A Dynamic Bayesian Network Click Model for Web Search Ranking

Olivier Chapelle
Yahoo! Labs
Santa Clara, CA
chap@yahoo-inc.com

Ya Zhang
Yahoo! Labs
Santa Clara, CA
yazhang@yahoo-inc.com

ABSTRACT

As with any application of machine learning, web search ranking requires labeled data. The labels usually come in the form of relevance assessments made by editors. Click logs can also provide an important source of implicit feedback and can be used as a cheap proxy for editorial labels. The main difficulty however comes from the so called *position bias* — urls appearing in lower positions are less likely to be clicked even if they are relevant. In this paper, we propose a Dynamic Bayesian Network which aims at providing us with unbiased estimation of the relevance from the click logs. Experiments show that the proposed click model outperforms other existing click models in predicting both click-through rate and relevance.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]; H.3.5 [**Online Information Services**]; I.2.6 [**ARTIFICIAL INTELLIGENCE**]: Learning; I.6.m [**SIMULATION AND MODELING**]: Miscellaneous

General Terms

Algorithms, Design, Experimentation

Keywords

Click-through rate, Click modeling, Web search, Ranking, Dynamic Bayesian network

1. INTRODUCTION

Web page ranking has been traditionally based on hand designed ranking functions such as BM25 [18]. With the inclusion of thousands of features for ranking, hand-tuning of ranking function becomes intractable. Several machine learning algorithms have been applied to automatically optimize ranking functions [4, 5]. Machine learned ranking requires a large number of training examples, with relevance labels indicating the degree of relevance for each query-document pair. The cost of the editorial labeling is usually quite expensive. Moreover, the relevance labels of the training examples could change over time. For example, if the query is time sensitive or recurrent (e.g. "www" or "presidential election"), a search engine is expected to return the most up-to-date documents/sites to the users. However, it would be prohibitive to keep all the relevance labels up to date. Click logs embed important information about user satisfaction with a search engine and can provide a highly valuable source of relevance information. Compare to editorial labels, clicks are much cheaper to obtain and always reflect current relevance.

Clicks have been used in multiple ways by a search engine: to tune search parameters, to evaluate different ranking functions [7, 13, 14, 15], or as signals to directly influence ranking [1, 13]. However, clicks are known to be biased, by the presentation order, the appearance (e.g. title and abstract) of the documents, and the reputation of individual sites. Many studies [8, 10] have attempted to account the position-bias of click. Carterette and Jones [7] proposed to model the relationship between clicks and relevance so that clicks can be used to unbiasedly evaluate search engine when lack of editorial relevance judgment. Other research [10, 21, 16] attempted to model user click behavior during search so that future clicks may be accurately predicted based on observations of past clicks.

Two different types of the click models are *position models* [8, 10, 17] and the *cascade model* [8]. A position model assumes that a click depends on both relevance and examination. Each rank has a certain probability of being examined, which decays by rank and depends only on rank. A click on a url[1] indicates that the url is examined and considered relevant by the user. However this model treats the individual urls in a search result page independently and fails to capture the interaction among urls in the examination probability. Take for example two equally relevant urls for a query: a user may only click on the top one, feel satisfied, and then leave the search result page. In this case, the positional bias cannot fully explain the lack of clicks for the second url.

The cascade model assumes that users examine the results sequentially and stop as soon as a relevant document is clicked. Here, the probability of examination is indirectly determined by two factors: the rank of the url and the relevance of all previous urls. The cascade model makes a strong assumption that there is only one click per search and hence it could not explain the abandoned search or search with more than one clicks. Even though the cascade model is quite restrictive, the authors of that paper showed that

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

[1]We refer to url (or equivalently document) as a shorthand for the entire display block consisting of the title, abstract and url of the corresponding result.

it can predict click-through rates (CTRs)[2] more accurately than the position models described above.

None of the above models distinguish perceived relevance and actual relevance[3]. Because users cannot examine the content of a document until they click on the url, the decision to click is made based on perceived relevance. While there is a strong correlation between perceived relevance and actual relevance, there are also many cases where they differ.

In this paper, a dynamic bayesian network (DBN) model is proposed to model the users' browsing behavior. As in the position model, we assume that a click occurs if and only if the user has examined the url and deemed it relevant. Similar to the cascade model, our model assumes that users make a linear transversal through the results and decide whether to click based on the perceived relevance of the document. The user chooses to examine the next url if he/she is unsatisfied with the clicked url (based on actual relevance). Our model differs from the cascade model in two aspects: 1. because a click does not necessarily mean that the user is satisfied with the clicked document, we attempt to distinguish the perceived relevance and actual relevance. 2. We do not limit the number of clicks that a user can make during a search.

We compare the proposed model with pervious models and show that the dynamic bayesian network based model outperforms the others. The predicted relevance for each url are then used in two ways: either as a feature in a ranking function or used as supplementary data to learn a ranking function. We show that the function learned with these predicted relevance is not far from being as good as a function trained with a large amount of editorial data. We further show that combining both type of data can lead to an even more accurate ranking function.

2. MODELING PRESENTATION BIAS

As explained above, the presentation bias refers to the fact that users are more likely to click on documents at the top of the ranking.

2.1 Position models

A popular class of methods for dealing with this presentation bias problem are the position based models [8, 10, 17]. A core assumption in these methods is that the user clicks on a link if the following two conditions are met: the user examined the url *and* found it relevant; in addition, the probability of examination depends only on the position. More precisely, given a url u at position p, the probability of a click is modeled through a hidden variable E denoting if u was examined or not:

$$\begin{aligned} &P(C=1|u,p) \\ &\quad = \sum_{e\in\{0,1\}} P(C=1|u,p,E=e)P(E=e|u,p) \\ &\quad = \underbrace{P(C=1|u,E=1)}_{:=\alpha_u}\underbrace{P(E=1|p)}_{:=\beta_p}. \end{aligned}$$

[2]The click-through rate of an url is defined as the ratio between the number of times this url was clicked and the number of times it was shown.

[3]Perceived relevance is the relevance of the url presented by a search engine. Actual relevance means the relevance of the landing page.

