Predicting Ads' Click-Through Rate with Decision Rules

Krzysztof Dembczyński
Poznan University of Technology
Piotrowo 2
Poznan 60-965, Poland

Wojciech Kotłowski
Poznan University of Technology
Piotrowo 2
Poznan 60-965, Poland

Dawid Weiss
Poznan University of Technology
Piotrowo 2
Poznan 60-965, Poland

{kdembczynski,wkotlowski,dweiss}@cs.put.poznan.pl

ABSTRACT

Paid advertisements displayed alongside search results constitute a major source of income for search companies. Optimizations leading to more clicks on ads are a target goal shared by advertisers and search engines. In this context, an ad's quality can be measured by the probability of it being clicked assuming it was noticed by the user (*click-through rate*, CTR). There are two problems here. First, the real CTR is heavily affected by the ad's position, not only by its content. Second, it is not possible to directly calculate CTR for new ads.

In this paper we build upon a large data set with real ad clicks and impressions, acquired thanks to Microsoft's Beyond Search program. First, we describe the data set and anomalies found inside it. We then address the problem of computing CTR for existing ads using maximum likelihood estimation. Finally, we present an algorithm learning an ensemble of decision rules that can be used for predicting the CTR for unseen ads and giving recommendations to improve ads' quality.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

On-line advertising, CTR prediction, decision rules, ensemble methods

1. INTRODUCTION

Search advertising is one of the major contributions to search engines' revenues, reaching $9.4 billion in 2006 compared to $5.75 billion in 2005 [13]. One of the advertising models concerns presenting contextual ads directly on query results pages. Advertisers are typically charged each time their ads are clicked (such an approach is called pay-per-click; two other approaches, pay-per-action and pay-per-impression are rarely used [10]). In this model we have two roles: the search engine and the advertisers. While the search engine seeks to maximize its revenue, advertisers compete with each other for the highest position (rank) of their ads. The revenue of a search engine is based on the number of clicks multiplied by the average cost per click (CPC) in a given period of time. The CPC has the form of a bid and is established through an auction [6]. The number of clicks, in turn, depends on how often a given ad is displayed and the probability that the user clicks on it (*click-through rate*, CTR). The latter is connected with a notion of *ad quality*—a high quality ad will have a corresponding high probability of collecting users' clicks.

The most common ad ranking strategy used by search engines depends on a combination of CPC and CTR. The CTR is usually unknown for rarely displayed or new ads and it cannot be estimated directly with historical data. Several methods for estimating and predicting the CTR have been proposed in literature: clustering existing ads and using the cluster's CTR [14] or building a logistic regression model using features extracted from statistics of existing ads [15].

2. MOTIVATION AND SCOPE OF WORK

We divided the work to be done into two parts:

- *estimating* the CTR value for existing ads from historical data (logs),
- *predicting* the CTR value for new or rarely displayed ads.

With respect to the first problem, the main issue is that in order to meaningfully compare ads, we would like to use a measure which depends only on the ad's content. Unfortunately, ads are displayed at different positions on a page—it is well known that this position and results page number (when search results are paged) has a dramatic impact on click-through rate [12]. In order to remove the influence of these factors [15] defines the CTR as a probability of clicking the ad provided it was seen. The probability of seeing an ad was taken from the so-called heat maps (experimental study of eye movements on pages with search results). The CTR was then estimated by simply dividing the number of times the ad was clicked by the estimated number of times it was seen. We propose a more formal approach, while keeping the same interpretation, and introduce a specific factorization of probabilities. Then, in order to estimate CTR, we use maximum likelihood estimation (MLE) to learn all model parameters directly from real historical data.

To address the second problem (CTR prediction) we propose a new prediction model in the form of ensemble of decision rules. The difference between this model and previous approaches is that:

- we try to make the model *interpretable* so that it is comprehensive for humans to analyze and possibly improve,

- we try to handle interactions between features and the context of the original query for which the ads are to be ranked.

Note that our focus is not limited to prediction performance. Assuming the model is comprehensive one can use it for making more general recommendations, hopefully improving the quality of ads. For example statements such as "use shorter title", but also recommendations for advertisers based on knowledge existing in the model—if the ad's keywords are "firewall" and "security" and the model predicts that ads containing the word "network" in their description would be scored higher, the system may recommend adding this word somewhere in the body of the ad (not necessarily to the set of keywords describing it). Such a model is distinctively different from keyword-suggestion systems because it gives recommendations in a broad scope (not limited to keywords).

Let us recall that we want to build a prediction model based on features that are strictly related to the quality of ads. These features could include: the number of words used in the title and in the body of an ad, the number of segments and length of the target URL, individual words and terms used in the title and the body and many other possibilities (see [15] for examples). Using these features we learn an ensemble of decision rules—an efficient and accurate combination of individual classifiers with human-understandable interpretation. A single rule is just a logical statement: "if [condition] then [decision]". The condition corresponds to the conjunction of logical relations imposed on features, and the decision is a number, either positive (increasing the estimated probability) or negative (decreasing the probability). Decision rules in an ensemble are combined by weighted majority voting to give accurate predictions.

