dimensionality chosen for the word-vectors. In order to evaluate different dimensionalities a set of test questions was used. The questions was provided in English from [22] and translated to Spanish using Google translate. A word-vector dimensionality of 100, 200, 300 and 400 was tried and the best results was achieved with 200 so all results presented use that.

Using the vector size 200 a number of custom tests were performed to see how well they are handled. In table 5.1 the questions are on the form $word2 + word3 - word1$ the five words with vectors closest to the question are returned in order of similarity to the question and if the correct answer is among them it has been boldfaced and the less similar words removed. As an example look at the first row. Here we have the question $espaa + pars - madrid$ and the closest word was $alemania$ which is wrong, the second closest word was $francia$ which is the expected result so we do not look at the next word.

The similarity measure was also tested with a series of custom tests. In table 5.2 is shown two words and their similarity. These words have been translated to English

Question	**Answer**
madrid is to *españa* as *parís* is to	*alemania,* ***francia***
estocolmo is to *suecia* as *londres* is to	*dinamarca, lituania, francia, islandia, holanda*
hermana is to *hermano* as *madre* is to	***padre***
ronaldo is to *cristiano* as *elvis* is to	***presley***
messi is to *futbolista* as *woods* is to	***golfista***
playa is to *la* as *coche* is to	*de, un, su, una,* ***el***

Table 5.1: Questions put to the Word2Vec model by simple arithmetic operations on the word-vectors. Correct answers are boldfaced

for ease of reference. On the first row we have *clothes* and *technician* with a similarity of -0.062 which means no similarity just as expected. On the second row are *clothes* and *shoes* with a similarity of 0.45. As clothes and shoes are both something you wear some similarity is expected. All in all the tests in table 5.2 give reasonable results which indicate this is a good measure.

Word one	**Word two**	**Similarity**
clothes	technician	-0.062
clothes	shoes	0.45
footbaler	ronaldo	0.41
messi	ronaldo	0.51
cristiano	ronaldo	0.76
telephone	mail	0.52
messi	telephone	-0.028
messi	elvis	0.028
stockholm	gothenburg	0.61
paris	gothenburg	0.43
paris	stockholm	0.50

Table 5.2: The similarity of two words. The words tested were in Spanish but translated to English here for easy reference

Word2Vec found many more of the keywords than WordNet did as can be seen in figure 5.1. It should be noted that many of the keywords not found are bi-grams or tri-grams witch are not handled by either Word2Vec or WordNet.

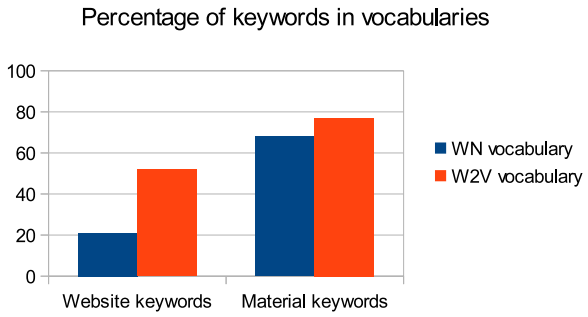

Figure 5.1: Shows how large percentage of the keywords exists in the vocabularies

Once Word2Vec was trained it's similarity measure, the cosine distance between two word vectors, was used in the same manner as WordNet's similarity measure was used. One difference is that WordNet gave a number between zero and one but Word2Vec gives a number between minus one and one so the scaling had to be done slightly differently to correctly map scores to parameter values. As with WordNet we call the different similarity values:

- W2VavgAvg
- W2VavgSum
- W2VavgMax
- W2VsumAvg
- W2VsumSum
- W2VsumMax
- W2VmaxAvg
- W2VmaxSum
- W2VmaxMax

The AlchemyAPI tool generated between one and one-hundred keywords for every web-page and sorted them according to relevance. This meant that maybe using all the keywords and giving them the same importance might not be the best solution. It was considered to use the weights provided by AlchemyAPI in the calculations, but in the interest of time the solution was to instead try to use only the top four words and treat them as equally important. In figure 5.2 it can be seen how many words are found in WordNet and Word2Vec respecivly for the websites. For Word2Vec only six of the fifty-six different websites had less than four keywords found while the number was eleven for WordNet.