The last equation made use of the following assumptions: there is no click if the user did not examine the url; if the url is examined, the probability of click depends only on its relevance; the probability of examination depends only the position. As a result the probability of a click is the product between two probabilities α_u and β_p: the first one models the relevance of the url to the query while the second one captures the position effect. Remember that our goal is to infer the relevance of an url based on the click logs. That is exactly what α_u represents: the perceived relevance of an url to the user, independent of the position. If we make the additional assumption that $\beta_1 = 1$ — that is the user always examine the first result — then α_u can be interpreted as an equivalent CTR at position 1, i.e. the CTR of that url had it been placed in the first position. Note that the query q is implicit here; more formally, we should write α_{uq} to stress the dependence to the query, but in the rest of paper we assume that the query is fixed.

2.1.1 COEC model

A cheap and straightforward approach is to estimate β_p as the aggregated CTR (over all queries and sessions) in position p. Suppose there are N sessions in which u appeared and for the i-th session $c_i \in \{0,1\}$ indicates if there was a click and p_i is the position in which the url u appeared. Then α_u is computed as [19]:

$$\alpha_u = \frac{\sum_{i=1}^N c_i}{\sum_{i=1}^N \beta_{p_i}}. \quad (1)$$

As in [19] we refer to this method as *clicks over expected clicks* (COEC), because the denominator can be seen as the number of "expected" clicks given the positions that the url appeared in.

The problem with the COEC model is that the estimation of β is biased. It would be valid if the search engine gives results in a random order. But since more relevant documents tend to appear higher in the ranking, the observed CTR at a given position captures not only the position bias, but also the typical relevance at this position.

2.1.2 Examination model

Another approach is to find α_u and β_p by maximum likelihood. Note that the urls need to have been shown in different positions for this approach (and other below) to be meaningful. Otherwise the solution is ill-defined. This makes sense because in order to capture the position effect, one needs to observe the CTR of the *same* url at different positions. That is usually the case because of the constant variations in a search engine. Given the vector β_p, the maximum likelihood solution for α_u is:

$$\alpha_u = \arg\max_{\alpha} \sum_{i=1}^N c_i \log(\alpha\beta_{p_i}) + (1-c_i)\log(1-\alpha\beta_{p_i}). \quad (2)$$

The vector β_p is estimated by an alternate (or joint) maximization of the likelihood between α_u and β_p.

A drawback of the above approach is that it can lead to $\alpha_u > 1$. This is not desirable since α_u is supposed to represent a probability. Instead of maximizing the likelihood directly, one can use the Expectation-Maximization (EM) algorithm where the hidden variables are the examination variables E [10]. This ensures that $\alpha_u \leq 1$. We used the

EM algorithm in our implementation of the the examination model.

2.1.3 Logistic model

Another alternative is to use a slightly different model related to logistic regression [8]:

$$P(C=1|u,p) := \frac{1}{1+\exp(-\tilde{\alpha_u}-\tilde{\beta_p})}. \quad (3)$$

The click probability is not a product of probabilities any longer, but it is still a function of the url and of the position. The main advantage is that it ensures that the resulting probability is always between 0 and 1; also the optimization is much easier since it is an unconstrained and jointly convex problem.

2.2 Cascade Model

Cascade model [8] differs from the above position models in that it considers the dependency among urls in a same search results page and model all clicks and skips simultaneously in a session. It assumes that the user views search results from top to bottom and decides whether to click each url. Once a click is issued, documents below the clicked result are not examined *regardless of the position*. With the cascade model, each document d, is either clicked with probability r_d (i.e. probability that the document is relevant) or skipped with probability (1-r_d). The cascade model assumes that a user who clicks never comes back, and a user who skips always continues. A click on the i-th document indicates: 1. the user must have decided to skip the ranks above; 2. the user deem the i-th document relevant. The probability of click on i-th document can thus be expressed as:

$$P(C_i=1) = r_i \prod_{j=1}^{i-1}(1-r_j). \quad (4)$$

3. DYNAMIC BAYESIAN NETWORK

We now introduce another model which considers the results set as a whole and takes into account the influence of the other urls while estimating the relevance of a given url from click logs. The reason to consider the relevance of other urls is the following: take for instance a relevant document in position 3; if both documents in position 1 and 2 are very relevant, it is likely that this document will have very few clicks; on the other hand, if the two top documents are irrelevant, it will have a lot of clicks. A click model depending only on the position will not be able to make the distinction between these two cases. We extend the idea of cascade model and propose a Dynamic Bayesian Network (DBN) [11] to model simultaneously the relevance of all documents.

3.1 Model

The Dynamic Bayesian Network that we propose is illustrated in figure 1. The sequence is over the documents in the search result list. For simplicity, we keep only the top 10 documents appearing in the first page of results, which means that the sequence goes from 1 to 10. The variables inside the box are defined at the session level, while those out of the box are defined at the query level. As before, we assume that the query is fixed.

For a given position i, in addition to the observed variable C_i indicating whether there was a click or not at this position, the following hidden binary variables are defined to model examination, perceived relevance, and actual relevance, respectively:

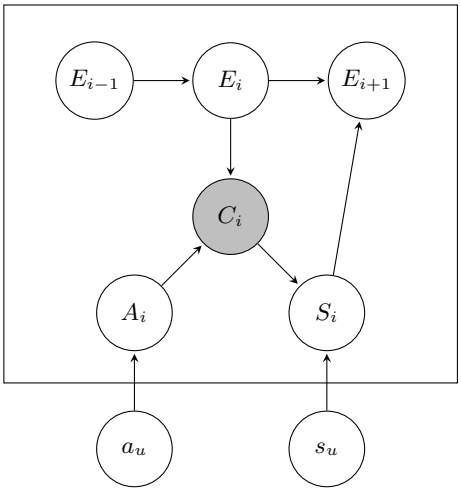

Figure 1: The DBN used for clicks modeling. C_i is the the only observed variable.

- E_i: did the user *examine* the url?
- A_i: was the user *attracted* by the url?
- S_i: was the user *satisfied* by the landing page?