The process of constructing an ensemble of decision rules utilizes techniques such as *boosting* [7] and *forward stagewise additive modeling* [11]. In each iteration a single rule is built based on the value of the error made by previously generated rules. A number of algorithms building rule ensembles are known: RuleFit [9], SLIPPER [3] or LRI [16] for example. In this study, we use an algorithm called ENDER [4].

All experiments in this work were possible because we could use a data set with real data granted to us by Microsoft as a part of Beyond Search—Semantic Computing and Internet Economics research grant. Unfortunately, this data set does not contain all the elements that have been previously used to derive ad features [15], particularly those related to ad quality (lack of keywords, titles and bodies of ads). We thus had to perform several simplifications that can possibly distort the results. The purpose of this study was, however, to show the potential of decision rule-based methodology rather than to give final results on Beyond Search data set. We think the initial outcome is so promising that it justifies the publication of results in their present shape.

The remainder of the paper is organized as follows. In Section 3 we briefly describe the data set used in the experiments. Section 4 discusses the problem of CTR estimation from real historical data and use of MLE for this purpose. In Section 5, we present an algorithm learning an ensemble of decision rules that can be used for predicting CTR of previously unseen ads and recommending ad quality improvements. We also show preliminary results we obtained on Beyond Search data. The last section concludes the paper and gives some ideas for future work.

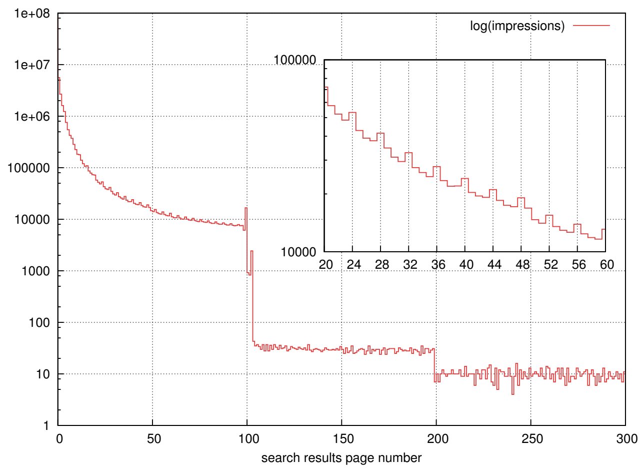

Figure 1: A histogram showing the count of impressions for each search results page. The zoom-in window shows patterns between page 20 and 60. Vertical axis is logarithmic.

3. BEYOND SEARCH DATA SET

The data set we used was granted by Microsoft as part of the Beyond Search research grant and contained two sources of information about ads:

- `impressions`—an excerpt from Microsoft Live! query log, sampled over a few months and filtered to remove any personal information,
- `clicks`—a log of click throughs for ads displayed on pages with search results.

The clicks file had a foreign key column to impressions file, allowing one to combine queries with ads and their positions on each results page. To simplify and speed up computations we used the map-reduce computing paradigm (specifically, we used the Hadoop project[1] running on a cluster of 10 quad core machines).

3.1 Data analysis and peculiarities

There was a total of 7 822 292 ad clicks and 101 171 081 impressions (response pages). We first combined these two data sets into a single "flat" form. Each impression was transformed into one or more records in the flat file (depending how many ads it contained). For each ad we recorded the information whether it was clicked or not and which position on the results page it occupied. The final flattened file contained 386 857 679 records and 1 989 709 unique ad identifiers (4.2 GB of block-compressed Hadoop record sequence file).

We noticed some intriguing anomalies in the distribution of samples. For example, impression contained information about the served search results page for a given query. Intuitively, the distribution of impressions for pages should be exponential (not many people browse past the first page of search results). In the data set this is confirmed to about the 100-th page, then there is a sudden drop and the number

[1]`http://hadoop.apache.org/`

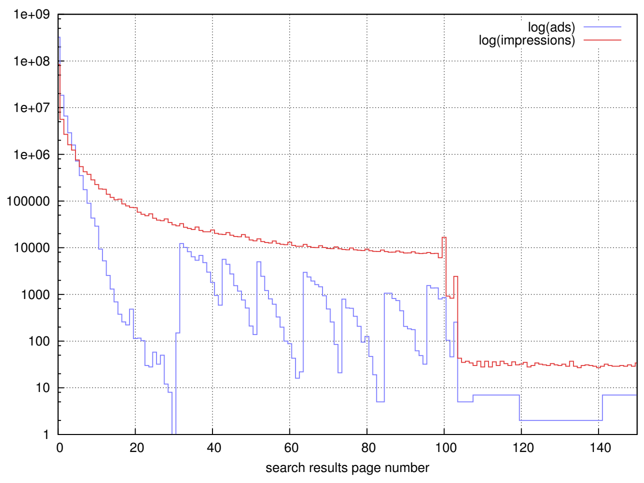

Figure 2: A histogram showing the count of impressions and ads for a range of result pages. Vertical axis is logarithmic.

of impressions remains nearly *constant* (see Figure 1). This seems odd and we tend to believe it can be some sort of an artifact left from the sampling method used to produce the data set (it is unlikely that these impressions were caused by human activity anyway). The zoom-in window in Figure 1 shows another interesting phenomenon—cyclic "bumps" every fourth page. This can be explained if one looks at how paging is implemented in Live!'s user interface—the maximum page jump in either direction is exactly 4.