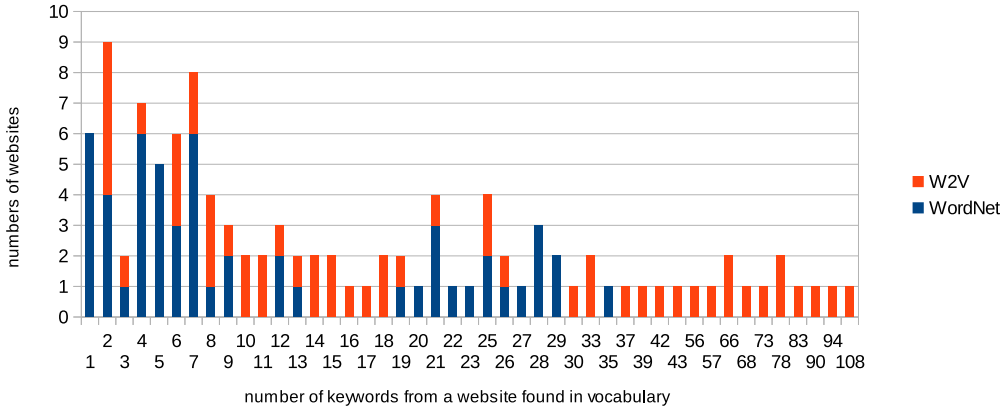

Figure 5.2: This graph shows how many of the websites contain a certain number of keywords from the vocabularies

5.3 Effects

The model use effects to model the behaviour of ads, described in section 2. The effects are based on the different covariates and here is a complete list of all effects that were tested in this thesis.

- **intercept** - the intercept effect is a single real value that does not depend any covariates instead it can be viewed as the average outcome of all impressions
- **placement** - from the placement id covariate
- **material** - from the material id covariate
- **material-placemet interaction** - interaction effect between material and placement ids
- **webCategory** - from the website category covariate
- **matCategory** - from the material category covariate
- **webCategory-matCategory interaction** - interaction effect between the two categories
- **webCategory-material interaction** - interaction effect website category and material
- **keywords** - an effect that combines the influence of all keywords in a website

- **similarities** - all the WordNet and Word2Vec covariates described earlier get their own effect with the same name as the covariate.
- **W2VavgAvg-webCategory interaction** - interaction effect between a similarity covariate and the website category
- **W2VavgAvg-material interaction** - interaction effect between a similarity covariate and the material
- **W2VavgAvg-placement interaction** - interaction effect between a similarity covariate and the placement

6

Results

The results from training the model using the different effects described in section 5.3 will be discussed in the following section. We will then take a closer look at the parameters of the similarity measures and investigate possible interactions between similarity and other covariates. These tests all use the *10/90 dataset*. Finally the tests performed on the *newAds dataset* will be presented.

6.1 The fit of the model

To have a baseline to compare against all tests used the material and placement effects. The results from training can be seen in tables 6.1 to 6.4. The higher the log-likelihood is the better the model fits the test data. As can be seen in table 6.1 the best result was achieved by using the keywords effect. The website category effect also yielded good results while the material category effect was more or less useless. The interaction between the two categories were very similar to simply using the website category. Indeed, if we use the website category and the category interaction the interaction effect's parameters become very small and have almost no effect on the log-likelihood.

In table 6.2 the results of the different similarity effects are presented. Effects based on Word2Vec generally performs slightly better than those based on WordNet with the exceptions of WNSumAvg and W2VMaxSum. For Word2Vec the best result is achieved by W2VavgAvg.

Finally a test using many of the best effects were performed. With the effects *material, placement, material-placement, webCategory, webCategory-material, W2VavgAvg and W2VavgAvg-material* the log-likelihood $-3526{,}5$ was reached.

Effects	Log-likelihood of test data
No added context	-3583,5
material-placement interaction	-3580,9
webCategory	-3539,1
webCategory-material interaction	-3535,4
webCategory-material interaction and webCategory	-3535,8
matCategory	-3583,4
webCategory-matCategory interaction	-3538,2
webCategory-matCategory interaction and webCategory	-3539,1
keywords	-3528,6

Table 6.1: The results of training using different effects derived from contextual data. All tests also use the material and placement simple effects.