The following equations describe the model:

$$A_i = 1, E_i = 1 \Leftrightarrow C_i = 1 \quad (5a)$$
$$P(A_i = 1) = a_u \quad (5b)$$
$$P(S_i = 1|C_i = 1) = s_u \quad (5c)$$
$$C_i = 0 \Rightarrow S_i = 0 \quad (5d)$$
$$S_i = 1 \Rightarrow E_{i+1} = 0 \quad (5e)$$
$$P(E_{i+1} = 1|E_i = 1, S_i = 0) = \gamma \quad (5f)$$
$$E_i = 0 \Rightarrow E_{i+1} = 0 \quad (5g)$$

As in the examination model, we assume that there is a click if and only if the user looked at the url and was attracted by it (5a). The probability of being attracted depends only on the url (5b). Similar to the cascade model, the user scans the urls linearly from top to bottom until he decides to stop. After the user clicks and visits the url, there is a certain probability that he will be satisfied by this url (5c). On the other hand, if he does not click, he will not be satisfied (5d). Once the user is satisfied by the url he has visited, he stops his search (5e). If the user is not satisfied by the current result, there is a probability $1-\gamma$ that the user abandons his search (5f) and a probability γ that the user examines the next url. In other words, γ measures the perseverance of the user[4]. If the user did not examine the position i, he will not examine the subsequent positions (5g). In addition, a_u and s_u have a beta prior. The choice of this prior is natural because the beta distribution is conjugate to the binomial distribution. It is clear that some of the assumptions are not realistic and we discuss in section 8 how to extend them. However, as shown in the experimental section, this model can already explain accurately the observed clicks.

[4]it would be better to define the perseverance γ at the user level, but we simply take the same value for all users.

Unlike the examination model, our model has *two* variables a_u and s_u related to the relevance of the document. The first one models the *perceived* relevance since it measures the probability of a click based on the url. The second one is the probability that the user is satisfied *given* that he has clicked on the link; so it can been understood as a "ratio" between *actual* and perceived relevance. Indeed, if we define the relevance of the url as the probability that the user is satisfied given that he has seen the url, we have:

$$\begin{aligned} r_u :=& P(S_i = 1|E_i = 1) \\ =& P(S_i = 1|C_i = 1)P(C_i = 1|E_i = 1) \\ =& a_u s_u \end{aligned} \quad (6)$$

As far we know, this is the first click model which attempts to model the actual relevance rather than the perceived relevance only.

3.2 Link with other models

The examination model can be seen as a special case of our model where the E_i are independent and have a distribution that only depends on the position. In that case the S_i are meaningless because they cannot be inferred.

The cascade model of [8] is a special case of our model with $\gamma = 1$ and $s_u = 1$. That is, the user keeps examining until he finds a document that appears relevant. He then clicks and stops.

In [7] the relevance of the documents is predicted from click logs and the CTRs of the *other documents* is used during the prediction. Their motivation is different (which is to evaluate the quality of search engine), but they address the same problem: the influence of other documents on CTRs. However the definition of CTRs in other positions is problematic if the set of documents changed between sessions.

Joachims [13] introduced the so-called *skip-above* pairs: when the user has not clicked at position i but has clicked at position $j > i$, it is an indication that the document in position j is preferred to the document in position i. We recover this type of pair with our model because in that case $1 = A_j > A_i = 0$. However, the problem with learning a ranking function with skip-above pairs is that one tends to learn the *reverse* function than the one in production (there are only "negative" instances in some sense).

For the position models, it has been observed that it is better to use different vectors β for different type of queries such as navigational vs informational[5]. The reason is that for navigational queries, the CTRs decay much faster with the position. But we argue that this decay is not a function of the query type, but of the quality of the top urls. For a navigational query, the top result is usually excellent and there are very few clicks in lower positions. Our click model captures this effect directly and does not need different browsing models for different type of queries.

3.3 Inference

The Dynamic Bayesian Network of figure 1 is just slightly more complex than a standard Hidden Markov Model (HMM) because of the conditional dependence between the hidden state at position $i + 1$ and the observation at position i given the hidden state at position i. The Expectaction-Maximization (EM) algorithm [9] is used to find the maximum likelihood estimate of the variables a_u and s_u and the forward-backward algorithm is used to to compute the posterior probabilities of the hidden variables. Please refer to the Appendix of the paper for a detailed derivation of the E-step and M-step, as well as a confidence computation for the latent variables. Even though we could have also estimated γ with EM, we simply treated γ as a configurable parameter for the model.

[5]A query is said to be navigational when the user intent is to reach a particular site [3].

4. EXPERIMENTS

Three types of experiments are performed to validate our model. First we evaluate the click model in terms of predicted CTR at position 1. Then we use the predicted relevance as signals in ranking. In a third type of experiments, we use the predicted relevance as targets to train a ranking function.

The click log is obtained from a commercial search engine. A session can be defined in various ways, but for all our experiments, it is defined as as follows. A session always has a unique user and unique query. It starts when a user issues a query and ends with 60 minutes idle time on the user side. For each session, we get the query, list of urls in result sets, and list of clicked urls. Simple normalization is applied to queries and urls. As explained before, we restrict ourselves to the top 10 urls on the first result page. Sessions for which the clicks are not in the same order as the ranking (for instance the user clicks first on the 4th link then on the 2nd link) have zero probability under our model. On average, roughly 3% of the sessions contain one or more out-of-the-order clicks. We could have decided to swap the click order for these sessions, but we simply discarded them. We also discarded all the queries for which we have less than 10 sessions.

4.1 Predicting click-through rate

We first assess how accurate is our proposed click model. For this task, we get 58M sessions and 682k unique queries from the click log of the UK market. A natural way to evaluate a click model would be to proceed as in [10]: for a given query, take some sessions for training and evaluate the likelihood of clicks on the other sessions. But our goal is different: we are not so much interested in click prediction, but more on how accurate the latent variables a_u and s_u are: they are indeed the variables that will be used later for ranking. It is difficult to assess the quality of s_u, but it is easy for a_u. Indeed a_u (like α_u for the examination model) is a prediction of the CTR in position 1. The following leave-one-out experimental protocol is used:

1: Retrieve all the sessions related to a given query;
2: Consider an url that appeared both in position 1 and some other positions;
3: Hold out as test sessions all the sessions in which that url appeared in position 1;
4: Train the model on the remaining sessions and predict a_u;
5: Compute the test CTR in position 1 on the held-out sessions;
6: Compute an error between these two quantities;
7: Average the error on all such urls and queries, weighted by the number of test sessions.