Once we found out that the distribution of impressions is quite unexpected, we looked at the distribution of ads with regard to the results page they appeared on. The outcome here was even more surprising, as shown in Figure 2. There seems to be little correlation between the number of impressions and ads shown on each result page. The jagged shape of ads' distribution cannot be easily explained as well.

3.2 Ads, their domains and queries

Another analysis focused on ads and their attributes. In theory, each ad in the clicks file contains an attribute describing its unique ID and a "domain" the ad points to. Our assumption that unique ad identifiers point to the same domain turned out to be false—majority of impressions concern ads whose identifiers point to multiple domains (see Table 1). Moreover, identical domains belong to non-unique ad IDs (see Table 2). Our conclusion was that ad identifiers actually represent *advertiser* identifier (or ad campaign identifier). This was further confirmed by the fact that queries leading to ads with a high number of domains were virtually random (contained no common subject or keywords).

We eventually restricted our further analysis to only those ads that belonged to exactly one domain. This limited data set was then filtered again, extracting ads that reached a given minimum number of impressions. The final results of this filtering and information about data sets is summarized in Table 3. Note the "long tail" of ads with a very small number of impressions—decreasing the number of unique ads four times does not affect the overall number of ad impressions much.

Table 1: Examples of unique ads with the highest number of domains (`www` and domain suffixes stripped).

domains	examples
4764	LivestockFarming, OhioLaborRelations, ReplacementSilver, DailyHoroscopeFree, PiesRecipes
713	hobbyfarms, tracking.searchmarketing, avenue, usplastic, 1000freeoffers
562	MarketingProducts, PortageCountyAuditor, Topographic-Map, PlotMaps, ItalianPeople
491	mo-media, lawdepot, replicon, shopzilla, cashunclaimed
470	rotarysystems, gccustomservices, tracking.searchmarketing, usplastic, s0b.bluestreak
429	weightlosspillguide, tracking.searchmarketing, rosenberryrooms, usplastic, s0b.bluestreak

Table 2: Examples of domains pointing to non-unique ads.

domain name	unique ad IDs
www.shopzilla.com	20032
www.bizrate.com	17592
www.pronto.com	12831
clickserve.dartsearch.net	11090
www.amazon.com	10828
clk.atdmt.com	7097
clickserve.cc-dt.com	5704
track.did-it.com	5264
rover.ebay.com	4767

4. CLICK-THROUGH RATE OF AN AD

Usually, the ads are ranked by the search engine according to the product:

$$\text{CPC} \times \text{CTR}$$

where CPC denotes *cost-per-click* and CTR is *click-through rate*. This policy of a search engine aims at maximizing its revenue. From the advertiser's point of view, higher values of CPC and CTR imply better position in the ranking, and higher probability that an ad will be seen by the search engine's user. The CPC has the form of a bid and is established using an auction mechanism, but the CTR depends mainly on the quality of the advertisement. The actual value of the CTR of a given ad is not easy to compute and in many cases has to be predicted. Moreover, the advertiser would

Table 3: Data sets after pruning to single-domain subsets. The first column indicates minimum number of impressions an ad had to reach to be included.

minimum impressions	unique ad IDs	impressions	sequence file size
all	463 202	110 955 561	2 639 938 600
10	373 437	110 553 933	2 627 028 530
30	279 786	108 851 279	2 578 935 391
60	211 419	105 916 306	2 499 814 466
200	101 529	93 538 958	2 180 492 901

like to improve the quality of its ads in order to increase their CTR. These two tasks, prediction of the CTR and ad quality improvement, are not trivial.

In the remainder of this section, we formalize the definition of CTR and the way it could be estimated for existing ads. Then in Section 5 we extend our analysis to CTR prediction for unseen ads and establishing a feedback loop between the prediction model and the advertiser for improving ads' quality.

4.1 Modeling CTR

Let us introduce a concise mathematical notation for modeling the CTR. Let us assume a set of n ad impressions, where i-th impression is described by the following variables: a_i—ad's ID, r_i—ad's position (rank) on the page, s_i—results page on which the ad appeared, c_i—binary variable indicating whether ad was clicked ($c_i = 1$) or not ($c_i = 0$).

To establish a probabilistic model for CTR prediction, we examine the assumption which is implicitly made when using the CTR for ads ranking. Namely, we assume that there exists a measure, expressed in terms of probability of clicking on the ad, depending only on the ad's content, not on its position on the search results page. This implies the following dependency: let $P(c = 1|a, s, r)$ be the probability of clicking the ad a on the position r and page s and let a and a' be any two ads. Then, the comparison of probabilities $P(c = 1|a, s, r)$ and $P(c = 1|a', s, r)$ should not depend on particular position r and page s, provided both r and s are the same for both ads. Otherwise if for some s, r we had $P(c = 1|a, s, r) > P(c = 1|a', s, r)$, while for some other s', r' we had $P(c = 1|a, s', r') < P(c = 1|a', s', r')$, then it would not be possible to meaningfully compare the ads only in term of the ad's content a. In other words, if we compare two ads on the same position and page, then the results of such comparison should not depend on the particular position and page value, assuming they are the same for both compared ads.