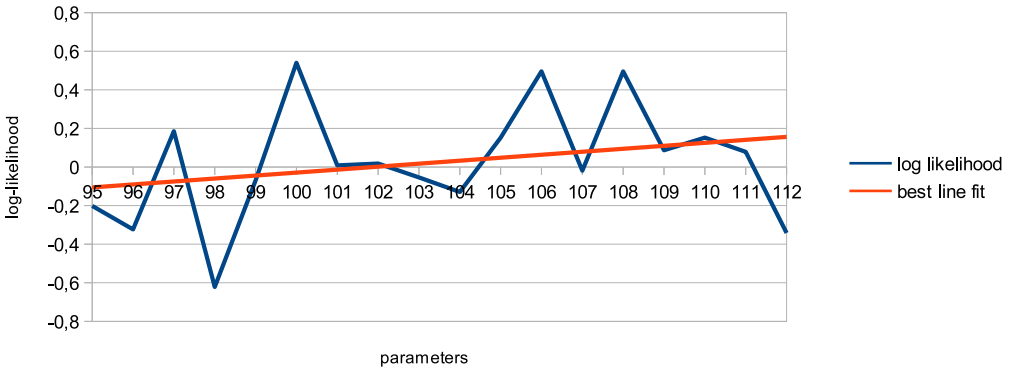

Figure 6.1: The values of the parameters of the effect W2VavgAvg using all keywords

6.2 Investigating similarity

The different versions of the Word2Vec similarity effect all have a small positive influence, except for W2VmaxSum in table 6.2 and W2VsumSum in table 6.3. The best result was achieved when using W2VavgAvg with all keywords and W2VavgSum, W2VavgAvg and W2VsumAvg with only the top 4 keywords. Looking at the parameters of these effects we can see in figures 6.1, 6.2, 6.3 and 6.4 that they all have a positive trend, meaning that as the similarity value increases so does the likelihood of clicks. But the other versions of the similarity effects all have smaller positive results and their parameters results in negative slopes as seen in figures 6.6, 6.5 and 6.7.

Effects	**Log-likelihood of test data**
No added context	-3583,5
WNavgAvg	-3561,0
WNmaxAvg	-3569,6
WNsumAvg	-3540,3
WNavgSum	-3585,3
WNmaxSum	-3578,8
WNsumSum	-3577,1
WNavgMax	-3576,0
WNmaxMax	-3571,6
WNsumMax	-3558,7
W2VavgAvg	-3557,3
W2VsumAvg	-3573,6
W2VavgSum	-3563,1
W2VmaxSum	-3631,1
W2VsumSum	-3570,0
W2VsumMax	-3558,3

Table 6.2: The results of training using different similarity effects derived from all keyword pairs. All tests also use the material and placement simple effects.

Effects	**Log-likelihood of test data**
No added context	-3583,5
W2VavgMax	-3564,6
W2VsumSum	-3582,1
W2VavgSum	-3545,3
W2VavgAvg	-3557,6
W2VsumAvg	-3548,3

Table 6.3: The results of training using different similarity effects using only the top 4 keywords from the web-page (and all from the material). All tests also use the material and placement simple effects.

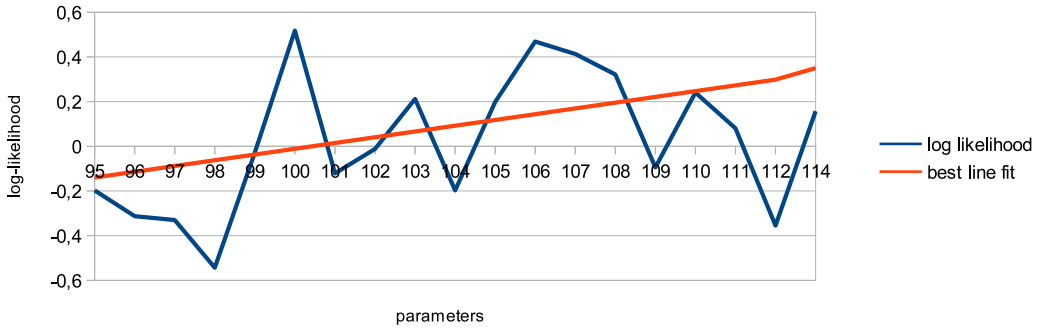

Figure 6.2: The values of the parameters of the effect W2VavgAvg using top four keywords