When computing the error between the actual CTR in position 1 and the predicted one, two types of error metrics

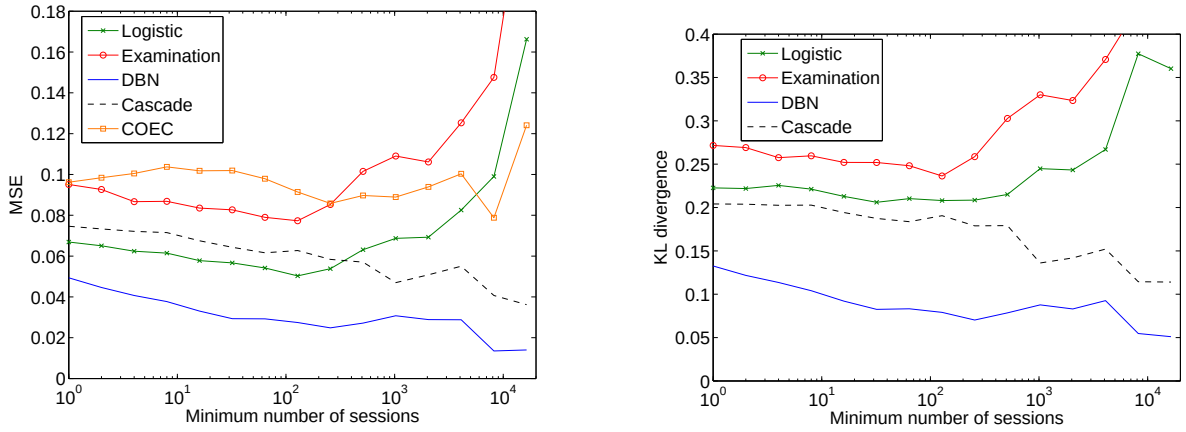

Figure 2: Accuracy of different clicks models in predicting the CTR at position 1 as measured by the mean squared error (left) and KL divergence (right). The x-axis shows the minimum number of training sessions that have been used for estimating this CTR.

are used: the Mean-Square-Error (MSE) and the KL divergence. Note that the KL divergence is, up to a constant value, the same as the likelihood of the clicks on the test sessions.

The DBN model requires the input of a parameter γ. We first perform the above leave-one-out procedure to empirically determine the optimal value of γ. The result is shown in figure 3. The DBN model reaches the smallest MSE when $\gamma = 0.9$, which indicates that users are persistent in finding relevant documents. For the rest of the experiments, we set γ to 0.9.

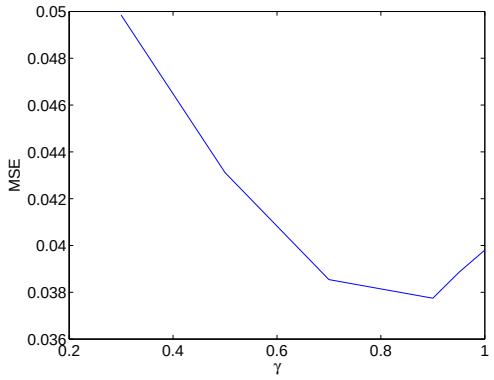

Figure 3: Mean squared error of the DBN model in predicting the CTR at position 1 as a function of γ.

We compare the DBN model with other click models described in section 2. The results are shown in figure 2. The COEC, examination, and logistic models are respectively described by equations (1), (2), and (3). For these methods we use the empirical Bayes method [6] with a beta prior to smooth the observed probabilities of clicks at different positions. This yields more stable and accurate results. More precisely, the CTRs at a given position p are assumed to be drawn from a beta distribution with parameters a_p and b_p. These parameters are found by maximum likelihood on the entire dataset. We then replace the raw CTRs of the form $\frac{N}{D}$ by their smoothed version $\frac{N+a_p}{D+a_p+b_p}$. The COEC model predicts sometimes CTR larger than 1: in that case, we clamp the prediction to 1 when computing the MSE error. However, even after clamping to 1, the KL divergence is infinite when the test CTR is smaller than 1. That is the reason why COEC does not appear in the right hand side of figure 2. As for the cascade model described by equation (4), it can only handle sessions which have exactly one click. When testing this method we thus discarded all the sessions which have zero or more than one click. While the cascade model is rather restricted by assuming that users are very persistent in examining until they find a relevant document, we see that it performs better than the COEC and examination model and comparably to the logistic model, indicating it is useful to consider the interaction between urls in the same search result page. These results are consistent with the ones reported in [8].

In figure 2, we broke down the errors as a function of the minimum number of sessions used for training. For instance 1000 means that the MSE has been computed on all the (query,urls) pairs such that the number of sessions in which this url does not appear in position 1 is at least 1000. When the number of sessions is large, the choice of the prior distributions on a_u and s_u does not play an important role. One would typically expect that the accuracy improves with the number of sessions. That is roughly the case for the cascade and our model, but surprisingly, the accuracy of the other models deteriorates when the number of sessions is very large. We investigated this issue by looking at the (query,url) pairs with a large error. An example was the query "myspace" and the url `www.myspace.com` in the UK market. The url in postion 1 should be `uk.myspace.com`, but because of variations in the ranking due to tests for instance, the url `www.myspace.com` appeared several times in position 1. For these sessions, the CTR was very high, 0.97, as expected. In other sessions, it appeared in position 2 and had a low CTR of 0.11. This low CTR is expected because the url in first position is then `uk.myspace.com` and most users do not even look at `www.myspace.com`. However, the logistic model predicted a CTR of 0.21. The predicted CTR

of the logistic model is understandable because in average there is roughly a factor 2 difference between the CTR of an url in position 1 and in position 2. The reason why there is a factor 9 for the query myspace is because the url shown in position 1 was already excellent and the users hardly looked at position 2. The examination and logistic model do not take this information into account but the DBN does: it predicted a CTR of 0.95 for `www.myspace.com`. Perfect urls in position 1 tend to happen more often for navigational queries. For this type of query, we have a lot of sessions; that explains why the bad performance of the position models is exacerbated when a large number of sessions is considered.