Such assumption immediately implies the following factorization of probabilities:

$$P(c = 1|a, s, r) = p_a(a)p_r(r)p_s(s), \qquad (1)$$

where $p_a(\cdot), p_r(\cdot), p_s(\cdot)$ are single-argument functions. Note that all three functions are defined up to the multiplication constant, since multiplying one of them by any value and dividing any other by the same value does not change the left hand side of (Eq. 1). Therefore we normalize them by setting $p_r(1) = 1$ and $p_s(1) = 1$. Then we obtain

$$p_a(a) = P(c = 1|a, s = 1, r = 1),$$

the probability of clicking the add a provided it is located on the first position and on the first page.

Equation 1 has a nice interpretation in terms of hidden variable model. Let h be some hidden binary variable, $h \in \{0, 1\}$ such that if $h = 0$ then the ad was definitely not clicked; moreover h depends on position and results page only, and $P(c = 1|a, s, r, h) = P(c = 1|a, h)$, i.e., clicking the ad depends on position and page only through h. In [2, 15] such variable was introduced implicitly with the following interpretation: $h = 1$ if and only if the ad a was seen by the user (of course, the particular interpretation is not important to the statistical analysis). The probability marginalized over h then becomes:

$$P(c = 1|a, s, r) = P(c = 1|a, h = 1)P(h = 1|s, r), \qquad (2)$$

(since $P(c = 1|a, h = 0) = 0$ from the definition of h) and we can identify $p(h = 1|s, r)$ as $p_r(r)p_s(s)$—the probability that an ad was seen given the position r and page s, and identify $p_a(a)$ as $P(c = 1|a, h = 1)$—the probability that ad was clicked provided it was seen. This will serve as an interpretation of CTR in this paper.

The only problem with the hidden variable interpretation is in the normalization of functions $p_r(r)$ and $p_s(s)$. We simply set $p_r(1) = p_s(1) = 1$, but it means that $P(h = 1|s = 1, r = 1) = 1$, i.e., the ad is always seen at the first position of the first page. One could argue whether this is true or not, however this issue is completely irrelevant, since what really matters is the comparison of CTRs for different ads, not their actual values. Therefore, any normalization would apply here just as well.

4.2 Maximum Likelihood Estimation of Model Parameters

In [15] the probability of seeing an ad for a given position was taken from the so-called heat map (eye tracking study). The CTR was then estimated by simply dividing the number of times the ad was clicked by the estimated number of times it was seen. The latter was estimated by adding the number of times the ad was clicked and the number of times it was displayed but not clicked weighted by the probabilities of being seen when not clicked. It is not clear from the paper how the probability of being seen when not clicked was estimated. Moreover, note that such an estimate is not equivalent to maximum likelihood estimate for the CTR.

Since we want to "let the data speak for itself", we do not use heat maps and estimate all the parameters $p_a(a)$, $p_r(r)$, $p_s(s)$ purely from the data set, employing maximum likelihood estimation (MLE) method for this, similarly to [2]. We maximize the following likelihood function:

$$L = \prod_{i=1}^{n} [p_a(a_i)p_r(r_i)p_s(s_i)]^{c_i} [1 - p_a(a_i)p_r(r_i)p_s(s_i)]^{1-c_i}$$

or equivalently, minimize the negative log-likelihood:

$$\begin{aligned} \ell &= -\sum_{i=1}^{n} \Big(I(c_i = 1) \log \big(p_a(a_i)p_r(r_i)p_s(s_i)\big) + \\ &+ I(c_i = 0) \log \big(1 - p_a(a_i)p_r(r_i)p_s(s_i)\big) \Big) \end{aligned}$$

where $I(\cdot)$ is the indicator function—$I(u) = 1$ if u is true, otherwise $I(u) = 0$. By setting the derivatives $\frac{\partial \ell}{\partial a_i}, \frac{\partial \ell}{\partial r_i}, \frac{\partial \ell}{\partial s_i}$ to 0, we obtain the following set of equations:

$$\sum_{i=1}^{n} I(a_i = a) \left(\frac{c_i}{p_a(a)} - \frac{(1 - c_i)p_r(r_i)p_s(s_i)}{1 - p_a(a)p_r(r_i)p_s(s_i)} \right) = 0 \quad (3)$$

$$\sum_{i=1}^{n} I(r_i = r) \left(\frac{c_i}{p_r(r)} - \frac{(1 - c_i)p_a(a_i)p_s(s_i)}{1 - p_a(a_i)p_r(r)p_s(s_i)} \right) = 0 \quad (4)$$

$$\sum_{i=1}^{n} I(s_i = s) \left(\frac{c_i}{p_s(s)} - \frac{(1 - c_i)p_a(a_i)p_r(r_i)}{1 - p_a(a_i)p_r(r_i)p_s(s)} \right) = 0 \quad (5)$$

for each ad a, position r and page s. It is not possible to solve these equations analytically. Therefore we perform the optimization similarly as in the Expectation-Maximization (EM) algorithm [5]: each variable is optimized with other variables being fixed, i.e., Equations 3–5 are solved separately. Note that each of these equations is a line-search