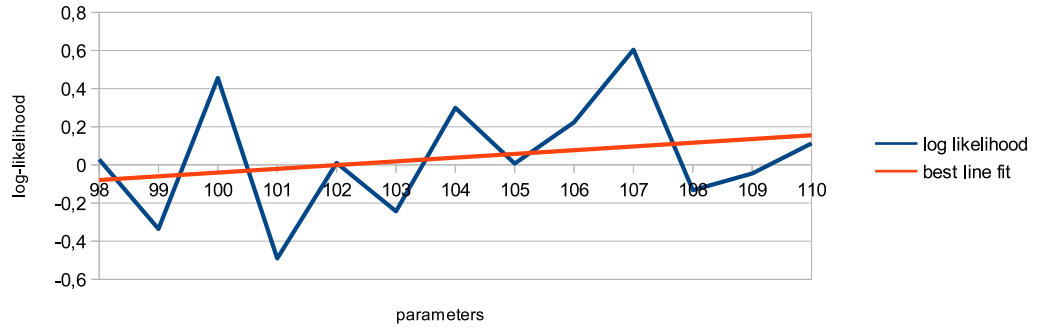

Figure 6.3: The values of the parameters of the effect W2VavgSum using top four keywords

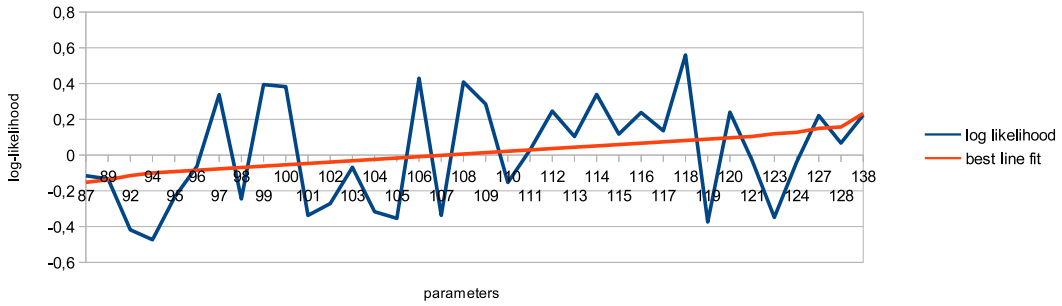

Figure 6.4: The values of the parameters of the effect W2VsumAvg using top four keywords

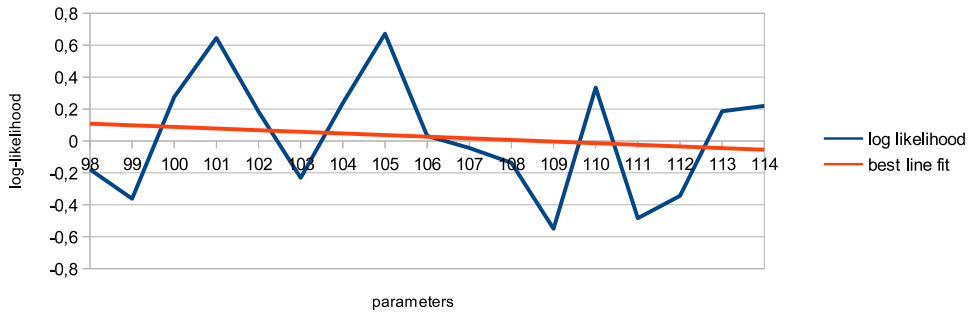

Figure 6.5: The values of the parameters of the effect W2VavgMax using top four keywords

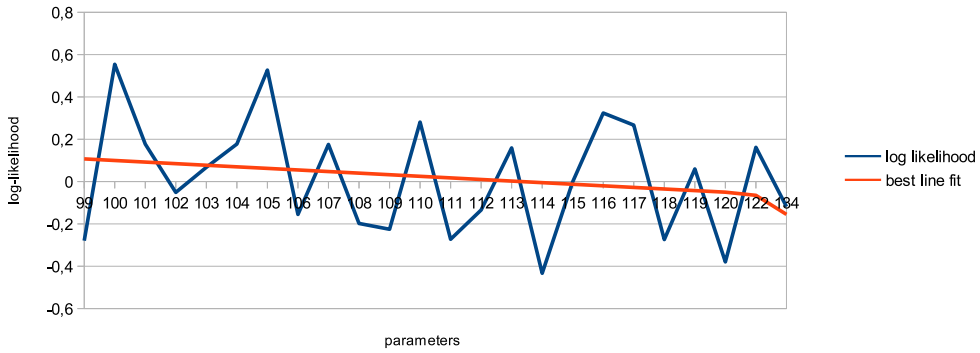

Figure 6.6: The values of the parameters of the effect W2VsumSum using all keywords