We have not done a formal comparison with the model of [10], but preliminary experiments show that it suffers from the same problem as the other position model described above.

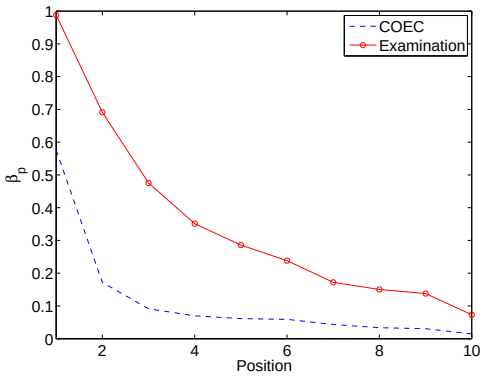

Figure 4: The β vector from COEC (1) and the examination (2) models. The former corresponds to the observed CTRs, while the latter aims at isolating the position effect in CTR modeling.

Of peripheral interest, but still noteworthy is figure 4 which compares the vector β for the two models of the form $P(C = 1|u, p) = \alpha_u \beta_p$. As explained in section 2, the values of β_p decay too fast for the COEC model, because this model does not make the distinction between the position effect and the fact that more relevant documents tend to appear at the top of the ranking. The logistic and examination models are able to make this distinction, intuitively by taking advantage of the position change of some of the urls in the click logs.

4.2 Predicted relevance as a ranking feature

The accuracy of CTR prediction may not directly translate to relevance. In the second set of experiments, we use the predicted relevance directly to rank urls. In this case, the relative order of click prediction is more important than its absolute values. We compare the DBN model with the cascade model and the logistic model. We also include in the comparison a baseline ranking function ϕ that uses many other ranking signals (e.g. BM25 scores). This function is typical of ranking functions used by web search engines.

The data is different from above: we considered only the queries for which we have editorial judgments and for which we have at least 10 sessions over a period of several months. This resulted in 3153 queries and 44.5M sessions.

The following experimental protocol is used:

1: Retrieve all the sessions related to a given query;
2: Consider all urls with editorial relevance judgments;
3: Train the model on the sessions and predict a_u and s_u for the urls;
4: Sort the urls with respect to the predicted relevance (6) given by the DBN model;
5: Compute the Normalized Discounted Cumulated Gain (NDCG) [12] at rank 5;
6: Average the $NDCG_5$ on all queries.

The results are shown in figure 5 where we broke down the $NDCG_5$ as a function of the minimum number of sessions required for a url. For instance, the minimum number of sessions of 1000 means that the $NDCG_5$ has been computed on all the (query,urls) pairs such that the number of sessions for this url is at least 1000. With the increasing number of sessions for each url, we expect the click prediction to be more accurate and more confident, leading to improved ranking of the urls. This is exactly what we have observed in figure 5 – for all the click models, $NDCG_5$ improves with the number of sessions. However, when we restrict the experiment to higher number of sessions for the urls, fewer urls are left for each query. In the extreme case, a query could contain only one url and the $NDCG_5$ would always be one. Indeed the average number of urls per query is 10.5 overall, but if we restrict ourselves to urls with more than 10,000 sessions, this number goes down to 8. The $NDCG_5$ is thus less discriminative when the number of sessions for each urls is very high and as a result, the performance of the baseline function is not constant. That is the reason why we also plot the $NDCG_5$ relative to the baseline function (right of figure 5) to remove the effect of varying number of urls per query.

In general, the DBN model is able to rank the urls better than the logistic model and the cascade model. As expected, with the increasing number of sessions for the urls (i.e. predictions are more confident), both the $NDCG_5$ and the relative $NDCG_5$ increases. As a special case of the DBN model ($\gamma = 1$ and $s_u = 1$), the cascade model behaves very similarly as the DBN model but with a lower $NDCG_5$, which confirms the necessity of introducing the notion of *satisfaction* s_u and *perseverance* γ. The cascade model suffers indeed from being able to only consider sessions with exactly one click. On the other hand, the logistic model behaves very differently than the cascade model and the DBN model. The difference of relative $NDCG_5$ between the DBN model and logistic model is large when the number of sessions is small. When the number of sessions becomes larger, the difference between DBN and logistic models gets smaller, mainly because there are less number of urls for each query.

Given the above observations, we then fix the minimum number of sessions to 10 and the minimum number of urls per query to 10. As a result 392 queries pass the criteria and on average each query contains 13.4 urls. We then compute $NDCG_5$ on this data set. As shown in table 1, the $NDCG_5$ for the DBN model is 5.8% and 2.4% better than that of the logistic model and the cascade model respectively. All the difference are statistically significant according to a Wilcoxon sign-rank test ($p \leq 0.001$). In order to quantify the influence of the satisfaction variable s_u, we also ranked according to a_u only instead of (6). The difference is only 0.5%, not statistically significant. We will discuss in section 7 an extension of our model resulting in a better modeling of the satisfaction.