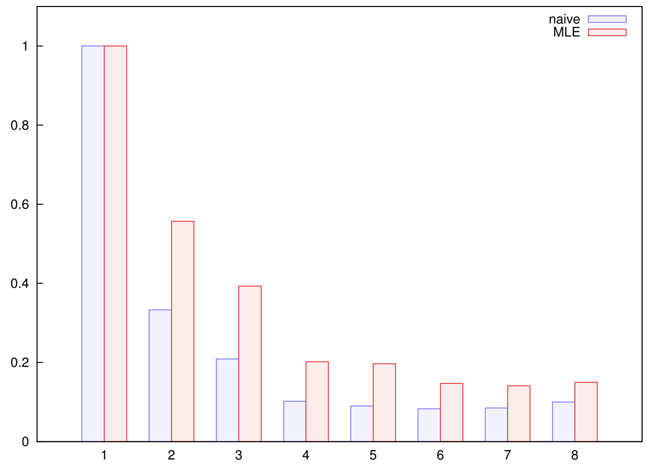

Figure 3: The likelihood of seeing the ad as a function of its position $p_r(r)$. Two estimation methods are considered: naïve and MLE.

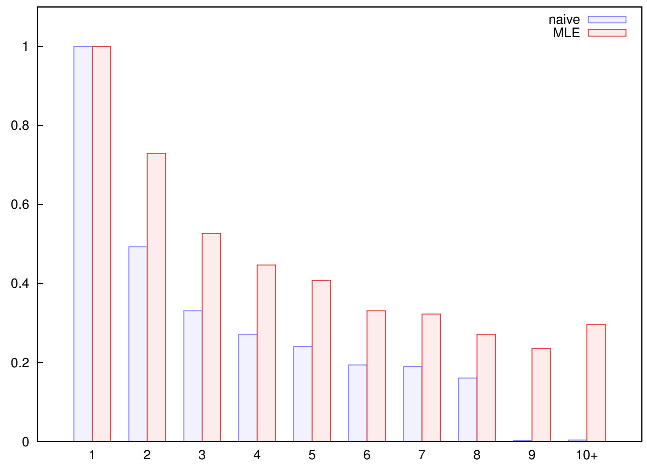

Figure 4: The likelihood of seeing the ad as a function of the ad's page $p_s(s)$.

convex optimization problem which can be easily solved numerically. The sequence of such problems converges to the local minimum of negative log-likelihood, since solving each problem always decreases the likelihood (unless we reach the minimum).

4.3 Results

To estimate the parameters $p_a(a)$, $p_r(r)$ and $p_s(s)$, we chose only a subset of the data set, consisting of single-domain ads which were displayed at least 200 times. This decreased the number of parameters to estimate (since for each ad, a single parameter $p_a(a)$ must be estimated), so it also increased the number of observations per parameter which in turn reduced the variance of the estimates. Although the threshold of 200 looks restrictive, this only slightly reduced the number of records n (number of impressions), while significantly reducing the number of unique ad IDs (see Table 3 for details). Next, we merged the numbers of results pages greater or equal to 10 into one single value "10+" to avoid the problems with only a few (or even none) observations for certain page numbers. We also noticed that the maximum ad position on a page was 8.

The algorithm converged within 30 iterations (taking about 2 hours on commodity hardware). We present the results for $p_r(r)$ and $p_s(s)$ in Figure 3 and 4 respectively (there are over 100 000 parameters $p_a(a)$, so we omit these). As it was already mentioned in Section 4.1, $p_r(r)$ and $p_s(s)$ are meaningful only up to some multiplication constant and are normalized by setting $p_r(1) = p_s(1) = 1$. We follow the "hidden variable" interpretation, so for each subsequent position r (or page s), the value $p_r(r)$ (or $p_s(s)$) denotes the decrease of probability of seeing the ad compared to the first position (first page); therefore we will call these values "likelihoods" instead of "probabilities". In both figures above we also present the results of naïve estimation method for comparison—this method estimates $p_r(r)$ as the fraction of clicked ads on position r; similarly it estimates $p_s(s)$ as the fraction of clicked ads on page s.

Figures 3 and 4 demonstrate that the likelihoods decrease both with position and results page (as expected) and the shape of this decrease is roughly exponential. Note the last, 8-th position has a slightly higher likelihood. This phenomenon may be caused by the fact that it is somewhat easier to focus on ads which are either on the top or on the bottom of the ads' block. What remains unclear to us is why the likelihood for pages "at least 10" (10+) is higher then for pages 9 and 8. We believe this must be related to anomalies found in the input data set (as shown in Figure 2). Returning to the analysis, note that the results of the naïve approach are "too optimistic" for the top position and page and "too pessimistic" for the bottom position and page. It follows from the fact that what naïve method actually estimates is the likelihood of *clicking on the ad*, not of *seeing* it. But the likelihood of clicking the ad decreases faster with increasing position/page than the likelihood of seeing the ad, since the pace of decrease is boosted by the decrease in ad's CTR (which is unknown). Concluding, the naïve method may lead to wrong conclusions about the estimated likelihoods.