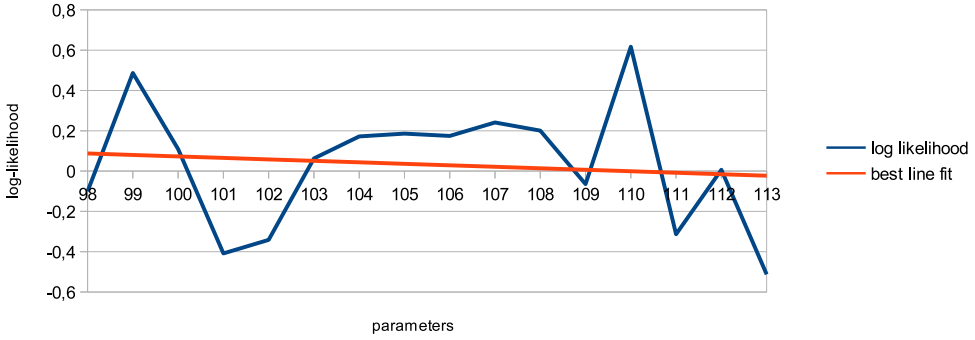

Figure 6.7: The values of the parameters of the effect W2VsumSum using top four keywords

6.2.1 Similarity interactions

It is also interesting to see if the similarity interacts with other covariates looking at table 6.4 we can see how the model is fitted when using different interactions. Only similarity interactions with W2VavgAvg was tested.

Effects	**Log-likelihood of test data**
W2VavgAvg and W2VavgAvg-material interaction	-3543,2
W2VavgAvg and W2VavgAvg-placement interaction	-3558,0
W2VavgAvg and webCategory	-3536,3
W2VavgAvg, webCategory and webCategory-W2VavgAvg interaction	-3536,7

Table 6.4: The results of interactions between W2VavgAvg and other covariates. All tests also use the material and placement simple effects.

The interaction W2VavgAvg-placement seems to have little or no added benefit compared to simply using W2VavgAvg the same can be said for W2VavgAvg-webCategory. W2VavgAvg-material on the other hand does improve the results slightly, this could indicate that not all materials are equally dependant on their similarity to the web-page. We look again at the parameters of the W2VavgAvg effect but this time when training together with the interaction effect W2VavgAvg-material, figure 6.9. Comparing this with the parameters for the W2VavgAvg effect without the interaction (figure 6.2, repeated in figure 6.8 for convenience), we can see that the best line fit is less steep in 6.9 but also that the average error of the line is smaller. The largest differences are found among the low parameters that are now less extreme. In figure 6.10 we can see the parameters of W2VavgAvg again, but this time with many other effects used in the training. As can be seen the parameters follow the best line fit better still.

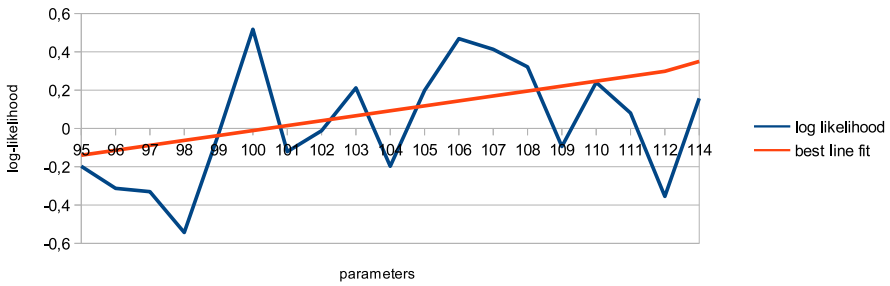

Figure 6.8: The values of the parameters of the effect W2VavgAvg using top four keywords