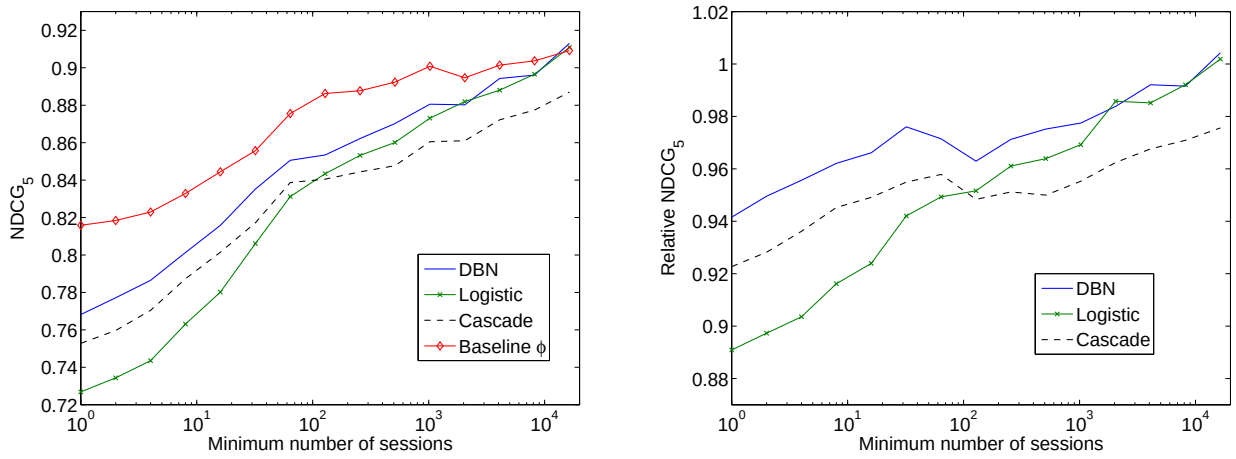

Figure 5: The predicted relevance is used as the only signal to rank urls. $NDCG_5$ is as a function of the minimum number of sessions for each url (left). The $NDCG_5$ is also plotted relative to the baseline function (right).

Table 1: $NDCG_5$ computed when requiring at least 10 sessions per url and at least 10 urls for each query. Right column shows the relative difference with respect to the DBN model.

Logistic	0.705	-5.8%
Cascade	0.73	-2.4%
DBN	**0.748**	–
DBN (a_u only)	0.744	-0.5%
Baseline ϕ	0.795	+6.3%

Compared to the baseline ranking function ϕ, the $NDCG_5$ for the DBN model is only 6.3% worse. Given the baseline function ϕ uses more than hundreds of ranking signals and is trained with rather large set of editorially labeled data, this indicates the predicted relevance are very accurate in terms of ranking.

We then try to improve the baseline function ϕ by using the predicted relevance from the DBN model as an additional ranking signal. About 0.8% $NDCG_5$ gain was achieved. Furthermore, this signal is observed to be one of the top 10 important ranking signals among hundreds of ranking signal, indicating the high correlation between the predicted relevance and relevance.

4.3 Learning a ranking function with predicted relevance

Machine learning for web search ranking has first been introduced in [4]; we follow here the gradient boosted decision trees framework applied to pairwise preferences [20]. We have two sets of pairwise preferences:

1. $\mathcal{P}_E$ comes from editorial judgments on 4180 queries and 126k urls, resulting in about 1M preference pairs;
2. $\mathcal{P}_C$ comes from our click model: we keep only 1.1M urls (corresponding to 420k unique queries) by filtering based on a threshold on the confidence (equation (8)); converting the relevance scores (equation (6)) into preferences yields about 2M pairs.

For each (query,url) pair we extract a feature vector $\mathbf{x}$. A pair $(\mathbf{x}_i, \mathbf{x}_j)$ in the preference set indicates that $\mathbf{x}_i$ is preferred to $\mathbf{x}_j$, which should ideally translates to $f(\mathbf{x}_i) > f(\mathbf{x}_j)$, where f is the ranking function.

The boosting algorithm optimizes the following objective function (see [20] for details on the boosting procedure):

$$\frac{1-\delta}{|\mathcal{P}_E|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{P}_E} \max(0, 1-(f(\mathbf{x}_i)-f(\mathbf{x}_j)))^2 + \frac{\delta}{|\mathcal{P}_C|} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{P}_C} \max(0, 1-(f(\mathbf{x}_i)-f(\mathbf{x}_j)))^2. \quad (7)$$

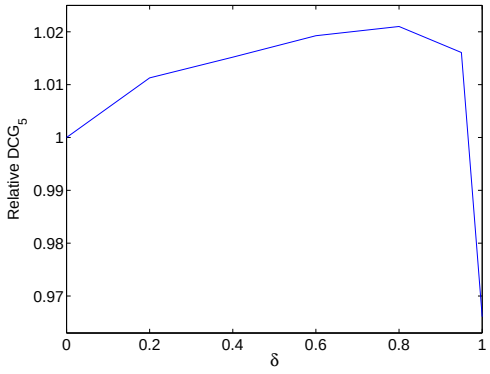

Figure 6: Relative DCG_5 for various values of δ (see equation (7)). The relative DCG_5 is normalized to be 1 for $\delta = 0$, which corresponds to learning only based on editorial judgments. On the other hand when $\delta = 1$ only click data are used for learning.

The objective function is thus a combination of editorial based and click based preference. The test set is a held

out set of editorial judgments on which the Discounted Cumulative Gain (DCG) at rank 5 is computed. The relative performance as a function of δ is plotted in figure 6.

We can draw two interesting conclusions from this plot:

1. Learning only from clicks, the DCG is only 4% worse than a standard model learned with editorial judgments; this is remarkable because in this experiment the set of editorial judgments is relatively large. This is an indication that learning from clicks can be very valuable for markets where there are few or no editorial judgments.

2. Combining both type of data lead to a 2% gain on DCG, which is considered substantial in the web search ranking community. So even in markets where a lot of editorial judgments are available, we can still leverage clicks to reach higher DCG.

Finally note that the evaluation has been done with an editorial metric. But because of the discrepancies between clicks and editorial judgments discussed below, we expect that our model trained on clicks would perform even better had we evaluated it on a click based metric.

5. A SIMPLIFIED MODEL

As we have shown in figure 3, the best prediction of the CTR at position 1 was obtained for $\gamma = 0.9$. But $\gamma = 1$ produces only slightly worse prediction. And this particular setting is interesting because in this case, the inference is much simpler. Indeed the user then keeps examining until he is satisfied, which means that the last click provided a satisfying result and the results below it were not examined. The forward-backwards algorithm and EM are thus not needed because there is no ambiguity on the examination variables: $E_1 = \cdots = E_\ell = 1$ and $E_{\ell+1} = \cdots = E_{10} = 0$ with ℓ being the position of the last click. The latent variables a_u and s_u are estimated using simple counting as described in algorithm 1.

Algorithm 1 Simplified model estimation for $\gamma = 1$.