The MLE can be thought of as the method of nivelating the influence of the position and search results page on the CTR. Notice that we end up with the probabilities factorized into three components $p_r(r)$, $p_s(s)$ and $p_a(a)$. In further analysis (prediction of the CTR for unseen ads) we are interested only in $p_a(a)$, because only this component contributes to the CTR. Therefore one can also think of MLE also as a method for preparing the data set: we obtain the CTR for each ad, which will play the role of the output (response) variable in the prediction problem described in sections below.

5. PREDICTING CLICK-THROUGH RATE

In order to predict the CTR we use an algorithm for learning an ensemble of decision rules [4]. Besides offering good prediction performance, this algorithm gives an interpretable result in the form of linear combination of decision rules $r_m(x)$:

$$f(x) = \sum_{m=0}^{M} r_m(x), \qquad (6)$$

where the resulting function $f(x)$ estimates the CTR of an ad ($p_a(a)$) described by the features in vector x. We used a training set $\{y_a, x_a\}_1^N$, in which each example corresponds to a single ad a, y_a is an estimate of the CTR computed using MLE (as it is shown in the previous section), and x_a contains features related to the ad quality. The choice of such features is caused by the fact that we would also like to use the prediction model for recommendations improving the quality of ads. Unfortunately, the data set we had access to did not contain elements such as the title and body of an ad and these are strictly connected to ad quality. We therefore assumed that similar words as those appearing in queries must have been used in titles and bodies of the ads. We approximated the body and title of each ad by taking all queries for which the ad was displayed and extracted about 1000 frequently recurring terms. This way we obtained something that we call a *description* of an ad. We used also several features based on ads' URLs: the domain suffix (".com", ".org", ".net", etc.), number of segments, length of the URL, etc. We obviously made some simplifications that can result in misleading rules. However, our goal in this study was to present a potential of the methodology rather than to give the final results on the Beyond Search data set.

Because, the prediction function $f(x)$ estimates the probability, we decided to minimize the squared-error loss on the training set (*empirical risk*):

$$R_{emp}(f) = \sum_{a=1}^{N} (y_a - f(x_a))^2$$

One could use other loss functions, like logit loss, but in this paper we limit our considerations to the squared-error loss. We will now shortly outline the algorithm for learning $f(x)$ (given as a linear combination of decision rules), show how to use these decision rules for quality improvement and give some experimental results.

5.1 Ensemble of Decision Rules

A decision rule is defined as:

$$r(x) = \alpha\Phi(x),$$

where Φ is a conjunction of elementary conditions constituting the condition of the rule, $\Phi(x)$ is a function indicating whether an object satisfies the condition of the rule, and α is a real non-zero value called decision or response of the rule. Notice that the decision rule takes only two values, $r(x) \in \{\alpha, 0\}$, depending whether x satisfies the condition part or not.

Construction of optimal combination of rules (Eq. 6) is a hard optimization problem. For this reason we follow forward stagewise additive modeling [11] that results in an iterative procedure in which rules are added one by one, greedily minimizing empirical risk. In the m-th iteration, we look for a rule:

$$r_m = \arg\min_{\Phi,\alpha} \sum_{a=1}^{N} L(y_a, f_{m-1}(x_a) + \alpha\Phi(x_a)), \qquad (7)$$

where $L(y, f)$ is a loss function and f_{m-1} is a prediction function after $m-1$ iterations. We assume that the first rule $r_0(x)$ covers all training examples, i.e., $\Phi(x_a) = 1$, $a = 1, \ldots, N$. Such a rule, referred to as *default*, plays a role of Bayes prior.

Since the exact solution of Equation 7 is computationally demanding, we proceed in two steps.

Step 1. We find Φ_m by minimizing a functional $\mathcal{L}_m(\Phi)$ in a greedy manner:

$$\Phi_m = \arg\min_{\Phi} \mathcal{L}_m(\Phi) \qquad (8)$$

The particular form of the functional is derived from (Eq. 7) and depends on the chosen loss function and minimization technique. In the case of the squared-error loss, there exists an exact form of $\mathcal{L}_m(\Phi)$ minimizing (Eq. 7) [8, 4]:

$$\mathcal{L}_m(\Phi) = -\frac{\left|\sum_{\Phi(x_a)=1}(y_a - f_{m-1}(x_a))\right|}{\sqrt{\sum_{a=1}^{N}\Phi(x_a)}}$$

The greedy procedure used for finding Φ resembles the way the decision trees are generated. Here, we look for only one branch instead of the whole decision tree. At the beginning, Φ_m is empty and in each next step an elementary expression is added to Φ_m until $\mathcal{L}_m(\Phi)$ cannot be decreased. Let us underline that a minimal value of $\mathcal{L}_m(\Phi)$ is a natural stop criterion, what differs this procedure from those used for decision trees generation. In order to improve both accuracy and computational complexity, this procedure is performed on a subsample of training examples of size $\eta \leq N$, drawn without replacement.