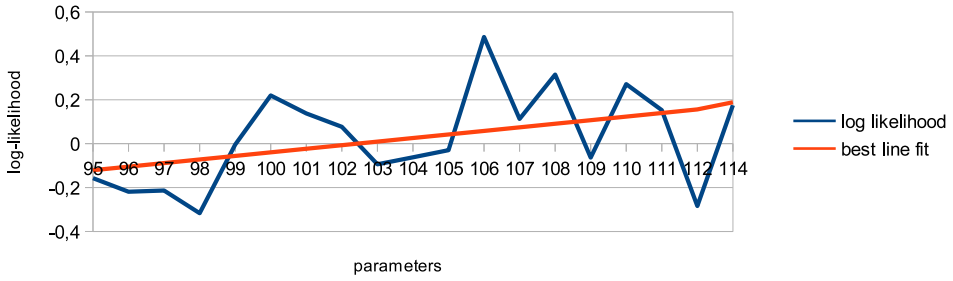

Figure 6.9: The values of the parameters of the effect W2VavgAvg using top four keywords and trained together with W2VavgAvg-material interaction effect as well as the usual material and placement effects

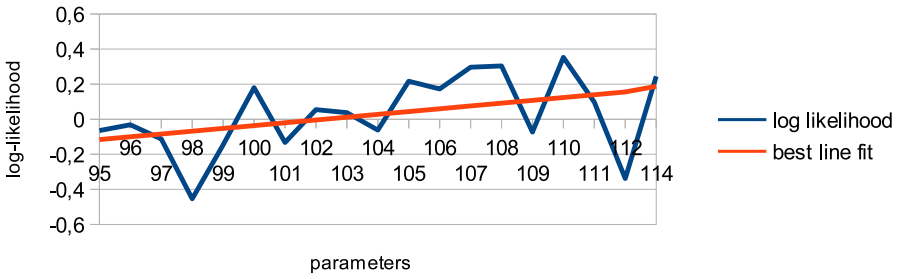

Figure 6.10: The values of the parameters of the effect W2VavgAvg when training with the effects: material, placement, material-placement, webCategory, webCategory-material, W2VavgAvg and W2VavgAvg-material

6.3 Tests on new materials

When testing on materials we have not trained on there is a large risk of over-fitting and indeed all the tests run show signs of this. As an example look at table 6.5 where training is done using the placement and material effects. For every iteration the model is fitted better to the training data but fits worse to the test data. In table 6.6 some of the other effects are tested and many of them are worse and still over-fitted. The only effect to perform good is the W2VavgAvg.

Iteration	Log-likelihood of training data	Log-likelihood of test data
0	-1250057,5167654	-2015,73435785121
1	-1249980,25429286	-2025,99360418356
2	-1249968,9349221	-2044,29116608447
3	-1249966,99429864	-2046,09018485268
4	-1249966,27953323	-2050,12627392327
5	-1249966,07638494	-2051,49687372202
6	-1249965,95484466	-2056,09533972916
7	-1249965,94776325	-2056,5064780244
8	-1249965,94602889	-2056,66062505361
9	-1249965,94548856	-2056,812688606

Table 6.5: The results of every iteration when training using only the material and placement simple effects. Notice that for every iteration the model fits better to the training data but worse to the test data.

Effect	Log-likelihood of test data
No added effects	-2056,8
W2VavgSum	-2062,1
W2VsumSum	-2049,5
W2VavgAvg	-2014,8
webCategory	-2062,1
W2VavgSum	-2062,1

Table 6.6: The results of different effects after training on the *newAds dataset*. All tests use the material and placement simple effects.

7

Discussion

Looking back at table 6.2 we can see that the results of the WordNet similarities is in general slightly poorer than those of Word2Vec. This could be because of the rather large difference in the number of keywords found in their vocabularies or it could have some other explanation. In any case I would not recommend using WordNet for this kind of task. Admeta's customers use many different languages and not all of them are as large as Spanish. There would be problems finding good WordNet translations and Word2Vec seems to outperform WordNet on a relatively common language. Word2Vec on the other hand is much easier to adapt to different languages. All that is needed is a lot of text in the target language and there are many such sources on-line, for instance Wikipedia.