Initialize a_u^N, a_u^D, s_u^N, s_u^D to 0 for all urls u associated with current query.
for all sessions **do**
  for all u above or at the last clicked url **do**
    $a_u^D \leftarrow a_u^D + 1$
  end for
  for all u that got clicked **do**
    $a_u^N \leftarrow a_u^N + 1$
    $s_u^D \leftarrow s_u^D + 1$
  end for
  $s_u^N \leftarrow s_u^N + 1$, where u is the last clicked url.
end for// $\alpha_a, \beta_a, \alpha_s, \beta_s$ are prior Beta parameters for a_u and s_u.
for all urls u **do**
  $a_u = (a_u^N + \alpha_a)/(a_u^D + \alpha_a + \beta_a)$.
  $s_u = (s_u^N + \alpha_s)/(s_u^D + \alpha_s + \beta_s)$.
end for

6. CLICKS VS EDITORIAL JUDGMENTS

In the last two sets of experiments, clicks are used as a proxy for editorial judgments. It is important to evaluate the correlation between relevance estimated by our click model and actual relevance given by editors. A natural measure is the number of contradicting pairs (related to Kendall's tau test). We converted all the editorial judgments in pairwise preferences and did the same from the relevance scores extracted from our model. On the intersection of both sets of pairs, there is a disagreement in the preference for 20% of the pairs. We investigated the reasons for these discrepancies and found that excluding errors in editorial judgments, theses reasons can be summarized into two main categories:

1: popularity is not necessary aligned with relevance;
2: clicks measure mostly perceived relevance, while editors judge the relevance of the landing page;

An example for the first category is the query "adobe": the home page `www.adobe.com` seems to be the most relevant url but most users click on the acrobat reader link. Another example query is "bank of america". Most users prefer to click on the online banking page `http://www.bankofamerica.com/onlinebanking/` while editors tend to consider the home page `http://www.bankofamerica.com` as the destination page for this query. Given this inherent discrepancy between relevance and clicks, we may never be able to close the gap completely. On the other hand, it may be useful to leverage the predicted relevance to refine the definition of relevance and hence the guideline for editorial relevance judgment.

The second type of inconsistency can be further divided in two sub-categories: cases where relevance of the search result snippets are very different from that of the landing page; and cases where users click based on the trustworthiness of the page rather than the relevance of the page. The first sub-category most often is related to the presentation of the title and summary of the url. An example query for the second sub-category is "travel insurance". While there are many small insurance companies focus on selling travel insurance (more relevance in terms of relevance judgment), the users still tend to click more often on the sites of branded insurance companies, where travel insurance is only a small fraction of their business.

In summary this study made clear that defining relevance is a complex matter and that clicks and editorial judgments are two related but distinct ways of answering the user needs.

7. EXTENSION

So far all the above experiments only considered web search results. In fact, a wealth of information has been blended into the search result page: the most noticeable one is sponsored search results.In addition, search engines nowadays tend to contain many links to help users to quickly navigate to their search destination, including related search, query mis-spelling suggestion and short cuts.

We discovered that in many cases, lack of clicks on search results is due to the fact that users have chosen to click on urls from one or more of the above sections. We thus would like to model the clicks on the entire search result page. We modify the DBN model slightly to consider the whole page clicks. Previously we only considered the top 10 retrieved results in a session. Here we define two virtual urls: the leading url defined as the urls at the top of search result page (e.g. sponsored search, spelling suggestion); the trailing url defined as the urls at the bottom of the search result page (e.g. pagination).

Clicks on the leading url may suggest that the users never examined the urls in the search result section; while clicks on the trailing url indicates the users most likely are unsatisfied with previous urls. Again, we use the predicted relevance from the refined DBN model to rank urls and compute $NDCG_5$. The results are summarized in table 2. As for table 1, we only keep urls with at least 10 sessions and the queries with at least 10 urls. The refined DBN model outperforms the original DBN model by 2.2% and the difference is statistically significant ($p \leq 0.001$).

Table 2: $NDCG_5$ for DBN when considering whole page relevance. Setting is the same as table 1. Right column shows the relative difference with respect to the improved DBN model.

DBN – 10 nodes	0.748	-2.2%
DBN – 12 nodes	**0.765**	–
DBN – 12 nodes (a_u only)	0.756	-1.2%
Baseline ϕ	0.795	+3.9%

In addition, for this improved DBN model, the satisfaction variables s_u seem to be better estimated – there is a 1.2% drop ($p \leq 0.001$) in accuracy if we rank according to a_u only (as compared to 0.5% for the original model, see table 1). This is probably because we now model clicks on the bottom of search page such as 'next': when a user click on the next button, it is likely that he is not satisfied with the last url he visited. Our original DBN ignored this fact and incorrectly attributed high satisfaction for this kind of url.

8. CONCLUSION AND FUTURE WORK

Extracting relevance information from click logs is a challenging but valuable task for web search ranking. In this paper we have proposed a novel click model based on dynamic bayesian network. The major contribution of the work is to introduce the notion of satisfaction to separately model the relevance of the landing page and perceived relevance at the search result page. We have demonstrated in this paper that the DBN model outperforms other click models.

There are several extensions which can improve the accuracy of our model. Other than the preliminary experiments we have done to consider urls in the entire search result page for click modeling, another extension is to incorporate the time users spent on a page, which is expected to be very helpful in predicting the user satisfaction. We can also allow the user to be satisfied even if he does not click (e.g. he might have fulfilled his request just by reading the abstract). In addition, the satisfaction variable can be continuous instead of binary: for informational queries, the user typically finds bits of information on each page and stops when his overall information need is fulfilled. This can be done by introducing a dependency between the S_i variables. Finally, a more challenging extension is to consider a non-linear examination model: this would require to model both forward and backward jumps.

Most of existing click modeling methods are biased by the search engine used to collect the clicks, and they mostly serve as 'positive feedback': if a document was never presented to the user, then the document would not be clicked. Another direction to extend the work would be to utilize the query smoothing to infer relevance for extra documents.

9. ACKNOWLEDGMENTS

The authors would like to thank Ralf Gutsche for valuable assistance on click data processing. The authors also wish to thank Georges Dupret, Narayanan Sadagopan and Belle Tseng for insightful discussions.