Step 2. We find α_m, the solution of the following line-search problem:

$$\alpha_m = \arg\min_{\alpha} \sum_{a=1}^{N} L(y_a, f_{m-1}(x_a) + \alpha\Phi_m(x_a)) \qquad (9)$$

In the case of the squared-error the computation of α_m is straightforward, because analytical expression for (Eq. 9) can be given:

$$\alpha_m = \frac{\sum_{\Phi(x_a)=1}(y_a - f_{m-1}(x_a))}{\sum_{a=1}^{N}\Phi(x_a)}$$

The obtained rule $r_m(x) = \alpha_m\Phi(x)_m$ is then shrunk towards previously generated rules, in order to improve the accuracy of the ensemble:

$$f_m(x) = f_{m-1}(x) + \nu \cdot r_m(x),$$

where $\nu \in (0, 1]$ is a shrinkage parameter.

The output of the procedure is the decision rule $r_m(x) = \alpha_m\Phi_m(x)$. Program listing in Algorithm 1 (referred to as ENDER) outlines how the ensemble of decision rules is constructed.

Algorithm 1 Ensemble of Decision Rules—ENDER

```
Input: set of training examples ${y_a, x_a}_1^N$,
       $M$—number of decision rules.
Output: ensemble of decision rules ${r_m(x)}_0^M$.

$f_0(x) = r_0(x) = 0$;
for $m = 1$ to $M$ do
  $\Phi_m = \arg\min_\Phi \mathcal{L}_m(\Phi)$
  $\alpha_m = \arg\min_\alpha \sum_{a=1}^N L(y_a, f_{m-1}(x_a) + \alpha\Phi_m(x_a)$
  $r_m(x) = \alpha_m \Phi_m(x)$
  $f_m(x) = f_{m-1} + \nu \cdot r_m(x)$
end for
```

Table 4: Mean squared error on test data.

classifier	MSE (1e-3)	improvement
Baseline	2.59	—
Ender	2.24	13.5%

5.2 Recommendation rules

The main advantage of decision rules is their simple and interpretable *if–then* form. Moreover, due to the conjunction of conditions, rules are tailored to model the interactions between attributes. That is why a prediction model based on decision rules can be used to give recommendations on how to improve the quality of an ad. From the advertiser's point of view the recommendation can be given in a contextual manner. If nearly all the conditions of a rule with a positive decision (increasing the CTR) appeared in a given ad then the system could suggest tweaking the missing part of the rule to increase the ad's score. On the contrary, if all the conditions of a rule with negative decision were satisfied the suggestion could be to remove one of the rule's conditions. Let us consider an exemplary decision rule:

IF *keywords* = {"firewall", "security"} AND *ad's title word* = "network" THEN *CTR is increased by 0.1*

The recommendation can consist of a hint of adding the word "network" to the title of the ad if keywords "firewall" and "security" are used. In other words, from the above decision rule one can derive the following recommendation rule:

IF *keywords* = {"firewall", "security"} THEN *ad's title word* = "network"

Note that rules of the form given above resemble association rules [1]. However, while association rules are focused on finding frequent itemsets (conditions), the ENDER algorithm seeks conditions with more predictive power.

5.3 Results

For the experiment we used only ads that had at least 200 impressions and were associated with one unique URL. We had in total 101 529 ads (see Table 3 for detailed numbers). Each ad was described by the features related to query terms and its URL. We split the data set into training, validation and testing parts in the proportion 25%, 25% and 50% respectively. We generated 1000 rules with $\nu = 0.1$ and $\eta = N/2$. The algorithm produced 1000 rules in 40 minutes on commodity hardware. Table 4 contains the mean squared error (MSE) on test data in a similar way as in [15]. We then compared the performance of ENDER with a baseline approach, which is just an average CTR over all training examples. One can observe that ENDER working on ad quality features achieved an improvement of 13.5%. This result is, however, not comparable to that presented in [15]—we used a different data set, specific data filtering and features (we do not have real titles and bodies of ads and had to approximate them). Figure 5 shows how the test error decreases with the number of rules. Notice that the algorithm does not overfit as the number of rules increases.

It is also interesting to see the rules generated by the algorithm and to check whether recommendations based on these rules can be formed. We present three example rules in Table 5.

Table 5: Exemplary decision rules generated from Beyond Search data set.

#	decision rule
1	IF *domain* = ".com" AND *#segments in url* = 3 AND *url length* $\in [11, 14]$ THEN *CTR is increased by 0.0087*
2	IF *term* = {*"electric", "install"*} AND *#segments in url* ≤ 4 AND *url length* $\in [5, 30]$ THEN *CTR is increased by 0.0051*
3	IF *term* = {*"women", "community", "books"*} AND *#segments in url* = 3 AND *url length* $\in [12, 26]$ THEN *CTR is increased by 0.02*

The first rule states that for ads whose domain is ".com", having standard number of segments and a normal URL length, the CTR is increased by 0.0087. The next two rules beside similar conditions concerning URL construction indicate associations between terms describing the ads. For example, the combination of terms "electric" and "install" increases the CTR. The same applies to terms "women", "community" and "books". Such rules, in our opinion, can be used in a recommendation system for ad quality improvement.

The rules have different support, i.e., they cover different fraction of training examples. Figure 6 presents a histogram of the number of rules with respect to percentage of covered objects. Note that almost half the number of rules is quite specific. The number of rules is then decreasing with increasing fraction of covered examples. It is interesting that for a fraction around 98%, the number of rules is slightly higher.