One of the best effects was the keywords effect. This was quite unexpected since it seems unlikely that individual words would have such a big impact on ad performance, but it probably has a simple explanation. As there were very few keywords that appeared in more than one of the tested web-pages the keywords effect acted as web-page effect rather than its intended use. To have a parameter for every web-page works well in this limited setting but would most likely be infeasible in Admeta's model.

I have found a few effects that seems to be useful. In particular categorizing the web-pages worked well as did the W2VavgAvg. I would say that W2VavgAvg is also the most intuitive of the similarity measures tested as it is simply the average of all web-page and material keyword pairs. W2VavgAvg also performed well for the *newAds dataset* which indicates that it should also be good for new ads where there are little historic data.

7.1 Future Work

There are many areas where it might be good to do more work. The results of categorizing the materials were very poor and it might be because the categories used does not

capture advertisements well. Maybe better results would be achieved if the categories were more fine grained and ad-driven, for instance things like ”clothes” or ”cars” rather than ”religion” or ”weather”. The website categorization while it had positive effects might have benefited from a deeper categorization as well. Another type of categorization that could have been tried is image categorization. Other information about the ads could be beneficial as well such as the target audience, gender, age-group or the like. Sun et al[24] discuss the differences between genders in on line advertisement. For instance an ad for a car insurance might be better to show on a page where many ads for middle aged men have been clicked, rather than on a page where ads targeted towards young teenagers have been successful. A large problem with these kind of variables is that they are very hard to extract automatically and would most likely have to be entered by the advertisers which could cause problems. To avoid some of this it could be plausible to use optical character recognition (OCR) to extract the text.

It would be good to look at a larger set of web-pages and materials as well as use more observations. As only a few web-pages and materials were used in the tests it may be that some results are inaccurate, or that some patterns of materials and web-pages that could have been found were missed. Using a larger dataset would also decrease the risk of over-fitting on the *newAds dataset*. It would also be interesting to look at the parameters of the *W2VavgAvg* similarity effect and see if the best line fit still has a positive slope and whether the parameters are closer to this slope than they were in the tests. If this is the case it could be that W2VavgAvg can be trained to a line rather than the more complex system with parameters. This would be good to save memory and space. It would also make it possible to not round the similarity values as much.

Another area to look at is the possibility of preprocessing the texts that are used for training Word2Vec. For instance it could be worthwhile to do entity recognition which means finding the entities in the texts and treating them as one word. This would mean that, as an example, we get a word vector for ”la liga” the Spanish top football division, instead of just the two vectors for ”la” and ”liga” separately which may be influenced by many other contexts.

It may also be good to employ the similarity measure between two texts proposed by Mihalcea et al[25]. It is similar to the *avgMax* that was tried in this thesis but gives greater weight to the similarity of words that are infrequent in the corpus.

8

Conclusion

In this project I have tested many different ways of using contextual information from ads and web-pages to improve the click rate predictions of on-line advertising. A categorization of web-pages was acquired using AlchemyAPI and this showed promising results.Two methods of measuring similarity between keywords were also tried. The first one used the distance between words in the lexical database WordNet to get a similarity and the second used vector representations of words trained by the tool Word2Vec. The similarities between keywords then had to be combined somehow to for the similarity between a web-page and ad. Many variations were tried but in the end the best and most intuitive method was to use the Word2Vec tool and take the average of the cosine-distances between the word-vector pairs.

I would recommend Admeta to continue their research in this area by looking further into these similarity values and perhaps try a measure similar to the one proposed by Mihalcea et al[25] were the keywords' frequencies in the language is also taken into account. Another thing that should be explored is a more fine grained categorization of both web-pages and ads.

Bibliography

[1] P. Chatterjee, D. L. Hoffman, T. P. Novak, Modeling the Clickstream: Implications for Web-Based Advertising Efforts, Marketing Science 22 (4) (2003) 520–541.
URL http://dx.doi.org/10.1287/mksc.22.4.520.24906

[2] C. Wang, P. Zhang, R. Choi, , M. D. Eredita, Understanding consumers attitude toward advertising, in: In Eighth Americas conf. on Information System, p. 1143–1148.