10. REFERENCES

[1] E. Agichtein, E. Brill, S. Dumais, and R. Ragno. Learning user interaction models for predicting web search result preferences. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR)*, pages 3–10, 2006.

[2] M. Beal and Z. Ghahraman. Variational bayesian learning of directed graphical models with hidden variables. *Bayesian Analysis*, 1(4):793–832, 2006.

[3] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[4] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

[5] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking svm to document retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in informat ion retrieval*, 2006.

[6] B. Carlin and T. Louis. *Bayes and Empirical Bayes Methods for Data Analysis*. Chapman & Hall/CRC, 2000.

[7] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 217–224. MIT Press, 2008.

[8] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM '08: Proceedings of the international conference on Web search and web data mining*, pages 87–94. ACM, 2008.

[9] N. M. Dempster, A. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:185–197, 1977.

[10] G. Dupret and B. Piwowarski. User browsing model to predict search engine click data from past observations. In *SIGIR 08: Proceedings of the 31st Annual International Conference on Research and Development in Information Retrieval*, 2008.

[11] Z. Ghahramani. Learning dynamic bayesian network. In C. L. Giles and M. Gori, editors, *Adaptive processing of temporal information*, Lecture notes in artificial intelligence. Springer-Verlag, 1998.

[12] K. Jarvelin and J. Kekalainen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[13] T. Joachims. Optimizing search engines using clickthrough data. In *ACM SIGKDD Conference on*

Knowledge Discovery and Data Mining (KDD), pages 133–142, 2002.

[14] T. Joachims. Evaluating retrieval peformance using clickthrough data. In *Text mining*, pages 79–96, 2003.

[15] T. Joachims, L. A. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161, 2005.

[16] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web (WWW)*, pages 521–530, 2007.

[17] 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. ACM, 2007.

[18] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in informat ion retrieval*, 1994.

[19] V. Zhang and R. Jones. Comparing click logs and editorial labels for training query rewriting. In *Query Log Analysis: Social And Technological Challenges. A workshop at the 16th International World Wide Web Conference*, 2007.

[20] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In *Advances in Neural Information Processing Systems 20*, pages 1697–1704. MIT Press, 2008.

[21] D. Zhou, L. Bolelli, J. Li, C. L. Giles, and H. Zha. Learning user clicks in web search. In *International Joint Conference on Artificial Intelligence (IJCAI07)*, 2007.

APPENDIX

We give here some details about the inference in our DBN outlined in section 3.3. Suppose that there are N sessions and denote A^j, S^j and E^j the vector of hidden variables associated with the j-th session. Also let d_i^j be the url in position i of the j-th session.

M step

Given some posterior distributions $Q(A_i^j)$ and $Q(S_i^j)$ on the hidden variables, the update of a_u and s_u are as follows:

$$a_u = \arg\max_a \sum_{j=1}^{N}\sum_{i=1}^{10} I(d_i^j = u)$$
$$\Big(Q(A_i^j = 0)\log(1-a) + Q(A_i^j = 1)\log(a)\Big) + \log P(a).$$

$$s_u = \arg\max_s \sum_{j=1}^{N}\sum_{i=1}^{10} I(d_i^j = u, C_i^j = 1)$$
$$\Big(Q(S_i^j = 0)\log(1-s) + Q(S_i^j = 1)\log(s)\Big) + \log P(s).$$

In the above equations, I is the indicator function; $P(a)$ and $P(s)$ are the prior beta distributions. We simply took a beta distribution with parameters (1,1), but these priors can be learned using a variational approximation [2]. The maximizers can of course be easily computed in closed form. Because of the priors, this EM algorithm does not converge to the maximum likelihood solution but to a mode of the posterior: it is a maximum a posteriori (MAP) solution.

E step

The M steps consists in computing the posterior probabilities:

$$Q(A_i^j) := P(A_i^j | C^j, a_u, s_u, \gamma)$$
$$Q(S_i^j) := P(S_i^j | C^j, a_u, s_u, \gamma).$$

In the rest of this section, we drop for convenience the conditioning on a_u, s_u and γ. As in the forward-backward algorithm, we define the following variables:

$$\alpha_i(e) = P(C_1^j, \ldots, C_{i-1}^j, E_i = e),$$
$$\beta_i(e) = P(C_i^j, \ldots, C_{10}^j | E_i = e)$$

And one can easily derived the recursion formula:

$$\alpha_{i+1}(e) = \sum_{e' \in \{0,1\}} \alpha_i(e') P(E_{i+1} = e, C_i | E_i = e').$$

$$\beta_{i-1}(e) = \sum_{e' \in \{0,1\}} \beta_i(e') P(E_i = e', C_{i-1} | E_{i-1} = e).$$

The conditional probabilities in the above equation are computed as follows:

$$P(E_{i+1}, C_i | E_i) = \sum_{s \in \{0,1\}} P(E_{i+1} | S_i = s, E_i) P(S_i = s | C_i) P(C_i | E_i).$$

Confidence

Remember that the latent variables a_u and s_u will later be used as targets for learning a ranking function. It is thus important to know the confidence associated with these values. A standard way of deriving a confidence is to compute the second derivative of the log likelihood function at the MAP solution. This can be seen as doing a Laplace approximation of the posterior distribution.

The second derivative in our case turns out to have a simple expression because $P(C^j | a_u, s_u, \gamma) = \sum_{a^j, e^j, s^j} P(C^j | A^j = a^j, E^j = e^j) P(A^j = a^j, E^j = e^j, S^j = s^j | a_u, s_u, \gamma)$ is linear in a_u and s_u. The result is simply the average squared gradient (similar equation stands for s_u):

$$\frac{\partial^2}{\partial a_u^2} \sum_{j=1}^{N} \log P(C^j | a_u, s_u, \gamma) = \sum_{j=1}^{N}\sum_{i=1}^{10} I(d_i^j = u) \left(\frac{Q(A_i^j = 1)}{a_u} - \frac{Q(A_i^j = 0)}{1 - a_u} \right)^2. \quad (8)$$