6. CONCLUSIONS

This paper describes initial results obtained from experiments on the Beyond Search data set. Our ultimate goal is to construct a powerful prediction algorithm for the CTR that can also be used in a recommendation system for improving ad quality. In this paper, we first described briefly the data set, we discussed the model of estimating CTR that takes into account the position and the page on which an ad was displayed. We used MLE for estimating the CTR and

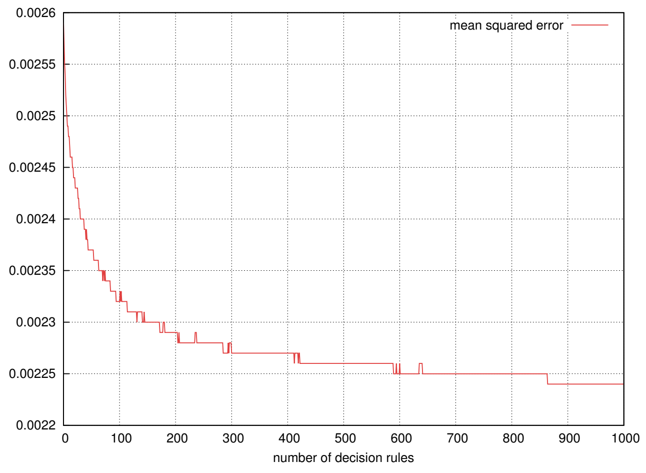

Figure 5: MSE curve on testing set up to 1000 rules.

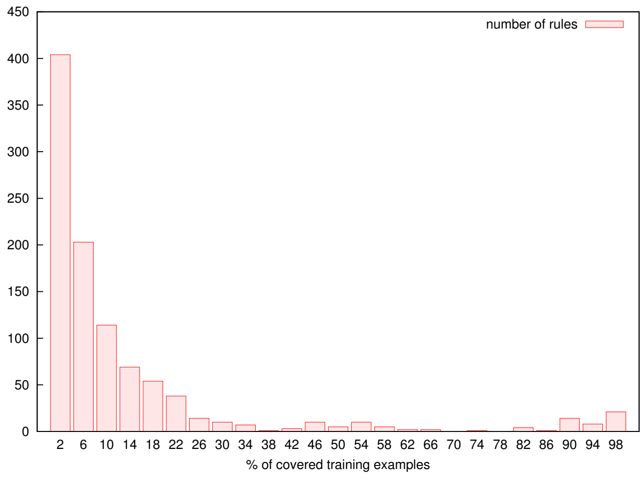

Figure 6: A histogram of number of rules with respect to percent of covered training examples.

the probability of an ad being seen on a given position and page. Finally, we presented initial studies in CTR prediction using ensemble of decision rules. The obtained rules can be further used in recommendations improving the ad quality. During the experiment we had to perform several simplifications because of the lack of important information related to ad quality (such as title and body of an ad). However, the initial results are quite promising.

In future work, we plan to gather information on ad titles and bodies to make the results more realistic and apply the ensemble of decision rules to the data set with all ads. More sophisticated methods of associating ads with their keywords would also be useful to work around the limitations of the Beyond Search data set.

Acknowledgments

We would like to thank Microsoft Research and Microsoft adCenter for supporting our research and delivering Beyond Search data set. We also thank all the contributors of the Hadoop project for their perfect work. For computations we used the HPC cluster at the Institute of Computing Science, Poznan University of Technology.

7. REFERENCES

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 1994.

[2] H. Becker, C. Meek, and D. M. Chickering. Modeling contextual factors of click rates. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, Canada*, pages 1310–1315. AAAI Press, 2007.

[3] W. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proc. of 16th National Conference on Artificial Intelligence*, pages 335–342, 1999.

[4] K. Dembczyński, W. Kotłowski, and R. Słowiński. Solving regression by learning an ensemble of decision rules. In *Artificial Intelligence and Soft Computing*, LNCS (LNAI), Zakopane, Poland, 2008. Springer.

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[6] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second price auction: Selling billions of dollars worth of keywords. *American Economic Review*, 97(1):242–259, 2007.

[7] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. of Computer and System Sciences*, 55(1):119–139, 1997.

[8] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[9] J. H. Friedman and B. E. Popescu. Predictive learning via rule ensembles. Stanford University technical report, Dept. of Statistics, 2005.

[10] J. Grimmelmann. The structure of search engine law. *Iowa Law Review*, 93, 2008.

[11] T. Hastie, R. Tibshirani, and J. H. Friedman. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2003.

[12] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 154–161, Salvador, Brazil, 2005.

[13] K. Newcomb. Search marketing shows strength in 2006. *searchenginewatch.com*, 2007.

[14] M. Regelson and D. C. Fain. Predicting click-through rate using keyword clusters. In *Proceedings of the Second Workshop on Sponsored Search Auctions*, 2006.

[15] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 521–530, New York, NY, USA, 2007. ACM.

[16] S. M. Weiss and N. Indurkhya. Lightweight rule induction. In *Proceedings of 17th International Conference on Machine Learning*, pages 1135–1142, 2000.