[3] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, E. Silva de Moura, Impedance Coupling in Content-targeted Advertising, in: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '05, ACM, New York, NY, USA, 2005, pp. 496–503.
URL http://doi.acm.org/10.1145/1076034.1076119

[4] A. Broder, M. Fontoura, V. Josifovski, L. Riedel, A Semantic Approach to Contextual Advertising, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, ACM, New York, NY, USA, 2007, pp. 559–566.
URL http://doi.acm.org/10.1145/1277741.1277837

[5] D. Chakrabarti, D. Agarwal, V. Josifovski, Contextual Advertising by Combining Relevance with Click Feedback, in: Proceedings of the 17th International Conference on World Wide Web, WWW '08, ACM, New York, NY, USA, 2008, pp. 417–426.
URL http://doi.acm.org/10.1145/1367497.1367554

[6] T. Wästerlid, Application of L-BFGS to a large-scale poisson MAP estimate, Master's thesis, Chalmers University of Thecnology, Gothernburg University (September 2012).

[7] A. Perry, A class of conjugate gradient algorithms with a two-step variable metric memory, Tech. rep., Graduate School of Management, Northwestern University, Evanston, Illinois (1977).

[8] J. F. Bonnans, J. C. Gilbert, C. Lemaréchal, C. A. Sagaztizábal, Numerical Optimization: Theoretical and Practical Aspects, Berlin Heidelberg, Springer, 2nd edition, 2006.

[9] R. Salakhidinov, A. Mnih, Probabilistic Matrix Factorization, in: Twenty-Second Annual Conference on Advancements in Neural Information Processing Systems, Vol. 20, NIPS, Hyatt Regency Vancouver, in Vancouver, B.C, Canada, 2008.

[10] F. Bond, Open Multilingual Wordnet (2013).
URL `compiling.hss.ntu.edu.sg/omw/(2014-05-05)`

[11] Princeton University, Word Net (2010).
URL `wordnet.princeton.edu(2014-05-05)`

[12] MCR, Multilingual Central Repository (2013).
URL `adimen.si.ehu.es/web/MCR(2014-05-05)`

[13] T. Pedersen, Similarity Measures (2014).
URL `cpansearch.perl.org/src/BTMCINNES/UMLS-Similarity-1.35/web/docs/similarity_measures.html(2014-05-05)`

[14] G. E. Hinton, J. L. McClelland, D. E. Rumelhart, Distributed representaions, Parrales distributed processing: Explorations in the microstructure of cognition, 1986Foundations, MIT Press.

[15] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient Estimation of Word Representation in Vector Space, in: Proceedings of Workshop at ICLR, 2013.

[16] Y. Bengio, New distributed probabilistic language models, Tech. rep., Département d'informatique et recherche opérationnelle, Université de Montréal (2002).

[17] F. Morin, Y. Bengio, Hierarchical probabilistic neural network language model, in: AISTATS'05, 2005, p. 246–252.

[18] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed Representations of Words and Phrases and their Compositionality, in: Proceedings of NIPS, 2013.

[19] T. Mikolov, W. tau Yih, G. Zweig, Linguistic Regularities in Continuous Space Word Representations, in: Proceedings of NAACL HLT, 2013.

[20] T. Mikolov, Q. V. Le, I. Sutskever, Exploiting Similarities among Languages for Machine Translation, CoRR abs/1309.4168.

[21] M. Kågebäck, O. Mogren, N. Tahmasebi, D. Dubhashi, Extractive Summarization using Continuous Vector Space Models, in: 2nd Workshop on Continuous Vector Space Models and their Compositionality CVSC 2014, Chalmers University of Technology.

[22] T. Mikilov, Word2vec (2013).
URL `https://code.google.com/p/word2vec(2014-05-05)`

[23] R. Řehůřek, P. Sojka, Software Framework for Topic Modelling with Large Corpora, in: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, ELRA, Valletta, Malta, 2010, pp. 45–50, `http://is.muni.cz/publication/884893/en`.

[24] Y. Sun, K. H. Lim, C. Jiang, J. Z. Peng, X. Chen, Do Males and Females Think in the Same Way? An Empirical Investigation on the Gender Differences in Web Advertising Evaluation, Comput. Hum. Behav. 26 (6) (2010) 1614–1624.
URL `http://dx.doi.org/10.1016/j.chb.2010.06.009`

[25] R. Mihalcea, C. Corley, C. Strapparava, Corpus-based and knowledge-based measures of text semantic similarity, in: IN AAAI'06, 2006, pp. 775–780.